1 2 90

UNIVERSIDADE D
COIMBRA

Rafael Lopes Belo Baptista

FRAMEDROP - MOBILE CLIENT

Dissertation in the context of the Master's in Informatics Engineering, specialization in Information Systems, advised by Professor Frederico Cerveira and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

July of 2023

1 2 9 0

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Rafael Lopes Belo Baptista

Framedrop - Mobile client

Dissertation in the context of the Master's in Informatics Engineering, specialization in Information Systems, advised by Professor Frederico Cerveira and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

July of 2023

Acknowledgements

First and foremost, I would like to express my deepest gratitude and appreciation to Professor Frederico Cerveira and Pink Room's CEO Mário Gago for their valuable insights, constructive feedback, and dedication to the quality of this work. A huge thank you to my colleagues from Pink Room for their support and mentorship throughout the project, and to my colleagues from Framedrop for always being available to contribute positively to the success of this work. Your kindness and passion for the profession was very important to motivate me to reach my goals.

To my family, I owe a debt of gratitude for their unconditional love, encouragement, and steadfast belief in my abilities. My father's example of sacrifice has taught me the importance of hard work and perseverance in achieving success. My mother's passion for her own pursuits has inspired me to find meaning and motivation in my work, even in the face of challenges. And to my sister, who has shown me that knowledge is a powerful tool on the path to success, I am grateful for the reminder to never stop learning and growing.

To my course friends that I own a lot, for making my journey through academic life incredible. Special thanks to Nuno Tiago, João Carnide, and Beatriz Ribeiro, who have been by my side from the beginning, offering genuine friendship and unwavering support. To Bernardo, Faria, Filipe, Gonçalo, Miguel, and Santana, thank you for the wonderful memories and the camaraderie that has made my academic life unforgettable. I would also like to acknowledge the support of my dear home friends Daniel Alves e Beatriz Mendes for the moments of comfort, incessant help and care.

I humbly extend my deepest appreciation to my girlfriend, Lúcia Santos, for her incomparable support and encouragement throughout the completion of this report. Her kindness and understanding have been a constant source of motivation, pushing me to strive for excellence. From all the valuable insights and suggestions, her contributions have significantly enhanced the quality of this work. I am incredibly grateful for her presence in my life, and I cannot thank her enough for being my shelter during this process. Her patience, understanding, and belief in me have truly made a difference, and I am fortunate to have her by my side.

For all that marked my journey positively, I really appreciate your presence. These past years have been amazing!

Abstract

Today's entertainment comes in all kinds of forms and segments, where finding the appropriate content can be challenging since it is dispersed in a tangled, ever-growing chaos of information. The goal of this dissertation is to create an Android mobile client application that focuses on providing entertainment through gaming content and promoting the interactions between gaming communities, which enables content creators to grow and brings fans closer together.

The proposed mobile application serves as a comprehensive solution, considering market trends and business factors while emphasizing the consumption of short-form gaming clips. The research conducted for this project encompassed a thorough analysis of market competitors and substitutes, along with a study of state-of-the-art architectural patterns. These findings guided the decision-making process, including the selection of video protocols and other technological considerations, which were crucial during the subsequent development phase.

The preparatory study for the development of this mobile application focused on the analysis of market competitors and substitutes, as well as a state of the art of the architectural patterns. Based on these findings, a decision was made regarding video protocols and other technological concerns to be used during the development phase. The development proceeded according to the plan outlined in this report, and presents the results of the implementation, testing and other processes that were conducted such as usability tests and optimization processes. The application's performance, functionality, and user experience were all taken into account, ensuring a well-rounded and refined end product.

Keywords

Entertainment, gaming, software development, mobile development, android application.

Resumo

Atualmente, o entretenimento surge em todo o tipo de formas e segmentos, onde encontrar o conteúdo apropriado pode ser um desafio, uma vez que está disperso num total caos de informação. O objetivo desta dissertação é criar uma aplicação cliente móvel que se concentre em fornecer entretenimento de conteúdos de jogos e promover as interações entre as suas comunidades, o que permite que os criadores de conteúdos cresçam e aproximem os fãs.

A aplicação móvel proposta serve como uma solução abrangente, tendo em conta as tendências do mercado e os fatores de negócio, ao mesmo tempo que enfatiza o consumo de clips de jogos de curta duração. A investigação realizada para este projeto englobou uma análise exaustiva dos concorrentes e substitutos do mercado, bem como um estudo dos padrões de arquitetura mais avançados. Estas conclusões orientaram o processo de tomada de decisões, incluindo a seleção de protocolos de vídeo e outras considerações tecnológicas, que foram cruciais durante a fase de desenvolvimento subsequente.

O estudo preparatório para o desenvolvimento desta aplicação móvel centrou-se na análise dos competidores e substitutos do mercado, bem como num estado da arte dos padrões de arquitetura. Com base nestas conclusões, foi tomada uma decisão relativamente aos protocolos de vídeo e outras preocupações tecnológicas a serem utilizadas durante a fase de desenvolvimento. O desenvolvimento decorreu conforme o plano delineado neste relatório e apresenta os resultados da implementação, dos testes e de outros processos que foram realizados, como os testes de usabilidade e os processos de otimização. O desempenho, a funcionalidade e a experiência do utilizador da aplicação foram todos tidos em conta, garantindo um produto final completo e refinado.

Palavras-chave

Entretenimento, jogos, desenvolvimento de software, desenvolvimento mobile, aplicação android.

Contents

Acronyms

AAB - Android App Bundle

AOT - Ahead-of-Time

ART - Android Runtime

API - Application Programming Interface

APK - Android Application Pack

CSS - Cascading Style Sheets

CPU - Central Processing Unit

GPS - Global Positioning System

GPU - Graphics Processing Unit

HTML - HyperText Markup Language

IOS - IDevice Operating System

JIT - Just-In-Time

JS - JavaScript

KMM - Kotlin Multiplatform Mobile

MVC - Model-View-Controller

MVI - Model-View-Intent

MVP - Model-View-Presenter

MVP - Minimum Viable Product

MVVM - Model-View-ViewModel

NFR - Non-Functional Requirement

OHA - Open Handset Alliance

PGO - Profile Guided Optimization

PO - Product Owner

QA - Quality Assurance

SDLC - Software Development Life Cycle

SM - Scrum Master

SSoT - Single Source of True

USB - Universal Serial Bus

UX - User Experience

List of Figures

List of Tables

Chapter 1

Introduction

The following dissertation is written in the context of the Master's degree in Informatics Engineering, specialization in Information Systems, from the University of Coimbra. Here we analyze the results of the curricular internship, proposed by Pink Room, in collaboration with Framedrop, to develop an Android mobile application.

1.1. Motivation

Oxford's dictionary defines the word “entertainment” as “the action of providing or being provided with amusement or enjoyment” [1]. This “action” is something that we often get involved in or, due to unpleasant situations, we involuntarily crave. Society and technology have evolved significantly, regarding entertainment, where, in the old days, this could come in the form of works of “art, music, theater, opera and the necessary paraphernalia for his extravagant celebrations, dances, banquets, hunting, falconry, horse riding” [2]. These sources of entertainment could only be shared if people were there to see it in person. Even though it’s still doable nowadays, these forms of entertainment have become more and more forgotten, with the emergence of other sources: the Internet.

XXI Century Entertainment

The Internet empowered user entertainment consumption around the world, by breaking the distance limitation, with “millions of users who are searching for both information and entertainment” [3], which drove to “an overload of entertainment available for them to consume because of how easy it is to upload content and share it” [3]. According to the Statista research department, “as of April 2022, there were more than five billion internet users worldwide, which is 63.10% of the global population” [4], resulting in a giant pool of digital content.

"The Internet is the most important single development in the history of human communication since the invention of call waiting."

Dave Barry

The Global Internet Phenomena Report of January 2022, done by SANDVINE, showed that the majority (53.72%) of Internet traffic comes from video streaming. "1 Terabyte per month power users are growing sharply – driven by gaming, videoconferencing and video streaming" [5].

The Rise of Smartphones and Streaming Media

It is undeniable that cell phones have undergone remarkable innovation over the years. Counting only from the year 2000, we can already notice that we went from a simple device that sent messages and received calls to something that is a fusion of several other devices we use regularly, but in a portable and appealing version.

The evolution of smartphones was parallel to video streaming, which appeared on June 24th 1993, when a live video stream featured the band "Severe Tire Damage". Performing from the patios of Xerox PARC[1], they broadcast onto the Internet Multicast Backbone (the "MBone") [6]. Because the world press was watching the broadcast, "Severe Tire Damage" was suddenly very popular. "Articles appeared in The Wall Street Journal, The New York Times, The Washington Post, The Boston Globe, and foreign media such as British Sky TV" [7], and one year later, the Rolling Stones were also doing their first live streaming Internet show.

Streaming media has become even more popular over the years, and in 2005, YouTube was launched. In October of that year, a video of Ronaldinho getting a pair of "golden boots" was hitting 1 million views. In December, the website was receiving 8 million views a day [8].

[1] Xerox PARC – Xerox Palo Alto Research Center (PARC) was a major research division of Xerox Corporation based in Palo Alto, California, in the United States. PARC was founded in 1970 and became a stand-alone company in 2002. It is famous for having been the birthplace of inventions such as the graphical user interface (GUI) for personal computers, used by Apple Computer in the Macintosh and later popularized by other operating systems.

Multimedia content creation and consumption was already beginning to dominate Internet traffic, driven by the creation of Justin.tv in 2006, which would later become Twitch. Netflix showed up in 2007 and Periscope in 2015, which allowed users to consume and share live videos via their mobile devices. Facebook enabled live streaming in 2016 and in 2020, more than 1/3 of all Internet users were gaming or social live streamers [9].

1.2. Pink Room

Pink Room is a software development studio that started in 2018 and focused on the mobile development of Android and iOS applications. Since then, it has worked in a variety of industries, such as Fintech, Medtech, and the Automotive Industry, and has specialized in building digital products for mobile platforms, using both native and cross-platform technologies.

1.3. Framedrop

Framedrop was founded in 2021 and develops cloud-based solutions for the live streaming industry - it relies on Artificial Intelligence, mainly Computer Vision and Audio Signal Analysis, to automate the process of content curation and clipping of long-form broadcasts. Their product is a web app that enables Twitch streamers to collect their highlights without having to exhaustively search through hours of stream content. This process of capturing highlights during streams is made using Artificial Intelligence and Machine Learning.

1.4. Objectives

During the course of this internship a mobile application is going to be developed, aligned with Framedrop's value proposition of providing entertainment to gaming communities in the form of short media content. This Android mobile application will allow users to engage with Twitch streamers' short highlight clips, by upvoting (liking), commenting and sharing, as well as claiming clips by unlocking them with in-app credits. Framedrop's application clips come from their server which stores the streamers' highlights collected during streaming events.

1.5. Document Structure

This document starts with a brief introduction on Chapter 1 that defines the motivations and the objectives for this project as well as a contextual perspective, regarding the background for the topic and the field of study.

Moving to Chapter 2, State-of-the-Art, where the different product competitors, substitutes, and the most recent features implemented at this stage are defined and analyzed. As a development approach, a state-of-the-art analysis regarding architectural patterns and technological stack is also described here, as well as the conclusions and decisions made through the process, in the context of this project. Decisions on the different aspects of the product are justified with the previous analysis detailed in the following chapters.

Chapter 3 describes the project's approach and decisions for specifying requirements, selecting development strategies, and other events that occur during the project life cycle, such as code reviews, quality assurance, and techniques for integrating, delivering, and deploying the various versions of the product.

In Chapter 4, the assumptions, constraints, non-functional requirements, functional requirements, and elements out of scope for the project are outlined and detailed, as well as the process of wireframing and prototyping.

In Chapter 5, the architecture of the application is thoroughly described and discussed. It provides a comprehensive overview of the design and layout of the system, including the different components and how they interact with one another.

Chapter 6 outlines the plan for this assignment, including the high-level estimated and accomplished plan for each semester.

Chapter 7 presents the outcomes of the development work, providing a comprehensive overview of the major challenges encountered during the process. It includes an application showcase that highlights the iterative nature of the project, as well as an examination of the testing procedures employed. Additionally, the chapter delves into the details of the usability testing conducted and the decisions that were informed by the results of the test.

Chapter 8 concludes this document, highlighting the challenges and learning acquired during the internship as well as the project's future work.

Chapter 2

State of the Art

The world of app development is constantly evolving, with new technologies and best practices emerging all the time. As a result, apps are now more powerful and feature-rich than ever before, offering users a wide range of functionality and convenience. From social media and entertainment apps to productivity tools and utilities, there is an app for almost any purpose. And with the app market growing at an incredible pace, there is no shortage of options for users to choose from. This chapter presents the study performed on the base concepts of gaming communities and a market analysis of competitor and substitute products on how they access user's needs and how Framedrop's application might become successful amidst the competition based on a discussion of their business model and functionalities.

2.1 Gaming Spectator Culture

Gaming experience can be divided into two main types, self-play and watching others play, although they are not independent from each other. Users have traditionally derived pleasure from watching others play, which in the early stages of the internet was limited to the "people's houses with friends" [10].

Internet speed and adoption proliferation enhanced the gaming communities since it was shown that in November 2021, "the typical mobile internet user enjoyed a median download speed of 29 Mbps, which is plenty fast enough to stream a 4K movie without any buffering" [11].

The science behind the act of watching others play can be found in the neuroscientific phenomenon of mirror neurons. These neurons, discovered over twenty years ago "in the ventral premotor region F5 of the macaque monkey" [12], are a part of how we generate our own actions and how we monitor and interpret actions from others [12]. These brain cells play a role in our capacity to mimic. The known concept of "learning by watching others" [13] is evidenced in the baby's ability to copy sounds and facial expressions. Many game viewers watch others play so they can learn new skills, and acquire new knowledge to improve their gaming experience. Other reasons why a spectator community is becoming popular are the fast access to gaming content, escapist need from daily routines, tension release and the difficulty of owning equipment to self-play, due to high costs, time, etc [14].

2.2 Market Competitors

The analysis of existent and significant market competitors, as well as potential competitors can be “an important input to forecasting future industry conditions” [15]. Despite the clear need for “sophisticated competitor analysis in strategy formulation” [15], such analysis can create dangerous assumptions. A competitor can be seen as a company that wants the same goals and produces “the same or a similar product” [16]. A good example of competitor products is Coca-Cola and Pepsi. If the user chooses a product that uses another technology but it can fulfill the same needs it is called a substitute product [17]. If the product is for example a restaurant, then a coffee shop would be a substitute product because it does not compete directly with the restaurant (it provides different goods), but users could achieve their goal of having a meal [18].

Companies can think that they know their competitors because they compete with them every day, or discourage the idea of competitors' systematic analysis. This lack of awareness of the competitors can be motivated by the difficulty of systematic collection of data, much of which is not easy to find, leading to informal impressions, conjectures and intuitions, which translates into an incorrect analysis of the competition [15].

2.2.1. Market Analysis

In order to develop a product that caters to the needs and preferences of our target audience, an analysis of market competitors and substitutes was conducted. For the competitor products, we analyzed Medal.tv[2], Outplayed[3], Hover[4], and Powder[5]. For the substitute products, our analysis englobed Youtube/Youtube Gaming[6], Facebook Gaming[7],

[2] Medal - https://medal.tv/

[3] Outplayed - https://outplayed.tv/

[4] Hover - https://hover.gg/app/discover

[5] Powder - https://powder.gg/

[6] Youtube - https://www.youtube.com/

[7] Facebook Gaming - https://www.facebook.com/gaming

Tiktok[8], and Caffeine.tv[9]. The extended analysis is presented in Appendix A. This comprehensive evaluation aimed to gain insights into the distinct product features offered by competitors and understand how these features align with the preferences of our shared end users. By closely examining the strategies employed by other market players, we were able to identify the key features that resonate with our target audience and determine the desired attributes for our own product.

2.2.2. Feature Analysis

Since Framedrop is developing a proof of concept application to analyze market attraction, the following features represent a Minimum Viable Product (MVP), with the goal of validating the product's value proposition and gathering feedback from early adopters as quickly as possible. This allows the company to make informed decisions about what features to add or remove, and how to improve the product based on real-world usage [19].

Authentication via Twitch

Since Framedrop works with Twitch streams highlights and user information, an authentication via Twitch can provide connectivity between applications to manage user data and provide security to the application.

Watch Clips

Representing the main feature of the application, users can watch clips from all streamers. Further work will focus on creating different clip consumption modes such as being suggested to the viewer, coming from a particular user or game, and coming from users being followed.

Interaction with clips

Users may like videos (represented in Framedrop with an "Upvote" expression), as well as engage in commenting and sharing. In upcoming developments, users will be empowered to duplicate and personalize videos according to their preferences, represented by a "Fork" symbol within Framedrop.

[8] TikTok - https://www.tiktok.com/

[9] Caffeine.tv - https://www.caffeine.tv/

Search for clips and users

Through the use of the mobile application, users can search for other users. As the application evolves, we will receive feedback to validate if this decision is adequate or if users tend to search for other characteristics (clips, specific game, tags, etc).

Claim clips

When users scroll through the app watching videos, some of them may be locked, waiting for a user to claim them (unlocked videos are public to everyone to watch). Unclaimed clips are submitted to an auction, so users can bid them with in-app tokens.

Clip contest

As the users interact with other user’s videos by claiming them, they are able to participate in contests for specific games and earn in-app tokens if their videos are placed in top scoring positions.

Top-up in-app wallet

As mentioned above, tokens are used to bid unclaimed clips. Therefore, users should be able to buy these tokens through the app using a conventional payment system.

2.3 Platform Analysis and Mobile Application

The company made a strategic decision during the project's inception to adopt the Kotlin programming language and exclusively focus on developing the mobile app for the Android platform. While this decision imposed certain constraints on the project, a thorough analysis was conducted to investigate the rationale behind this choice and better comprehend the dynamics of the two major mobile platforms.

Android and iOS are the two dominant mobile operating systems on the market today. Both have their own unique features and capabilities, and each has its own loyal user base.

One of the key differences between Android and iOS is the market share of each platform. According to StatCounter, at the time of this report, Android has a market share of 67.56% globally, while iOS has a market share of 31.60% [20]. This means that there are significantly more Android devices in use around the world than iOS devices.

Another key difference between Android and iOS is the range of devices available on each platform. Google acquired the Android operating system from Android Inc. in 2005, with the goal of developing a user interface for touchscreen devices. This development was carried out in collaboration with the Open Handset Alliance (OHA), a group of developers sponsored by Google. OHA consists of a number of partner companies, including Samsung, HTC, Lenovo, and other smartphone manufacturers [21], so the Android devices are available at a variety of price points, mostly due to market competition. This means that consumers have a lot of choices when it comes to selecting an Android device, in contrast, iOS devices are only made by Apple.

When it comes to app stores, Android and iOS both have a large selection of apps available. However, there are some key differences between the two platforms. For example, the Google Play Store has a larger selection of apps overall, being the largest app store with 3.3 million Android apps as of the first quarter of 2022. In the same period, the App Store hosted 2.11 million apps, which makes it the second-largest on the market [22].

One of the main consequences of the disparity between App Store and Play Store number of available applications is due to the deploying process to these platforms. The submission process is similar, where publishers must first create a developer account in order to submit their apps to the respective platforms. However, it is not free to register as a developer. Google Play requires a one-time payment of $25, while the annual fee for Apple's platform is $99. These fees may vary based on the region and currency. It is important to note that maintaining developer status on Apple's platform requires the annual payment to be made [22]. Apple also performs a strict app review process, beginning with the analysis of each app element, “which include dimensions, design, keywords” [22], among others. The app review process on Google Play is faster than on other platforms, where the standard time to be reviewed and approved on Google Play Store is typically three days, however, it can take longer for certain developer accounts [23]. Although developers need to follow some guidelines, according to the Google Play Store App Criteria, they are not so severe compared to Apple’s [23].

Framedrop's mobile application will be built using Steve Blank's Customer Development framework, which consists of four steps that include identifying the need(s) that customers have, creating a product to fulfill that need(s), testing the most effective methods for acquiring and converting customers, and properly allocating resources within the organization to meet the demand for the product [24]. Therefore, in order to determine whether the product meets the needs of customers, it is important to choose a platform with a large number of users (the larger the sample size, the more accurate the results). Since we lack access to the specific number of Twitch users on each platform, it is necessary to make the assumption that the Android platform potentially harbors a greater number of users. This assumption is based on the global dominance of Android, with a larger user base compared to iOS. This said, the first version of the product should be developed for Android.

When developing mobile applications, there are two main approaches, native and hybrid. Native apps are developed specifically for one platform, using the platform's native programming language and tools. Hybrid applications, on the other hand, are developed using web technologies such as HyperText Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript (JS), or more modern frameworks like React, and can be deployed on multiple platforms.

One of the main advantages of native apps is that they are typically faster and more responsive than hybrid apps, as they are designed to take full advantage of the capabilities of the platform they are running on, such as the camera, Global Positioning System (GPS), and push notifications. Hybrid apps, on the other hand, have the advantage of being able to run on multiple platforms with minimal changes, however, hybrid apps may not perform as well as native apps, and they may not have access to all the features and functionalities of the device [25].

Hybrid apps can present some challenges when it comes to User Experience (UX). Native mobile apps tend to offer a more seamless and enjoyable UX due to features such as smooth scrolling and support for specific gestures like swipes. They also have more advanced interface animations and effects, as well as faster performance. This type of app development allows both developers and users to take full advantage of the capabilities of the software and operating system [26]. While platforms like "React Native” and “Flutter" have made progress in improving UX for hybrid apps, there are still some limitations, such as slower loading times and slower performance. Additionally, apps like these “often look like a custom website with a user interface of a mobile app” [26].

Following the determination to develop the application for the Android platform, the choice of language fell upon Kotlin. This decision was driven by the fact that Kotlin was officially endorsed as the recommended language for Android app development by Google at its I/O conference in 2019 [27].

2.4 Android App Architecture

In a world that is constantly changing and evolving, it is important that mobile applications can easily adapt to transformations, but, to achieve this result, programmers need to follow some architectural principles.

2.4.1. Android App Components

The process of defining the architecture for an application involves deciding which components to use, as Android offers a wide range of libraries that were tailored to meet the specific needs of a project.

Android mobile applications were usually developed based on XML (eXtensible Markup Language), to define the layout of the app's user interface and is typically associated with the Activities that are the main screens that users interact with in an Android app. They represent the UI and business logic of an app, and are responsible for handling user input and displaying the appropriate UI to the user. When an Activity is launched, the Android system reads the layout resource file and creates the views and view hierarchy defined in the file to display on the screen. These views may be divided in Android Fragments that are modular components used to represent part of an Activity in an Android app. They divide the user interface of an app into smaller, reusable pieces that can be combined to create complex layouts.

Another approach is the use of Jetpack Compose, since it is a new declarative UI toolkit for Android that allows developers to build apps faster and more efficiently. It is based on the idea of composing small, reusable widgets to build a complete UI, rather than using complex layouts and view hierarchies.

Considering the components responsible for managing the data for an app's UI, architectural patterns suggest a ViewModel class designed to store and manage UI-related data in a lifecycle-conscious way. It allows data to survive configuration changes such as screen rotations, and it also helps to decouple the UI from the underlying data source. Elements that are aware of their lifecycle, such as ViewModelScope, can cancel any ongoing work when the ViewModel is no longer in use. Using Kotlin Coroutines to run asynchronous, non-blocking tasks in a background thread, it is possible to update the UI once the task is finished without interrupting the UI's operation.

2.4.2. Architectural Principles

For high-level architecture, we are following a Clean Architecture approach, created by Robert C. Martin [28]. It is a software design paradigm that advocates separating the business logic of a software system from its presentation and infrastructure concerns.

Good programming practices dictate that developers should strive to write clear and well-structured code that is easy to read and understand. The SOLID design principles describe how to arrange the functions and data structures, and how they should be interconnected, so they can tolerate change, and provide a base for the software system. To help achieve this goal and improve the overall design of a software system, it is important to follow other principles such as Separation of Concerns [29], Single Source of Truth [30], and Unidirectional Data Flow [31].

Separation of concerns, in the context of mobile applications, is the ability to divide an entire application into different parts and give different responsibilities (concerns) to each of them. When concerns are well-separated, there are more opportunities to upgrade and reuse the code, as well as independent development.

Single Source of Truth is the practice of ensuring that the data is stored exactly once, by designating a data source as the central source of truth for the rest of the application, so all actions regarding data are done within that data source.

Unidirectional Data Flow is a technique found in reactive programming, which means the data has one and only one way to be transferred to other parts of the application, allowing better control of the processes.

SOLID Principles

The first principle dictates that each software module “has one, and only one reason to change” [28], which means that each module should be responsible for only one actor, and change according to him. A module is a cohesive set of functions and data structures and an actor could be defined as a group of stakeholders that want the same things, and require the same changes, to that module. For example, if two different actors use the same set of functions to perform the same action but one requires a different result, it could cause a problem to the other actor that uses the same set of functions and does not have any problem with them. Therefore, it is important to separate the modules, so they can only be responsible for one actor, and change without interfering with other actors [28].

The Open-Closed principle is a design philosophy that suggests that software classes should be designed in a way that allows them to be easily extended to meet new requirements without requiring modification of their existing code. An example of this principle is the authentication process in the Framedrop app, which currently only supports authentication through Twitch. If a client requests the ability to authenticate through Google, for instance, simply adding an "if" statement to the existing code is not a proper solution because it may not scale well as the number of authentication methods increase. Instead, the authentication logic should be abstracted so that it can be extended to support other methods without requiring modifications to the existing code.

The Liskov Substitution Principle states that if a program is using a base class, then every derived class should perform the same tasks as the base class. In the Framedrop app, the clips can have two states, claimed or unclaimed, and the interactions that the user can perform with them are different. Assuming that we implement a base class “Clip” that gets the claimer and the creator of that clip, and derive from that class the subclasses “ClaimedClip” and “UnclaimedClip”, we expect that these subclasses perform the same as the base class. The problem is that unclaimed clips do not have a claimer, so the program will generate an error. According to this principle, it is important to structure inheritance so that all subclasses can substitute the base class without affecting the correctness of the program.

The principle mentioned above, will reflect on the Interface Segregation Principle, which states that classes should not be forced to implement methods that they do not use. Using the same example for claimed and unclaimed clips, we should not force the “UnclaimedClip” class to implement the method to get the claimer because it does not use it.

The Dependency Inversion Principle suggests that high-level modules of an application should not depend on volatile concrete elements such as low-level modules, since they are frequently changing, but rather both should depend on abstractions.

These principles take an important role in architectural design by guiding developers into better programming approaches, but they do not show how the code should be implemented. Architectural patterns specify components and behaviors that, if implemented correctly, will lead to better code.

2.4.3. Architectural Patterns

Since Martin’s approach to Clean Architecture can be applied to any type of application, the software engineer and architect, Fernando Cejas, wrote extensively on

Android development and proposed a Clean architectural approach specifically tailored for Android development where it describes the use of dependency injection tools and frameworks to manage these dependencies, which follows the Dependency Inversion Principle. Additionally, Cejas' Android Clean Architecture includes specific guidelines for how to structure an Android project, such as where to place different types of code and how to organize packages [32].

Considering Martin's high-level Clean Architecture and Cejas' application to Android development, Android officially recommends that an application should have at least two layers: UI and Data, with an intermediate optional layer, Domain, that acts like a middle-man, managing the interactions between the other two layers [33].

The Data layer contains the data structures and exposes the data to the rest of the app. The Domain layer promotes the separation of concerns, since the business logic presented in this layer can be divided into use cases that represent a set of actions that interact with the Data model in order to retrieve information. For example, a use case could represent an action of getting data from the Data layer, that can perform a request to a remote API, and retrieve it to the View. The View layer refers to the interface that the user interacts with. It is responsible for presenting the data to the user and handling his input. The View layer is typically implemented as a user interface (UI) and is separated from the business logic of the application [33].

Having this philosophy present for different application demands, some architectural patterns were created in order to solve a particular set of problems.

Model-View-Controller (MVC)

The Model-View-Controller (MVC) pattern was first introduced in 1979 and was designed to come up with a solution for managing complex components [34]. It is a popular architectural pattern that separates an application into three main logical components: Model, View, and Controller, represented in Figure 1.

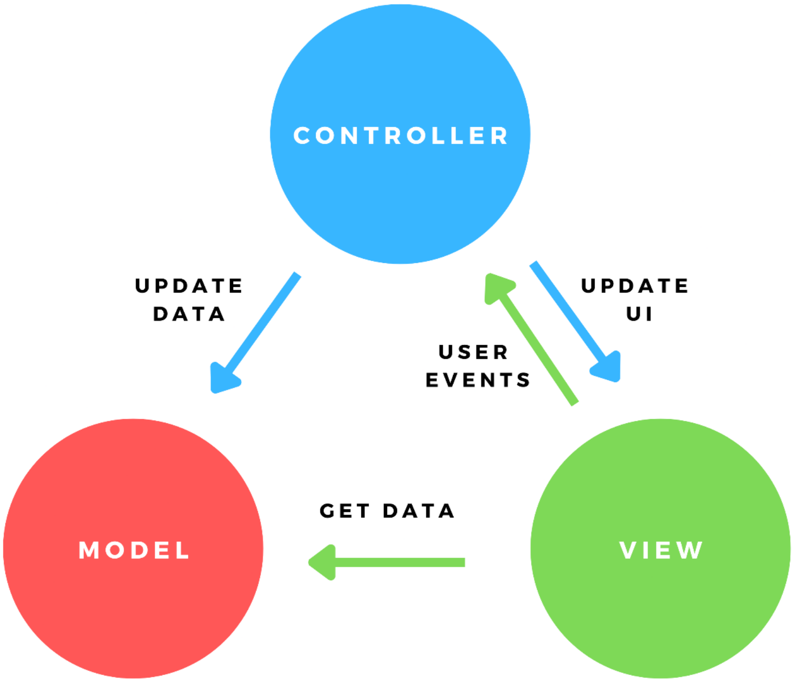


Figure 1. MVC architecture diagram

The Model is the domain element since it is responsible for handling the data structures, as well as managing all business logic [35]. The View component (or User Interface) is responsible for data representation. The Controller manages the user interactions with the View by accessing and changing the Model. To facilitate the communication between the views and the Model, the Controller component is responsible for making updates to the Model, which removes that logic from the View. In a mobile application, usually the View is implemented using XML and the Controller is the Activity or Fragment, leaving the Model to the data structures. Unit testing can be challenging, since the Controller is placed on Android classes that provide less testing flexibility. Additionally, as the View is dependent on both the Controller and the Model, any changes made to the UI logic may require updates to multiple classes, which reduces the pattern's flexibility [36].

Model-View-Presenter (MVP)

This pattern originated in the early 1990s at Taligent with a joint venture of Apple, IBM, and Hewlett-Packard [37]. As mentioned above, in MVC there is a connection between the View and the Model. As described in Figure 2, MVP architecture breaks this connection handling all the logic regarding the UI with the Presenter.

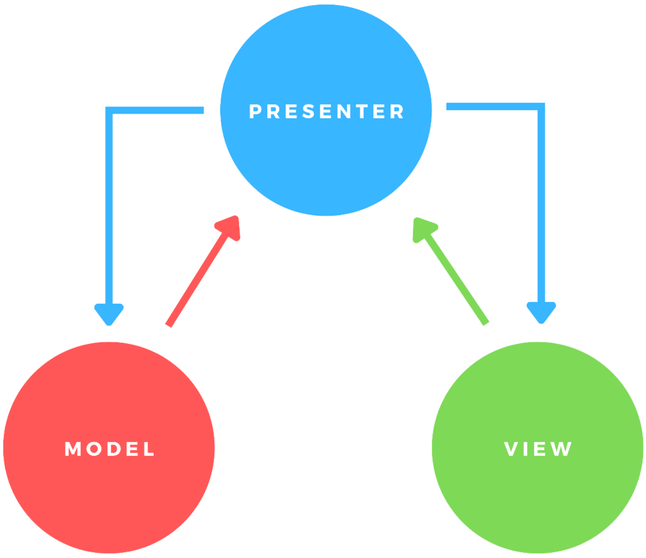


Figure 2. MVP architecture diagram

This leaves the View component exclusively for the rendering of the data bound by the Model and capturing user's input and the Presenter becomes the middle-man that performs all the necessary modifications in the two other components. The View and Presenter have a closely interrelated role, as a result, they need to have a reference of each other.

Although we can achieve highly decoupled components, having a reference in the View and Presenter components can lead to boilerplate code that architectures like MVC dismiss.

Model-View-Intent (MVI)

MVI is an architectural design pattern based on reactive programming. The goal is to have less complex, more testable, and easier to maintain code. The MVI schema may look similar to MVP, but the main difference lies in the implementation of the components and their interactions in the app. For instance, Models in MVI represent a state and in MVP they represent data. This happens because, in MVI architecture there is no need to send data from the Model to the Controller/Presenter to be rendered by the View, as described in Figure 3.

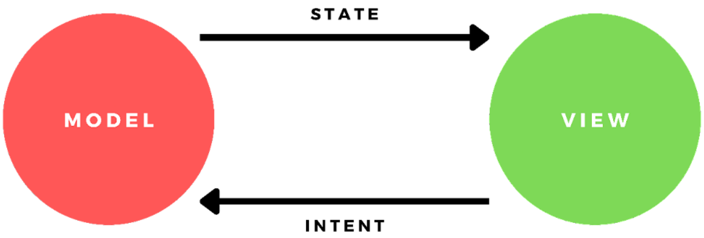


Figure 3. MVI architecture diagram

The Model has one overall state and it is immutable, which follows the principle of Single Source of Truth, because it is updated from only one place and emitted to the View that observes the state to be rendered [38]. MVI uses Intents that can be translated as the "intention to do something" [38], which means that every action that the user has with the UI triggers an Intent. These actions will perform changes at the Model component, updating the state. This process chain follows the Unidirectional Data Flow and the Separation of Concerns.

Model-View-ViewModel (MVVM)

In MVVM, the Views and the ViewModel are not strongly connected, since the View has a reference to the ViewModel while the ViewModel has no information about the View, as presented in Figure 4. This happens because "the View directly binds to properties on the ViewModel to send and receive updates" [39]. The MVVM pattern was designed to make event-driven programming of user interfaces simpler. While in the MVP pattern, the Presenter would directly instruct the View on what to display, in MVVM the ViewModel provides streams of events to which the Views can bind. This way, the ViewModel no longer needs to maintain a reference to the View, as the Presenter would in MVP [40].

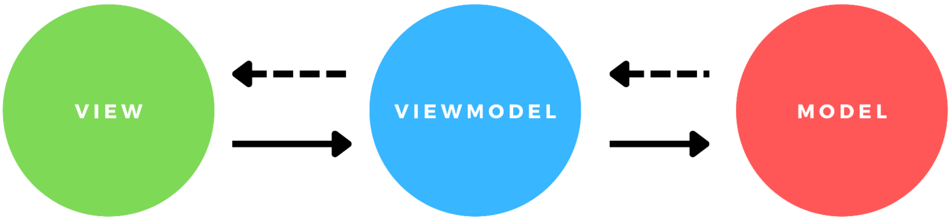


Figure 4. MVVM architecture diagram

Since the ViewModel "is completely separated from the UI or any Android classes" [56], having a decoupled system eases the testing process and provides the creation of effective tests compared to patterns like MVC that create bigger, slower and flaky tests, due to a higher dependency between components [41].

2.4.4. Summary

The analysis of the different architectural patterns and the advantages and disadvantages of each aims to provide a better decision of the "best-fit" for the application that is being developed. The decision is not always exclusively about the project industry and requirements, as it may have to do with constraints such as duration, budget, work methodologies, among others.

There is a clear set of characteristics that we want the architecture to provide. It should have low coupling, which means that there are few dependencies between components, making them mostly independent. Loosely coupled components are easier to develop, maintain, and test, because there is no need to affect the other components [42], [43]. High cohesion is another characteristic that should be present in the architecture, to make sure that the components are created with a single, well-focused purpose. These components tend to be easier to maintain and because of low purpose complexity, could be reusable [44].

Application

For the specifications of this project, we considered that MVVM architectural pattern is the best-fit, since it is a well-established design pattern for building complex user interfaces. It is particularly well-suited for Android mobile applications due to its ability to easily support the separation of concerns required for good application design easing the process of testing the different parts of the app in isolation.

Although unit testing can be performed in architectures like MVC and MVP, the low level of component isolation when compared to MVVM, increases the difficulty to test as the project evolves. In MVC, the controller handles both the UI and the business logic, making it harder to test the business logic separately. In MVP, the presenter handles the business logic and updates the view, making it harder to test the business logic and the view separately. Since this mobile application is composed of a considerable number of screens, remote requests to an API, and mechanisms of payment, providing low coupling between components will ease the process of testing, as well as maintaining and developing new app features. MVI was also a good candidate to be the architecture pattern of choice, except that it features one more layer than MVVM, with all user intentions having to be mapped first. This extra layer would create more boilerplate code that we felt was unnecessary.

User Interface

The user interface is going to be developed using Jetpack Compose, a modern toolkit for building native Android UI. It is an alternative to the traditional XML-based layout approach in Android. This can make it easier to write and maintain the UI, as well as give more control over its appearance and behavior. It is based on a declarative style of programming, where it focuses on describing what it is trying to achieve, rather than how to achieve it, often used to build user interfaces and to specify data transformations [45].

Framedrop's mobile application is more focused in the UI and user interactions with the View layer, since most of the data modifications are communicated to the Framedrop server, reducing the complexity of low-level layers. Being said, we need an architectural pattern that provides a good separation of the complexity of the higher layers. That said, using an architecture such as MVVM, it is possible to separate components that are responsible for handling the data and logic that drives the View.

The use of a ViewModel instead of a Presenter is due to the fact that the Jetpack Compose library is already designed to take advantage of the use of a ViewModel and its life cycle. Another advantage of using a ViewModel is the use of states since the ViewModel does not have a reference to the view, however, the View has a reference to the ViewModel states and updates the UI when they change [46].

When following an MVVM approach, this pattern usually separates the different states responsible to perform changes to the UI components of the screen. This can cause a lot of boilerplate code, since it is necessary to instantiate the specific state every time we want to perform changes to the UI, and repeat the process to every state. Since we are diverging from a single source of truth (SSoT) approach when it comes to the states, it was decided to use a principle of MVI's architecture that follows the SSoT principle by merging all the pieces of data into one state class. This way, all the states are stored in a single object, and to access it on the View, we only need to instantiate it once, removing all the boilerplate code and following the SSoT principle [38].

2.5 Technological Research

The implementation of certain features in Framedrop's mobile app can be challenging due to the use of technologies with specific requirements, since they require a deeper analysis to conclude which libraries are a best-fit for that set of problems.

This exercise is important because it provides information about the choices that were made and it can be used to guide stakeholders through their decisions, making the development process more transparent.

2.5.1. Authentication with Twitch

Twitch uses OAuth 2.0 for authentication [47], and has its own API that provides integration to third-party applications. The documentation presented in the Twitch Developers website describes three types of flows to authenticate:

- Implicit grant flow - This flow is used in applications that do not use a server. For example, a client-side JavaScript app or mobile app.
- Authorization code grant flow - This flow is used if the application uses a server that can securely store a client secret, and can make server-to-server requests to the Twitch API.
- Client credentials grant flow - This flow is meant for apps that only need an app access token, but follow the same specifications of "Authorization code grant flow" [48].

Twitch documentation referred to an SDK that could ease the process of dealing with the OAuth 2.0 flow. However, after analyzing the documentation, we concluded that there was no information about that tool, which should come in the form of a library that presents customized methods to that specific platform.

Twitch documentation also referred to the use of an OpenID Connect (OIDC) protocol that defines optional mechanisms for robust signing and encryption, and integrates the OAuth 2.0 capabilities [49]. Since authentication communications must be secure, the integration of this protocol in our application would ensure a security layer to the authentication process.

Implementing this protocol from scratch could generate security breaches, so we also analyzed a set of libraries that implement this protocol, where we decided to use the AppAuth library that is recommended by Google to help to implement the OAuth 2.0 flow [50].

AppAuth for Android is a client SDK for communicating with OAuth 2.0 and OpenID Connect providers that strives to directly map the requests and responses of those services. In addition to mapping the raw protocol flows, it still provides convenient methods to assist with common tasks (for example, to perform an action with fresh tokens) [51].

Framedrop's application will use Framedrop's server to authenticate to Twitch, so the process of authentication will not follow the steps provided by the Twitch Developers guide mentioned above. Since the application contacts the server to perform the authentication, there is no direct connection to the Twitch authentication server.

A representation of this flow is described in Figure 5.

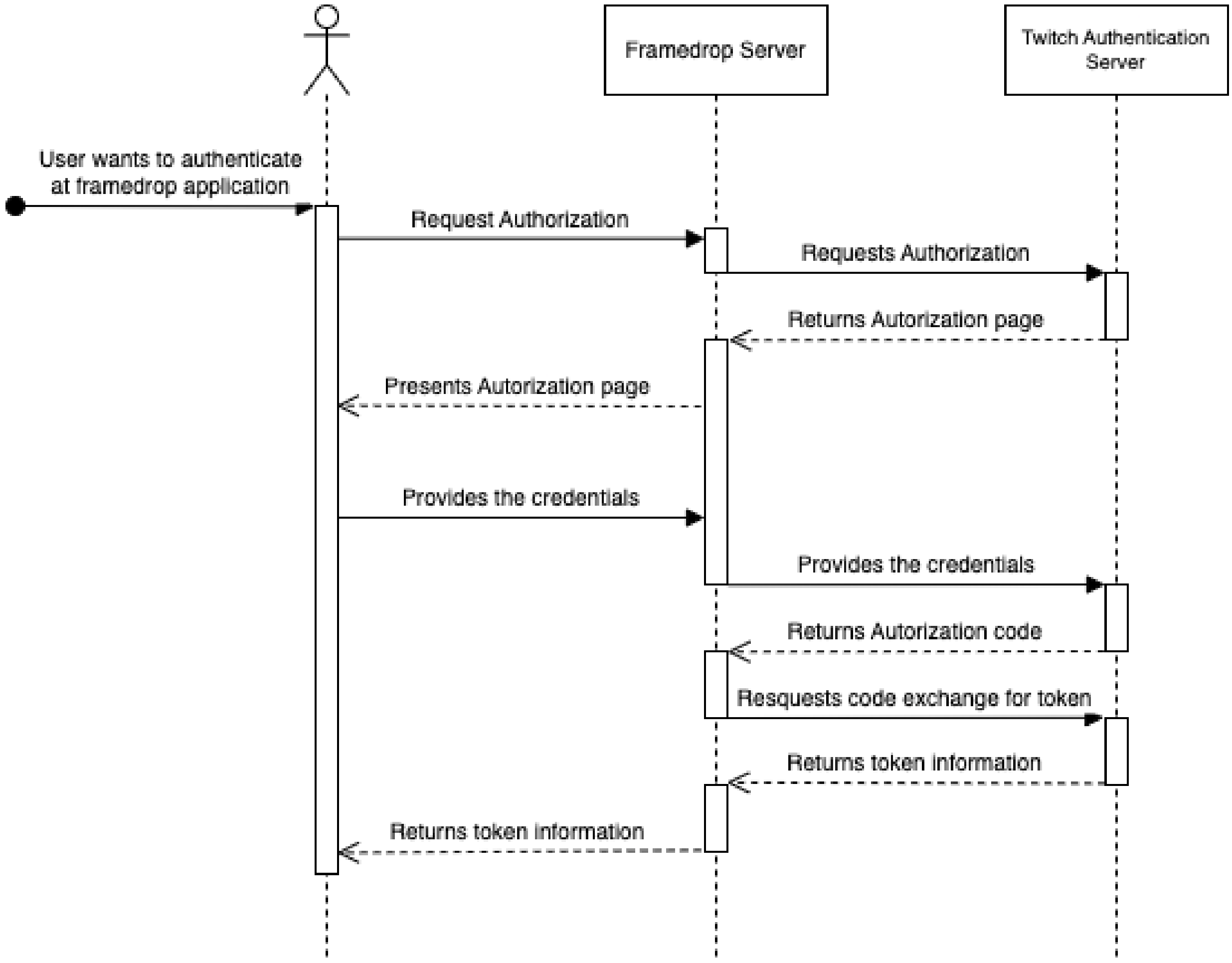

Framedrop Server
Twitch Authentication Server
User wants to authenticate at framedrop application
Request Authorization
Requests Authorization
Returns Autorization page
Presents Autorization page
Provides the credentials
Provides the credentials
Returns Autorization code
Resquests code exchange for token
Returns token information
Returns token information

Figure 5. Framedrop's mobile app Authentication with Twitch Sequence Diagram

In the initial stages of development, the Framedrop team was engaged in an assessment of potential implementation solutions for the authentication flow. To preempt any delays in the mobile app's progress, I took the initiative to develop the authentication with

Twitch feature by creating a local server that facilitated the authentication process with the Twitch service. Consequently, recognizing its suitability and fulfillment of the necessary conditions, the Framedrop team integrated the implementation of the local server into their infrastructure.

2.5.2. Video Protocol and Media Player

Framedrop uses the (HTTP Live Streaming) HLS communication protocol to share content with its web app. Below follows an analysis of HLS video players that are compatible with Kotlin.

Researching through a various number of well known applications like Facebook, Instagram, TikTok, etc, we realize that the media players used by this big corporation are not for general use, which means that they are built from scratch to fulfill the needs of each company.

Nevertheless, there are some must-have features that are useful for Framedrop's application and could be used for comparison in media player research.

- **Compatibility with HLS** - Framedrop uses HLS protocol, so its required a player with HLS compatibility.
- **Portrait mode** - Gaming content usually comes in landscape scale, so it is important that the player provides a way to crop the video into portrait mode.
- **Hide controls** - To enhance user experience, videos must be free of any component that could overlay them. Other applications already implement this feature such as Instagram, Facebook, TikTok, among others.
- **Autoplay** - When the users navigate between videos, they must start without any additional step.
- **Restart on end** - Videos may loop while they are exposed to consumption. Stopping a video at the end adds no value to the user experience.
- **Customizable UI** - Since Framedrop has its own design, it is important to customize the UI to their needs.

Here is a list of features that could be useful for future application updates.

- **Advertisement insertion** - Framedrop may want some form of monetization through ads that may display between clips in the app.
- **Download content** - It could be given the possibility to users to download content.

- **Analytics** - Media player analytics may be necessary.

Since Framedrop uses HLS, the research will be around players compatible with HLS, which reduces the search field and the margin of choice for a player who does not meet the needs. The next section describes an analysis of the positive and negative aspects of media players features.

ExoPlayer[10]

It is an application level media player for Android that supports media types like HLS, DASH, SmoothStreaming, Progressive and RTSP. It is an open source project alternative to Android's MediaPlayer API, that is easier to customize and extend, and can be updated through Play Store application updates [52].

Including this player in the app may increase the Android Application Pack (APK) size by a few hundred kilobytes, which means that the application will take up more memory when installed on a mobile device. Nevertheless, ExoPlayer has less variation across different Android devices and versions, and allows for customization to specific use cases by replacing components with custom implementations. It also supports playlists, various media formats, Widevine common encryption on Android 4.4 and higher, and has well-detailed documentation with a solid amount of snippet examples. Additionally, the player can be updated along with the application.

NAGRA Player[11]

NAGRA's player is an operator-centric OTT (over-the-top) player, that can detect and prioritize between the available DRMs[12]. It provides an SDK for Android Java and Kotlin's applications, which wraps and extends the media player API, providing a similar interface to the Android video view [53].

While the SDK for NAGRA is available for Java and Kotlin, the documentation is only provided in Java. Some essential features are not provided by NAGRA, although they can be resolved with a workaround. On the positive side, once it is set up, the code base of NAGRA player appears easy to work with and allows customization to specific use cases by

[10] ExoPlayer - https://exoplayer.dev/

[11] Nagra Connect Player - https://docs.nagra.com/doc/home/cpanddoc524x

[12] DRM - Digital Rights Management. Helps prevent copyright infringement by restricting the dissemination of copied digital content, while securing and managing copyrights and trademarks.

* Ad insertion is done on the server side, which means that the server sends through HLS manifest when to display the ads.

replacing components with custom implementations. It also supports playlists and various media formats.

Bitmovin Player[13]

Bitmovin's playback solution enables video streaming with a focus on high quality and testing. It gives the possibility to analyze the data through a dashboard portal to receive insights into how playback is performing for viewers.

Bitmovin player offers a 30-day free trial that includes a limited-time player for free. To unlock the full version, a monthly fee of $249 must be paid. However, it does not include video resize modes, and a portrait mode is a required feature. To set up the media player, a player license key and a personal license key are necessary. Although it is a paid media player, it offers simple integration with an application, and its documentation is well-described with a great amount of tutorials and code examples. It allows customization to specific use cases by replacing components with custom implementations and supports playlists and various media formats.

NexPlayer[14]

NexPlayer has over 15 years of experience in video streaming, focusing on delivering media content across all devices. The company integrates its service into other big company products like Hisense, PlayStation, Xbox, Nintendo, among others. They are currently bringing live streaming to the metaverse, integrating with Unreal Engine and Unity [54].

For the context of this project, NexPlayer does not provide some of the necessary features and needs to use third-party entities to enable Ad insertion. Nonetheless, NexPlayer provides well-described documentation with a great number of tutorials and code examples. It supports playlists and various media formats.

Summary

To summarize the analysis of the different media players, Table 1 presents the features that are useful for that Framedrop's mobile application media player, and the players that support which of them.

[13] Bitmovin - https://bitmovin.com/docs/player/getting-started/android
[14] NexPlayer - https://nexplayer.github.io/Android-SDK/#/README

Feature	ExoPlayer	NAGRA	Bitmovin	NexPlayer
Portrait mode	✓	✓	✓	✓
Hide controls	✓	✗	✗	✗
Autoplay	✓	✗	✗	✗
Restart on end	✓	✗	✗	✗
Customizable UI	✓	✓	✓	✗
Advertisement insertion	✓	✓	✓	✓
Download content	✓	✓	✗	✗
Analytics	✓	✓	✓	✓

Table 1. Media players feature comparison

From the analysis made in this document, it is possible to see that the ExoPlayer is the best-fit solution for this specific problem. Not only does it meet all the feature requirements without using third-party entities or workarounds, but also because of its popularity. Google developed ExoPlayer for use in YouTube, Google Play Movie, Google Photos, Youtube gaming, and Google Play Newsstand before releasing it to developers [55]. In 2018, there were more than 140.000 applications in the Play Store that use ExoPlayer [56].

Another important characteristic of this player is the ability to update itself along with the application, which means that it becomes a part of the application, so we can ship the same exact version of ExoPlayer through the different Android releases, which provides a consistent experience. Another important advantage of this model is feature addition that can be supported through all versions of Android, because there are no dependencies on low level media APIs or anything like that in the platform. An example of this advantage can be seen in Android's MediaPlayer which is implemented in Android's operating system and not on the app itself as we may see in Figure 6. When adding a new feature, in MediaPlayer, it will only be available for subsequent releases of Android [56].

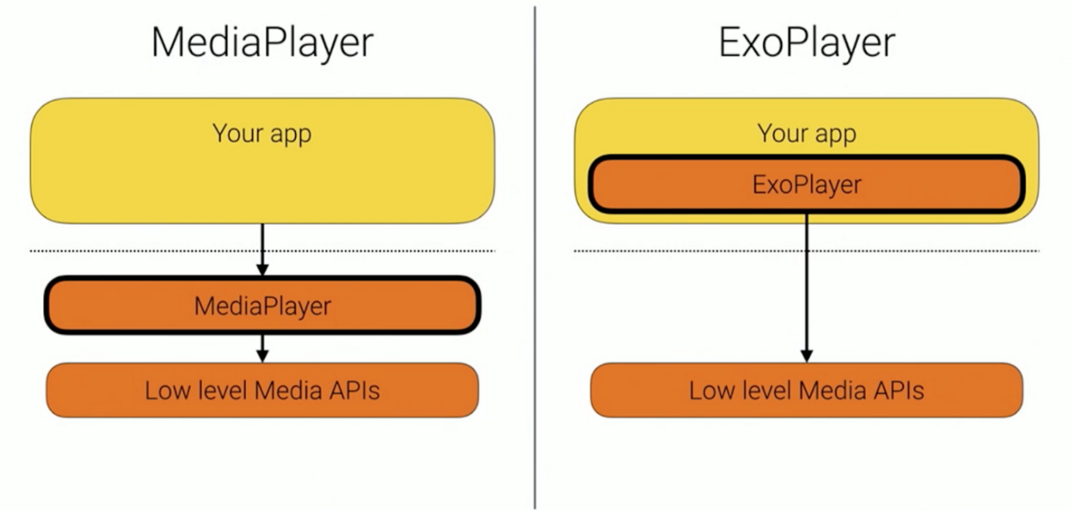


Figure 6. Media Player and ExoPlayer layer of implementation [81]

2.5.3. Infinite Clip Scroll

As mentioned in the video player analysis, the mobile application requests clips from the Framedrop server in order to display them to the users. Thinking about a smooth user experience, it is wrong to assume that the videos are only rendered once they are active on the screen. If the videos are only rendered immediately before being consumed, the rendering times aligned to other delays caused by the scroll animation and video start playing will not guarantee a pleasant experience to the users.

Since the videos can not be rendered at the time they appear on the screen, we need to follow TikTok's approach of "preloading (loading the next video ahead of time) and pre-rendering (rendering the first frame of the video ahead of time)" [57]. The company accomplished this feature based on the use of a class Surface that is designed to be used with a separate rendering thread to enable smooth animations and reduce the potential for dropped frames [58]. By using this component, they could render the clips in the background and expose the video already rendered when it becomes in focus [57].

2.5.4. Payments

Framedrop's mobile application integrates in-app tokens that are used to trade for unclaimed clips, where the user becomes the claimer of that specific clip and every user that forks it to his profile will have the claimer's name attached to it. In the first version of the app, the claim price is fixed, leaving out of the scope the process of bidding for a clip. Further iterations will focus on providing a payment feature for in-app tokens and an auction feature, given the ability to bid a specific clip instead of the first-come-first-serve approach of the first version.

The payment functionality is a critical aspect of the development process as it involves handling users' money, which carries a high level of risk. During the initial phases of development, the Framedrop team faced uncertainty regarding the payment service to be utilized, as they engaged in discussions to identify the most suitable service aligned with their business values. The sequence diagram depicted in Figure 7 illustrates the payment flow for purchasing tokens and acquiring a clip based on the available information collected during that period.

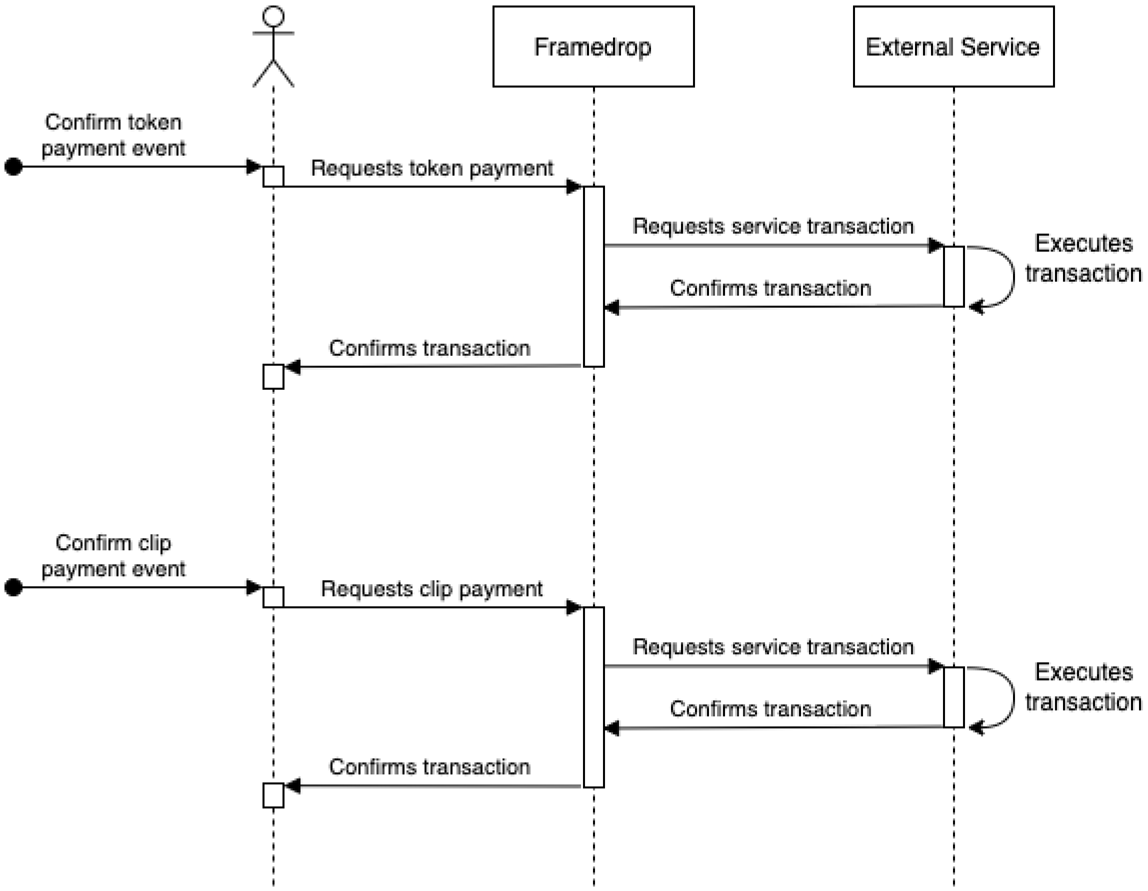


Figure 7. Payment sequence diagram for token and clip payment.

In further stages of development, the Framedrop team reached the determination to incorporate the Paypal service for in-app payments. With this new information and how the flow would perform, it was determined that no alterations to the existing sequence flow were required, as it aligned precisely with our initial projections.

2.6 CI/CD

Before Grady Booch proposed the term Continuous Integration (CI) in 1991, developers had to go through the tremendous effort of "integrating" a new feature code that they had been working on for a couple of days or months. To do that, they had to merge their changes into a mainstream repository, which had inevitably been changed, since there were other developers working on other features at the same time. The code was only integrated into the main repository once the feature was completed, taking a long period of time, which increased the number of integration errors, leading to conflicts like bugs or failures that may take a serious time and effort to correct ("integration hell"[15]) [59], [60]. Grady Booch describes that this process "should not be a single 'big bang' integration event" [61].

2.6.1. Continuous Integration (CI)

CI minimizes and helps eliminate code drift between the mainline branch and a feature under development, greatly reducing the risk of integration [62]. Although this process was not advocated for several times a day, the software development methodology, extreme programming (XP), embraced the idea of CI and encouraged integrating code multiple times per day. The idea was to turn the conventional software process, plan, analyze, design, implement, and test, into an iterative process, instead of a waterfall methodology that follows a unidirectional flow throughout that process. By doing this "a little at a time" [63], and following some XP principles, new code could be integrated with the current system after "no more than a few hours" [63].

A research for the best suitable CI tool for Framedrop's project was performed, respecting pricing, with the ideal scenario being a free tool. The results are represented in Table 2.

[15] Integration Hell - https://wiki.c2.com/?IntegrationHell

Name	Description	Pricing
Jenkins	An open-source automation server that helps automate parts of the software development process. It can be used to build, test, and deploy software projects [64].	Free, but it needs a hosting service to be set up
GitLab CI	A continuous integration tool built into the GitLab code repository platform. It can be used to build, test, and deploy software projects [65].	Free plan with 400 minutes per month
Bitrise	Bitrise is a cloud-based continuous integration and continuous deployment (CI/CD) platform that is designed to automate the process of building, testing, and deploying mobile apps [66].	Free plan with 300 minutes per month
CircleCI	A cloud-based continuous integration and delivery platform that can be used to build, test, and deploy software projects [67].	Free plan with 6000 minutes per month when using Docker
GitHub Actions	GitHub Actions is a continuous integration and continuous deployment platform that allows developers to automate their build, test, and deployment pipelines [68].	Free plan with 1000 minutes per month

Table 2. Continuous Integration tool price analysis

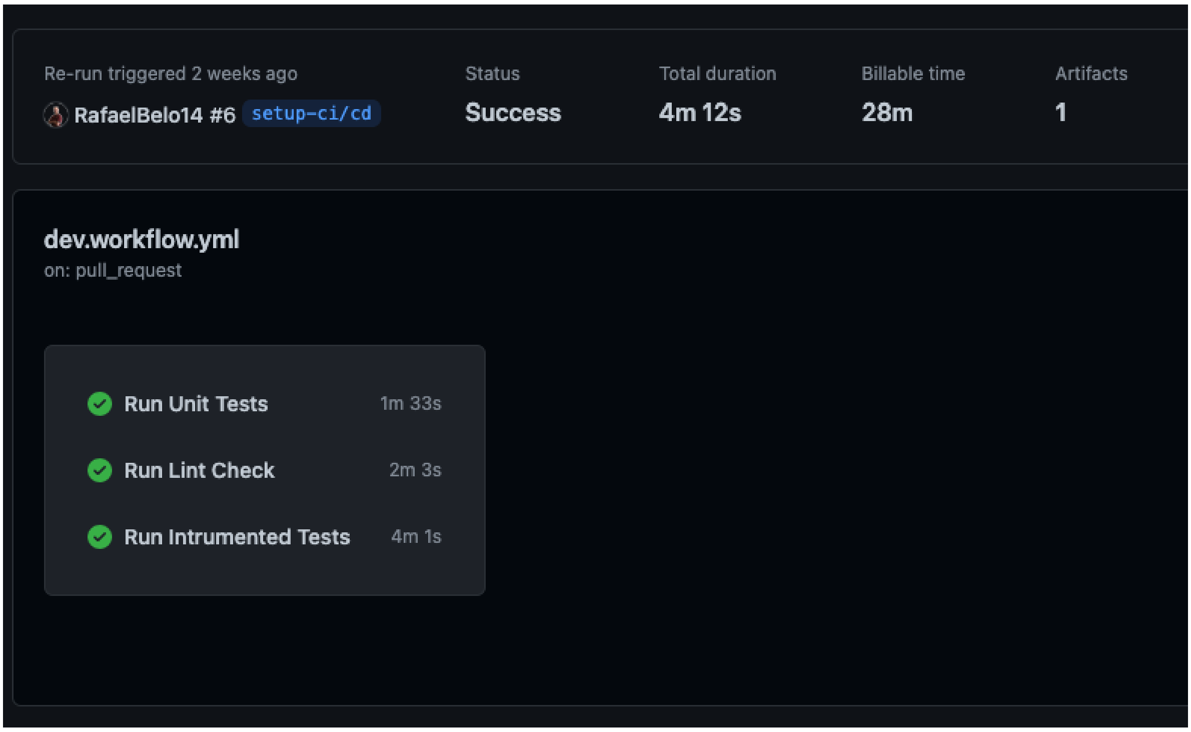
From the analysis made on Table 4, there are a set of CI tools that could be considered for the project, so the decision was based on build time per month and convenience. When comparing the results, GitHub Actions stands out from the other tools, since we have the advantage of setting CI/CD on the same platform where the repository is stored, not having the additional step of getting an external platform just for that. An additional advantage is that it offers a substantial amount of time for building projects, as compared to Bitrise and GitLab CI. Additionally, it eliminates the need for hosting or searching for a hosting service, as it runs on GitHub's virtual machines.

GitHub Actions

GitHub Actions is a tool that helps software development process automation by providing a platform for continuous integration and continuous deployment (CI/CD). With this tool, it is possible to create different workflows with specific actions that need to be performed in different branches. For example, if the code that will be used to generate a AAB to be deployed in the Play Store is on a branch called “production”, and the code that will be used to generate an APK that is being tested in Quality Assurance is on a branch “release”, these branches need to deploy their code to different environments, the production environment (Play Store), and an internal environment to access QA, respectively.

Github workflows are initiated when specific activities in the repository are made (e.g. pull request, merge, push). Jobs are created to represent the steps of the workflow, with each step being either a shell script or an action that is executed in sequence and depends on the successful completion of the previous step. If all steps in a job are running on the same runner[16], data can be shared between steps [68].

The flow for the continuous integration pipeline is described below, as shown in Figure 8.



Re-run triggered 2 weeks ago	Status	Total duration	Billable time	Artifacts
RafaelBelo14 #6 `setup-ci/cd`	Success	4m 12s	28m	1

dev.workflow.yml
on: pull_request

✓ Run Unit Tests 1m 33s
✓ Run Lint Check 2m 3s
✓ Run Intrumented Tests 4m 1s

Figure 8. Continuous Integration Pipeline

Instrumented testing was integrated with Firebase Test Lab. This cloud-based testing service enables testing the mobile application on a wide range of devices and configurations,

[16] A runner is a machine that is responsible for executing the steps in a workflow when the workflow is triggered. A runner can only run one job at a time. GitHub provides runners for Ubuntu Linux, Microsoft Windows, and macOS to run your workflows [93].

mitigating the need to set up and maintain a physical test lab, providing detailed test results and logs that ease the debugging process.

2.6.2. Continuous Deployment (CD)

Even though successful continuous integration provides regularly built, tested, and merged code to a shared repository, minimizing the number of conflicts, it does not minimize the effort of deploying the app to a live staging environment or to the customer [69], which is still a manual process.

Manual tasks that are repeated often lead to errors and inconsistencies, which can decrease the return on investment and waste valuable IT resources. For instance, manual code analysis can allow technical problems to accumulate, testing can frequently miss regressions and other issues until the software is deployed, managing infrastructure can introduce anomalies in environment configuration and slow down issue resolution, and deployments can be time-consuming and introduce risks from unintentional mistakes [70].

Automating this process with continuous deployment means that changes can be automatically bug tested and deployed to a live staging environment, or directly to the customer [69].

Firebase will also be used in Continuous Deployment, providing a staging environment using App Distribution service, to distribute the application to perform Quality Assurance. Every new version that is deployed to that environment will trigger an email notification to the testers, so they can test the application, providing crash reports and feedback.

The final step of the Continuous Deployment flow is deploying to the Play Store, which is also automated with GitHub actions, using the Google Developer API, to perform the upload of an Android App Bundle (AAB), a publish format that includes all your app’s compiled code and resources [71]. The complete flow is described in Figure 9 and Figure 10.

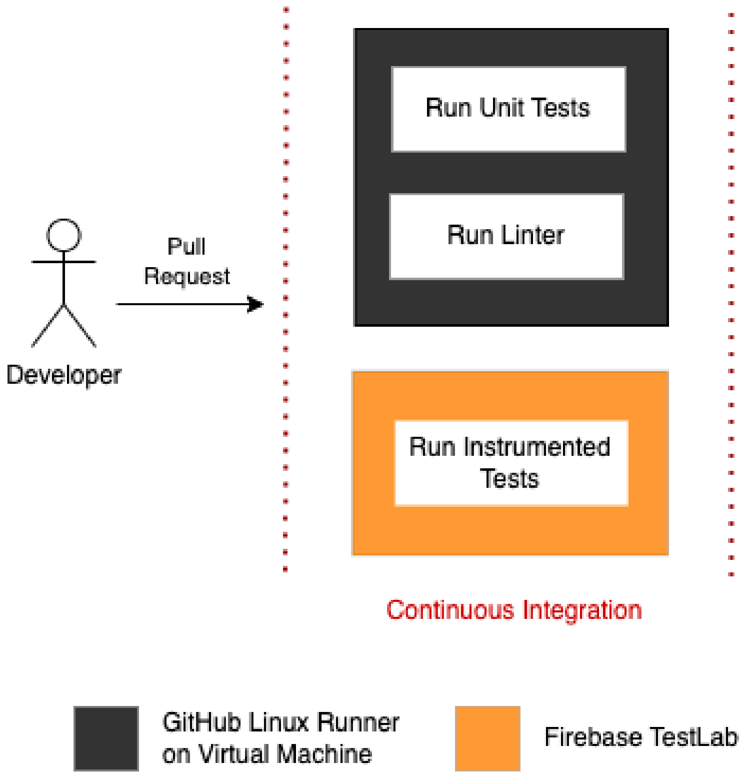


Figure 9. Continuous Integration flow diagram

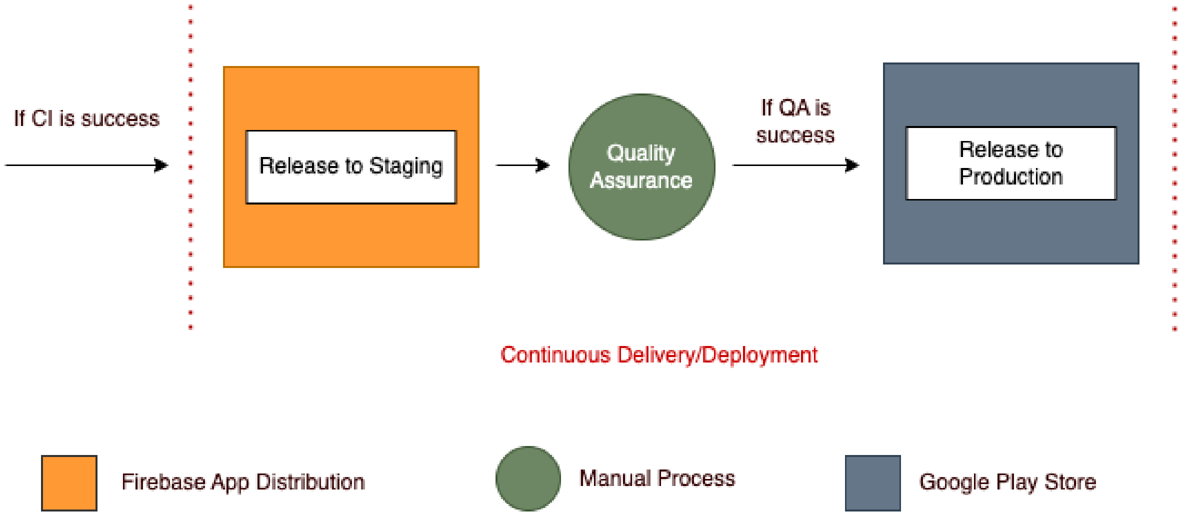


Figure 10. Deployment flow diagram

Chapter 3

Approach

Our approach followed an adaptation of Scrum, an agile development methodology that is characterized by a focus on rapid iteration and continuous deployment, with the goal of delivering high-quality software quickly and efficiently. It is based on the Agile Manifesto, which values individuals and interactions over processes and tools, working solutions over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan [72]. It divides work into small, manageable chunks called "sprints", and each of them will be planned according to the previous one. This happens because Scrum is an empirical process, meaning that it can not be planned upfront, but rather learned by doing, and then feeding that information back into the process [73].

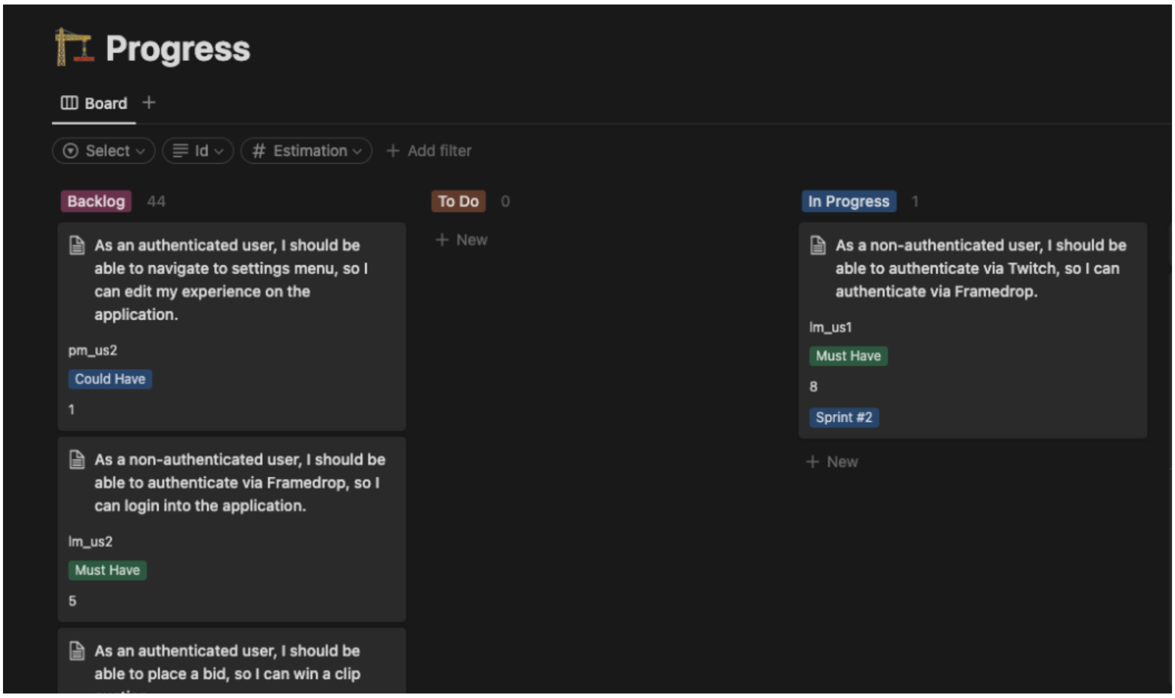
In the Scrum methodology, the three key roles are the Product Owner (PO), Scrum Master (SM), and development team. João Diogo Costa, the founder of Framedrop, served as the Product Owner, representing stakeholders and defining the product's vision. He prioritized the product backlog items, ensuring alignment with the overall vision and stakeholder needs. Working closely with the development team, he could adjust requirements and establish acceptance criteria for backlog items, guiding the team's work and ensuring compliance with standards.

Mário Gago assumed the role of Scrum Master, responsible for facilitating and coaching the team in adhering to the Scrum methodology. As a software engineer, he offered guidance on engineering practices, promoted principles, and removed obstacles to enable the team to deliver high-quality products. Collaborating closely with the development team, the Scrum Master ensured successful product delivery and effective teamwork.

Assuming the role of the development team, my primary responsibility was to develop the product. I acquired the necessary skills and expertise to fulfill the sprint's work requirements. Representing the development team, I made final architectural and technological decisions, maintaining continuous communication with the Product Owner and Scrum Master to discuss future development steps.

3.1 Sprints

Sprints are integral to the Scrum methodology for managing complex projects. They offer several benefits such as goal clarity, time-bound focus, enhanced planning and collaboration, adaptable course correction, and continuous deployment. User stories, previously prioritized and added to the product backlog, are selected for development during each sprint. These backlog items then progress through stages of development, review, code review, continuous integration, and continuous deployment. The process is visualized using Notion, allowing the Product Owner and Scrum Master to provide feedback and oversee the overall planning.



Backlog 44

As an authenticated user, I should be able to navigate to settings menu, so I can edit my experience on the application.
pm_us2
Could Have
1

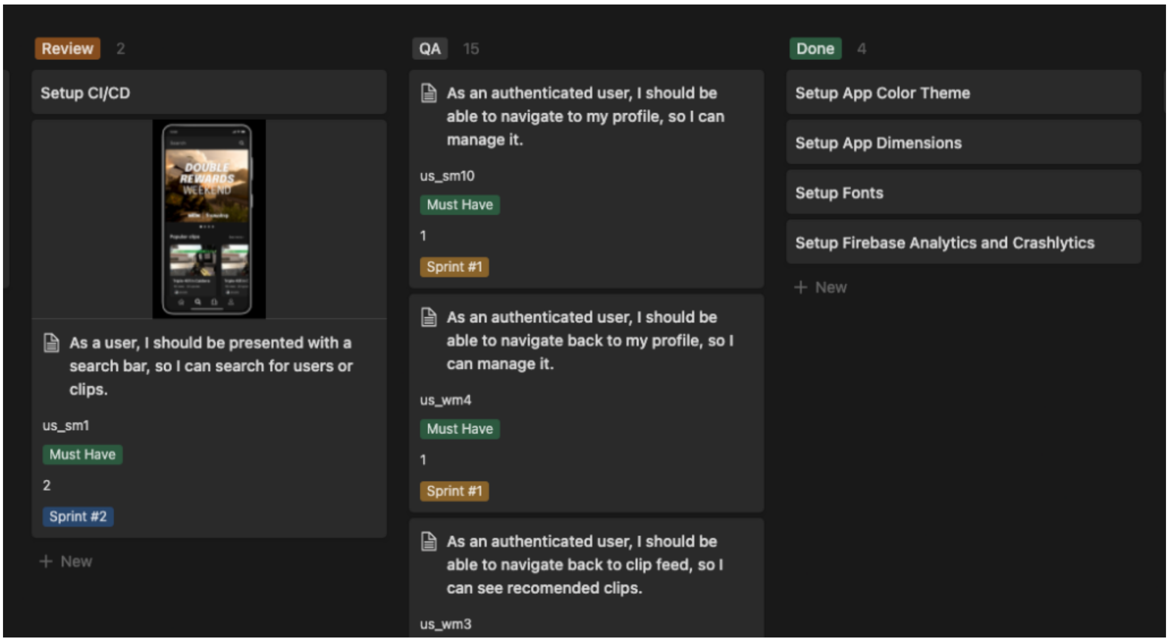
As a non-authenticated user, I should be able to authenticate via Framedrop, so I can login into the application.
lm_us2
Must Have
5

As an authenticated user, I should be able to place a bid, so I can win a clip

In Progress 1

As a non-authenticated user, I should be able to authenticate via Twitch, so I can authenticate via Framedrop.
lm_us1
Must Have
8
Sprint #2

Figure 11. Notion board with app progress (part 1)



Review 2

Setup CI/CD

As a user, I should be presented with a search bar, so I can search for users or clips.

us_sm1

Must Have

2

Sprint #2

+ New

QA 15

As an authenticated user, I should be able to navigate to my profile, so I can manage it.

us_sm10

Must Have

1

Sprint #1

As an authenticated user, I should be able to navigate back to my profile, so I can manage it.

us_wm4

Must Have

1

Sprint #1

As an authenticated user, I should be able to navigate back to clip feed, so I can see recomended clips.

us_wm3

Done 4

Setup App Color Theme

Setup App Dimensions

Setup Fonts

Setup Firebase Analytics and Crashlytics

+ New

Figure 12. Notion board with app progress (part 2)

3.2 Scrum Events

Scrum includes events that help the team organize and track their work. Some events were merged together like Planning and Retrospective, and Review and Showcase meetings.

3.2.1. Sprint Review and Showcase

Sprint review and showcase meetings were conducted every two weeks to evaluate the work completed at the end of each sprint. While typically separate events in the Scrum methodology, they were merged into a single meeting for this project. Two separate meetings were held in the academic environment, one with the University adviser, Scrum Master, and developer, and another with the Product Owner, Scrum Master, and developer. The purpose of these meetings was to review and showcase the development process, with the added benefit of receiving suggestions from the University adviser regarding internship reports and other relevant documents.

3.2.2. Sprint Planning and Retrospective

Sprint planning and retrospective meetings were combined into a single event, occurring every two weeks. Attended by the Product Owner, Scrum Master, and developer, these meetings served to evaluate the project's progress, address any obstacles or impediments, and plan for the upcoming sprint. Due to the ongoing development of Framedrop's server, these meetings assisted in prioritizing backlog items, many of which rely on server endpoints.

3.2.3. Daily

Daily stand-up meetings, lasting no more than 15 minutes, were conducted by the Pink Room company at a fixed time and location. Attended by the developer, Scrum Master, and all team members, these meetings served as a platform to share progress updates, address any blockers or issues, and plan for the day ahead. While not exclusive to the internship project, these meetings were utilized to provide progress updates and seek support as needed. The inclusive nature of these meetings allowed any Pink Room team member to assist in the development process, thereby reducing the workload of the Scrum Master and minimizing obstacles that may impede progress.

3.3 Requirements definition methodology

Requirements elicitation is non-trivial because requirements can not usually be gathered from the user and customer by just asking them what the system should or not do. Through an elicitation process, the requirements also need to be analyzed, modeled, or specified, so that the final set of requirements specifies quality attributes and functionalities for the system.

Since the project follows the Scrum methodology, the requirements were presented in the form of user stories, which represent a short, informal description of a feature or piece of functionality that a user of the system or application would like to see. User stories are used to capture the requirements for a new system or application, and are typically written from the perspective of the user, following a “who”, “what” and “why” structure that identifies the actor, the goal that is trying to achieve, and the benefits of completing the goal, respectively. An example of a user story is presented in Figure 15.

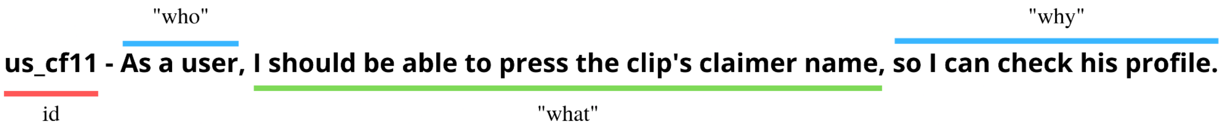

us_cf11 - As a user, I should be able to press the clip's claimer name, so I can check his profile.

Figure 13. Example of a user story structure from clip feed menu

User stories are an important part of the requirements definition process, as they help to ensure that the final product meets the needs of the users and stakeholders. They are also useful for planning and tracking the work for each sprint in agile development, as they provide a clear and concise description of the functionality that needs to be delivered.

Scrum defines two artifacts, product burndown chart and sprint burndown chart, one representing the collection of user points of all the user stories throughout the development of the application and the other only related to each sprint. For the context of this assignment, the sprint burndown chart was placed out of the scope of deliverables since it could limit the developer's flexibility to respond to changes in priorities or unexpected problems that arise during the sprint, as well as if the estimates are unrealistic, it could lead to missed deadlines and a breakdown between the developer and the stakeholders.

Before the development phase all the requirements were defined and translated into the form of user stories that were placed into the product backlog. With this method of having all the requirements in the beginning, it eased the process of re-prioritizing, improving the responsiveness to changes in priorities or unexpected problems that arise during the project.

Overall, while a sprint backlog can be a useful tool for planning and tracking the work for each sprint, a product backlog provides a longer-term view of the project and improved communication and collaboration, which helps to better understand the overall scope and complexity of the work.

3.4 Estimation

Effective time management is essential in project planning, involving the estimation of task durations and overall project duration. It is important to consider potential delays, risks, and uncertainties to provide accurate time estimates. Initially, hours were used for estimating user stories, but this approach led to disparities between estimates and actual time. To address this, a story points approach was adopted.

Story points are unitless and represent the effort and complexity of completing a user story, allowing for relative comparison [74]. They minimize disparities and are contextual to each developer's interpretation of effort. Planning poker, using a fibonacci-based sequence, was utilized to lower uncertainty. Each team member provided individual estimates, which were then discussed and agreed upon. The fibonacci-like scale is capped at thirteen to align with agile methodology's focus on small increments.

User stories with high estimates were further analyzed, and if necessary, divided into sub-requirements for re-estimation with lower values. When uncertain between two values, the higher one was chosen to manage risk and stakeholder expectations. This approach improved accuracy and empowered individual decision-making. Planning poker resolved issues of excessive discussion and dominance of effort estimation [75].

3.5 Development Process

Until now some principles of Agile development were discussed like Scrum events, product and sprint backlog and user stories that help to organize and manage the development planning. At the stage of development, it is important to follow other principles that will improve code quality and testability.

Effective use of efficient testing methods at any stage of the Software Development Life Cycle (SDLC) can be a challenge, since there are several testing techniques that can be used at different phases of testing. The selection process should not only consider subjective knowledge, but also objective information in order to make the most appropriate choice of testing techniques based on the requirements [76]. As the project grows, the codebase and the number of tests grow with it, so it is a good practice that tests run every time a new feature is developed, because this new added code may create defects on previous code. This process can become exhaustive, because it is repeated every time something on the codebase changes. Automating this step with continuous integration, which integrates the new code with the existing one after running the already defined tests, ensures that there are no unnoticed defects. Since we are using git as version control, it is important that on every failed test, the integration is rejected, so it does not affect the stable codebase.

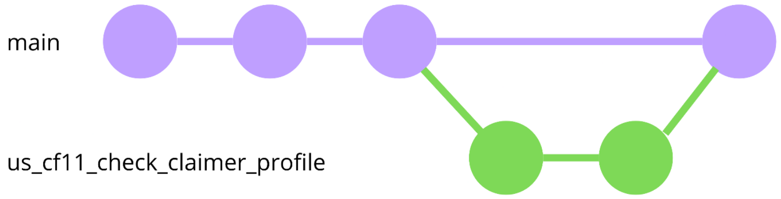
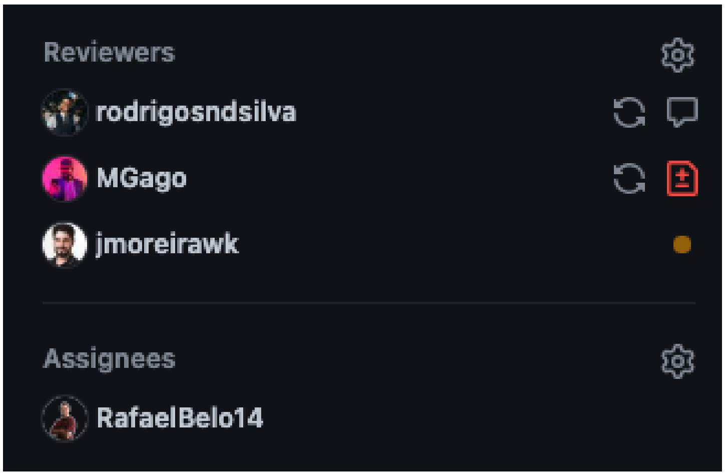


Figure 14. Feature branch behavior example

The process of adding the new code to the codebase can be assured using versioning with feature branching, allowing developers to work on multiple features concurrently without affecting the main branch of the codebase. Figure 14 describes the process of creating a feature branch, working on the code by committing to git, and merging again to the main branch. This process integrates automating testing because every commit triggers the continuous integrations pipeline that runs the tests to check if the code has defects. By encapsulating all the committed changes, it provides a fast way to debug and fix the code, because there are committed several small parts of code instead of a full feature.

3.6 Code Review and Quality Assurance

When a feature passes all tests, a Pull Request is created so another developer can review it. For this project, it was settled that if at least two reviewers approve the code, as it is described in Figure 15, it could be merged into the main branch. The reviewers were not always the same, so all the Pink Room elements could review the code, which improves code quality, since different elements may detect different details in the code and less blockers for the developer.



Reviewers

rodrigosndsilva

MGago

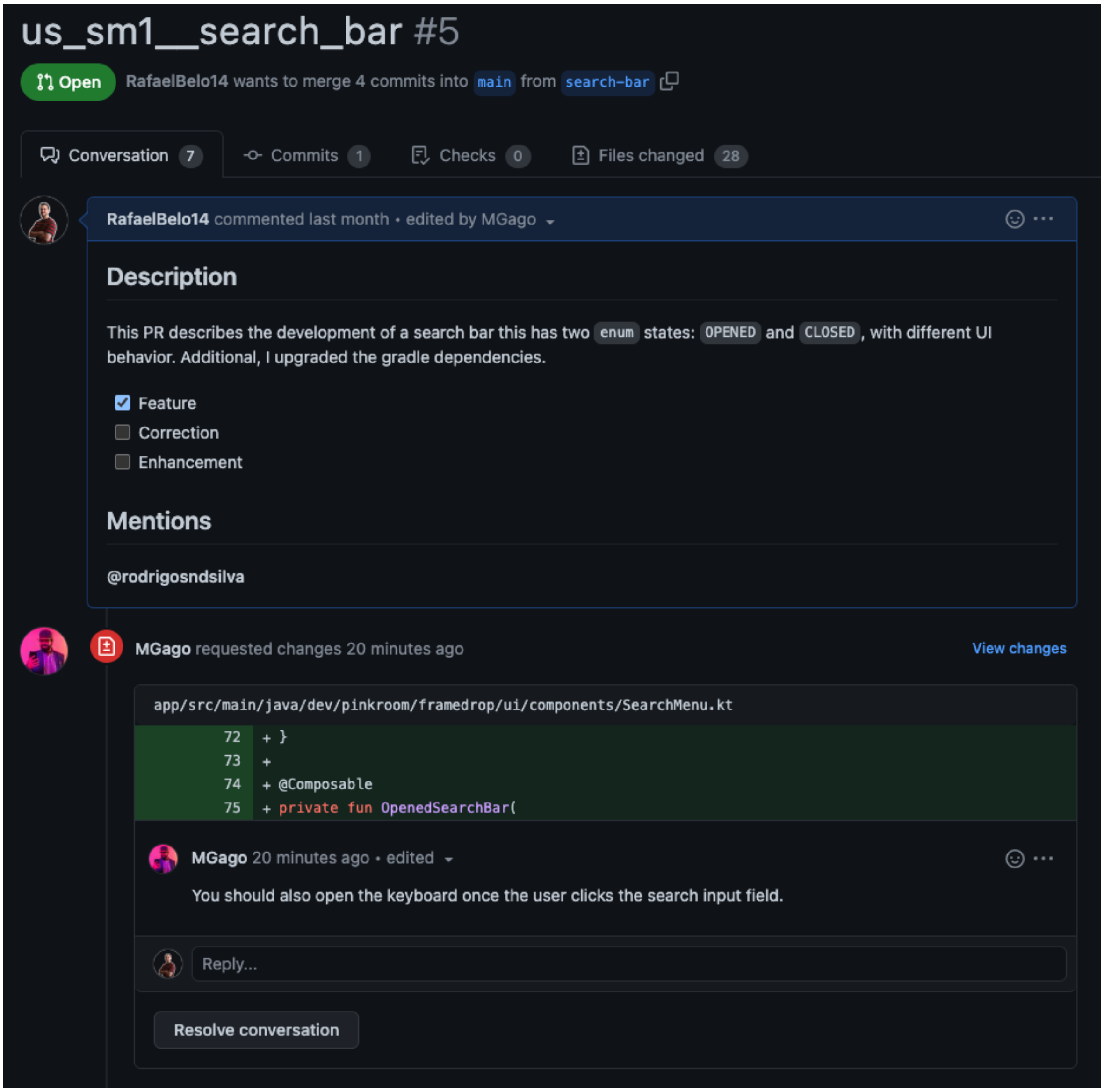
jmoreirawk

Assignees

RafaelBelo14

Figure 15. Code Reviewers and Assignees

If the code met the requirement specifications, the reviewer could accept the code which the developer merges to the main branch. If the code needed changes, the reviewer could leave comments about some aspects that may not be aligned with the purpose of that feature or some advice about code style or optimization.



us_sm1__search_bar #5

RafaelBelo14 wants to merge 4 commits into `main` from `search-bar`

RafaelBelo14 commented last month • edited by MGago

Description

This PR describes the development of a search bar this has two `enum` states: `OPENED` and `CLOSED`, with different UI behavior. Additional, I upgraded the gradle dependencies.

☑ Feature
□ Correction
□ Enhancement

Mentions

@rodrigosndsilva

MGago requested changes 20 minutes ago

app/src/main/java/dev/pinkroom/framedrop/ui/components/SearchMenu.kt

```
+ }
+
+ @Composable
+ private fun OpenedSearchBar(
```

MGago 20 minutes ago • edited

You should also open the keyboard once the user clicks the search input field.

Figure 16. Example of a Pull Request

In Figure 16, a Pull Request has been reviewed by the Scrum Master, Mário Gago, who requested changes. Since it was defined that every Pull Request would only be merged to the main branch if at least two reviewers approve the code, in this example, it still needs to be reviewed by at least one more element. When the code goes to the main branch it does not mean that the work is complete, since there is always a margin to evolve. Agile methodology is represented as a cyclical process, so a feature that was sent to the main branch can go back to the development stage, because it was not optimized and there is a margin to enhance, or because changes were requested or because it could potentiate future bugs. In this process, a new feature branch is created and the steps are repeated.

All the APK versions generated from the code that is sent to the main branch should be tested in quality assurance. Since code reviews are an internal process, and Agile methodologies bring the customers closer to the development, it is important to provide a way so they can test and give feedback about the application. As we defined in the context of this internship, the PO integrates as an element of the quality assurance team, analyzing if the developed features meet the requirements and to test potential bugs or defects. If a problem is found, the team reports it and the issues must be prioritized to understand if it is an urgent event that needs to be fixed right away or enqueued to a later fix. Only after that process the feature can be delivered to a staging environment, which is still an internal environment and aims to test under the same conditions of a production environment.

3.7 Delivery and Deployment

During development time the PO, SM, and developer should meet every two weeks to ensure that the development meets the requirements by showcasing the app and to discuss next approaches to the project, like prioritization of items in product backlog or converting feedback to new product backlog items. The features that were approved after the previous processes were deployed to the production environment, available to the end users. This process was automated by using a continuous deployment pipeline that is responsible to deploy the app to the specific environment. This should allow the release of small pieces of code that can be easier to troubleshoot in case of a problem and provide feedback loop acceleration.

3.8 Risks and Mitigation

Defining the threshold of success (ToS), followed by risk statements (condition and consequence) helped to understand issues/concerns that could stop the project from being successful (achieve the ToS). Risks were mostly defined using free-form brainstorming, based on knowledge of the project specifications and development experience. Before defining a mitigation plan for the identified risks, those must be subject to analysis to better understand the risk by determining its expected impact, probability, and timeframe. The levels of these attributes will be defined further.

3.8.1. Threshold of Success

When defining the number of conditions that must be met to consider a project a success, the areas typically considered are: scope, budget, schedule, and quality. Also, these conditions must be specific, measurable, and time bound. A good approach for building a ToS is to draw a failure picture, by listing possible paths that guide the project to failure. After this step, it becomes easier to convert these statements into minimum conditions to success. A good Threshold of Success is represented by 3-4 SMART[17] goals [77]. Table 3 represents the result of the methodology described above.

Sad path	What failed?	Minimum condition
The project was delivered without the "must have" requirements	"Must have" requirements	Deliver the project with all "must have" requirements
The project was not delivered on schedule	Schedule	Deliver the project on schedule
The project did not satisfy quality attributes	Quality attributes	The project satisfied quality attributes

Table 3. Results of SMART methodology

Building SMART goals with these minimum conditions requires a set of attributes that ensure a good goal definition. Since this project is being developed as the scope of an internship, the maximum delivery time is the end of the internship. Following SMART framework, the next goals represent the minimum condition to project success:

- Deliver all "must requirements" present in the Software Requirements Specification before the end of the internship.
- Deliver the project until the end of the internship.
- Product satisfies all the quality attributes present in the Software Requirements Specification until the end of the internship.

[17] SMART represents Specific, Measurable, Attainable, Relevant, and Timebound, and it is one of the most effective goal-setting strategies, because they clearly define all the different parameters involved in completing a task [104].

With the ToS defined, the next step is to define issues/concerns that could become a barrier to project success.

3.8.2. Risk Analysis

The majority of Framedrop's application operations depend on Framedrop's server API, since the data is stored in the Framedrop database and is only accessed from the server. Although they have their own database, Framedrop is dependent on Twitch, since the data stored in Framedrop's database comes from there. Since the server API was being developed in parallel with the mobile application, the majority of end-points were not available when development started and have been made available as development of the app went along. It is worth pointing out that the design process took place during the beginning of the semester, and was already finished at the time of development.

The statements mentioned above represent potential risks to the project's success. Converting them into risk statements (condition and consequence format) will facilitate their analysis in terms of impact, likelihood and the time from its identification to time that is required to deal with this risk.

- Framedrop's API server is not developed at the time of app development; might delay the development of the mobile application taking longer than was originally expected;
- Twitch blocks Framedrop's dependency to its services; might compromise all the application purpose;
- Framedrop's design presents flaws; might delay the development of the mobile application taking longer than was originally expected;
- The developer has less experience with the development language used in the project; might delay the development time and/or compromise application architecture or quality;
- Developed code is difficult to test; might compromise quality attributes.

With the risks identified, it becomes possible to identify their attributes. Impact attributes can be defined in three levels:

1. Marginal - the ToS can be reached without great difficulty;
2. Critical - the ToS can be reached, but with great effort/cost;
3. Catastrophic - the ToS can not be reached.

The likelihood can be defined in three level, as well:

1. Low or unlikely - < 40%;
2. Medium - between 40% and 70% ;
3. High or likely - > 70%.

For last, time bound can be defined in three levels, since it is considered that the time from identification to when one is required to deal with this risk is considered long for more than 3 months.

1. Short - < 1 month;
2. Medium - between 1 and 3 months;
3. Long - > 3 months.

Risk priority can be represented using a tridimensional risk matrix, since there are three attributes presented in Table 4.

Risk	Impact	Likelihood	Time Bound
Framedrop’s API server is not developed at the time of app development; might delay the development of the mobile application taking longer than was originally expected because some requirements are dependent on server end-points.	2	2	2
Twitch blocks Framedrop’s dependency to its services; might compromise all the application purpose.	3	1	3
Framedrop’s design presents flaws; might delay the development of the mobile application taking longer than was originally expected.	1	2	1
The developer has less experience with the development language used in the project; might delay the development time and/or compromise application architecture or quality.	2	1	2

Developed code is difficult to test; might compromise quality attributes.	2	1	2
Framedrop's project is dependent on the client availability; might delay the development of the mobile application.	3	1	1

Table 4. Risk Statements analyzed by impact, likelihood and time bound

Converting the data from the table below to a tridimensional risk matrix, represented in Figure 17, becomes easier to rank the risk statements by their priority level and conclude what risks are needed to be addressed first.

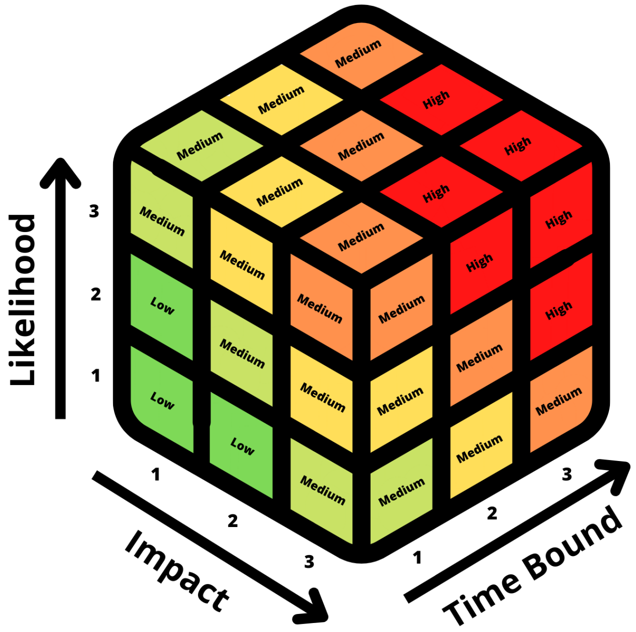


Figure 17. Tridimensional Matrix to measure risk priority

A comprehensive plan to manage, eliminate, or limit potential setbacks was developed in order to proactively address potential risks and minimize their impact on the overall success of the project.

3.8.3. Mitigation Plan

For each of the risks defined above a mitigation plan was developed which described the steps involved to mitigate the respective risk. Below, we described the top three mitigation strategies for the two highest priority risks. The complete plan for each risk is presented in Appendix C.

“Framedrop’s API server is not developed at the time of app development; might delay the development of the mobile application taking longer than was originally expected.”

1. Identify the specific requirements that are dependent on the server end-points, and prioritize them in terms of their impact on the overall development of the mobile application.
2. Develop a contingency plan for each of the identified requirements, in order to minimize their impact on the development timeline. For example, this could involve identifying alternative solutions or temporary workarounds that can be implemented until the API server is ready.
3. Work closely with the team responsible for developing the API server to ensure that they are aware of the dependencies on their work and to establish a clear timeline for the development and testing of the server.

“Twitch blocks Framedrop’s dependency to its services; might compromise all the application purpose.”

1. Identify and evaluate alternative services or platforms that could be used in place of Twitch, in order to maintain the functionality of the application. This could involve researching and comparing the features and capabilities of different platforms, as well as seeking input from key stakeholders and users of the application.
2. Develop a contingency plan for implementing the use of an alternative service or platform, including any necessary changes to the application's code or design. This should include steps to ensure a smooth transition from Twitch to the new platform, with minimal disruption to the application's functionality or user experience.
3. Work closely with the team responsible for implementing the use of the alternative platform, to ensure that they are aware of the potential risks and the steps being taken to mitigate them. This should include regular communication and collaboration to ensure that the transition is carried out smoothly and efficiently.

Chapter 4

Requirements

At the time of requirements specification, different categories of requirements were defined regarding functional requirements that were not related to system functionalities, but to quality attributes (non-functional requirements), assumptions, and constraints.

The functional requirements represented the Minimum Feature Set (MFS), since we are following a Minimum Viable Product (MVP) approach, reducing engineering waste to get the product to the customers sooner [78]. Throughout the project, there were some variations in the requirements, depending on the evolution of the users' needs, either adding new functionalities or removing them. In general, by following an agile methodology, we could have more flexibility to respond to the changes imposed on us.

Non-functional requirements (NFR) were defined considering Framedrop's needs, since they have another product and want to keep the same quality attributes. It was important that these attributes were measurable, so we could validate how well the system satisfies the needs of its stakeholders.

Software project assumptions are statements that are made about a project based on incomplete information or a lack of certainty. These assumptions were made in order to move forward with the project and can be revised during its development.

Constraints were defined based on resource limitations and business value propositions which restrict the freedom of finding solutions to certain aspects of the development. They may also be related to the competitive landscape and the needs of the target market, which restricts the degree of freedom of application features on the project scope, since it represents a specific audience and not all the features will be considered useful.

4.1 Out of Scope

For this assignment, it was left out of the scope an iOS mobile app (since it was only developed for the Android operating system), user clip recommendations based on previous interactions, the possibility to bid for an unclaimed clip (substituted for fixed costs and first-come-first-served approach), the notification menu, and the built-in editor to edit clips when they are “forked”. It is important to notice that, for future work, these requirements are planned to be re-analyzed and considered to the project.

4.2 Assumptions

Framedrop’s team has established the following assumptions for the project:

- Users want to authenticate through Twitch because they want to keep their information, such as their username and avatar, so that their followers on Twitch can recognize them.
- Users would like to be able to connect to the Framedrop mobile application through multiple devices with a single account.

4.3 Constraints

For this project there were a set of constraints that were taken into account. The minimum API level must be 24, which ensures a cumulative usage of 95.80% of Android devices. Although API level 21 ensures a cumulative usage of 99.20%, this choice was based on the rapid evolution of technology, and by the time the application is released, some older devices could stop supporting recent libraries [79]. As already mentioned, the Pink Room company has imposed the use of Kotlin for the development of the app, which is also the language recommended by Google and used by the company in their work.

Finally, the streaming communications are made using only the HLS protocol and any communications with third-party systems must be protected by encryption when circulating sensitive data for the system. These types of data include any information that could compromise the security of the system, user authentication data such as access tokens and refresh tokens, or personal data of users.

4.4 Non-Functional Requirements

Non-functional requirements are an important aspect of software development that impact the overall quality and effectiveness of an application. In the context of this assignment, the quality attributes that were considered to play a critical role in determining the user experience and the overall success of a software product were availability, security, usability, performance, and testability. The tables presented below represent the scenarios from the different quality attributes.

Availability is a key non-functional requirement that refers to the ability of an application to be available and functional for users at all times, as it is formalized in Table 5. Ensuring high availability is important for maintaining user satisfaction and ensuring that an application is able to meet the needs of its users.

ID	qa1
Actor	App
Trigger	Watch clips
Environment	Network slow/failing
Artifact	App
Response	Network failure when performing an action that communicates with Framedrop’s server
Response Measure	App does not crash

Table 5. Quality attribute 1

Security is another critical non-functional requirement, particularly in today's digital age where sensitive data is at risk of being accessed by unauthorized individuals. Implementing strong security measures, such as secure authentication, is essential for protecting user data and ensuring the integrity of an application, as it is mentioned in Table 6.

ID	qa2
Actor	App
Trigger	Authentication
Environment	Normal Operation
Artifact	Data
Response	Authentication successful
Response Measure	Sensitive information (email/username and password) never

	circulates again over the internet while the user is signed in

Table 6. Quality attribute 2

Usability is also an important factor to consider in the development process. Applications that are difficult to use or navigate can be frustrating for users, leading to a poor user experience. By designing interfaces considering the customers that will use them, will ensure that applications are easy to use and that users can accomplish their goals efficiently. Table 7 presentes this quality attribute.

ID	qa3
Actor	App
Trigger	Navigate from the clip feed to the profile page
Environment	Runtime
Artifact	App
Response	Open profile page
Response Measure	Users are able to navigate from point clip feed to point profile page in one click

Table 7. Quality attribute 3

Performance is another key non-functional requirement that can impact the user experience. Applications that are slow or prone to crashing can be frustrating for users and can lead to a poor overall impression of the product. Jakob Nielsen wrote in the "Usability Engineering" book that users have an attention limit of 1 second, and in the cases that the waiting time passes this limit, it should provide feedback to the user, since they do not know what is happening in the background [80]. The specification of this quality attribute is presented in Table 8.

ID	qa4
Actor	User
Trigger	Scroll between clips

Environment	Normal operation
Artifact	App
Response	Clip is rendered and start playing
Response Measure	Less than 1 second of latency

Table 8. Quality attribute 4

Finally, testability is an essential aspect of the development process. By writing comprehensive unit and integration tests, developers can ensure the reliability and correctness of their code, leading to a higher-quality product, as mentioned in Table 9. By following good architectural patterns that take advantage of low coupling components, and a continuous integration pipeline that runs the tests for every chunk of code that is developed, we can ensure that the application is less susceptible to defects.

ID	qa5
Actor	Tester
Trigger	Merge code to main branch
Environment	During development
Artifact	Code
Response	Build, run tests and notify in case of defects
Response Measure	Tests pass and merge process is completed

Table 9. Quality attribute 5

Overall, non-functional requirements are a crucial aspect of software development that can significantly impact the user experience and the overall success of a product. By considering these requirements throughout the development process, we can ensure that the application is reliable, secure, and easy to use, leading to a positive user experience and a successful product.

4.5 Functional Requirements

The application has 65 user stories scattered through the different system menus. While developing the Software Requirements Specification documented, annexed to this document in Appendix D, every user story followed a set of fields that described it, and served as a guide through the development process, described in Table 10.

Unique ID	An unique identifier, easy to read and that can identify the requirement and its location in the system.
Title	Presented in a user-story format.
Notes	An optional field that could serve as a guide when it is difficult to describe the requirement and can lead to ambiguity.
Priority	Following a MoSCoW (Must Have, Should Have, Could Have) technique that helps to prioritize the development.
Dependency	Identification of another set of user stories that this user story is dependent on. It helps to understand what needs to be completed in order to develop the respective requirement.
Estimation	An estimation of the development using a fibonacci-like numerical sequence. Helps to choose what user stories to develop next, based on the recommended user story points for the sprint.
Wireframe Reference	A reference to the wireframe that integrates this user story. Helps to guide between the different screens.

Table 10. User Story fields

Table 11 describes the functional requirements related to the profile menu. The complete list is described in Appendix D.

ID	Priority	Estimation
As an authenticated user, I should be presented with my profile information, so I can manage it.	Must Have	8
As an authenticated user, I should be able to navigate to the settings menu, so I can edit my experience on the application.	Could Have	1
As an authenticated user, I should be able to press to navigate to my edit profile menu, so I can edit my profile information.	Could Have	1
As an authenticated user, I should be able to navigate to the	Could Have	1

wallet menu, so I can top-up my wallet.		
As an authenticated user, I should be able to press the Twitch icon, so I can navigate to my Twitch account.	Could Have	2
As an authenticated user, I should be able to switch between "Streamed", "Claimed", "Forked", and "Upvotes" tabs, so I can see the content from each of them.	Must Have	3
As an authenticated user, I should be able to interact with a clip, so I can watch it.	Must Have	5
As an authenticated user, I should be able to navigate to clip feed, so I can view recommended clips.	Must Have	1
As an authenticated user, I should be able to navigate to the search menu, so I can search for users or videos.	Must Have	1
As an authenticated user, I should be able to navigate to the clip contest menu, so I can see clip classifications of a game that I like.	Must Have	1

Table 11. Profile Menu User Stories

4.6 Wireframing and Prototyping

During the project, we used wireframes to clearly define the requirements for the mobile app. This step was crucial in refining the app's user interface and user experience with the PO before actually building it. To do this, we were involved in a Product Design Sprint that, where we followed Jake Knapp's, Google Ventures, Design Sprint framework template which allowed us to rapidly prototype and test ideas in a short period of time, enabling better decisions and complex problem solving efficiently [81]. This framework helped us to identify and address potential problems early on by gathering feedback from real users and incorporating it into our design [82]. The complete description of this process is presented in Appendix E.

Chapter 5

Architecture

To describe the software structure, we used Simon Brown's C4 model that provides an effective way to represent the different layers of detail, and will create the diagrams in Archi, mapping ArchiMate concepts to the C4 meta model. The reason why we did not use ArchiMate is the fact that it could be time consuming if the person who's reviewing it is not familiar with the language, since connections are not labeled and are oriented to different goals [83]. The C4 model is based on an "abstraction-first" approach, reflected in four levels: Context, Containers, Components, and Code.

For the purpose of this work, we only described the first three levels, since the Code level does not add the appropriate level of detail. Other modeling languages could be taken into account, like the standard Unified Modeling Language (UML), but to visualize the whole application architecture, "would have to be drawn and brought together" [84], which can be confusing and time costly, especially for larger architectures [84]. The C4 model is flexible enough to be used in agile development, and features component reusability and maintainability when used with software modeling tools like Archi.

The mapping concepts are presented in Table 12.

C4	ArchiMate
Person	Business Actor
Software System and Container	Application Component
Component	Application Function
Relationship	Triggering Relationship

Table 12. C4 concepts mapped to AchiMate

5.1 System Context

Figure 18 represents the system context for the Framedrop mobile application, which in the C4 model represents the environment in which the system operates, including the other systems and components that interact with it. It is used to help understand and communicate the context in which the system exists and how it fits into the larger system landscape.

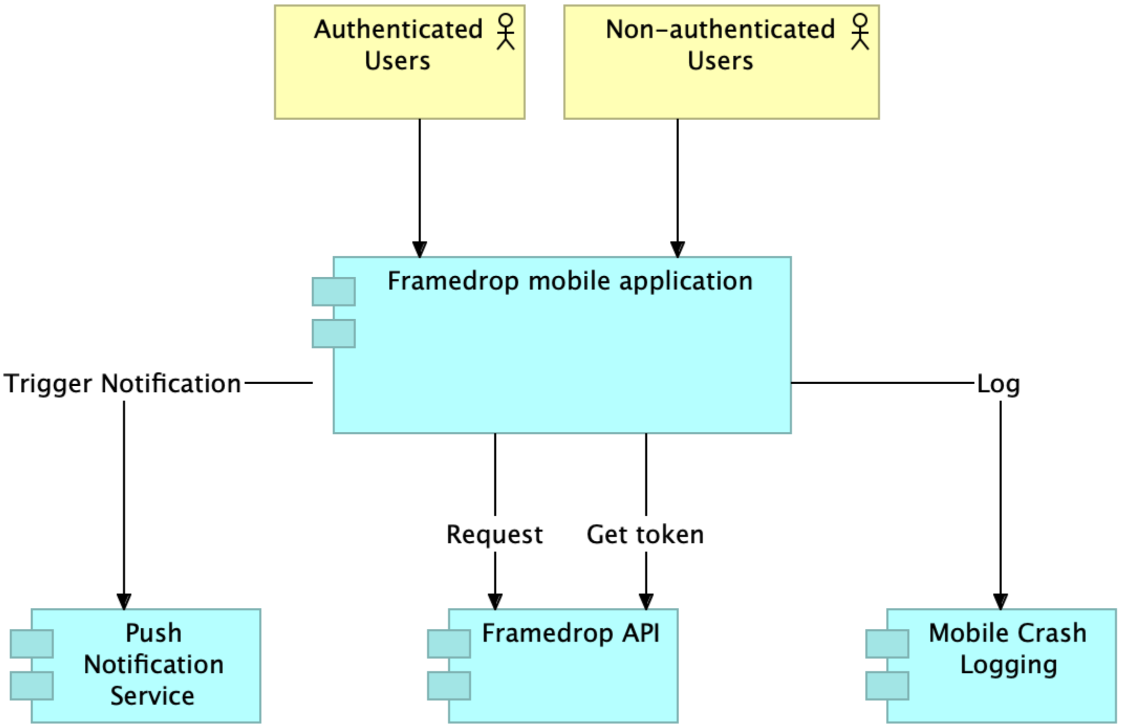
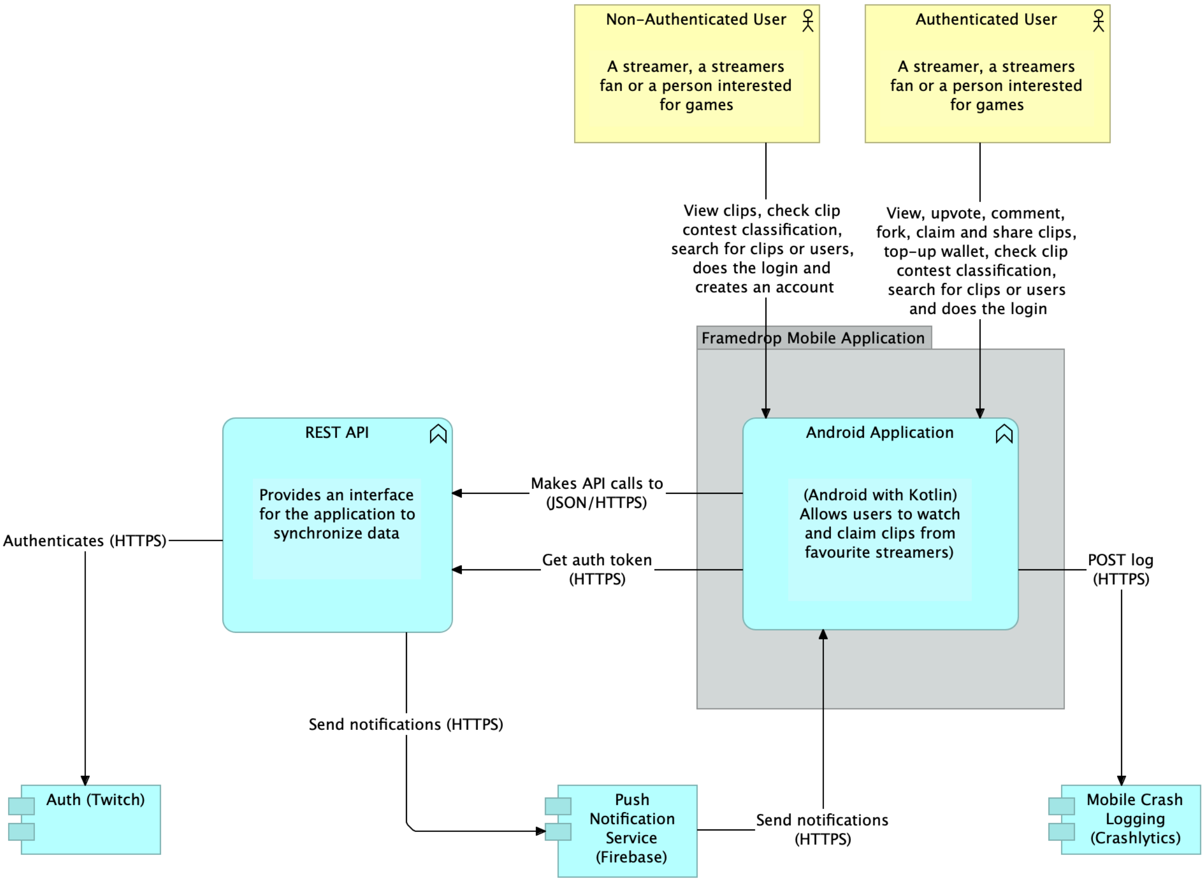


Figure 18. System context layer

The actors that interact with the system are authenticated and non-authenticated users, and the system interacts with the Framedrop server in form of requests to the API and to perform authentication. As we described in Chapter 2, the server deals with the process of authenticating with Twitch and retrieving the token and refresh token. As a push notification server, we use Firebase, which allows us to send notifications to the mobile application, and for mobile crash logs, we use Firebase Crashlytics service.

5.2 Containers

Figure 19 describes the Container diagram that consists of a high-level architecture that demonstrates how responsibilities are divided among its components, and how they communicate with each other [85].



Non-Authenticated User
A streamer, a streamers fan or a person interested for games
Authenticated User
A streamer, a streamers fan or a person interested for games
View clips, check clip contest classification, search for clips or users, does the login and creates an account
View, upvote, comment, fork, claim and share clips, top-up wallet, check clip contest classification, search for clips or users and does the login
Framedrop Mobile Application
REST API
Provides an interface for the application to synchronize data
Android Application
(Android with Kotlin) Allows users to watch and claim clips from favourite streamers)
Makes API calls to (JSON/HTTPS)
Get auth token (HTTPS)
Authenticates (HTTPS)
POST log (HTTPS)
Send notifications (HTTPS)
Send notifications (HTTPS)
Auth (Twitch)
Push Notification Service (Firebase)
Mobile Crash Logging (Crashlytics)

Figure 19. Containers layer

5.3 Components

Our Android mobile application uses the MVVM architectural pattern that was decided after an analysis described in Chapter 2, taking the advantage of low decoupled components and high cohesion. Following the clean architecture principles, the architecture follows three main layers, presentation, domain and data. The presentation layer includes the View and ViewModel components, where we defined a different architectural approach that is also described in Chapter 2, using an MVVM approach with some concepts of the MVI pattern, providing a Single Source of True for the states that the View component observes.

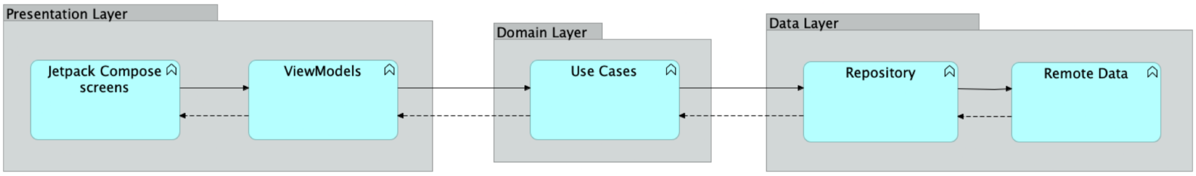


Figure 20. Components layer

Figure 20 details the architecture of Framedrop's application and the interactions between the different components. As it can be seen, the ViewModel acts as an intermediary between the View and the Domain layer. It receives a user input from the View, processes it triggering the appropriate use case, which contacts the Data layer and returns the results to the ViewModel. The ViewModel will then change its state and the View, which observes that state, will update accordingly.

Although the MVVM pattern helps to organize and structure code in a clean way, it can still result in boilerplate code, particularly when accessing data from local or remote databases.

The repository pattern as used in Android mobile development today, is based on the same concept of data separation and abstraction of the one introduced by Robert C. Martin [86]. In this context, the pattern is used to provide a clean separation between the data access code and the UI code by introducing a Repository class, which acts as a mediator between the Data layer and the Domain layer. This class is responsible for loading the data from the local or remote data source and providing it to the ViewModel when it is requested. This allows the ViewModel to remain decoupled from the underlying data source, making it easy to test and maintain.

Chapter 6

Planning

This chapter provides an overview of the high-level plan for the development process, milestones, and progress of the project. As the project follows agile methodologies, it is important to note that a detailed plan cannot be provided. However, key milestones have been identified to offer valuable insights into the development journey.

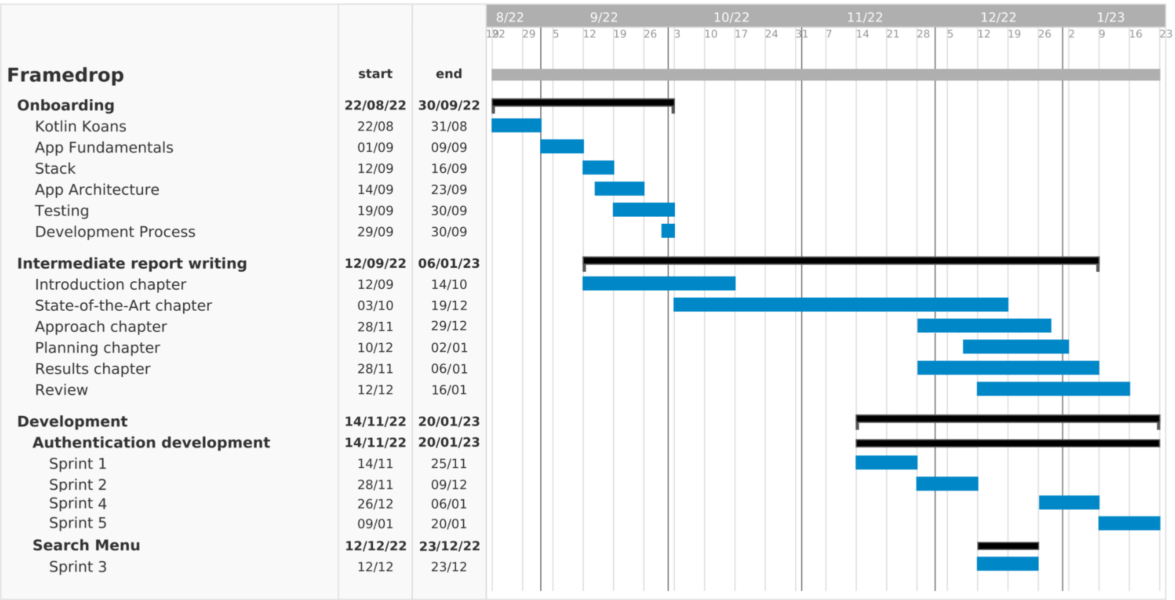
6.1 High Level Plan

The development process was planned to consist of 16 sprints, as we were following a two week sprint duration and had already begun development in the first semester. The following information covers the plan for the first and second semesters.

6.1.1. First Semester

The semester started as planned with the onboarding and first phases of writing the internship report. When the development started, some issues regarding workload started to influence the planning. Writing, developing and attending university courses became a challenge to balance in terms of work, so the tasks became stacked to the end of the semester, being the month of November the one with more intense university assignments and deliverables.

The development process was slowed down due to a shortage of available working time and delays in the development of the Framedrop API server. In addition, we encountered an unexpected problem with authentication, as the documentation did not align with the actual specifications. This required a thorough analysis of the authentication process. The results of the real process are described in Figure 21.

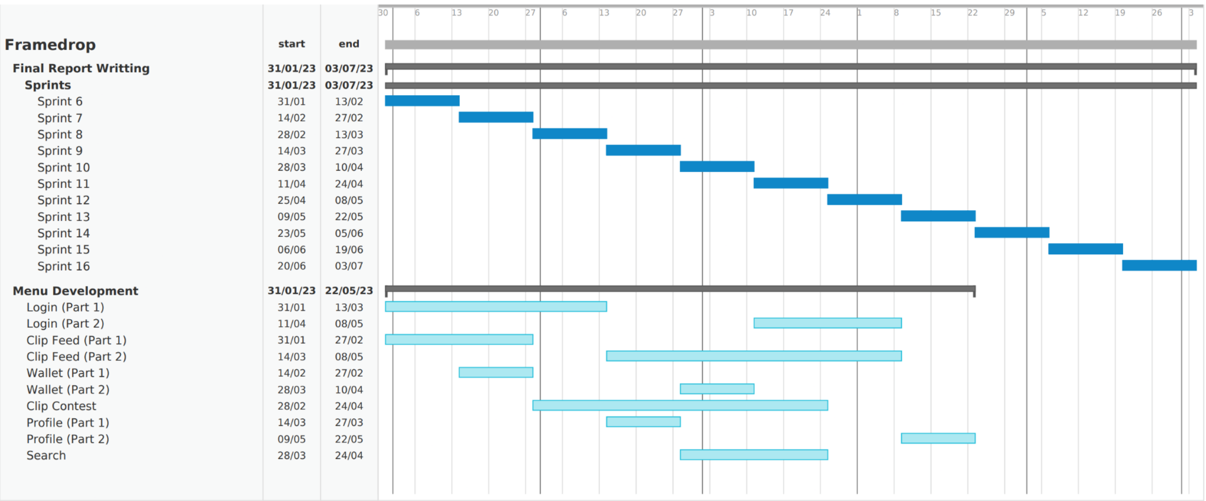


Framedrop	start	end
Onboarding	**22/08/22**	**30/09/22**
Kotlin Koans	22/08	31/08
App Fundamentals	01/09	09/09
Stack	12/09	16/09
App Architecture	14/09	23/09
Testing	19/09	30/09
Development Process	29/09	30/09
Intermediate report writing	**12/09/22**	**06/01/23**
Introduction chapter	12/09	14/10
State-of-the-Art chapter	03/10	19/12
Approach chapter	28/11	29/12
Planning chapter	10/12	02/01
Results chapter	28/11	06/01
Review	12/12	16/01
Development	**14/11/22**	**20/01/23**
Authentication development	**14/11/22**	**20/01/23**
Sprint 1	14/11	25/11
Sprint 2	28/11	09/12
Sprint 4	26/12	06/01
Sprint 5	09/01	20/01
Search Menu	**12/12/22**	**23/12/22**
Sprint 3	12/12	23/12

Figure 21. Real task plan for the first semester

6.1.2. Second Semester

As we embrace an agile development methodology centered around component iteration, our planning for the second semester perfectly aligned with this philosophy. It's worth noting that almost none of the implemented system components were developed from start to finish without requiring iterative enhancements. Figure 22 illustrates the progress made during these iterations, showcasing the evolution and refinement of the application.



Framedrop	start	end
Final Report Writting	**31/01/23**	**03/07/23**
Sprints	**31/01/23**	**03/07/23**
Sprint 6	31/01	13/02
Sprint 7	14/02	27/02
Sprint 8	28/02	13/03
Sprint 9	14/03	27/03
Sprint 10	28/03	10/04
Sprint 11	11/04	24/04
Sprint 12	25/04	08/05
Sprint 13	09/05	22/05
Sprint 14	23/05	05/06
Sprint 15	06/06	19/06
Sprint 16	20/06	03/07
Menu Development	**31/01/23**	**22/05/23**
Login (Part 1)	31/01	13/03
Login (Part 2)	11/04	08/05
Clip Feed (Part 1)	31/01	27/02
Clip Feed (Part 2)	14/03	08/05
Wallet (Part 1)	14/02	27/02
Wallet (Part 2)	28/03	10/04
Clip Contest	28/02	24/04
Profile (Part 1)	14/03	27/03
Profile (Part 2)	09/05	22/05
Search	28/03	24/04

Figure 22. Real task plan for the second semester

6.2 Sprint Level Plan

Since the user stories were already estimated, and even though these estimations can be refined during sprint planning, in every iteration a set of items in the product backlog was selected by priority and developed, taking into account the choice of the items' story points so that the sum would be close to the expected number of story points for the sprint.

Another important aspect to be mentioned is that during the first semester of the internship, the effort of theoretical and practical work was balanced to prevent one from jeopardizing the quality of the other, which directly impacted the product burndown chart, where the number of story points spent on development were usually below the velocity recommended for the sprint. The velocity is a measure used to help to determine the appropriate amount of work to commit to in each sprint.

The product burndown story points was calculated using a full time development approach along 16 sprints, and in the first semester, the work was made under part-time conditions.

The development process initiated in the first semester facilitated a smooth transition to full-time development, thanks to the groundwork already laid in the project base. As a result of these decisions, the development process accelerated, with the majority of sprint velocity exceeding the recommended pace, as depicted in Figure 23. This figure also illustrates the burndown chart of the requirements implementation process, which aids in analyzing the alignment between estimated and achieved goals.

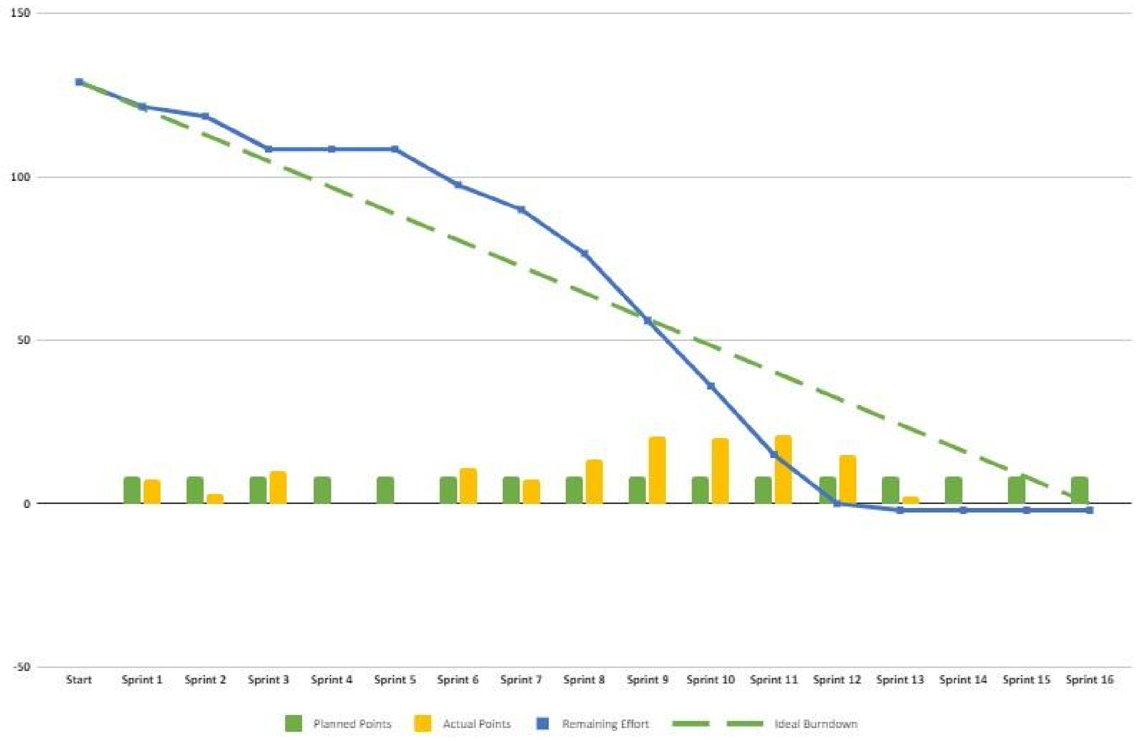


Figure 23. Product Burndown Chart artifact

Figure 24 represents only the progress of the first semester.

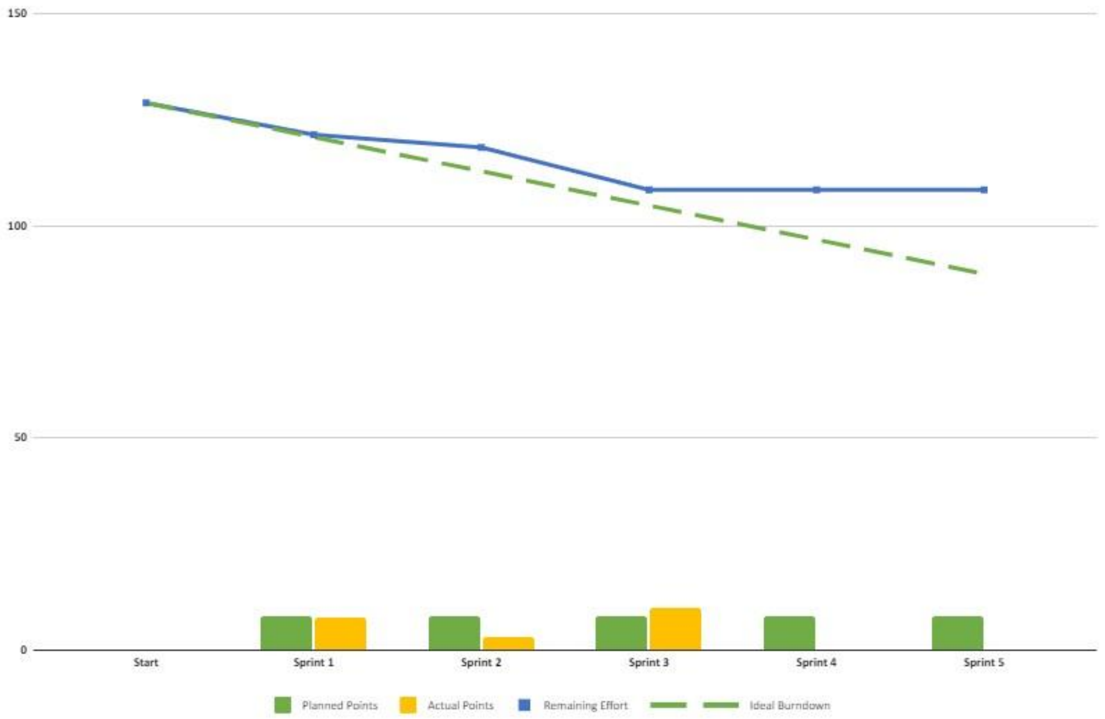


Figure 24. Product Burndown Chart diagram for the first semester

Beginning the development process in the first semester allowed a faster learning process, although balancing that part of the internship with theoretical work and classes raised some implementation issues like delays and incapacity to produce in such a short schedule window. Figure 25 represents only the progress of the second semester.

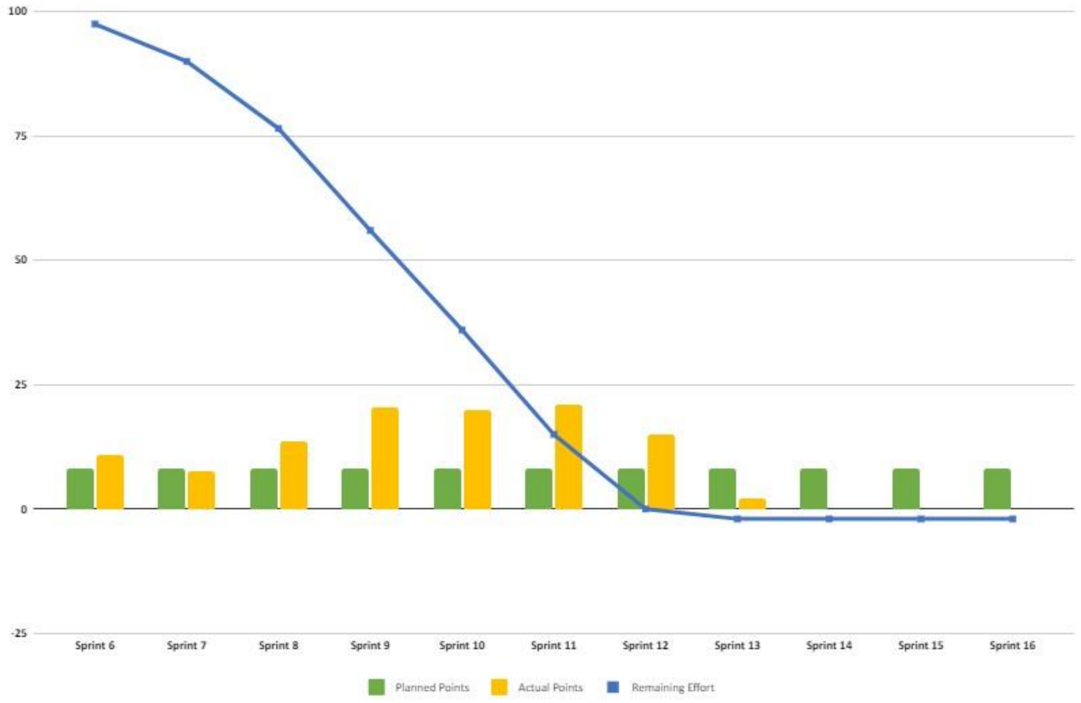


Figure 25. Product Burndown Chart diagram for the second semester

Overall, the requirements estimation process can be considered a success, with only one user story surpassing the anticipated point value. The utilization of an agile approach played a crucial role in achieving this outcome, as it allowed for the continuous analysis and re-estimation of backlog items in each sprint. Such adjustments were necessary due to various factors, such as alterations in the behavior of specific components. Consequently, upon concluding the development phase, we exceeded the projected point total by 2 points. This discrepancy arose primarily from significant design and logic modifications to the login component, which made it challenging to accurately forecast the required implementation effort.

The detailed estimations can be found in Appendix B, describing the specific user story's point usage, per sprint.

Chapter 7

Results

This chapter provides an overview of the outcomes achieved during the development process. It discusses the major challenges we encountered and the decisions we made along the way. Additionally, this chapter explores the testing and validation phase, including the testing criteria and important component unit and instrumentation tests that were performed. To assess the suitability of the Framedrop application for end customers, we conducted a usability test, the details of which are described below. Based on the results of the usability test, we made adjustments to a few features and implemented an optimization process, which is also covered in this chapter. The final product is an application that not only obeys the proposed functional and non-functional requirements but goes further and provides a usable and optimized experience to the end user.

7.1 Development

This project involved a total of 65 user stories, as outlined in Appendix F. The user stories went through several iterations, as the Framedrop team refined their product. Each sprint involved prioritizing the user stories to determine if they were still relevant for development. Notably, the decision was made to exclude the implementation of components related to claiming clips through an auction system. Instead, the team settled on a first-come, first-serve approach, for the purpose of this MVP.

During the development process, additional user stories were added, such as incorporating a countdown timer in the clip contest menu to display the remaining time and including a logout button for users to log out of the application, for example. In the end, a total of 49 user stories were successfully implemented in the scope of this internship. The remaining 16 user stories were not implemented, with 12 of them being deemed irrelevant for this application and the remaining 4 awaiting the development of corresponding endpoints by the Framedrop team.

Despite these unreleased features, they were not considered critical for this particular release, as all the essential features for the minimum viable product were successfully implemented, and the application is already deployed in internal testing on Play Store. The next steps will encompass the release to the next phases - close testing, open testing, and production.

7.1.1. Application Showcase

When launching the application for the first time, users are greeted with a login menu (Figure 26) offering two authentication options: Twitch or Guest login. Opting for Twitch authentication allows users to access additional features within the app, while selecting Guest login restricts certain interactions. When logging via Twitch, the creator code menu is presented (Figure 27) so the user can enter an optional creator code that will add more credits to his wallet balance, if it is the first time he is logging to the application. To complete the login process, users need to click the "Continue with Twitch" button, which opens a WebView featuring the Twitch authentication service (Figure 28).

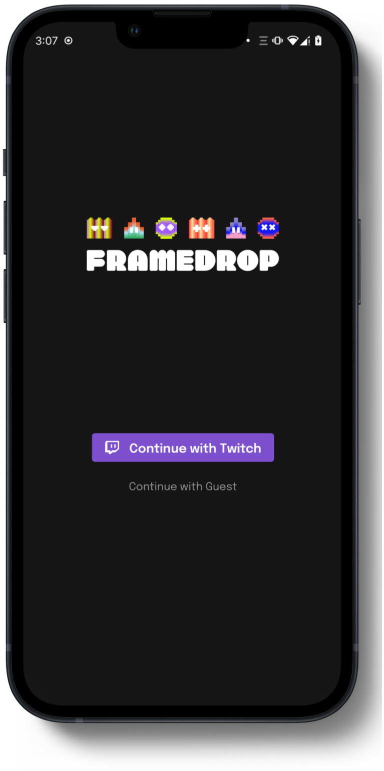
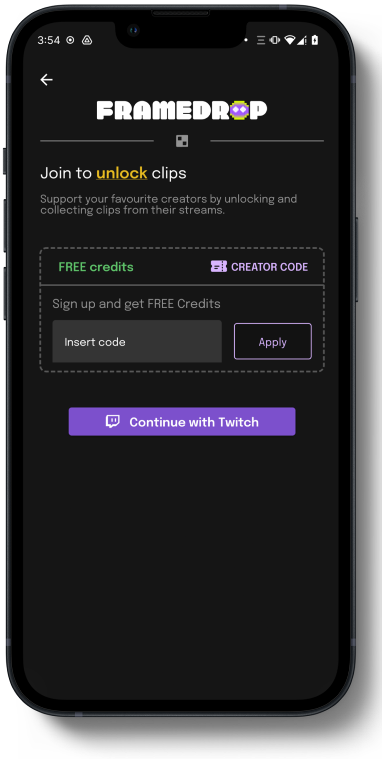


Figure 26. Login menu



Join to unlock clips

Support your favourite creators by unlocking and collecting clips from their streams.

Sign up and get FREE Credits

Figure 27. Creator Code menu

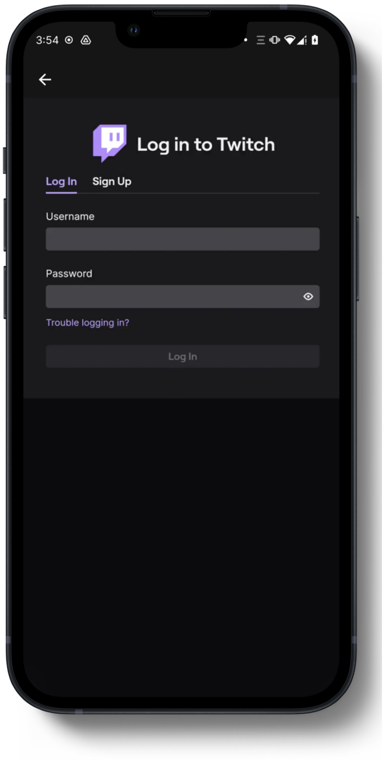


Figure 28. Login WebView

After proceeding past the login menu, users are presented with the Clip Feed menu (Figure 29), providing them with the ability to watch clips and engage with them through actions such as upvoting, commenting, and sharing.

Additionally, they can navigate to the Wallet menu (Figure 30) to add funds to their application wallet. The Wallet menu offers three predefined amount choices, but users also have the option to input a custom amount. To complete the payment process, a WebView is displayed, allowing users to finalize the transaction using the PayPal service.

Another way to access the Wallet menu is through the Profile menu (Figure 31). The Profile menu not only presents user information but also showcases the clips they have streamed, unlocked, and upvoted. Furthermore, it includes buttons to navigate to the user's Twitch profile and to log out of the application.

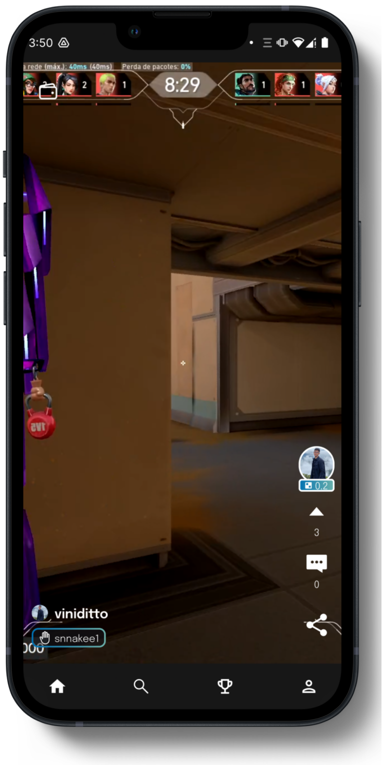
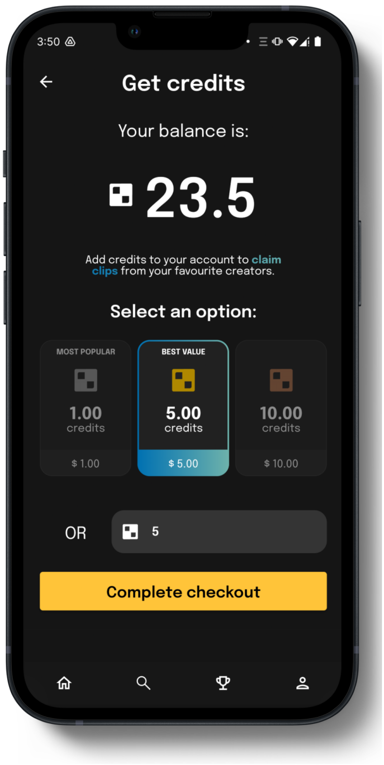


Figure 29. Clip Feed menu


Get credits

Your balance is:

23.5

Add credits to your account to claim clips from your favourite creators.

Select an option:

Figure 30. Wallet menu

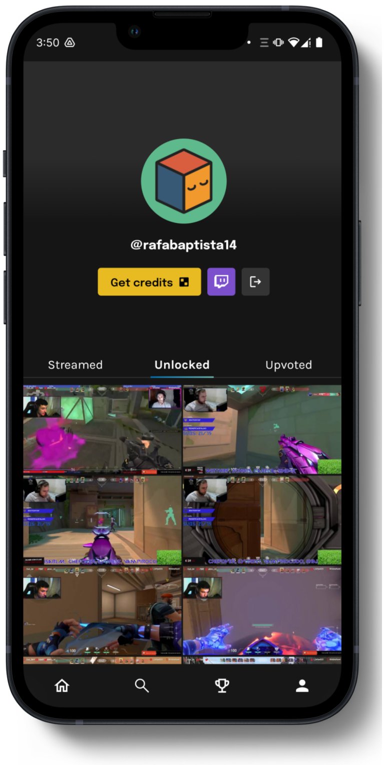


Figure 31. Profile menu

When accessing the Search menu, users have the ability to search for other users, as depicted in Figure 32. By clicking on the corresponding item from the search results, they can easily navigate to the selected user's profile.

Lastly, upon entering the Clip Contest menu, users are presented with a list of supported games (currently, Framedrop supports two games, as shown in Figure 33). Upon selecting a specific game, users are directed to the Clip Feed Scoreboard menu (Figure 34), where they can view the scoreboard for each contest period along with the remaining time. Users can switch between contest periods by clicking on the desired period or swiping horizontally through the clip list. Additionally, they can utilize the "My clips only toggle" to exclusively display their own clips (whether streamed or unlocked).

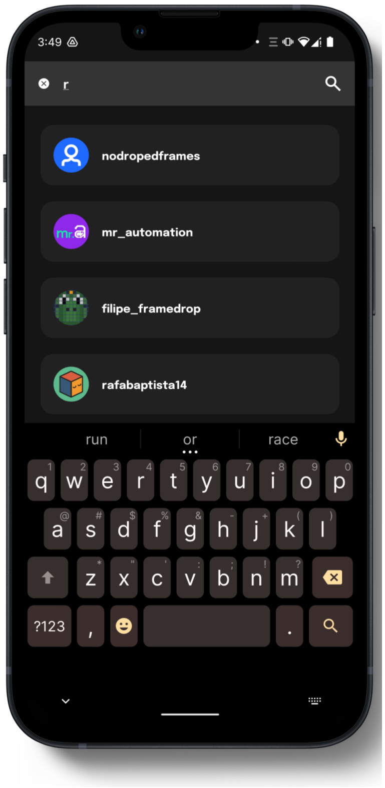


Figure 32. Search menu

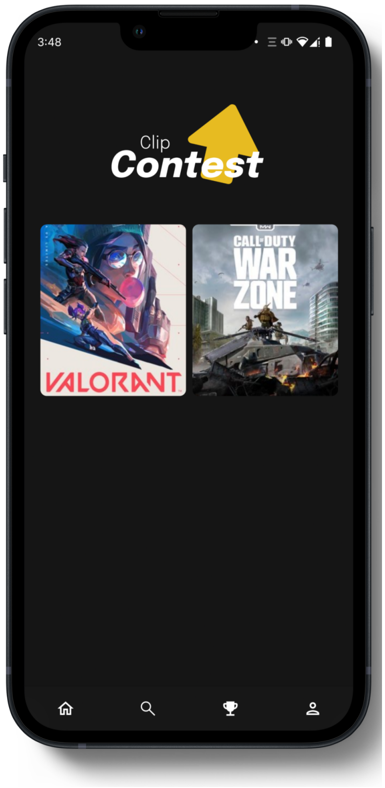


Figure 33. Clip Contest games menu

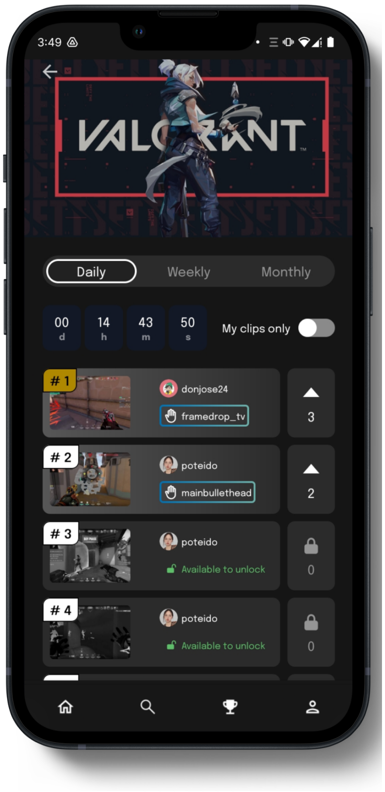


Figure 34. Clip Contest scoreboard menu

7.1.2. Major Challenges

During the implementation of the project, we encountered specific challenges related to working with complex components like ExoPlayer. Extensive investigation and research were required to ensure proper utilization of this technology. From the initial implementation to the optimization process, ExoPlayer proved to be a versatile tool with numerous possibilities. As a general tool that can be integrated into various environments, it provides a high level of customization, allowing developers to tailor it to their specific systems.

Since our app utilizes the HLS video format, we needed to delve into ExoPlayer HLS customization. As HLS employs adaptive bitrate streaming, we had to incorporate bandwidth awareness into ExoPlayer. Fortunately, the player already had predefined methods for this purpose. Additionally, since our app focuses on short-form content, one important feature was the ability for videos to loop seamlessly after playback. ExoPlayer offered a solution by implementing flags with different reproduction modes.

However, we encountered an issue when receiving videos from the API, as they often had extended durations compared to the desired length for the app. For example, if a video was supposed to be 24 seconds long in the app, it might be received from the API with a duration of 58 seconds. The API also provided starting and ending positions, allowing us to seek the player to the specified position. However, due to the extended duration, achieving seamless looping became problematic. Fortunately, our analysis of the ExoPlayer documentation revealed a message system that triggers events at specific positions in the video timeline. This discovery turned out to be the best-fit solution for triggering the loop. When the playback time matches the ending position, an event is triggered, seeking the player back to the starting position and enabling continuous looping of the video.

While the current solution is deemed the most optimal, the clips still experience a buffering time when looping, a drawback that would be minimized if the clips matched the exact duration displayed in the application. When presenting this issue to the client, the Framedrop team acknowledged its existence but considered it relatively minor. They clarified that reducing the clip duration to precisely match the video length would pose complications with video compression due to the utilized codecs.

Additionally, several iterations were carried out to refine the user experience of various app components. The login UI, as well as the UI for the Clip Contest and Profile Screen, were among the components that were aimed at and underwent multiple improvements to enhance their usability. In the previous implementation, the app would redirect the user to the mobile's browser window, initiating a cumbersome back-and-forth process where the browser would eventually redirect the user back to the app. However, with the new implementation, a WebView is integrated directly within the app, eliminating the need for external app redirections and streamlining the user experience.

The introduction of a WebView directly within the app brings several benefits to both the users and the Framedrop application itself. Firstly, this new implementation offers a seamless authentication process. Users no longer need to leave the app and open a separate browser, ensuring a smoother transition and eliminating potential confusion or distractions caused by switching between different applications. Moreover, by keeping users within the app during the authentication process, the new implementation enhances security and privacy. External browser windows may retain browsing history or leave temporary data behind, which can pose security risks. With the WebView approach, sensitive authentication data remains within the app's controlled environment, reducing the likelihood of information leaks or unauthorized access.

The Clip Contest and Profile menu presented a significant challenge due to the abundance of components, which made it difficult to navigate through the lists on these screens. To address this issue, a decision was made to hide some of the items located above the list, retaining only the essential ones, smoothly and seamlessly. To tackle this problem, extensive research was conducted by examining similar events in popular apps like Instagram[18] and Twitter[19]. The objective was to determine which elements should be hidden or preserved and their respective positions on the screen. Subsequently, the focus shifted towards finding suitable implementation solutions.

As Jetpack Compose is a relatively new technology, there is a limited availability of libraries that facilitate this specific action. Unfortunately, none of the existing libraries were capable of addressing our specific issue, being adopted a manual implementation approach. An Android Toolbar was created to consolidate the desired components for hiding or repositioning, and an animation was applied to this Toolbar when users scrolled through the screen. The resulting animation can be visualized in Figure 35, illustrating the transition achieved during scrolling in the Clip Contest menu.

[18] Instagram - https://www.instagram.com/

[19] Twitter - https://twitter.com/

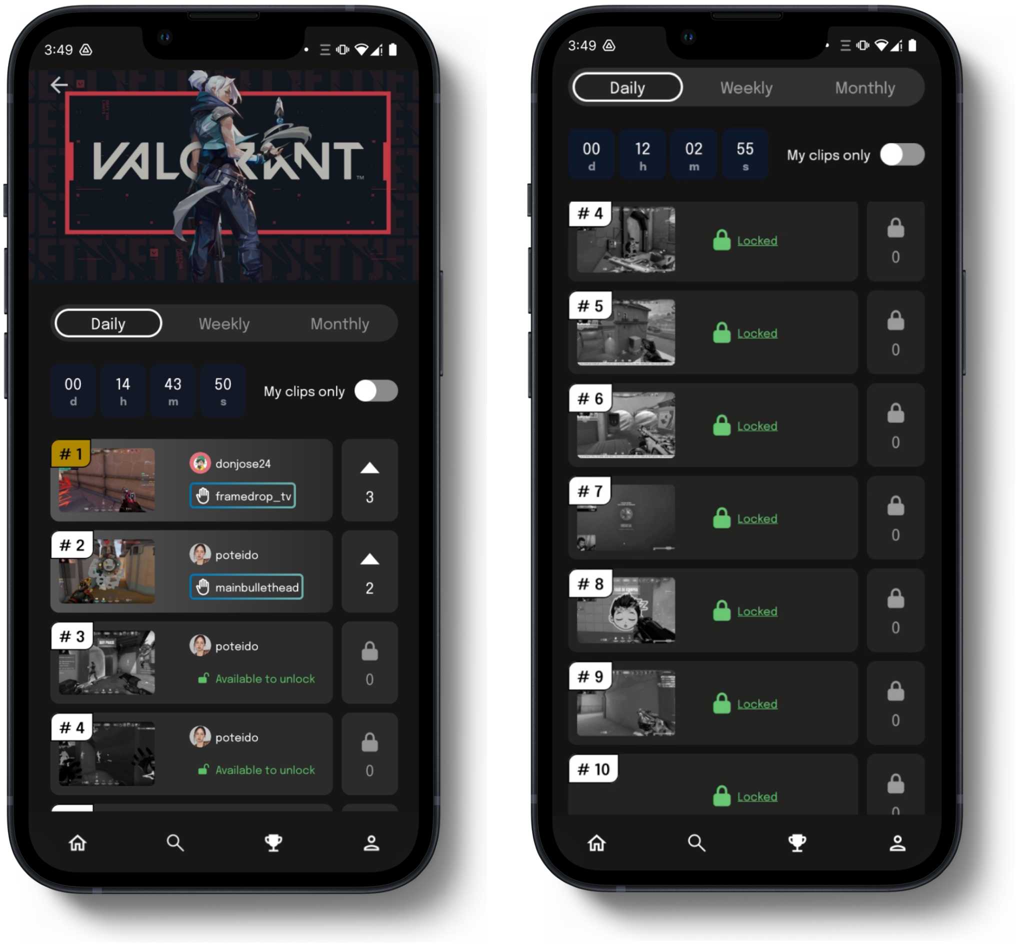


Figure 35. Scroll transition between extended and collapsed toolbar

This solution received an enthusiastic response from the Pink Room team, who proposed the development of a library that would facilitate the animation of an Android Toolbar. This library would allow for the collapse and expansion of the Toolbar by selectively hiding or revealing components on the screen. As part of my future work in the company, I will be creating this open-source library with the intention of making a positive contribution to the Android community.

7.2 Tests and Validation

In the context of this project, it was decided to test the critical paths, which are the most important and high-risk areas of the software, as they represent the functionalities that are most critical to the software's success and user satisfaction. This approach has the advantage of increasing the coverage of important functionality, by ensuring that the software meets its requirements and the critical bugs are identified and fixed, and is also cost-effective, since we are not testing all possible paths, which reduces the overall testing effort and resources required. Hence, our decision is to ensure a minimum test coverage of 90% for the critical paths as they constitute the primary focus of the testing process. This high coverage level will mitigate issues within the high-risk areas of the application, thereby enhancing the overall user experience.

7.2.1. Unit Tests

As our testing criteria we defined that we would be unit testing only the critical paths, namely the login flow, clip interactions in the clip feed, and payments in the wallet menu. The login menu offers two authentication methods, with the Twitch method being particularly important as it relies on a third-party service that could potentially go down unnoticed, resulting in a crash of the application.

Given that the primary user interactions involve watching videos and engaging with them through actions such as upvoting, claiming, commenting, and sharing, it is crucial to thoroughly test the underlying code for these events. This comprehensive testing is necessary to prevent any crashes that could disrupt the user experience.

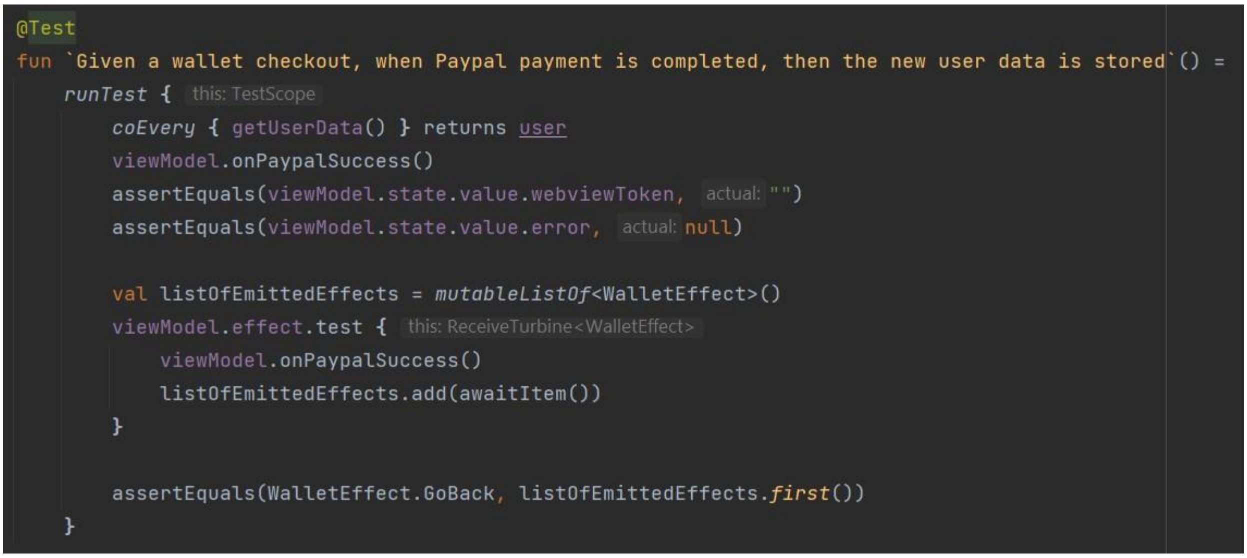
Lastly, although the payment process is handled by a third-party service like PayPal, and we cannot directly test the specific code associated with it, it is essential that the application appropriately responds to the information received from the PayPal service after completing the payment. Whether the payment is successful or not, the app should update the user's wallet balance accordingly and close the WebView used for the payment process.

Table 13 presents the coverage percentages of the mentioned components. It is crucial to acknowledge that the ViewModel classes under testing include these components, with the Clip Feed ViewModel encompassing more than just the critical components mentioned. Consequently, the coverage percentages may not be very close to 100% as expected, unlike the Wallet and Login ViewModel, which fully represent the component.

Critical Component	Class Coverage (%)	Method Coverage (%)	Line Coverage (%)
Login ViewModel	100%	100%	100%
Wallet ViewModel	100%	100%	100%
Clip Feed ViewModel	68%	78%	76%

Table 13. Critical component’s unit test coverage

Figure 36 illustrates a unit test example that verifies the successful storage of user data, including his wallet balance, following a completed PayPal checkout. Additionally, it asserts that the user is redirected to the previous screen – the screen he was on before accessing the wallet menu.



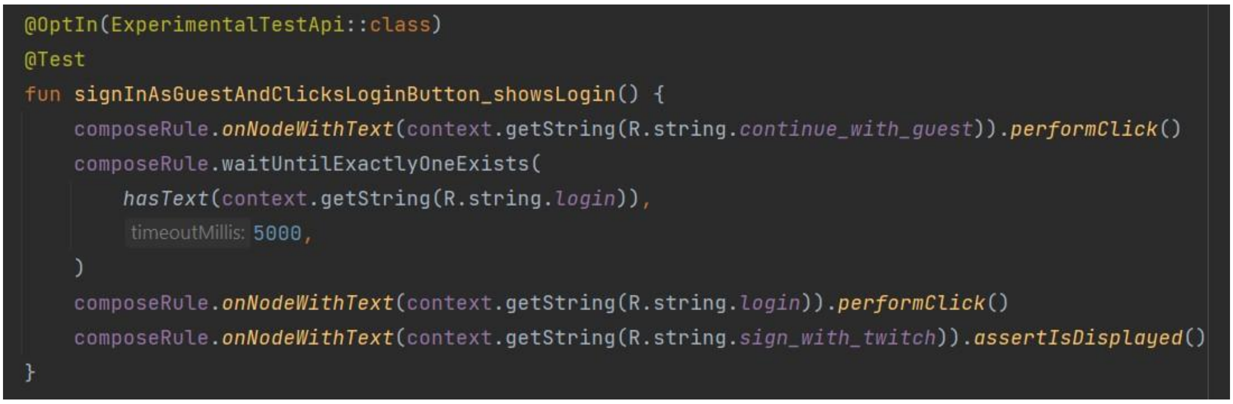
```
@Test
fun `Given a wallet checkout, when Paypal payment is completed, then the new user data is stored`() =
    runTest {  this: TestScope
        coEvery { getUserData() } returns user
        viewModel.onPaypalSuccess()
        assertEquals(viewModel.state.value.webviewToken,  actual: "")
        assertEquals(viewModel.state.value.error,  actual: null)

        val listOfEmittedEffects = mutableListOf<WalletEffect>()
        viewModel.effect.test {  this: ReceiveTurbine<WalletEffect>
            viewModel.onPaypalSuccess()
            listOfEmittedEffects.add(awaitItem())
        }

        assertEquals(WalletEffect.GoBack, listOfEmittedEffects.first())
    }
```

Figure 36. Wallet PayPal success unit test

7.2.2. Instrumentation Tests

During the development of the mobile application, our testing approach underwent adaptation to accommodate the unique circumstances surrounding the project. As we neared the completion of the development phase, the Framedrop company announced a significant transformation, including an upcoming rebranding process. In light of this announcement, our testing focus shifted primarily towards unit testing, considering that the core logic of the app was not expected to undergo substantial changes. This decision was driven by the awareness of forthcoming modifications to the application's user interface, leading us to prioritize resource and time allocation, avoiding unnecessary UI testing that would become irrelevant due to the imminent UI overhaul.

While unit testing took precedence, we still conducted some UI testing as part of our academic environment to enhance our understanding of this specific testing domain. Our aim was not to comprehensively validate the UI in its current state, but rather to gain practical knowledge and insights into UI testing methodologies. This approach allowed us to familiarize ourselves with the intricacies of UI testing and explore its potential impact on the overall testing process. Figure 37 presents an example of an instrumentation test that asserts that the login pop-up is displayed when a non-authenticated user clicks on the login button in the Clip Feed menu.



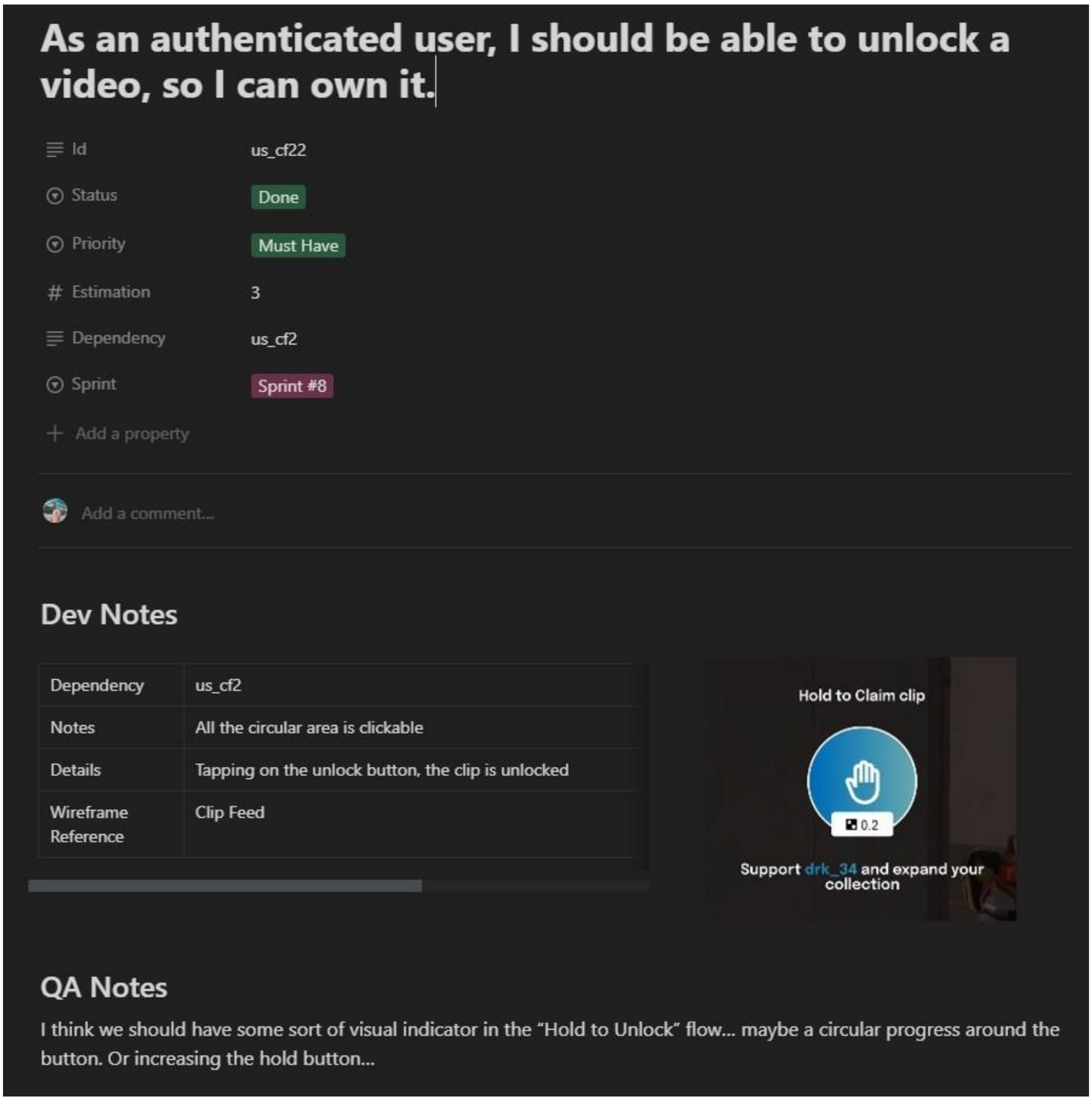
```
@OptIn(ExperimentalTestApi::class)
@Test
fun signInAsGuestAndClicksLoginButton_showsLogin() {
    composeRule.onNodeWithText(context.getString(R.string.continue_with_guest)).performClick()
    composeRule.waitUntilExactlyOneExists(
        hasText(context.getString(R.string.login)),
        timeoutMillis: 5000,
    )
    composeRule.onNodeWithText(context.getString(R.string.login)).performClick()
    composeRule.onNodeWithText(context.getString(R.string.sign_with_twitch)).assertIsDisplayed()
}
```

Figure 37. Login Pop-up Showing Test

7.2.3. Quality Assurance

As described in Chapter 3, Subchapter 3.6, the Quality Assurance (QA) process was an important step to ensure that the implemented code fulfilled the requirements and to identify any bugs that may have gone unnoticed during the code review phase. Since we were using the Notion tool to store the backlog items and manage the user stories progress through the different phases of development, it was also used to store the QA review from the Product Owner. This review may represent any issues that have been identified during the tests, as well as the actions to be taken to resolve them. Since every user-story had its own card, the QA information regarding specific user-stories was stored separately, leading to a more organized and manageable environment.

An example of a user-story card is presented in Figure 38, that informs how the application must react to the implemented code, under “Dev Notes”, and the QA review under “QA Notes”.



As an authenticated user, I should be able to unlock a video, so I can own it.

Id	us_cf22
Status	Done
Priority	Must Have
Estimation	3
Dependency	us_cf2
Sprint	Sprint #8

Dev Notes

Dependency	us_cf2
Notes	All the circular area is clickable
Details	Tapping on the unlock button, the clip is unlocked
Wireframe Reference	Clip Feed

QA Notes

I think we should have some sort of visual indicator in the "Hold to Unlock" flow... maybe a circular progress around the button. Or increasing the hold button...

Figure 38. User story with changes requested after QA

7.3 Usability Test

The primary objective of this test was to conduct a comprehensive evaluation of the overall usability and user experience of the mobile app, with a specific emphasis on identifying potential issues and gathering valuable feedback from users. In order to achieve this goal, a carefully selected sample of 5 participants was invited to participate in the test, and their feedback was systematically collected and analyzed.

In the field of usability testing, there is a prevailing tendency among some companies to dismiss the significance of such tests due to misconceptions that they are resource-intensive endeavors necessitating a large number of participants. Renowned usability expert Jakob Nielsen has highlighted this concern, stating that many companies mistakenly believe that the number of usability problems detected will stabilize as new participants are involved, making additional feedback redundant. Nielsen has explained this phenomenon through the application of Poisson Distribution, which elucidates the probability of achieving a given number of successes in a series of trials. His research has led to the conclusion that at least 15 users are required to unearth all the usability problems inherent in a design. However, it should be noted that the utilization of 5 participants strikes a balance between optimizing resources and attaining meaningful feedback [87].

While some companies may find it feasible to conduct a single usability test with a larger pool of 15 participants, it is essential to emphasize that the present evaluation of the Framedrop app is part of a series of planned usability tests in subsequent stages [88].

7.3.1. Preparation

To perform the usability tests, a structured approach was adopted, consisting of three distinct sections. The first section involved an introduction, where the goals and objectives of the test session were explained to the participants, presented in Appendix G. This introduction served to set the stage and provide participants with an understanding of what would unfold during the testing process.

Following the introduction, the second section entailed a series of general questions that were posed to the participants, presented in Appendix H. These questions aimed to gather additional information about the participants, including their personal background and their habits related to gaming consumption. By eliciting such information, a clearer context could be established, allowing for a better understanding of how the participants' backgrounds and preferences might influence their experiences with the app. The answers are presented in Appendix I.

The third section constituted the core of the usability test. Here, participants were presented with the app and provided with a scripted set of tasks to perform:

- **Task 1:** Watch 4 clips in Clip Feed and upvote 2 of them.
- **Task 2:** Navigate to the owner’s profile of the 10th positioned clip in Clip Contest of the game Valorant.
- **Task 3:** Top up the wallet with 10 credits. Stop when the app asks for payment information.
- **Task 4:** Claim a clip.

The script served as a guide, ensuring consistency across the test sessions. The tasks were thoughtfully crafted to cover a range of user interactions and encompassed typical app usage scenarios. This approach allowed for a comprehensive evaluation of the app's usability, as participants engaged with the app and completed the assigned tasks.

7.3.2. Results

Throughout the participant's task completion process, meticulous documentation of their interactions was essential, ensuring a comprehensive understanding of their behaviors. Continuous communication with the participant played a crucial role in capturing and comprehending their actions effectively. At the conclusion of each task, any encountered problems or challenges faced by the participant, as well as their suggestions, were recorded. To facilitate this process, a set of predefined questions was employed to encourage an open conversation and elicit valuable insights.

Upon completion of the designated tasks, participants were provided with a few minutes to freely explore the app, during which they were encouraged to provide additional feedback. This phase was of utmost importance as it allowed for the documentation of any issues or suggestions that may have arisen in areas not explicitly addressed during the structured usability test.

The 3 problems with the most severity are presented in Table 14, and the complete list is presented in Appendix J. The list of suggestions are presented in Table 15.

Problems	Severity	Possible Solutions
Videos are slow to play and buffer several times	High	Change clips buffer size. Framedrop team changes video store service
Transition between videos not very fluid	High	Move some logic to another thread
Confirm that the user wants to unlock the clip, to avoid accidental touches	High	Create a confirmation pop-up or create a long-press animation

Table 14. Usability Test most severe problems

Suggestions	Implementation Effort	Will it be included in the current release?
After opening the keyboard, you can hide it by touching anywhere on the screen	Low	Yes
Show mini clips of the user in the clip that is locked (similar to youtube videos, when the video ends)	High	No

Table 15. Usability Test gathered suggestions

Usability tests play a crucial role in the development process by shedding light on potential issues that may have been overlooked during the initial stages. These tests provide valuable insights into how users interact with the product and identify areas where improvements can be made to enhance the overall user experience. They also offer a unique opportunity to evaluate the product's functionality, ease of use, and effectiveness in meeting user needs. By involving real users in the testing process, we gain valuable feedback on their preferences, pain points, and expectations.

With these insights in hand, a refactoring process started where we modified the necessary code and implemented the recommended changes. Since some of the highest priority problems relied on the application's performance, it was decided to investigate app optimization mechanisms. From this analysis, a report was developed measuring the app performance before and after applying the optimization mechanisms.

In order to assess performance, the decision was made to utilize the Android Macrobenchmark library, which offered the necessary functionalities for measuring app startup time and scrolling. This tool was selected due to its official support from Google and its development and maintenance by the Android team, ensuring compatibility with the latest Android version. Its design aims to deliver consistent and accurate results across diverse devices and Android versions, taking into consideration various factors that can influence app startup time, such as device specifications, system resources, and background processes. By providing dependable measurements, it obviates the need for third-party benchmarking tools. Moreover, its integration with Android Studio allows for convenient execution of benchmark tests and analysis of results within the development environment [89].

7.4 App Optimization

The optimization efforts were directed towards enhancing app performance and minimizing frame drops during playback. This involved a four-step process: macrobenchmarking the base code, generating a baseline profile, optimizing the views, and refining the ExoPlayer implementation. Each step introduced specific optimizations to improve the overall performance of the app.

Macrobenchmarking is a comprehensive performance testing approach employed to evaluate the overall system performance of Android devices. It provides a holistic view of how Android devices perform under real-world scenarios and workloads. During macrobenchmarking, factors such as CPU and GPU performance, memory usage, battery consumption, and thermal management are taken into consideration as they significantly impact the device's performance. To ensure reliable and accurate results, the tests were performed in 5 devices ranging from 2018 to 2022, giving a wide variety of software and hardware.

In order to maintain accuracy and reliability, real devices were used for all tests instead of emulators. This approach aimed to capture real-world performance characteristics and variations. To minimize potential interference and resource allocation issues, the test devices were devoid of any background apps or processes. This created a clean testing environment, isolating the performance of the targeted task from any potential background activities.

Additionally, a wired USB connection was employed to connect the devices for testing, avoiding potential network-related latencies or fluctuations that could impact the test results. This choice ensured a stable and reliable connection, allowing the study to focus solely on the intrinsic performance capabilities of the devices.

The testing methodology revolved around running the same task for 100 iterations. This approach was adopted to assess the consistency and stability of the task's performance across multiple runs.

7.4.1. Macrobenchmarking the Base Code

Initially, the mobile app's base code, without any optimization, was macrobenchmarked. The purpose of this step was to establish a baseline performance level and identify areas for improvement.

7.4.2. Generating a Baseline Profile

Baseline Profiles enhance the speed at which code executes, resulting in an improvement of “30% from the first launch” [90]. This is achieved by bypassing interpretation and just-in-time (JIT) compilation steps for the relevant code paths.

When an app or library includes a Baseline Profile, the Android Runtime (ART) can optimize specific code paths using Ahead-of-Time (AOT) compilation. As a result, every new user and every app update can benefit from these performance enhancements. This optimization technique, known as Profile Guided Optimization (PGO), enables apps to optimize startup time, reduce instances of lag during user interactions, and enhance overall runtime performance from the moment the app is first launched. [90]

7.4.3. Optimizing the Views

At the start of the internship, no prior experience with Jetpack Compose was possessed. Consequently, the next objective was to enhance the app's views by leveraging the increased familiarity and practice with the technology, enabling the code to be refactored using a more effective approach. This involved streamlining the layout hierarchy, reducing redundant view updates, and enhancing the rendering pipeline. The optimization efforts went unnoticed, indicating a highly positive code review process and learning journey, as minimal mistakes were made from the beginning.

7.4.4. Optimizing the ExoPlayer Implementation

As video playback performance was crucial for Framedrop, the next step involved optimizing the ExoPlayer implementation. This optimization focused on enhancing video decoding, buffering, and rendering processes. Techniques such as hardware acceleration, adaptive streaming, and buffer management were employed to reduce frame drops, improve playback smoothness, and minimize resource consumption.

In the Framedrop app, it is important to understand that the performance of video optimization is inherently dependent on the server's capabilities. Currently, the Framedrop team stores the videos in S3 buckets on AWS, which introduces a latency issue due to the geographical location of the servers in the United States.

S3 buckets are designed primarily for storage rather than transmission of data. As a result, when users access the videos from the app, there is a noticeable delay in video playback due to the round trip time (RTT) required for data to travel from the user to the servers and back. This latency typically ranges from 100 to 120 milliseconds, impacting the user experience.

To address this latency concern and improve performance, the Framedrop team should consider implementing a Content Delivery Network (CDN) on top of the S3 buckets. By deploying a CDN, an on-edge server can be established closer to the app users. This means that when a user requests a video, it would be transmitted from a closer server, reducing the distance data needs to travel and consequently minimizing latency.

7.4.5. Results

The optimized app is the result of the three stages of optimization with baseline profiles, view optimization, and exoplayer optimization. The following tests were made in order to acquire the startup time of the application and the product. The main difference is that for the application, we focused solely on the components developed by the development team. Consequently, the timing measurement concluded before the application initiated the process of requesting the video list for the Clip Feed menu. On the other hand, the product's timing encompassed the complete flow until the first requested video started playing.

The third test aimed to evaluate the smoothness of the applications and detect any UI jank. In Android, the user interface is rendered by generating frames from the app and displaying them on the screen. If the app encounters slow UI rendering, frames may be skipped, resulting in a recurring flicker known as jank [91]. To evaluate this, we analyzed two key metrics: "Frame Duration" and "Frame Overrun." The first indicates the time required to render a frame, while the second denotes the amount of time by which a frame misses its deadline. Positive numbers signify dropped frames and visible jank or stutter, while negative numbers indicate frames that are faster than the deadline [92].

These results are presented using the 50th, 90th, 95th, and 99th percentile, where the 95th and 99th percentile are the most crucial metrics. This is because the 95th percentile response time denotes the time it takes for a request to be processed and is slower than 95% of all other requests. To illustrate, if there are 100 requests, the 95th percentile response time corresponds to the time it takes for the 95th slowest request to be fulfilled. The same applies to the 99th percentile. This concludes that if the app presents a low 99th percentile response time, it may indicate it is functioning efficiently even during periods of high workload [93].

The complete analysis is presented in Appendix K, with the results of each device. In Figure 39 to Figure 44 is shown the optimization results from every device.

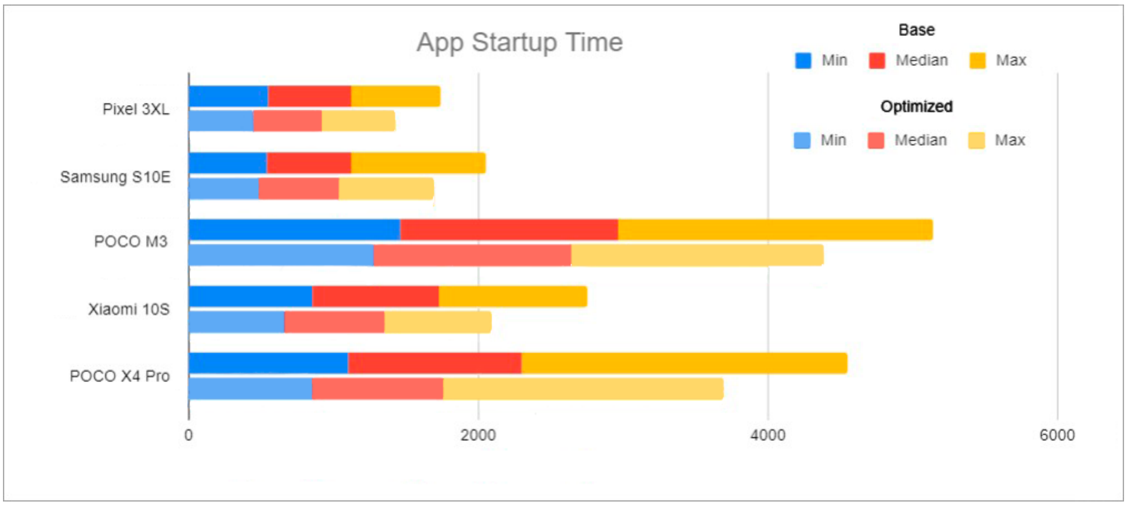


Figure 39. App Startup Time Results

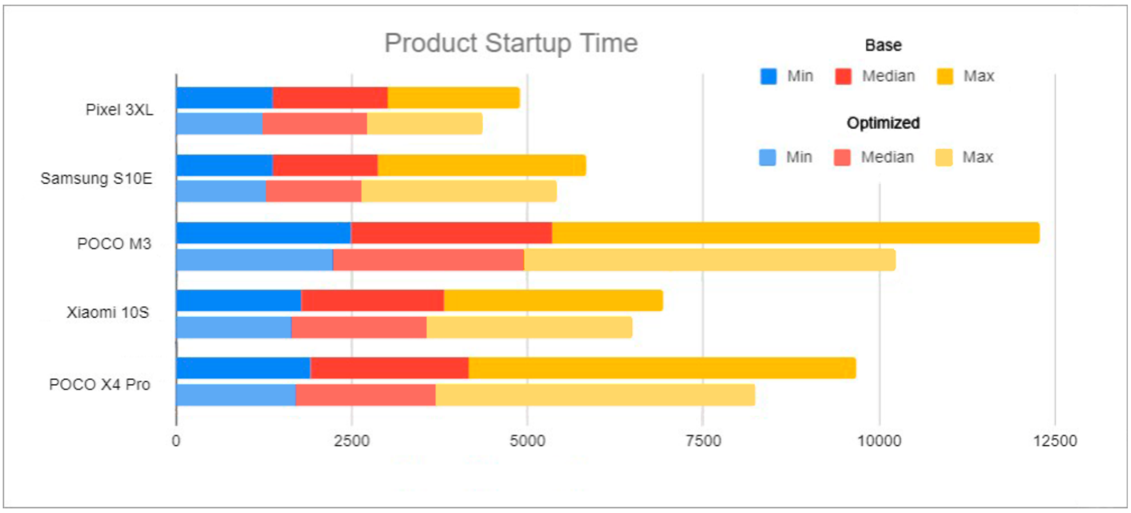


Figure 40. Product Startup Time Results

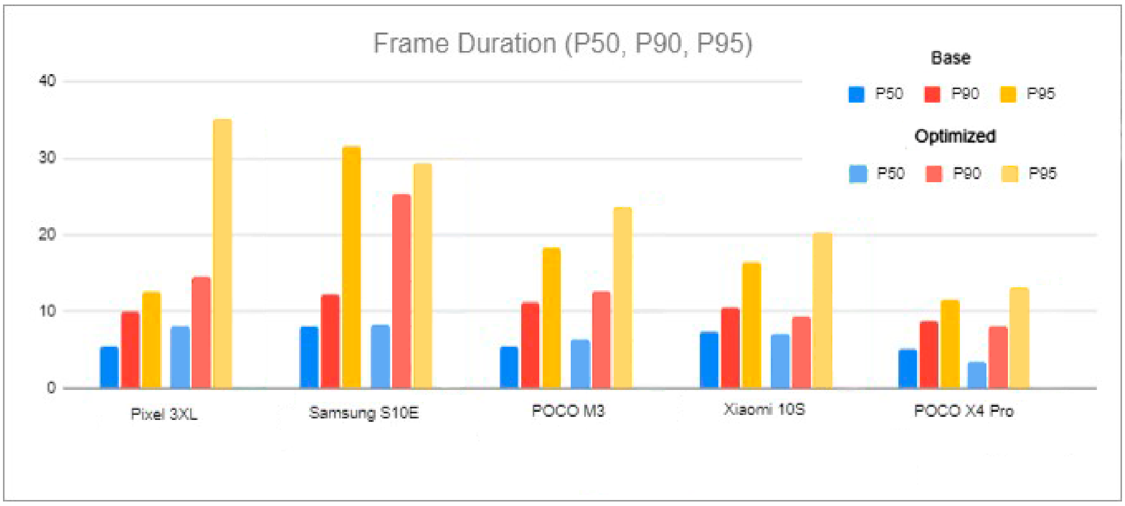


Figure 41. Frame Duration Results (P50, P90, P95)

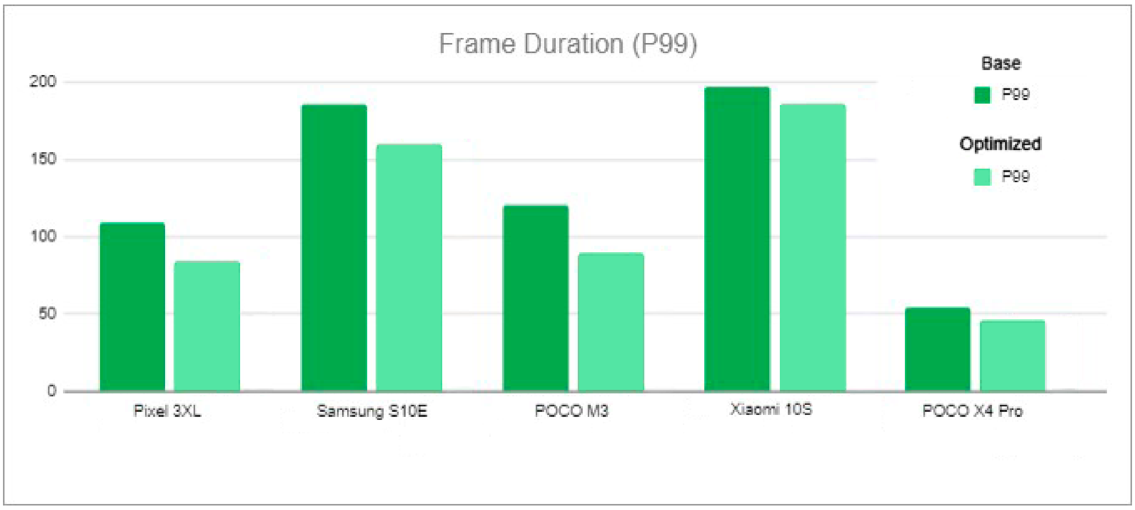


Figure 42. Frame Duration Results (P99)

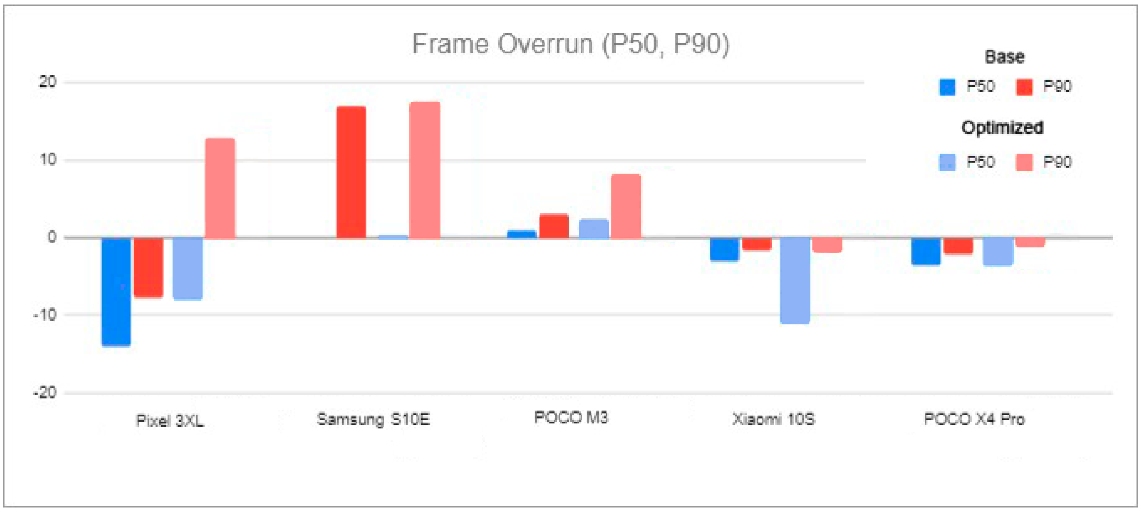


Figure 43. Frame Overrun Results (P50, P90)

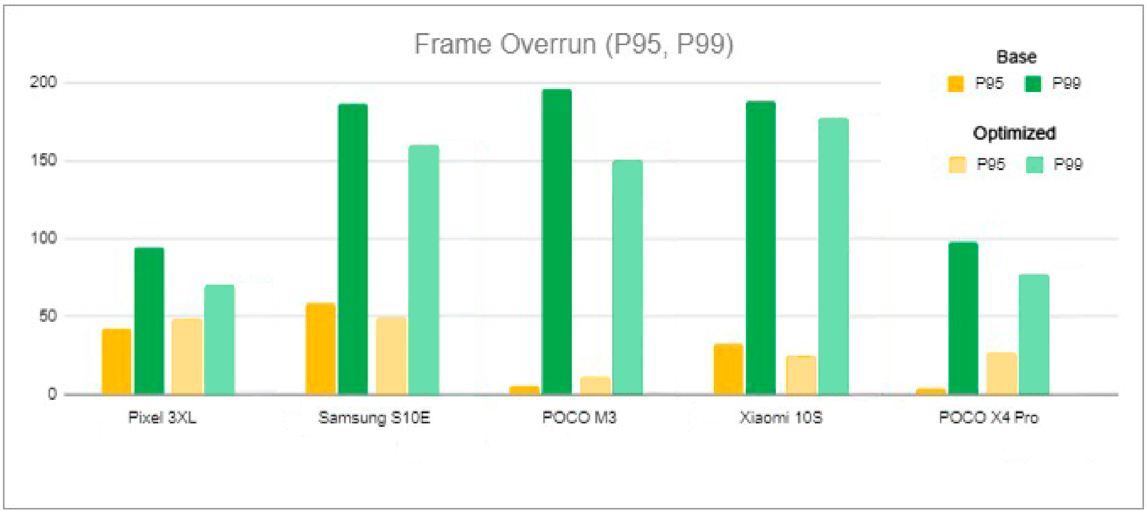


Figure 44. Frame Overrun Results (P95, P99)

Based on the results obtained from the Android Macrobenchmarking tests, we are pleased with the outcomes, as they demonstrated positive performance improvements for our app. However, we recognize that optimization is an ongoing process, and we remain committed to further enhancing the app's performance.

The constantly evolving nature of technology means that new libraries and frameworks are being developed every day, presenting exciting opportunities to leverage advancements and potentially boost Framedrop’s app performance even further. We acknowledge the importance of staying up-to-date with these developments and exploring their potential benefits for our application.

In the context of our academic environment, the results obtained from the Macrobenchmarking tests were encouraging. They provided valuable insights into the topic of app measurement and optimization, offering an excellent opportunity to deepen my understanding and knowledge in this field.

Chapter 8

Conclusion and Future Work

This dissertation presents the outcomes of the curricular internship conducted in collaboration with Pink Room and Framedrop. The internship focused on the development of an Android mobile application that offers gaming content for entertainment through short-form gaming clips. The project provided valuable insights and lessons learned that can be applied in future projects within this field.

Reflecting on the progress made during the first semester, initiating development early posed a significant challenge. However, it enabled me to be involved in the entire product development process from the beginning and acquire additional skills through an extended learning period. Balancing university coursework with research and development work proved to be one of the major hurdles. Juggling these responsibilities required considerable effort and dedication, since I had to ensure that I dedicated sufficient time and effort to each aspect of the project, striving to perform to the best of my abilities. This experience allowed me to compare the academic environment with real-world client expectations, equipping me with the skills to navigate rapid changes in project direction.

The second semester presented its own set of challenges, but it was comparatively less stressful due to the groundwork laid in the previous phase. Building upon the foundation of the developed application and leveraging enhanced mobile development experience positively influenced the internship's results. The client's requirements for a mobile application were successfully met, and the project had already entered the pre-production phase.

Completing a project within the given time frame without compromising on quality posed a challenge, as described by the "iron triangle" of project management. The iron triangle concept asserts that three interconnected factors—time, cost, and quality—must be balanced. Not compromising on the time frame may require increased costs or decreased quality to meet the deadline. Similarly, not compromising on the cost could extend the time frame or reduce the quality. The same applies to maintaining quality, which may necessitate extending the time frame or increasing the cost. By prioritizing and striking a balance among these three factors, we deemed the project a success. We achieved the project's scope within the allocated time frame while ensuring high quality and satisfying all stakeholders. Importantly, it should be noted that the success of the project was not contingent upon reinventing the wheel for developing features similar to existing applications. By studying and analyzing these applications, we were able to adopt optimized techniques that had been refined over time.

The development phase offered more than just coding skills. It provided an opportunity to learn about Quality Assurance, automated Testing, Usability Testing, and optimization mechanisms for Android development, but that, generally speaking, are Software Engineering concepts that can be applied in any Software project. Engaging in these aspects of the development process allowed me to grow as an engineer, as these are concepts, tools and processes that I will for sure come across again in the future.

Perhaps the biggest challenge I encountered during this project was my struggle to engage with technical articles and documentation. In the past, I had difficulty maintaining focus and motivation when reading technical materials, often delaying or avoiding them altogether. However, this project helped me overcome this obstacle and appreciate the value of staying updated with the latest advancements in mobile development. I recognized that in order to deliver a high-quality and functional application, it was crucial to be aware of the latest technologies and best practices in the field. Consequently, I consciously allocated time each day for reading and research, and to my surprise, I became more interested in the subject matter.

Given that certain requirements were not incorporated into this final application release, our forthcoming efforts will focus on assessing the market's interest in the product and establishing the subsequent set of requirements for implementation based on the obtained results. As a developer, my primary responsibility will revolve around incessantly enhancing the application and resolving any issues that are reported by end-users.

Concluding my internship, I would like to cover the importance of inspiration, because I truly believe that “the most important thing is to try and inspire people so that they can be great at whatever they want to do” - Kobe Bryant. And that is what this internship was to me, as it inspired me to be a better engineer and inspire others to be better too. Working within a company that upholds these values and consistently pushes me to transcend my capabilities on a daily basis, not only as an engineer but as a human being, was truly important to the success of this project and the perseverance of future work.

References

[1] "entertainment." Accessed: Oct. 02, 2022. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/entertainment

[2] "History of entertainment – HiSoUR – Hi So You Are." https://www.hisour.com/history-of-entertainment-35999/ (accessed Oct. 02, 2022).

[3] R. Digicult, "How the Internet has Helped Entertainment Evolve • Digicult | Digital Art, Design and Culture," *Digicult | Digital Art, Design and Culture*, Mar. 09, 2018. https://digicult.it/digimag/internet-helped-entertainment-evolve/ (accessed Oct. 02, 2022).

[4] "Internet and social media users in the world 2022," *Statista*. https://www.statista.com/statistics/617136/digital-population-worldwide/ (accessed Oct. 02, 2022).

[5] Sandvine, "Global Internet Phenomena." https://www.sandvine.com/phenomena (accessed Oct. 02, 2022).

[6] "Severe Tire Damage." https://www.std.org/text/live.html (accessed Oct. 02, 2022).

[7] "LiveVideoStack » An interview with Severe Tire Damage: The first live band on the Internet." https://www.livevideostack.cn/news/an-interview-with-severe-tire-damage-the-first-live-band-on-the-internet/ (accessed Oct. 02, 2022).

[8] P. Leskin, "YouTube is 15 years old. Here's a timeline of how YouTube was founded, its rise to video behemoth, and its biggest controversies along way," *Business Insider*. https://www.businessinsider.com/history-of-youtube-in-photos-2015-10 (accessed Oct. 02, 2022).

[9] T. Ruether, "History of Streaming Media [Infographic]," *Wowza*, Apr. 25, 2022. https://www.wowza.com/blog/history-of-streaming-media (accessed Oct. 02, 2022).

[10] L. J. Cabeza-Ramírez, S. M. Sánchez-Cañizares, F. J. Fuentes-García, and L. M. Santos-Roldán, "Exploring the connection between playing video games and watching video game streaming: Relationships with potential problematic uses," *Comput. Hum. Behav.*, vol. 128, p. 107130, Mar. 2022, doi: 10.1016/j.chb.2021.107130.

[11] "Digital 2022: Internet Connection Speeds Accelerate," *DataReportal – Global Digital Insights*. https://datareportal.com/reports/digital-2022-internet-connection-speeds (accessed Nov. 04, 2022).

[12] J. M. Kilner and R. N. Lemon, "What We Know Currently about Mirror Neurons," *Curr. Biol.*, vol. 23, no. 23, pp. R1057–R1062, Dec. 2013, doi: 10.1016/j.cub.2013.10.051.

[13] "How Observational Learning Affects Behavior," *Verywell Mind*. https://www.verywellmind.com/what-is-observational-learning-2795402 (accessed Jan. 04, 2023).

[14] C. Weightman, "Crucible: the science behind why watching others playing video games has become so popular," *The Conversation*. http://theconversation.com/crucible-the-science-behind-why-watching-others-playing-vid

eo-games-has-become-so-popular-139190 (accessed Nov. 08, 2022).

[15] M. E. Porter, *Competitive Strategy: Techniques for Analyzing Industries and Competitors*, First. New York, NY 10020: The Free Press, 1980.

[16] “What is the difference between competitors and substitutes? | - FintechAsia,” Feb. 05, 2022. https://fintechasia.net/2022/02/05/what-is-the-difference-between-competitors-and-substitutes/ (accessed Oct. 18, 2022).

[17] H. B. Review, “THE FIVE COMPETITIVE FORCES THAT by Michael E. Porter,” Jan. 2008.

[18] “What is the difference between competitors and substitutes? - Business Questions,” Oct. 09, 2022. https://businessqs.com/what-is-the-difference-between-competitors-and-substitutes/ (accessed Oct. 18, 2022).

[19] “Produto viável mínimo,” *Wikipédia, a enciclopédia livre*. Oct. 05, 2022. Accessed: Dec. 26, 2022. [Online]. Available: https://pt.wikipedia.org/w/index.php?title=Produto_vi%C3%A1vel_m%C3%ADnimo&oldid=64516352

[20] “Mobile Operating System Market Share Worldwide,” *StatCounter Global Stats*. https://gs.statcounter.com/os-market-share/mobile/worldwide (accessed Dec. 30, 2022).

[21] “Open Handset Alliance,” *Wikipedia*. Oct. 17, 2022. Accessed: Dec. 30, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Open_Handset_Alliance&oldid=1116652164

[22] “Apple App Store vs Google Play Store (2022 Comparison),” Jun. 29, 2021. https://cybercrew.uk/software/app-store-vs-play-store/ (accessed Dec. 30, 2022).

[23] “How Long Does It Take for an App Store to Approve Your App,” *Appy Pie*, Aug. 06, 2020. https://www.appypie.com/how-long-does-it-take-to-publish-apps (accessed Dec. 30, 2022).

[24] “What is Customer Development?,” *Agile Alliance* |, Jul. 15, 2017. https://www.agilealliance.org/glossary/customer-development/ (accessed Jan. 07, 2023).

[25] “How to Choose the Right Type of Mobile App: Native, Web or Hybrid,” *The Interaction Design Foundation*. https://www.interaction-design.org/literature/article/native-vs-hybrid-vs-responsive-what-app-flavour-is-best-for-you (accessed Jan. 05, 2023).

[26] A. Tapcrew, “User Experience in Native vs Hybrid App Development: | Tapcrew.” https://tapcrew.com/user-experience-in-native-vs-hybrid-app-development/ (accessed Dec. 30, 2022).

[27] “Kotlin for Android | Kotlin,” *Kotlin Help*. https://kotlinlang.org/docsandroid-overview.html (accessed Oct. 03, 2022).

[28] R. A. Martin, *Clean Architecture*, 1st edition (13 Sept. 2017). United States: Financial Times Prentice Hall.

[29] “Separation of concerns,” *Wikipedia*. Jul. 16, 2022. Accessed: Oct. 03, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Separation_of_concerns&oldid=1098544788

[30] sina.rahimi, “Single source of truth,” *Medium*, Jul. 20, 2019. https://medium.com/@sina.rahimi/single-source-of-truth-with-mvvm-retrofit2-livedata-rxjava-and-room-in-repository-pattern-f5304f39175 (accessed Oct. 03, 2022).

[31] “Unidirectional Data Flow,” *GeeksforGeeks*, Apr. 26, 2019. https://www.geeksforgeeks.org/unidirectional-data-flow/ (accessed Oct. 03, 2022).

[32] F. Cejas, “Architecting Android…The clean way?,” *Fernando Cejas*, Sep. 03, 2014. http://fernandocejas.com/2014/09/03/architecting-android-the-clean-way/ (accessed Jan. 06, 2023).

[33] “Guide to app architecture | Android Developers.” https://developer.android.com/topic/architecture (accessed Oct. 03, 2022).

[34] “MVC Architecture – What is a Model View Controller Framework?,” *freeCodeCamp.org*, Sep. 24, 2021. https://www.freecodecamp.org/news/mvc-architecture-what-is-a-model-view-controller-framework/ (accessed Oct. 04, 2022).

[35] M. Gago, “Development of a platform for promotion, reading and recommendation of short stories,” University of Coimbra, Coimbra, 2018.

[36] F. Muntenescu, “Android Architecture Patterns Part 1: Model-View-Controller,” *upday devs*, Nov. 01, 2016. https://medium.com/upday-devs/android-architecture-patterns-part-1-model-view-controller-3baecef5f2b6 (accessed Jan. 10, 2023).

[37] M. Phan, “MVP architectural pattern.” http://ducmanhphan.github.io/2019-08-05-MVP-architectural-pattern/ (accessed Oct. 04, 2022).

[38] R. Gazzah, “MVI Architecture With Android,” *The Startup*, Nov. 06, 2020. https://medium.com/swlh/mvi-architecture-with-android-fcde123e3c4a (accessed Oct. 06, 2022).

[39] “Model–view–viewmodel,” *Wikipedia*. Sep. 09, 2022. Accessed: Oct. 07, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Model%E2%80%93view%E2%80%93viewmodel&oldid=1109295015

[40] F. Muntenescu, “Android Architecture Patterns Part 3: Model-View-ViewModel,” *upday devs*, Nov. 04, 2016. https://medium.com/upday-devs/android-architecture-patterns-part-3-model-view-viewmodel-e7eeee76b73b (accessed Oct. 07, 2022).

[41] “Fundamentals of testing Android apps,” *Android Developers*. https://developer.android.com/training/testing/fundamentals (accessed Jan. 10, 2023).

[42] “Coupling in Java,” *GeeksforGeeks*, Nov. 09, 2017. https://www.geeksforgeeks.org/coupling-in-java/ (accessed Oct. 14, 2022).

[43] “Low coupling - Functional Kotlin [Book].” https://www.oreilly.com/library/view/functional-kotlin/9781788476485/401a0d5f-dcc7-4f4a-b343-ff9790fffe30.xhtml (accessed Oct. 14, 2022).

[44] “Cohesion in Java,” *GeeksforGeeks*, Oct. 31, 2017. https://www.geeksforgeeks.org/cohesion-in-java/ (accessed Oct. 14, 2022).

[45] M. Yiğit, “Say Hello to Jetpack Compose and Compare with XML,” *Medium*, Jan. 24, 2022. https://blog.kotlin-academy.com/say-hello-to-jetpack-compose-and-compare-with-xml-6bc6053aec13 (accessed Jan. 03, 2023).

[46] “Use Kotlin coroutines with lifecycle-aware components | Android Developers.”

https://developer.android.com/topic/libraries/architecture/coroutines (accessed Jan. 03, 2023).

[47] “Twitch API,” *Twitch Developers*, Nov. 29, 2022. https://dev.twitch.tv/api/ (accessed Dec. 01, 2022).

[48] “Using OIDC to get OAuth Access Tokens | Twitch Developers.” https://dev.twitch.tv/docs/authentication/getting-tokens-oidc (accessed Dec. 01, 2022).

[49] “OpenID Connect | OpenID,” Aug. 01, 2011. https://openid.net/connect/ (accessed Dec. 01, 2022).

[50] “OAuth 2.0 for Mobile & Desktop Apps | Authorization,” *Google Developers*. https://developers.google.com/identity/protocols/oauth2/native-app (accessed Jan. 04, 2023).

[51] “openid/AppAuth-Android: Android client SDK for communicating with OAuth 2.0 and OpenID Connect providers.” https://github.com/openid/AppAuth-Android (accessed Dec. 01, 2022).

[52] “ExoPlayer.” https://exoplayer.dev/ (accessed Nov. 21, 2022).

[53] “Android SDK 5 Player Features.” https://docs.nagra.com/doc/home/cpanddoc524x/android-sdk-5-player-features (accessed Nov. 21, 2022).

[54] “NexPlayer | The Premium Multiscreen Player SDK for Video Apps,” *NexPlayer*. https://nexplayersdk.com/ (accessed Nov. 23, 2022).

[55] M. Chayabanjonglerd, “Playing video by ExoPlayer,” *Medium*, Dec. 22, 2021. https://medium.com/fungjai/playing-video-by-exoplayer-b97903be0b33 (accessed Nov. 23, 2022).

[56] *ExoPlayer: Flexible media playback for Android (Google I/O ’17)*, (May 18, 2017). Accessed: Nov. 23, 2022. [Online Video]. Available: https://www.youtube.com/watch?v=jAZn-J1I8Eg

[57] “Precise Improvements: How TikTok Enhanced its Video Social Experience on Android,” *Android Developers Blog*. https://android-developers.googleblog.com/2022/08/precise-improvements-how-tiktok-enhanced-its-social-experience-on-android.html (accessed Jan. 03, 2023).

[58] “Surface,” *Android Developers*. https://developer.android.com/reference/android/view/Surface (accessed Jan. 03, 2023).

[59] D. R. B. Svensson and D. M. Staron, “Chalmers University of Technology University of Gothenburg”.

[60] “Continuous Integration,” *martinfowler.com*. https://martinfowler.com/articles/continuousIntegration.html (accessed Dec. 20, 2022).

[61] G. Booch, R. A. Maksimchuk, M. W. Engle, B. J. Young, J. Connallen, and K. A. Houston, “Object-oriented analysis and design with applications, third edition,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 33, no. 5, pp. 29–29, Aug. 2008, doi: 10.1145/1402521.1413138.

[62] J. Hall, “A brief history of CI/CD,” *Jonathan Hall*. https://jhall.io/archive/2021/09/26/a-brief-history-of-ci/cd/ (accessed Dec. 20, 2022).

[63] K. Beck, “Embracing change with extreme programming,” *Computer*, vol. 32, no. 10, pp. 70–77, Oct. 1999, doi: 10.1109/2.796139.

[64] “Jenkins,” *Jenkins*. https://www.jenkins.io/ (accessed Dec. 20, 2022).

[65] “GitLab CI/CD | GitLab.” https://docs.gitlab.com/ee/ci/ (accessed Dec. 20, 2022).

[66] “Bitrise | Mobile DevOps to Maximize App Impact.” https://bitrise.io (accessed Jan. 10, 2023).

[67] “Continuous Integration and Delivery,” *CircleCI*. https://circleci.com/ (accessed Dec. 20, 2022).

[68] “Understanding GitHub Actions,” *GitHub Docs*. https://ghdocs-prod.azurewebsites.net/en/actions/learn-github-actions/understanding-github-actions (accessed Dec. 20, 2022).

[69] “What is CI/CD?” https://www.redhat.com/en/topics/devops/what-is-ci-cd (accessed Dec. 20, 2022).

[70] “continuous-deployment,” Aug. 06, 2021. https://www.ibm.com/cloud/learn/continuous-deployment (accessed Dec. 20, 2022).

[71] “About Android App Bundles,” *Android Developers*. https://developer.android.com/guide/app-bundle (accessed Jan. 06, 2023).

[72] “Manifesto for Agile Software Development.” https://agilemanifesto.org/ (accessed Dec. 26, 2022).

[73] Atlassian, “Sprint Planning,” *Atlassian*. https://www.atlassian.com/agile/scrum/sprint-planning (accessed Dec. 27, 2022).

[74] “Story Points vs. Hours: The Relationship and the Difference | LinearB,” Sep. 21, 2021. https://linearb.io/blog/story-points-vs-hours/ (accessed Dec. 27, 2022).

[75] N. Pathak, “INNOVATION ROOTS,” *INNOVATION ROOTS*, Aug. 25, 2018. https://innoroo.com/blog/2018/08/25/interview-with-james-grenning/www.innoroo.com (accessed Dec. 27, 2022).

[76] M. Victor and N. Upadhyay, “Selection of Software Testing Technique: A Multi Criteria Decision Making Approach,” in *Trends in Computer Science, Engineering and Information Technology*, D. Nagamalai, E. Renault, and M. Dhanuskodi, Eds., in Communications in Computer and Information Science. Berlin, Heidelberg: Springer, 2011, pp. 453–462. doi: 10.1007/978-3-642-24043-0_46.

[77] -, “Threshold of Success.” https://www.neverletdown.net/2010/01/threshold-of-success.html (accessed Dec. 14, 2022).

[78] “Steve Blank Perfection By Subtraction – The Minimum Feature Set,” *Steve Blank*, Mar. 04, 2010. https://steveblank.com/2010/03/04/perfection-by-subtraction-the-minimum-feature-set/ (accessed Dec. 28, 2022).

[79] E. Belinski, “Android API Levels.” https://apilevels.com/ (accessed Jan. 03, 2023).

[80] J. Nielsen, *Usability Engineering*. Morgan Kaufmann Publishers Inc.340 Pine Street, Sixth FloorSan FranciscoCAUnited States, 1994.

[81] J. Knapp, *How to Solve Big Problems and Test New Ideas in Just Five Days*, First edition March 2016. Simon & Schuster Audio; Unabridged edition.

[82] “The Design Sprint — GV.” http://www.gv.com/sprint (accessed Oct. 10, 2022).

[83] “C4 Model, Architecture Viewpoint and Archi 4.7 – Archi.” https://www.archimatetool.com/blog/2020/04/18/c4-model-architecture-viewpoint-and-archi-4-7/ (accessed Nov. 02, 2022).

[84] M. Weidmann, “The four C’s of software architecture,” *devlix Blog*, Feb. 01, 2021. https://medium.com/devlix-blog/the-four-cs-of-software-architecture-58a784bdb19

(accessed Nov. 02, 2022).

[85] “The C4 model for visualising software architecture.” https://c4model.com/ (accessed Nov. 02, 2022).

[86] “P of EAA: Repository.” https://martinfowler.com/eaaCatalog/repository.html (accessed Jan. 03, 2023).

[87] “Why 5 is the magic number for UX usability testing | Inside Design Blog.” https://www.invisionapp.com/inside-design/ux-usability-research-testing/ (accessed May 31, 2023).

[88] J. Nielsen, “Why You Only Need to Test with 5 Users,” *Nielsen Norman Group*, Apr. 18, 2000. https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/ (accessed May 31, 2023).

[89] “Benchmark your app | App quality,” *Android Developers*. https://developer.android.com/topic/performance/benchmarking/benchmarking-overview (accessed Jun. 05, 2023).

[90] “Baseline Profiles | App quality,” *Android Developers*. https://developer.android.com/topic/performance/baselineprofiles/overview (accessed May 31, 2023).

[91] “UI jank detection | Android Studio,” *Android Developers*. https://developer.android.com/studio/profile/jank-detection (accessed Jun. 05, 2023).

[92] “Capture Macrobenchmark metrics | App quality,” *Android Developers*. https://developer.android.com/topic/performance/benchmarking/macrobenchmark-metrics (accessed Jun. 05, 2023).

[93] vikas kumar, “The 95th and 99th percentiles are the most crucial application metrics,” *Medium*, Apr. 04, 2023. https://medium.com/@vikaskumar4793/the-95th-and-99th-percentiles-are-the-most-crucial-application-metrics-33085d2d3e34 (accessed Jun. 30, 2023).

[94] Medal [@medal_tv], “In Case You Missed It (ICYMI) Powered by AI, Medal can now automatically create clips and bookmarks as soon as key moments happen in your game. This new option can be found in Settings > Clips or search Enable ICYMI in the Search bar! https://t.co/LJxcSu8BiE,” *Twitter*, Jul. 29, 2020. https://twitter.com/medal_tv/status/1288584748818980865 (accessed Jan. 04, 2023).

[95] “Record and Clip PC and Mobile Games | Medal.tv,” *Medal.tv | #1 Free Clip Platform*. https://medal.tv/features/clipping (accessed Oct. 19, 2022).

[96] “The Best Gaming Video Editor - Ever | Medal.tv.” https://medal.tv/features/editing (accessed Oct. 19, 2022).

[97] “Outplayed,” *Overwolf | Tech for developers who love gaming*. https://www.overwolf.com/app/Overwolf-Outplayed (accessed Oct. 19, 2022).

[98] T. 29 posts https://thefranswah-studio com Owner and O. of T. Studio, “Hover.gg - The TikTok For Streamers,” *TheFranswah Studio*, Feb. 21, 2022. https://thefranswah-studio.com/hover-gg-the-tiktok-for-streamers/ (accessed Oct. 19, 2022).

[99] “What Was Youtube Gaming and Why Was it Discontinued?” https://www.failory.com/google/youtube-gaming (accessed Oct. 03, 2022).

[100] “Top 6 Twitch Competitors - Digiaide.com,” Oct. 05, 2021. https://digiaide.com/twitch-competitors/ (accessed Oct. 03, 2022).

[101] W. Geyser, “Top Facebook Gaming Stats Every Brand Should Know in 2022,”

Influencer Marketing Hub, Jul. 02, 2020. https://influencermarketinghub.com/facebook-gaming-stats/ (accessed Oct. 03, 2022).

[102] “20 Essential TikTok Statistics You Need to Know in 2022,” *The Social Shepherd*. https://thesocialshepherd.com/blog/tiktok-statistics (accessed Nov. 08, 2022).

[103] “Most downloaded apps worldwide 2021,” *Statista*. https://www.statista.com/statistics/1285960/top-downloaded-mobile-apps-worldwide/ (accessed Nov. 08, 2022).

[104] “TikTok,” *Business Model Toolbox*. https://bmtoolbox.net/stories/tiktok/ (accessed Nov. 08, 2022).

[105] “TikTok Categories Grabbing the Most Views,” *IZEA*, Feb. 14, 2022. https://izea.com/resources/tiktok-categories-grabbing-the-most-views/ (accessed Nov. 08, 2022).

[106] *Caffeine.TV: The Twitch Competitor Backed By Drake | Forbes*, (Feb. 26, 2020). Accessed: Jan. 04, 2023. [Online Video]. Available: https://www.youtube.com/watch?v=dVnhUdpAIw4

[107] “Caffeine Review,” *PCMAG*. https://www.pcmag.com/reviews/caffeine (accessed Oct. 03, 2022).

[108] “caffeine.tv vs twitch.tv Traffic Comparison,” *Similarweb*. https://www.similarweb.com/website/caffeine.tv/vs/twitch.tv/ (accessed Oct. 03, 2022).

[109] D. Chmielewski, “Behind Streaming Upstart Caffeine’s Plan To Take On Twitch: Draft Drake,” *Forbes*. https://www.forbes.com/sites/dawnchmielewski/2020/02/11/behind-streaming-upstart-caffeines-plan-to-take-on-twitch-draft-drake/ (accessed Oct. 03, 2022).

[110] “The Design Sprint.” https://www.thesprintbook.com/the-design-sprint (accessed Oct. 10, 2022).

[111] “Jake Knapp’s Design Sprint template | Miroverse.” https://miro.com/miroverse/design-sprint-jake-knapp/ (accessed Oct. 10, 2022).

[112] “DAU/MAU Ratio.” https://www.klipfolio.com/metrics/saas/dau-mau-ratio (accessed Oct. 10, 2022).

[113] N. Eyal, *Hooked: how to build habit-forming products*. 375 Hudson Street: Penguin Group.

[114] “Product Design Sprint Process, Methods, Tools and Templates: A Complete Guide to Running a Design Sprint.” https://www.netguru.com/blog/product-design-sprint-methods-tools-templates (accessed Oct. 12, 2022)

Appendixes

Appendix A - Market Analysis

Market Competitors

Medal[20]

For Windows, Mac, IOS, Android

It is a digital tool with over 7 million users which lets them share clips of themselves playing video games and join in video chats. Users can select games that interest them from a menu, as well as which players they want to follow and chat with. It is a convenient way for like-minded gamers to socialize, share, and talk about their best gaming moments. Medal has its own AI for capturing user highlights titled “In Case You Missed It” [94] (ICYMI) that stores all the clips gathered in the user's profile so he can edit and share them [95]. Medal’s game clips editor has a complete suite of easy-to-use and customizable filters, stickers, GIFs, and sound effects to add to the footage [96].

It is free to use, but it has another paid plan, “Medal Premium” which increases upload clip size, removes watermark and credits, among other bragging rights features.

Outplayed[21]

For Windows

Outplayed is a video capturing app that tracks and records the best moments, but can be manually recorded on demand. Users can edit the recorded clips using the platform’s editor and share it across social networks. Up to the date of this thesis, it supports over 400 games [97]. It is free to use.

[20] Medal - https://medal.tv/

[21] Outplayed - https://outplayed.tv/

Hover[22]

For Web, IOS, Android

Hover allows sharing clips and videos on the platform with emphasis and focus on gamers. It integrates directly with Twitch allowing users to share any clip from the user's Twitch channel from the last 2 weeks. Viewers can navigate directly to a streamer’s Twitch profile by pressing the Twitch button when viewing a clip in Hover. The platform has its own clip cloud making it easy to manage PC and phone clips. Hover still integrates directly with Xbox and Twitter, and indirectly with PlayStation and Switch accounts, because these are pulled by linking Twitter to Hover [98]. It is free to use.

Powder[23]

For Windows, IOS, Android

Powder provides a highlight tracker for PC and mobile devices that gathers the best moments from users' gaming sessions, from 27 different games (up to the date of this thesis). Powder’s editor allows gamers to turn their game clips into shareable content and submit them for challenges in the app and get rewarded for it. As a social network, it allows users to scroll through the highlights of other Powder creators, discover games, and join their communities. It is free to use, and allows users to get rewarded for winning challenges.

Feature Comparison

From a brief description of product competitors, it is unquestionable the similarities between them, since they aim for the same goals. In Table 1, the list of the features supported by each product is presented. Most of these products rely on a strong connection between their mobile and web applications, and since Framedrop also has its own, it is interesting to list the features of the web application as well.

[22] Hover - https://hover.gg/app/discover

[23] Powder - https://powder.gg/

Game Recording on PC	✓	✓	✗	✓
Game Recording on Mobile	✓	✗	✗	✓
AI highlight tracker	✓	✓	✗	✓
Camera overlay in live-recording	✓	✗	✗	✗
Record only game sounds (discard notification of third-party apps)	✓	✓	✗	✓
Built-in clip editor	✓	✗	✓	✓
Trim or montage clips	✓	✗	✓	✓
Add effects/music to clips	✓	✗	✗	✓
Tag friends in captured videos	✓	✗	✗	✗
Cloud for storing captured clips	✓	✓	✓	✓
Directly share embed link for social networks	✓	✗	✗	✓
Watch clips	✓	✗	✓	✓
Like/comment/share/follow user	✓	✗	✓	✓
Download clip	✓	✗	✗	✗

Authenticate with Username and Password	✓	✗	✓	✓
Authenticate with Discord	✓	✗	✗	✗
Authenticate with TikTok	✓	✗	✗	✗
Authenticate with Twitch	✗	✗	✗	✓
Authenticate with Steam	✓	✗	✗	✗
Authenticate with Twitter	✓	✗	✗	✗
Authenticate with Riot Games	✓	✗	✗	✗
Authenticate with Facebook	✓	✗	✗	✗
Authenticate with Apple ID	✓	✗	✗	✓
Authenticate with Xbox	✓	✗	✗	✗
Check other user activities	✓	✗	✗	✗
Follow clips from a specific game	✓	✗	✓	✗
Built-in chat function	✓	✗	✗	✗
Search for users, games and tags	✓	✗	✓	✓
Watch videos in full size with mobile in landscape mode	✓	✗	✗	✓

Redirect to Twitch profile directly from clip (no need to go tho his profile)	✗	✗	✓	✗
Upload a clip	✓	✗	✓	✓
Automatically import Twitch clips	✗	✗	✓	✗
Information if the user is live on Twitch in real time	✗	✗	✓	✗
See recommended clips	✓	✗	✓	✗
See followed user clips	✓	✗	✓	✗
See clips from users that are live on Twitch at the moment	✗	✗	✓	✗
Receive more profile exposure on the app by interacting with it	✗	✗	✓	✗
Edit profile information	✓	✗	✓	✓
Challenges for best clips of a specific category	✓	✗	✓	✓
System of rewards/badges for the user, by engaging with the platform	✓	✗	✗	✓
Block user	✓	✗	✓	✓
Report Clip	✓	✗	✓	✓
Reformat clip size to appear streamer face and clip in portrait mode	✗	✗	✓	✗

Table 1. Product competitors feature comparison

Market Substitutes

Youtube/Youtube Gaming[24]

For Web, Windows, Mac, IOS, Android

Youtube Gaming was a separate YouTube app aimed at the gaming community. The app also had popular features such as the Super Chat, dedicated Game Pages, and even Channel Membership [99]. It was launched in 2015, and its main goal was to provide exclusive game-related videos. Due to YouTube main app popularity, numbers showed that YouTube gaming was not doing so well, causing the app to be discontinued in 2019. Luckily, when merged into the main app as “Gaming Vertical”, the app became popular again. This new version contained all the features of the old one such as the Super Chat, dedicated Game Pages, and even Channel Membership [100].

Facebook Gaming[25]

For Web, IOS, Android

Taking advantage of Twitch popularity, Facebook Gaming was launched in 2018, getting its own tab in Facebook's original app. The popular platform accounts for 3% of the total hours watched in live streaming, and this equals to 356 million streaming hours that have been watched by viewers [100]. In 2018 and 2019, Facebook gaming was far behind its present competitors, Twitch and YouTube Gaming. Contrary to expectations of its cadence in the livestreaming business, according to Streamlabs, 2021 saw the social media giant blow, with 1.06 billion gaming hours [101], taking its direct competitor, YouTube Gaming with 1.37 billion gaming hours. This platform is getting an important share of the market, as the

[24] Youtube - https://www.youtube.com/

[25] Facebook Gaming - https://www.facebook.com/gaming

time goes by. This is happening due to the acquisition of big influencers and streamers, coming from other platforms and because of "Gaming's Talent Acquisitions" which attracted users that wanted to improve their game skills.

TikTok[26]

For Web, IOS, Android

TikTok is a social media app dedicated to short-form (15 to 60 seconds) content creation and consumption. In September 2021, the app reached one billion active users [102], and according to Statistica[27], it was the most mobile downloaded app in 2021 with 656 million downloads [103].

TikTok's mission is to inspire users creativity, leading to a mobile short-form content entertainment. With their artificial intelligence technology, users receive desired content without the need to explicitly search for that. The Revenue model of the app is through in-app gift purchases. Users can exchange real money for "TikTokens" and use them to show their interest for a specific creator [104].

The company does not operate in a specific area of entertainment, as the application offers a variety of categories, such as dance, fitness, fashion, market and so on [105].

Caffeine.tv[28]

For Web, IOS, Android

"It is a more refined version of the application Twitch" [106]. Launched in 2016, with a core objective to refine real time chat between the streamer and the viewers. The company did not focus primarily on gaming livestream, embracing other varieties of categories in its product, which did not "quite match those that just stick to video games" [107].

Although Twitch had opened live streaming to other categories as well, they still provided good quality to gaming. Caffeine offers low quality resolution for gaming, and it is only compatible with about 900 PC games (as of June 18, 2020) [107]. Nowadays, Caffeine.tv is not a threat to Twitch, since in August 2022, they had 337.7 thousand visits

[26] TikTok - https://www.tiktok.com/
[27] Statistica - https://www.statista.com/
[28] Caffeine.tv - https://www.caffeine.tv/

against 1.3 billion visits on Twitch [108], but Keighran's vision for the future of entertainment, which he calls "social broadcasting", attracted a lot of investors and triggered a joint venture, Caffeine Studios, where the core is to produce esports, sports and live entertainment from the Fox studio lot in Los Angeles [109].

Feature Comparison

As it was done for the competing products, the Table 2 shows the features that each substitute product integrates in its constitution.

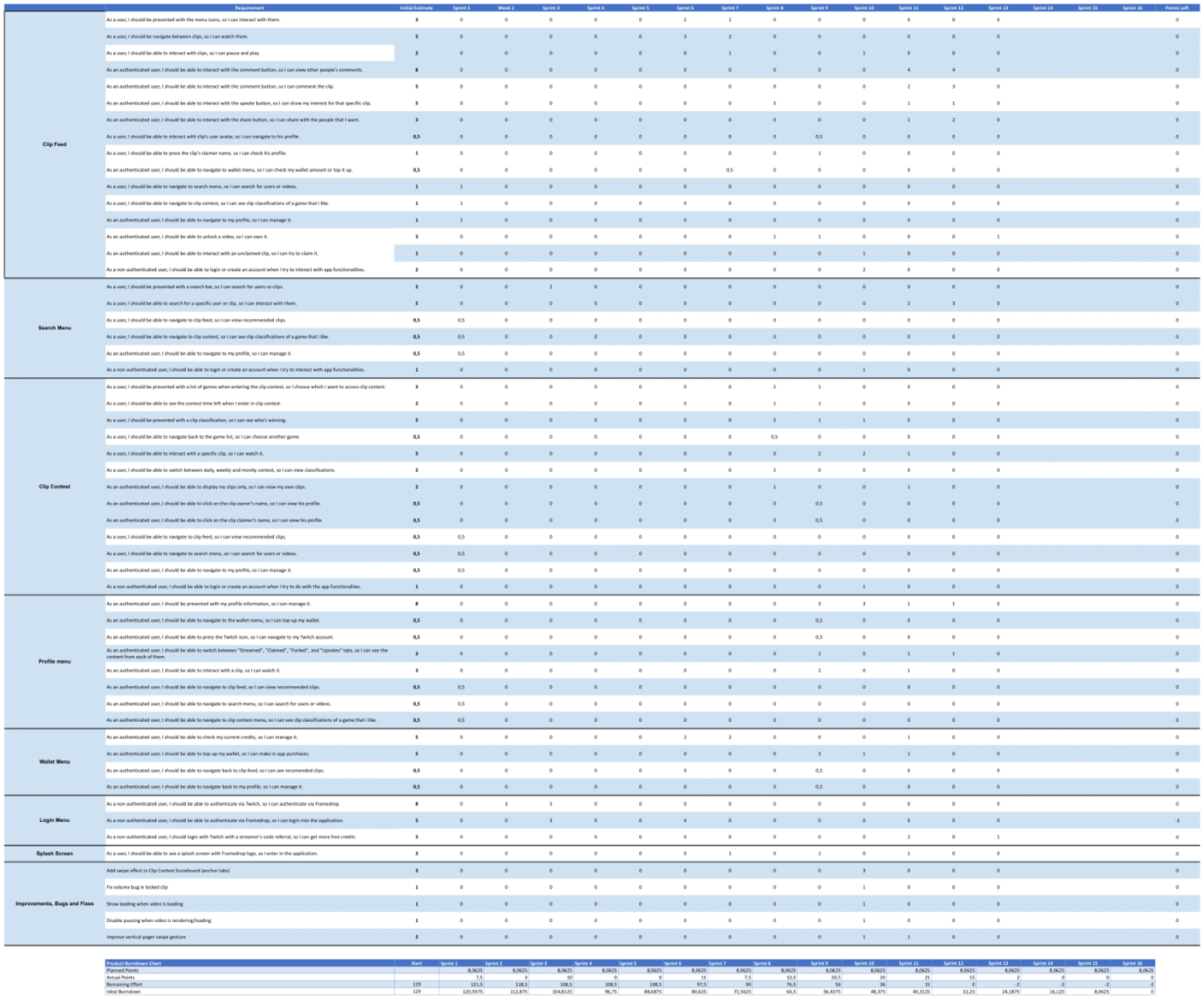
	YouTube		TikTok	
Game Recording on PC	✗	✗	✗	✗
Game Recording on Mobile	✗	✗	✗	✗
AI highlight tracker	✗	✗	✗	✗
Camera overlay in live-recording	✗	✗	✗	✗
Record only game sounds (discard notification of third-party apps)	✓	✓	✗	✓
Built-in clip editor	✓	✗	✓	✗
Trim or montage clips	✓	✗	✓	✗
Add effects/music to clips	✓	✗	✓	✗
Tag friends in captured videos	✓	✓	✓	✓
Cloud for storing captured clips	✓	✗	✗	✗
Directly share embed link for social networks	✓	✓	✓	✓

Watch clips	✓	✓	✓	✓
Like/comment/share/follow user	✓	✓	✓	✓
Download clip	✓	✗	✓	✗
Authenticate with Username and Password	✓	✓	✓	✓
Authenticate with Discord	✗	✗	✗	✗
Authenticate with TikTok	✗	✗	✗	✗
Authenticate with Twitch	✗	✗	✗	✗
Authenticate with Steam	✗	✗	✗	✗
Authenticate with Twitter	✗	✗	✓	✓
Authenticate with Riot Games	✗	✗	✗	✗
Authenticate with Facebook	✗	✗	✓	✓
Authenticate with Apple ID	✗	✗	✓	✗
Authenticate with Xbox	✗	✗	✗	✗
Check other user activities	✗	✗	✗	✗
Follow clips from a specific game	✗	✓	✗	✗
Built-in chat function	✗	✓	✓	✗
Search for users, games and tags	✓	✓	✓	✓
Watch videos in full size with mobile in landscape mode	✓	✓	✗	✓
Redirect to Twitch profile directly from clip (no need to	✗	✗	✗	✗

go tho his profile)				
Upload a clip	✓	✓	✓	✓
Automatically import Twitch clips	✗	✗	✗	✗
Information if the user is live on Twitch in real time	✗	✗	✗	✗
See recommended clips	✓	✓	✓	✓
See followed user clips	✓	✓	✓	✓
See clips from users that are live on Twitch at the moment	✗	✗	✗	✗
Receive more profile exposure on the app by interacting with it	✗	✗	✗	✗
Edit profile information	✓	✓	✓	✓
Challenges for best clips of a specific category	✗	✗	✗	✗
System of rewards/badges for the user, by engaging with the platform	✗	✗	✗	✗
Block user	✓	✓	✓	✓
Report Clip	✓	✓	✓	✓
Reformat clip size to appear streamer face and clip in portrait mode	✗	✗	✗	✗

Table 2. Product substitutes feature comparison

Appendix B - Product Burn



	Requirement	Initial Estimate	Sprint 1	Week 2	Sprint 3	Sprint 4	Sprint 5	Sprint 6	Sprint 7	Sprint 8	Sprint 9	Sprint 10	Sprint 11	Sprint 12	Sprint 13	Sprint 14	Sprint 15	Sprint 16	Points Left
Clip Feed	As a user, I should be presented with the menu icons, so I can interact with them.	3	0	0	0	0	0	2	1	0	0	0	0	0	0				0
	As a user, I should be navigate between clips, so I can watch them.	5	0	0	0	0	0	3	2	0	0	0	0	0	0				0
	As a user, I should be able to interact with clips, so I can pause and play.	2	0	0	0	0	0	0	1	0	0	1	0	0	0				0
	As an authenticated user, I should be able to interact with the comment button, so I can view other people's comments.	8	0	0	0	0	0	0	0	0	0	0	4	4	0				0
	As an authenticated user, I should be able to interact with the comment button, so I can comment the clip.	5	0	0	0	0	0	0	0	0	0	0	2	3	0				0
	As an authenticated user, I should be able to interact with the upvote button, so I can show my interest for that specific clip.	5	0	0	0	0	0	0	0	3	0	0	1	1	0				0
	As an authenticated user, I should be able to interact with the share button, so I can share with the people that I want.	3	0	0	0	0	0	0	0	0	0	0	1	2	0				0
	As a user, I should be able to interact with clip's user avatar, so I can navigate to his profile.	0,5	0	0	0	0	0	0	0	0	0,5	0	0	0	0				0
	As a user, I should be able to press the clip's claimer name, so I can check his profile.	1	0	0	0	0	0	0	0	0	1	0	0	0	0				0
	As an authenticated user, I should be able to navigate to wallet menu, so I can check my wallet amount or top it up.	0,5	0	0	0	0	0	0	0,5	0	0	0	0	0	0				0
	As a user, I should be able to navigate to search menu, so I can search for users or videos.	1	1	0	0	0	0	0	0	0	0	0	0	0	0				0
	As a user, I should be able to navigate to clip contest, so I can see clip classifications of a game that I like.	1	1	0	0	0	0	0	0	0	0	0	0	0	0				0
	As an authenticated user, I should be able to navigate to my profile, so I can manage it.	1	1	0	0	0	0	0	0	0	0	0	0	0	0				0
	As an authenticated user, I should be able to unlock a video, so I can own it.	3	0	0	0	0	0	0	0	1	1	0	0	0	1				0
	As an authenticated user, I should be able to interact with an unclaimed clip, so I can try to claim it.	1	0	0	0	0	0	0	0	0	0	1	0	0	0				0
	As a non-authenticated user, I should be able to login or create an account when I try to interact with app functionalities.	2	0	0	0	0	0	0	0	0	0	2	0	0	0				0
Search Menu	As a user, I should be presented with a search bar, so I can search for users or clips.	2	0	0	2	0	0	0	0	0	0	0	0	0	0				0
	As a user, I should be able to search for a specific user or clip, so I can interact with them.	5	0	0	0	0	0	0	0	0	0	0	2	3	0				0
	As a user, I should be able to navigate to clip feed, so I can view recommended clips.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As a user, I should be able to navigate to clip contest, so I can see clip classifications of a game that I like.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As an authenticated user, I should be able to navigate to my profile, so I can manage it.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As a non-authenticated user, I should be able to login or create an account when I try to interact with app functionalities.	1	0	0	0	0	0	0	0	0	0	1	0	0	0				0
Clip Contest	As a user, I should be presented with a list of games when entering the clip contest, so I choose which I want to access clip contest.	3	0	0	0	0	0	0	0	2	1	0	0	0	0				0
	As a user, I should be able to see the contest time left when I enter in clip contest.	2	0	0	0	0	0	0	0	1	1	0	0	0	0				0
	As a user, I should be presented with a clip classification, so I can see who's winning.	5	0	0	0	0	0	0	0	3	1	1	0	0	0				0
	As a user, I should be able to navigate back to the game list, so I can choose another game.	0,5	0	0	0	0	0	0	0	0,5	0	0	0	0	0				0
	As a user, I should be able to interact with a specific clip, so I can watch it.	5	0	0	0	0	0	0	0	0	2	2	1	0	0				0
	As a user, I should be able to switch between daily, weekly and montly contest, so I can view classifications.	2	0	0	0	0	0	0	0	2	0	0	0	0	0				0
	As an authenticated user, I should be able to display my clips only, so I can view my own clips.	2	0	0	0	0	0	0	0	1	0	0	1	0	0				0
	As an authenticated user, I should be able to click on the clip owner's name, so I can view his profile.	0,5	0	0	0	0	0	0	0	0	0,5	0	0	0	0				0
	As an authenticated user, I should be able to click on the clip claimer's name, so I can view his profile.	0,5	0	0	0	0	0	0	0	0	0,5	0	0	0	0				0
	As a user, I should be able to navigate to clip feed, so I can view recommended clips.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As a user, I should be able to navigate to search menu, so I can search for users or videos.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As an authenticated user, I should be able to navigate to my profile, so I can manage it.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As a non-authenticated user, I should be able to login or create an account when I try to do with the app functionalites.	1	0	0	0	0	0	0	0	0	0	1	0	0	0				0
Profile menu	As an authenticated user, I should be presented with my profile information, so I can manage it.	8	0	0	0	0	0	0	0	0	3	3	1	1	0				0
	As an authenticated user, I should be able to navigate to the wallet menu, so I can top-up my wallet.	0,5	0	0	0	0	0	0	0	0	0,5	0	0	0	0				0
	As an authenticated user, I should be able to press the Twitch icon, so I can navigate to my Twitch account.	0,5	0	0	0	0	0	0	0	0	0,5	0	0	0	0				0
	As an authenticated user, I should be able to switch beteewn "Streamed", "Claimed", "Forked", and "Upvotes" tabs, so I can see the content from each of them.	3	0	0	0	0	0	0	0	0	1	0	1	1	0				0
	As an authenticated user, I should be able to interact with a clip, so I can watch it.	3	0	0	0	0	0	0	0	0	2	0	1	0	0				0
	As an authenticated user, I should be able to navigate to clip feed, so I can view recommended clips.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As an authenticated user, I should be able to navigate to search menu, so I can search for users or videos.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
	As an authenticated user, I should be able to navigate to clip contest menu, so I can see clip classifications of a game that I like.	0,5	0,5	0	0	0	0	0	0	0	0	0	0	0	0				0
Wallet Menu	As an authenticated user, I should be able to check my current credits, so I can manage it.	5	0	0	0	0	0	2	2	0	0	0	1	0	0				0
	As an authenticated user, I should be able to top-up my wallet, so I can make in-app purchases.	5	0	0	0	0	0	0	0	0	3	1	1	0	0				0
	As an authenticated user, I should be able to navigate back to clip feed, so I can see recomended clips.	0,5	0	0	0	0	0	0	0	0	0,5	0	0	0	0				0
	As an authenticated user, I should be able to navigate back to my profile, so I can manage it.	0,5	0	0	0	0	0	0	0	0	0,5	0	0	0	0				0
Login Menu	As a non-authenticated user, I should be able to authenticate via Twitch, so I can authenticate via Framedrop.	8	0	3	5	0	0	0	0	0	0	0	0	0	0				0
	As a non-authenticated user, I should be able to authenticate via Framedrop, so I can login into the application.	5	0	0	3	0	0	4	0	0	0	0	0	0	0				-2
	As a non-authenticated user, I should login with Twitch with a streamer's code referral, so I can get more free credits	3	0	0	0	0	0	0	0	0	0	0	2	0	1				0
Splash Screen	As a user, I should be able to see a splash screen with Framedrop logo, as I enter in the application.	3	0	0	0	0	0	0	1	0	1	0	1	0	0				0
Improvements, Bugs and Fixes	Add swipe effect in Clip Contest Scoreboard (anchor tabs)	3	0	0	0	0	0	0	0	0	0	3	0	0	0				0
	Fix volume bug in locked clip	1	0	0	0	0	0	0	0	0	0	1	0	0	0				0
	Show loading when video is loading	1	0	0	0	0	0	0	0	0	0	1	0	0	0				0
	Disable pausing when video is rendering/loading	1	0	0	0	0	0	0	0	0	0	1	0	0	0				0
	Improve vertical pager swipe gesture	2	0	0	0	0	0	0	0	0	0	1	1	0	0				0

Product Burndown Chart	Start	Sprint 1	Sprint 2	Sprint 3	Sprint 4	Sprint 5	Sprint 6	Sprint 7	Sprint 8	Sprint 9	Sprint 10	Sprint 11	Sprint 12	Sprint 13	Sprint 14	Sprint 15	Sprint 16
Planned Points		8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625	8,0625
Actual Points		7,5	3	10	0	0	11	7,5	13,5	20,5	20	21	15	2	0	0	0
Remaining Effort	129	121,5	118,5	108,5	108,5	108,5	97,5	90	76,5	56	36	15	0	-2	-2	-2	-2
Ideal Burndown	129	120,9375	112,875	104,8125	96,75	88,6875	80,625	72,5625	64,5	56,4375	48,375	40,3125	32,25	24,1875	16,125	8,0625	0

Appendix C - Mitigation Plan

“Framedrop’s API server is not developed at the time of app development; might delay the development of the mobile application taking longer than was originally expected because some requirements are dependent on server end-points.”

1. Identify the specific requirements that are dependent on the server end-points, and prioritize them in terms of their impact on the overall development of the mobile application.

2. Develop a contingency plan for each of the identified requirements, in order to minimize their impact on the development timeline. For example, this could involve identifying alternative solutions or temporary workarounds that can be implemented until the API server is ready.

3. Work closely with the team responsible for developing the API server to ensure that they are aware of the dependencies on their work and to establish a clear timeline for the development and testing of the server.

4. Regularly review the progress of the API server development and adjust the development plan for the mobile application accordingly, in order to ensure that the overall timeline remains on track.

5. Regularly communicate with all relevant stakeholders, including project managers, developers, and other key personnel, to ensure that everyone is aware of the potential risks and the steps being taken to mitigate them.

“Twitch blocks Framedrop’s dependency to its services; might compromise all the application purpose.”

1. Identify and evaluate alternative services or platforms that could be used in place of Twitch, in order to maintain the functionality of the application. This could involve researching and comparing the features and capabilities of different platforms, as well as seeking input from key stakeholders and users of the application.

2. Develop a contingency plan for implementing the use of an alternative service or platform, including any necessary changes to the application's code or design. This should include steps to ensure a smooth transition from Twitch to the new platform, with minimal disruption to the application's functionality or user experience.

3. Work closely with the team responsible for implementing the use of the alternative platform, to ensure that they are aware of the potential risks and the steps being taken to mitigate them. This should include regular communication and collaboration to ensure that the transition is carried out smoothly and efficiently.

4. Regularly review the progress of the transition to the alternative platform, and adjust the plan as necessary in order to ensure that the application remains functional and successful. This could involve conducting user testing and gathering feedback to identify any potential issues or concerns, and taking steps to address them.

“Framedrop’s design presents flaws; might delay the development of the mobile application taking longer than was originally expected.”

1. Identify the specific design flaws that are causing or have the potential to cause delays in the development process. This could involve conducting a thorough review of the design, seeking input from key stakeholders and developers, and identifying any potential issues or areas of concern.

2. Develop a contingency plan for addressing the identified design flaws, in order to minimize their impact on the overall development timeline. This could involve

implementing alternative solutions or redesigning certain elements of the application in order to resolve the issues.

3. Work closely with the design team to ensure that they are aware of the identified flaws and the steps being taken to address them. This should include regular communication and collaboration to ensure that the design is updated and refined as necessary.

4. Regularly review the progress of the design updates and the development process, and adjust the plan as necessary in order to ensure that the overall timeline remains on track. This could involve conducting user testing and gathering feedback to identify any potential issues or concerns, and taking steps to address them.

"The developer has low knowledge about the development language used in the project; might delay the development time and/or compromise architectural or quality requirements."

1. Identify the specific areas of the project where the developer's lack of knowledge is causing or has the potential to cause delays or other issues. This could involve conducting a thorough review of the developer's work and seeking input from other team members and stakeholders.

2. Develop a plan for providing the developer with the necessary training and support to improve their knowledge and skills in the relevant development language. This could involve providing access to online resources, assigning a mentor or experienced team member to provide guidance, or enrolling the developer in a relevant training course or program.

3. Monitor the developer's progress and provide ongoing support and guidance as necessary, in order to ensure that they are able to effectively contribute to the project.

This should include regular communication and feedback to identify any areas where additional support or training may be needed.

4. Regularly review the overall progress of the project and the developer's contributions, and adjust the plan as necessary in order to ensure that the project remains on track and meets all architectural and quality requirements. This could involve conducting regular code reviews, user testing, and other forms of quality assurance to identify and address any potential issues.

“Developed code is difficult to test; might compromise quality attributes.”

1. Identify the specific areas of the code that are difficult to test, and the reasons why they are causing issues. This could involve conducting a thorough review of the code, seeking input from other team members and stakeholders, and identifying any potential challenges or obstacles to testing.

2. Develop a plan for addressing the identified challenges to testing, in order to ensure that the code can be effectively tested and that the quality attributes are maintained. This could involve implementing changes to the code itself, such as refactoring or restructuring, or developing additional tools or processes to support testing.

3. Work closely with the development and testing teams to ensure that they are aware of the challenges to testing and the steps being taken to address them. This should include regular communication and collaboration to ensure that the code is updated and tested as necessary.

4. Regularly review the overall progress of the project, including the testing of the code, and adjust the plan as necessary in order to ensure that the project remains on track and meets all quality requirements. This could involve conducting regular code reviews, user testing, and other forms of quality assurance to identify and address any potential issues.

“Framedrop’s project is dependent on the client availability; might delay the development of the mobile application.”

1. Identify the specific areas of the project where the client's availability is necessary, and the reasons why their availability is critical to the project's success. This could involve conducting a thorough review of the project plan and timeline, and seeking input from other team members and stakeholders.

2. Develop a contingency plan for addressing potential delays or other issues that may arise due to the client's availability. This could involve identifying alternative solutions or workarounds, or establishing clear communication and coordination processes to ensure that the project remains on track even if the client is not immediately available.

3. Work closely with the client to ensure that they are aware of the project's dependencies on their availability, and to establish a clear plan for communication and coordination. This should include regular communication and collaboration to ensure that the project remains on track and that any potential issues are addressed in a timely manner.

4. Regularly review the overall progress of the project, including the client's availability, and adjust the plan as necessary in order to ensure that the project remains on track and successful. This could involve conducting regular project reviews and status updates, and taking steps to address any potential delays or other issues that arise.

Appendix D - Requirements

Login Menu

In this menu, users can authenticate via their Twitch account to access the application. At this stage of development, there are no other ways to authenticate to the application.

The user stories are the following:

us_lm1	**As a non-authenticated user, I should be able to authenticate via Twitch, so I can authenticate via Framedrop.**
Priority	Must Have
Estimation	8
Dependency	-
Notes	-
Wireframe Reference	-

us_lm2	**As a non-authenticated user, I should be able to authenticate via Framedrop, so I can login into the application.**
Priority	Must Have
Estimation	5
Dependency	-
Notes	-
Wireframe Reference	-

Clip Feed Menu

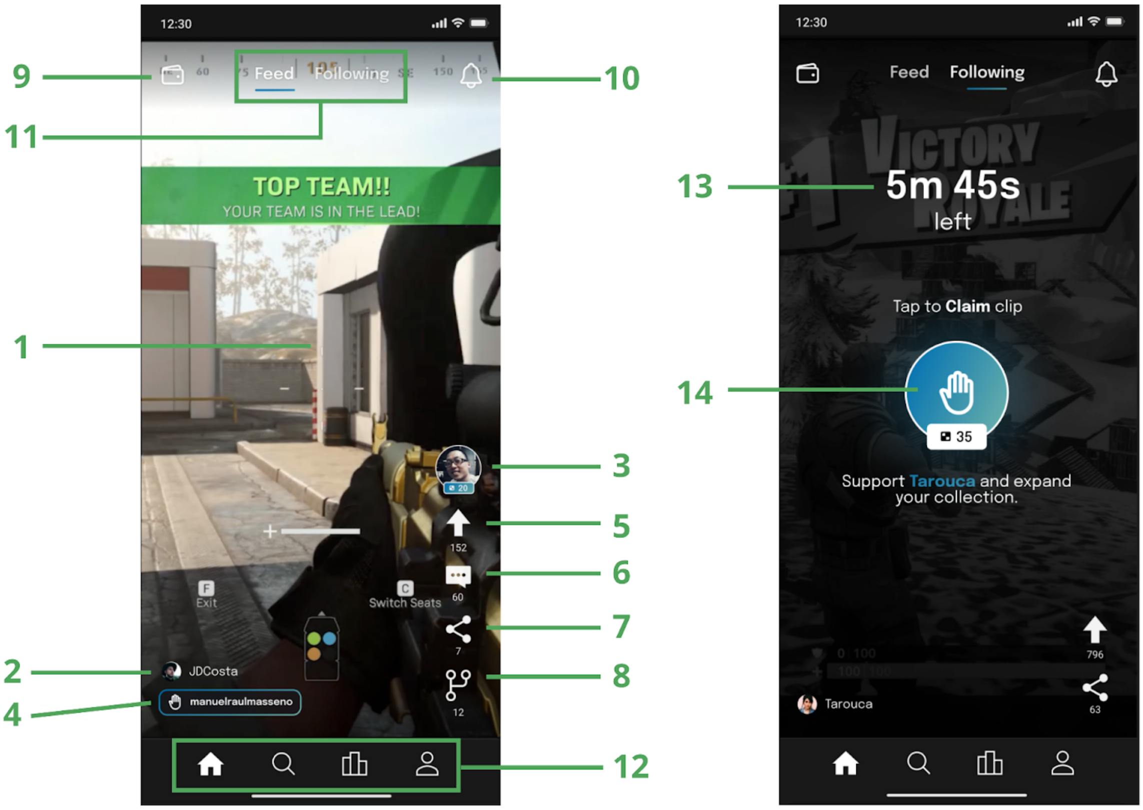


Figure 1. Clip Feed Menu

This menu represents the main menu of the application where users can consume gaming content by scrolling down the screen (1). For every clip being watched, it is displayed its creator (2), the owner of that clip (3) and the person who claimed it (4). The users can upvote (5), comment (6), share (7) and fork the clip (8). They can navigate to the wallet menu (9) and notification center (10) or change the tab from recommended clips feed to followed users clips (11) or vice-versa. Like in every menu, there is a bottom navigation bar, to navigate to the other menus.

The next tables represent the user stories for this menu:

us_cf1	**As an authenticated and a non-authenticated user, I should be presented with a clip when entering the clip feed, so I can watch it.**

Priority	Must Have
Estimation	3
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf2	**As a user, I should be able to navigate between clips, so I can watch them.**
Priority	Must Have
Estimation	5
Dependency	us_cf1
Notes	-
Wireframe Reference	Clip Feed

us_cf3	**As a user, I should be able to consume clips, so I can be entertained.**
Priority	Must Have
Estimation	5
Dependency	us_cf1
Notes	-
Wireframe Reference	Clip Feed

us_cf4	**As an authenticated user, I should be able to interact with the comment button, so I can view other people's comments.**
Priority	Must Have

Estimation	8
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf5	**As an authenticated user, I should be able to interact with the comment button, so I can comment on the clip.**
Priority	Must Have
Estimation	5
Dependency	us_cf4
Notes	-
Wireframe Reference	Clip Feed

us_cf6	**As an authenticated user, I should be able to interact with a comment button, so I can like other people's comments.**
Priority	Could Have
Estimation	3
Dependency	us_cf4
Notes	-
Wireframe Reference	Clip Feed

us_cf7	**As an authenticated user, I should be able to interact with the upvote button, so I can show my interest for that specific clip.**
Priority	Must Have
Estimation	5

Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf8	**As an authenticated user, I should be able to interact with the share button, so I can share with the people that I want.**
Priority	Should Have
Estimation	3
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf9	**As an authenticated user, I should be able to interact with the fork button, so I can have a personalized clip copy done by me on my profile.**
Priority	Could Have
Estimation	2
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf10	**As a user, I should be able to interact with the clip's user avatar, so I can navigate to his profile.**
Priority	Must Have
Estimation	1

Dependency	us_cf1
Notes	-
Wireframe Reference	Clip Feed

us_cf11	**As a user, I should be able to press the clip's claimer name, so I can check his profile.**
Priority	Must Have
Estimation	1
Dependency	us_cf1
Notes	-
Wireframe Reference	Clip Feed

us_cf12	**As an authenticated user, I should be able to navigate to the notification center, so I can check who interacted with me.**
Priority	Could Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf13	**As an authenticated user, I should be able to navigate to the "Following" tab, so I can only see clips from followed users.**
Priority	Must Have
Estimation	1
Dependency	-

Notes	-
Wireframe Reference	Clip Feed

us_cf14	**As an authenticated user, I should be able to navigate to the "Feed" tab, so I can see all recommended clips from all users.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf15	**As an authenticated user, I should be able to navigate to the wallet menu, so I can check my wallet amount or top it up.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf16	**As a user, I should be able to navigate to the search menu, so I can search for users or videos.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-

Wireframe Reference	Clip Feed

us_cf17	**As a user, I should be able to navigate to clip contest, so I can see clip classifications of a game that I like.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf18	**As an authenticated user, I should be able to navigate to my profile, so I can manage it.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf19	**As an authenticated user, I should be able to interact with an unclaimed clip, so I can try to claim it.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-

Wireframe Reference	Clip Feed

us_cf20	**As an authenticated user, I should be able to see the auction time left and last bid made on an unsigned clip, so I can manage my bid.**
Priority	Must Have
Estimation	5
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

us_cf21	**As a non-authenticated user, I should be able to login or create an account when I try to interact with app functionalities.**
Priority	Must Have
Estimation	2
Dependency	-
Notes	-
Wireframe Reference	Clip Feed

Search Menu

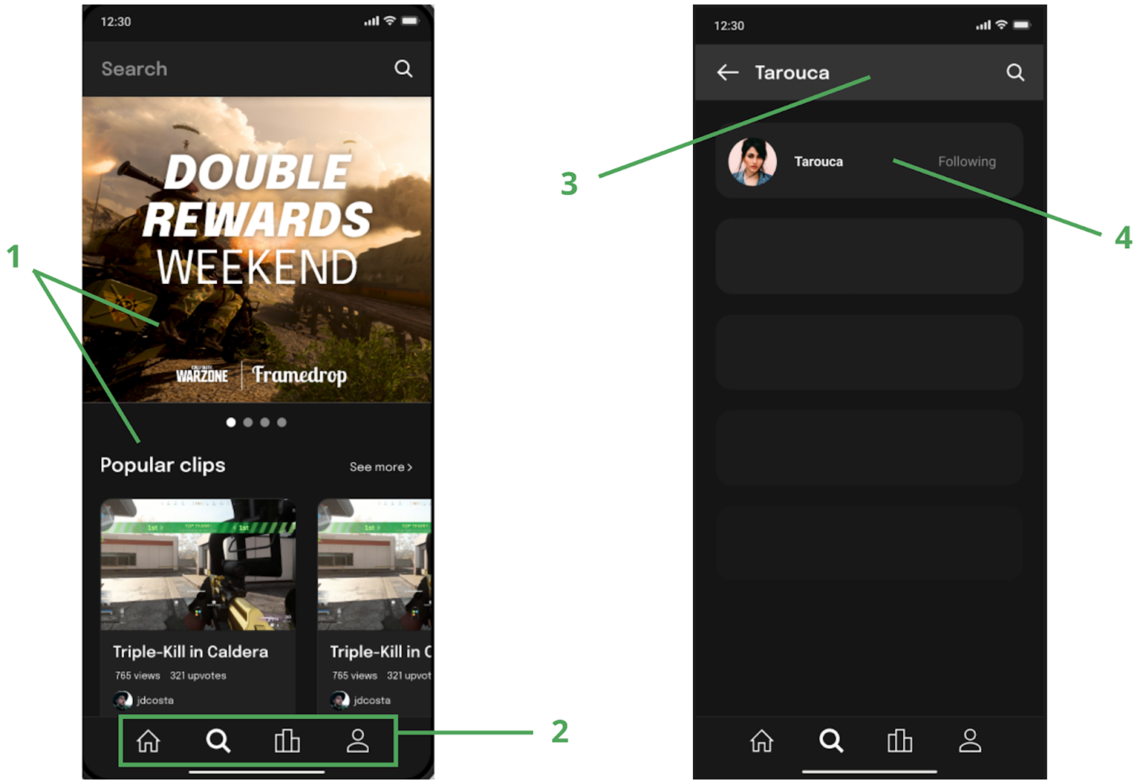


Figure 2. Search Menu

The Search Menu was designed with the goal of providing users a way to search for specific users or clips (3 and 4), as well as discover new games, watch popular clips, streamers "on the rise", among others (1).

The user stories for this menu are described above:

us_sm1	**As a user, I should be presented with a search bar, so I can search for users or clips.**
Priority	Must Have
Estimation	2
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm2	**As a user, I should be presented with popular clips, so I can explore it.**
Priority	Could Have
Estimation	3
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm3	**As a user, I should be presented with popular games, so I can explore it.**
Priority	Could Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm4	**As a user, I should be presented with recommended clips, so I can explore it.**
Priority	Could Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm5	**As a user, I should be presented with streamers on the rise, so I can explore them.**
Priority	Could Have
Estimation	2
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm6	**As a user, I should be presented with clips waiting to be unlocked, so I can try to claim them.**
Priority	Could Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm7	**As a user, I should be able to search for a specific user or clip, so I can interact with them.**
Priority	Could Have
Estimation	5
Dependency	us_sm1
Notes	-
Wireframe Reference	Search Menu

us_sm8	**As a user, I should be able to navigate to clip feed, so I can view recommended clips.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm9	**As a user, I should be able to navigate to clip contest, so I can see clip classifications of a game that I like.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm10	**As an authenticated user, I should be able to navigate to my profile, so I can manage it.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Search Menu

us_sm11	**As a non-authenticated user, I should be able to login or create an account when I try to interact with app functionalities.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Search Menu

Clip Contest Menu

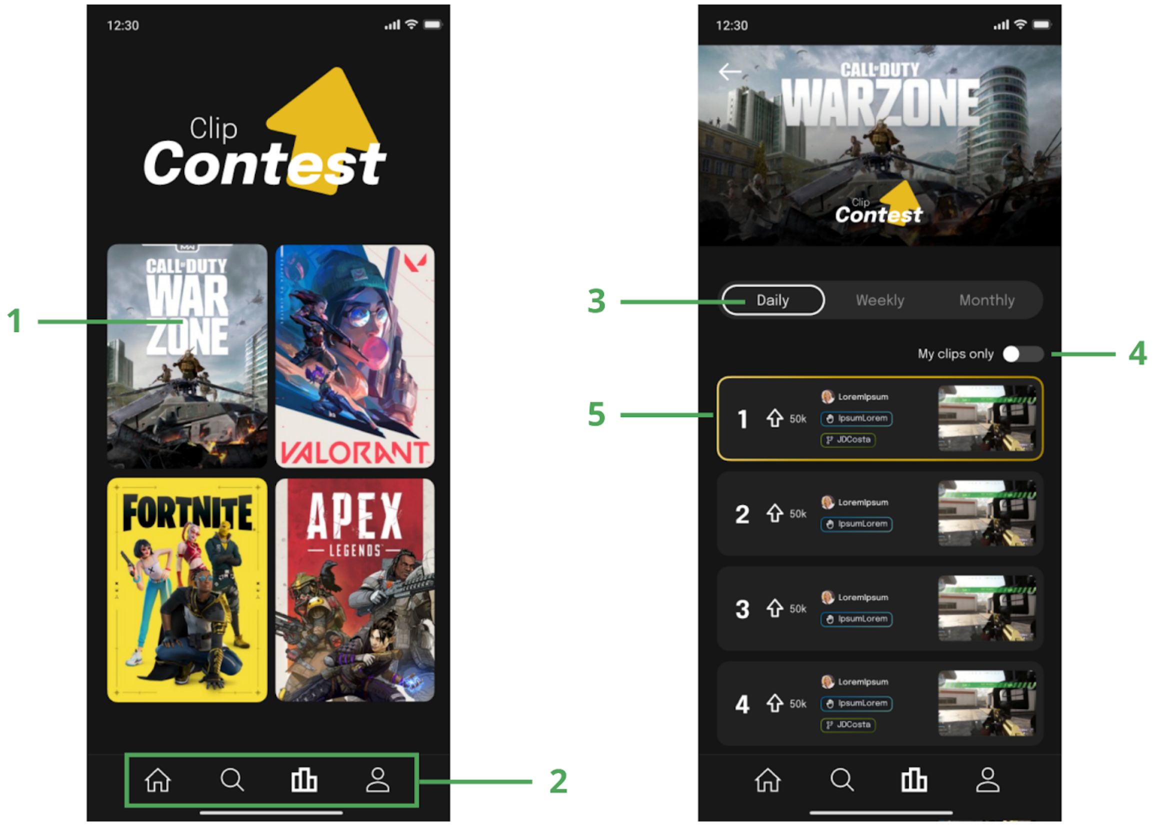


Figure 3. Clip Contest Menu

In the Clip Contest Menu, users are presented with a first screen to choose between a variety of games, to access their contest (1). When in the contest screen, they can switch between "daily", "weekly" or "monthly" classifications (3), and interact with the scoreboard, by watching the clip or navigating to users profiles (5), or they may want to check only their clips (4).

The user stories are the following ones:

us_cc1	**As a user, I should be presented with a list of games when entering the clip contest, so I choose which I want to access clip contest.**
Priority	Must Have
Estimation	3
Dependency	-
Notes	-
Wireframe Reference	Clip Contest Menu

us_cc2	**As a user, I should be presented with a clip classification, so I can see who's winning.**
Priority	Must Have
Estimation	5
Dependency	us_cc1
Notes	-
Wireframe Reference	Clip Contest Menu

us_cc3	**As a user, I should be able to navigate back to the game list, so I can choose another game.**
Priority	Must Have
Estimation	1

Dependency	us_cc1
Notes	-
Wireframe Reference	Clip Contest Menu

us_cc4	**As a user, I should be able to interact with a specific clip, so I can watch it.**
Priority	Must Have
Estimation	5
Dependency	us_cc2
Notes	-
Wireframe Reference	Clip Contest Menu

us_cc5	**As a user, I should be able to switch between daily, weekly and monthly contest, so I can view classifications.**
Priority	Must Have
Estimation	2
Dependency	us_cc1
Notes	-
Wireframe Reference	Clip Contest Menu

us_cc6	**As an authenticated user, I should be able to display my clips only, so I can view my own clips.**
Priority	Must Have
Estimation	2
Dependency	us_cc1

Notes	-
Wireframe Reference	Clip Contest Menu

us_cc7	As a user, I should be able to navigate to clip feed, so I can view recommended clips.
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Contest Menu

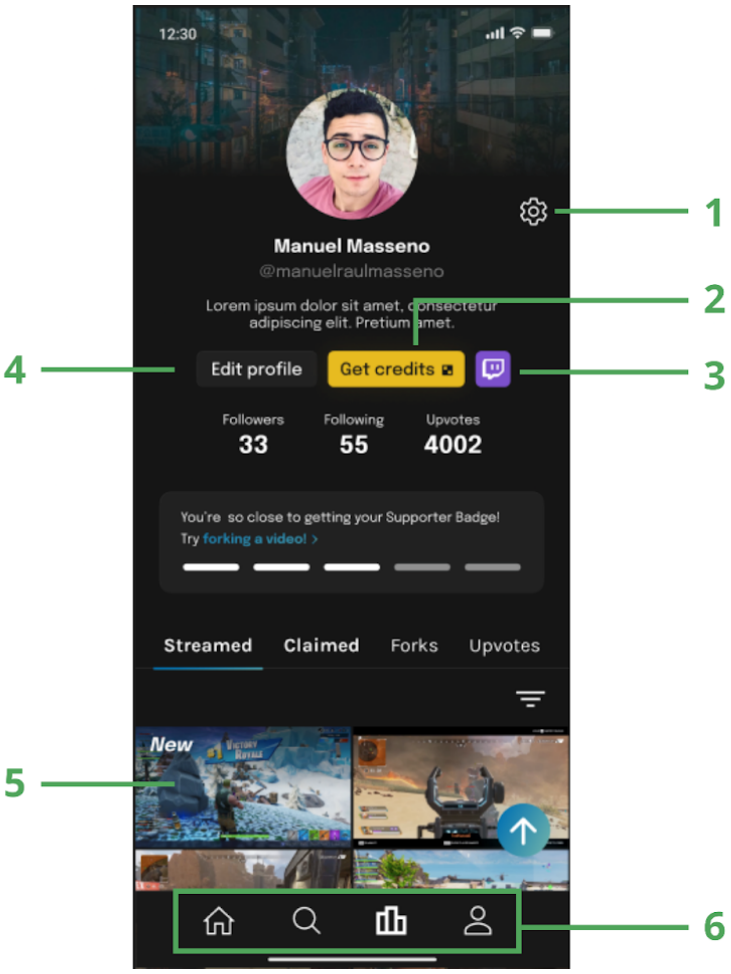
us_cc8	As a user, I should be able to navigate to search menu, so I can search for users or videos.
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Contest Menu

us_cc9	As an authenticated user, I should be able to navigate to my profile, so I can manage it.
Priority	Must Have
Estimation	1
Dependency	-
Notes	-

Wireframe Reference	Clip Contest Menu

us_cc10	**As a non-authenticated user, I should be able to login or create an account when I try to do with the app functionalities.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Clip Contest Menu

Profile Menu



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pretium amet.

You're so close to getting your Supporter Badge! Try forking a video! >

Figure 4. Profile Menu

This menu represents the user profile and, without some elements (1, 2 and 4), it represents the profile menu of a third user (not the owner of that account). In their own profile, users can edit it (1 and 2), navigate to the wallet menu to get more credits (2), navigate to their Twitch profile (3) and watch their own clips (5). They may also filter their clips by “Streamed”, “Claimed”, “Forks” and “Upvotes”. These four categories will present different content (5).

Above is a list of the user stories:

us_pm1	**As an authenticated user, I should be presented with my profile information, so I can manage it.**

Priority	Must Have
Estimation	8
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm2	**As an authenticated user, I should be able to navigate to the settings menu, so I can edit my experience on the application.**
Priority	Could Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm3	**As an authenticated user, I should be able to press to navigate to my edit profile menu, so I can edit my profile information.**
Priority	Could Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm4	**As an authenticated user, I should be able to navigate to the wallet menu, so I can top-up my wallet.**
Priority	Could Have

Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm5	**As an authenticated user, I should be able to press the Twitch icon, so I can navigate to my Twitch account.**
Priority	Could Have
Estimation	2
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm6	**As an authenticated user, I should be able to switch between "Streamed", "Claimed", "Forked", and "Upvotes" tabs, so I can see the content from each of them.**
Priority	Could Have
Estimation	3
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm7	**As an authenticated user, I should be able to interact with a clip, so I can watch it.**
Priority	Must Have

Estimation	5
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

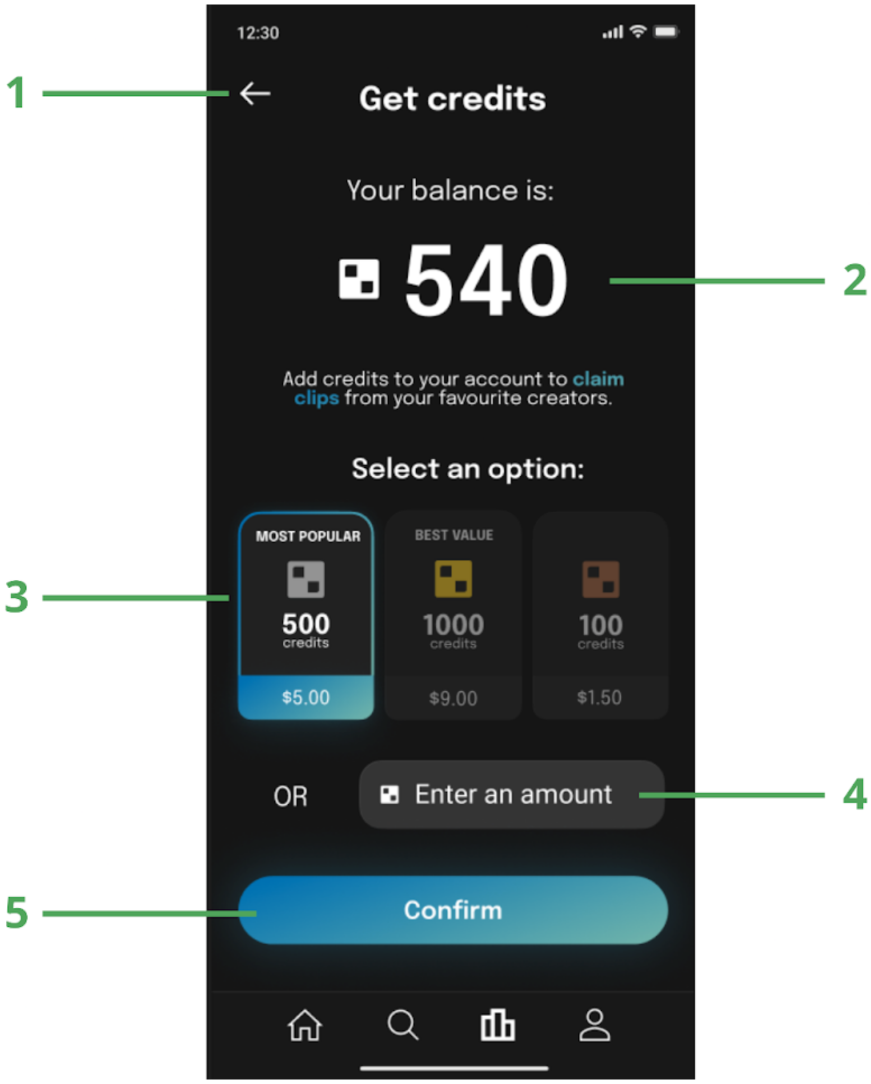
us_pm8	**As an authenticated user, I should be able to navigate to clip feed, so I can view recommended clips.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm9	**As an authenticated user, I should be able to navigate to the search menu, so I can search for users or videos.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Profile Menu

us_pm10	**As an authenticated user, I should be able to navigate to the clip contest menu, so I can see clip classifications of a game that I like.**
Priority	Must Have
Estimation	1

Dependency	-
Notes	-
Wireframe Reference	Profile Menu

Wallet Menu



Get credits

Your balance is:

540

Add credits to your account to claim clips from your favourite creators.

Select an option:

Figure 5. Wallet Menu

To top up the in-app wallet, users can select one of the standard amount options (3) or enter a custom value on their own (4), and confirm the transaction (5). Users can also check their balance in this menu (1).

The list of user stories is the following:

us_wm1	**As an authenticated user, I should be able to check my current credits, so I can manage it.**
Priority	Must Have
Estimation	5
Dependency	-
Notes	-
Wireframe Reference	Wallet Menu

us_wm2	**As an authenticated user, I should be able to top-up my wallet, so I can make in-app purchases.**
Priority	Must Have
Estimation	20
Dependency	-
Notes	This user story has an estimated time over the maximum defined. This means that it still needs to be decomposed in sub-requirements, but we still don’t have any information about the payment option to define the process.
Wireframe Reference	Wallet Menu

us_wm3	**As an authenticated user, I should be able to navigate back to clip feed, so I can see recommended clips.**
Priority	Must Have

Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Wallet Menu

us_wm4	**As an authenticated user, I should be able to navigate back to my profile, so I can manage it.**
Priority	Must Have
Estimation	1
Dependency	-
Notes	-
Wireframe Reference	Wallet Menu

Clip Auction Menu

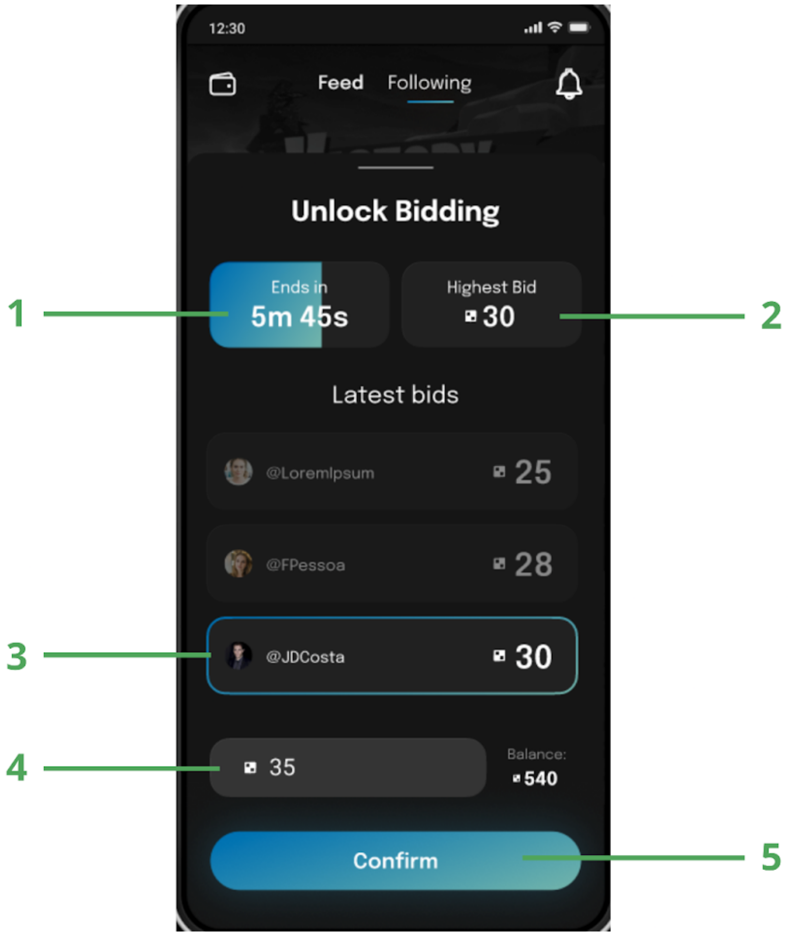


Figure 6. Clip Auction Menu

When trying to claim an unsigned clip, users are presented with an auction menu, where they can check the remaining auction time, and the highest bid at that time, as well as the latest bidders. They can also place a bid by entering a value and confirming the transaction.

Appendix E - Product Design Sprint

1. Introduction

The purpose of this document is to present an overview of the Product Design Sprint of Framedrop’s mobile application. It focuses on a brief explanation of what the company provides as a service and expectations regarding the mobile application, the tools and methodologies used to perform it as well as the artifacts developed.

2. Sprint Duration

The original document states that the period to complete the Product Design Sprint is 5 days, divided by 5 stages: Map, Sketch, Decide, Prototype and Test [110].

In this case, the process was led by one of Framedrop's summer interns, Manuel Masseno, and due to the proximity of the end of the summer internship, the Product Design Sprint was reduced to two days, leaving Prototyping and Testing out of scope.

3. First Day

The plan for the first day was to define a long-term goal, metrics that could measure if the long-term goal was achieved and the risks that could prevent from achieving it. By the end of the day, we mapped all the interactions between customers and the product. For doing this, all the potential users were taken into account as well as the different forms of discovering and interacting with the product. All the information gathered before would help, after a round of interviews with experts on the subject, to identify, on the map, the core problem to be solved, transforming it into an opportunity.

3.1. Introduction

As the first day of the Product Design Sprint, it was necessary to define the team's roles and responsibilities. The two most important roles were Sprint Leader and Decider, given to Framedrop’s summer intern Manuel Masseno and company CEO João Diogo Costa, respectively. The Sprint Leader was responsible to conduct the sprint, control time and facilitate the decisions of the group. The Decider took the responsibility to make all key decisions and on the occasions where the group was splitted between choices.

Define a long-term goal

As the name implies, the objective of this task was to agree on the project's 12-18 months mission [111], assuming that everything went perfectly. The group ideas were likely similar, focusing on providing entertainment for the viewers and revenue for the content providers. The most voted idea was the one that we concluded that was more focused on the product sustainability, which was "100% running an app with a stable and growing user base and some big streamers using it actively". With this long-term goal, it would be possible to continue revenuing streamers and providing content to the viewers.

Define metrics to measure if the long-term was achieved

To perform this task, each person drafted two metrics, assuming once again that everything went perfectly. Since the long-term goal was focusing on product maintainability, the metrics followed the same path, with enhancement on the number of viewers and streamers per day or active on the platform.

After a voting round, there were two metrics that the group considered the most important:

- DUA/MAU ratio is over 20% - which means Daily Active Users to Monthly Active Users ratio, and represents how active users are on a daily basis. This metric presents the number of days that the user was engaged with the service sufficient time to consider him an active user [112]. "A higher DAU/MAU Ratio generally indicates high stickiness, meaning users consistently return to the app" [112].
- 10% of the streamers with over 1000 concurrent viewers are using it - having a good base of big streamers is important and represents a strong relation of users with the service.

Define risks

Assuming a worst case scenario, the group tried to figure out what might prevent from reaching the long-term goal and metrics defined on the previous exercises. Using a "yes or no" approach, the questions came down to streamer and viewer engagement, whether the platform's content would be addictive enough (like other existing ones, e.g. TikTok) and whether streamers would trust their image with Framedrop.

After a round of debating and improving the presented questions, we came down to two risks, regarding the possibility of designing a “super engaging app” and the possibility of competing for attention with other market competitors such Youtube. Launching a product in a market with big competitors could be a challenge, especially when trying to break users' old engagement routines and make them interact with our product. Even if they interact with the app, there’s another challenge that we need to face: retain users and make them engage with the app for as long as it takes for it to become a habit [113].

Map the product

In the original framework, the exercise “Asking Experts” would come before mapping the product, having more information to make the Map [111]. In our case, we preferred to follow the book’s author preference and use the Map as a reference for the conversations, because there was already an initial idea of the application screen flow.

The initial steps to define the Map were the following:

- Define customers;
- Define the discovery path, which means the moments where the customer learns and starts using the app;
- Define the core experience, which represents the flow of actions that the customer can perform through the app.

Customers could be different actors with different intentions of engaging with the app. The most high-level customers would be streamers and fans. From them we can define other actors like:

- Casual Viewer - it doesn’t have the particular fanaticism for a streamer;
- Creator fan - it has a particular fanaticism for a streamer or a group of streamers;
- Creator supporter - like the name implies, aims to support creator’s content;
- Trader - is a user that want to unlock and sell clips to top up his wallet;
- Montage Editor - edits is own clips on top of streamers clips;
- Staff - streamer’s staff that maintains the streamer’s profile.

After the analysis of the different end-users, it was necessary to define the various channels where they could discover Framedrop and start engaging with it. This exposure could be done from three approaches: Streamers word-of-mouth, brand publicity and brand exposure by reaching out to customers. Streamers would provide the majority of exposure, because they have a niche of end-users following them and have a variety of tools to communicate with them like social networks, Discord community and during streaming. The next interactions are presented in the image below.

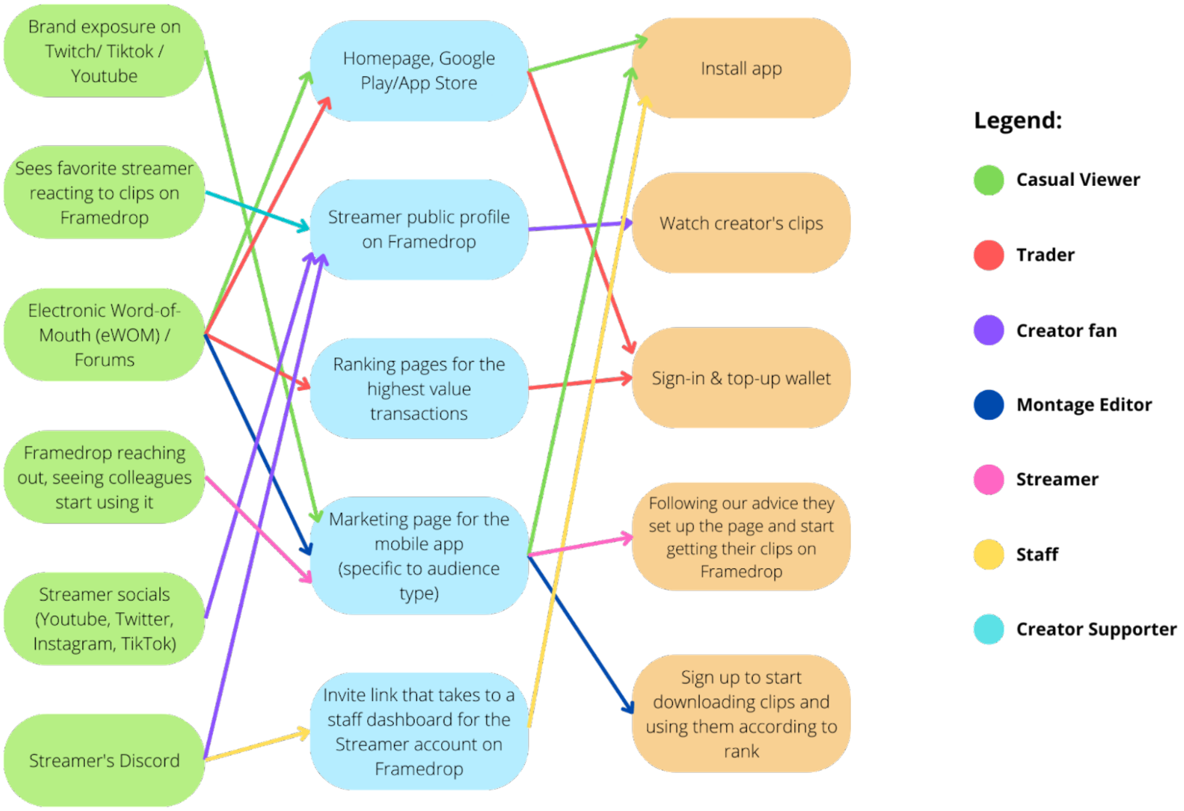

Brand exposure on Twitch/ Tiktok / Youtube
Sees favorite streamer reacting to clips on Framedrop
Electronic Word-of-Mouth (eWOM) / Forums
Framedrop reaching out, seeing colleagues start using it
Streamer socials (Youtube, Twitter, Instagram, TikTok)
Streamer's Discord
Homepage, Google Play/App Store
Streamer public profile on Framedrop
Ranking pages for the highest value transactions
Marketing page for the mobile app (specific to audience type)
Invite link that takes to a staff dashboard for the Streamer account on Framedrop
Install app
Watch creator's clips
Sign-in & top-up wallet
Following our advice they set up the page and start getting their clips on Framedrop
Sign up to start downloading clips and using them according to rank
Legend:
Casual Viewer
Trader
Creator fan
Montage Editor
Streamer
Staff
Creator Supporter

Figure 1. Map of interactions channels between possible end-users and Framedrop

The core experience, which represents the user walkthrough in the app was defined aiming to a target moment: unlocking a clip. This action is the most important to Framedrop because users paying to unlock the clip revenues the company with a percentage of that transaction, and it's a sign that the users are engaging with the app. To get users to perform this action, it was taken into account the minimal steps needed (number of screens or clicks) to do it and the moments after it, that should encourage repetition. With this information, the flow is described below.

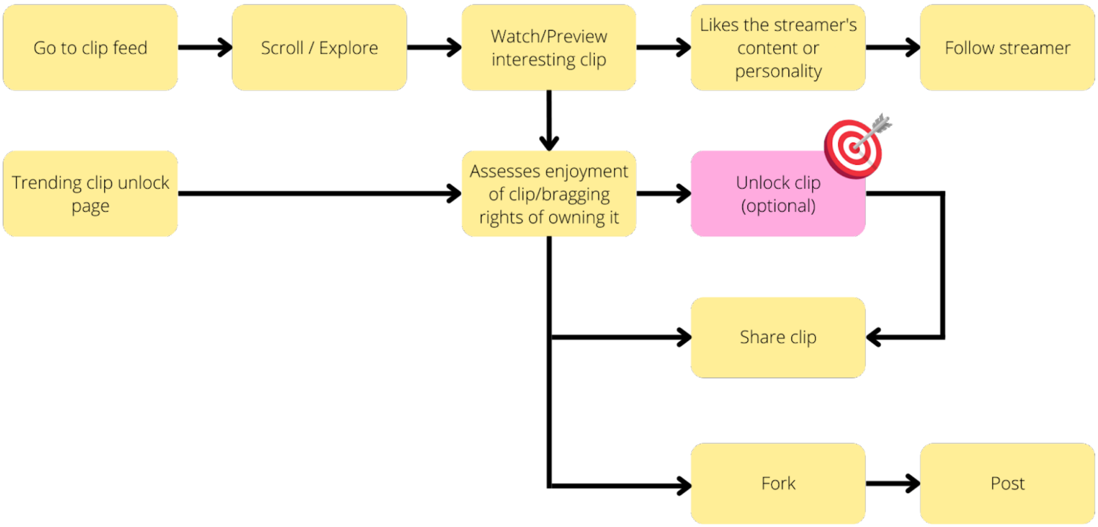

Go to clip feed
Scroll / Explore
Watch/Preview interesting clip
Likes the streamer's content or personality
Follow streamer
Trending clip unlock page
Assesses enjoyment of clip/bragging rights of owning it
Unlock clip (optional)
Share clip
Fork
Post

Figure 2. Application flow and target moment

Identify core problems

After mapping the product, we conducted the “Ask Experts” meeting where we learned about streamer’s routines, engagement with fans, communication channels, tools used in stream to interact with fans and other aspects regarding Twitch and the community. This interview was complemented with an exercise where we drafted some “How Might We” notes that consisted of problems that we could convert into opportunities, valuing the product. The vote picked the following questions as the most important:

1. “How might we do an UX that focus on viewer's karma points (eg. exponentiate bragging rights)?”
2. “How might we provide further value and bragging rights in a user’s profile?”

3.2. Conclusion

At the end of the day, we identified what we considered the most important aspects regarding the application, collecting as much information as possible from the exercises performed during the day and felt ready to move to the Sketch and Decide section.

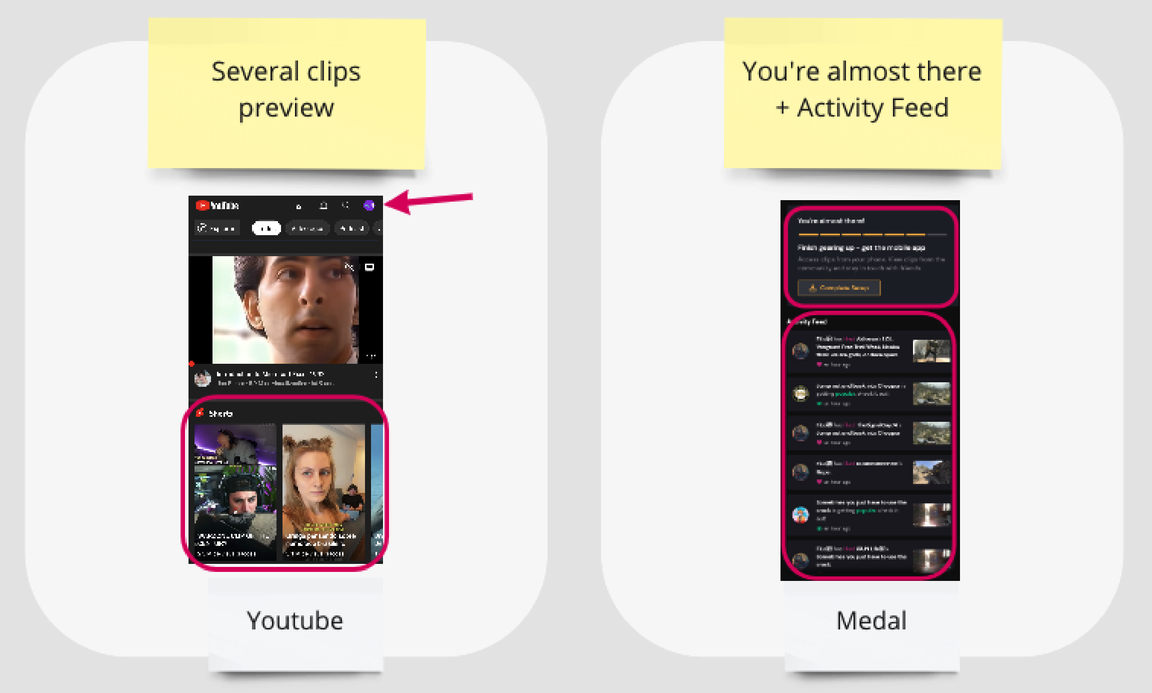
4. Second Day

The second day was mainly focused on finding a solution to the target moment identified on the first day. The original document used the time only for sketching ideas, reviewing the existing ones and improving [82], but with a short time, the plan was to sketch possible detailed, opinionated solutions and decide the best ideas for the prototype.

4.1. Introduction

The original framework tells that the Sketch section is only for the target moment, identified on the earlier sections, but as we were designing the product as a whole, we focused our Sketch section to all the application screens and the target moment.

The first exercise aimed to find inspiring solutions from other industries or, in other words, components that could be beneficial to implement in Framedrop. It was given 3 minutes to gather as much information as possible for this task. Some examples are presented below.



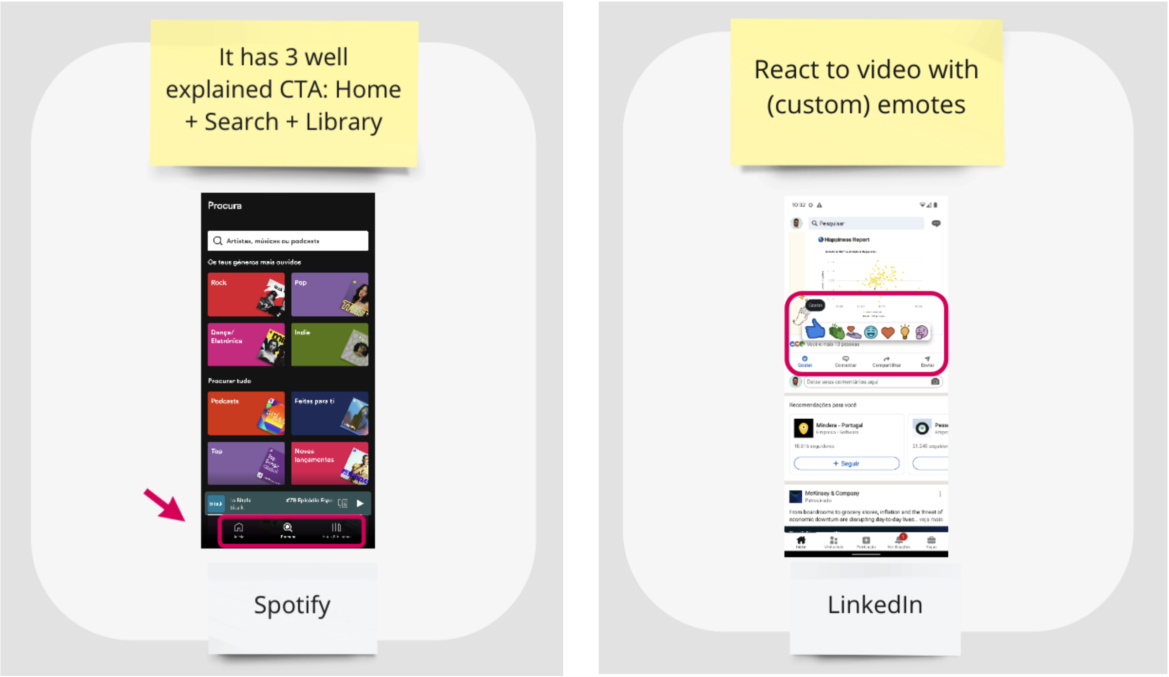

It has 3 well explained CTA: Home + Search + Library
Spotify
React to video with (custom) emotes
LinkedIn

Figure 3. Component inspiration gathered during the exercise

With all these components as inspiration, was "time to sketch" where each team element would brainstorm some ideas on a piece of paper and create sketches of the different application screens. These drafts were anonymous so it was important to have a significant amount of detail to be well interpreted by the team.

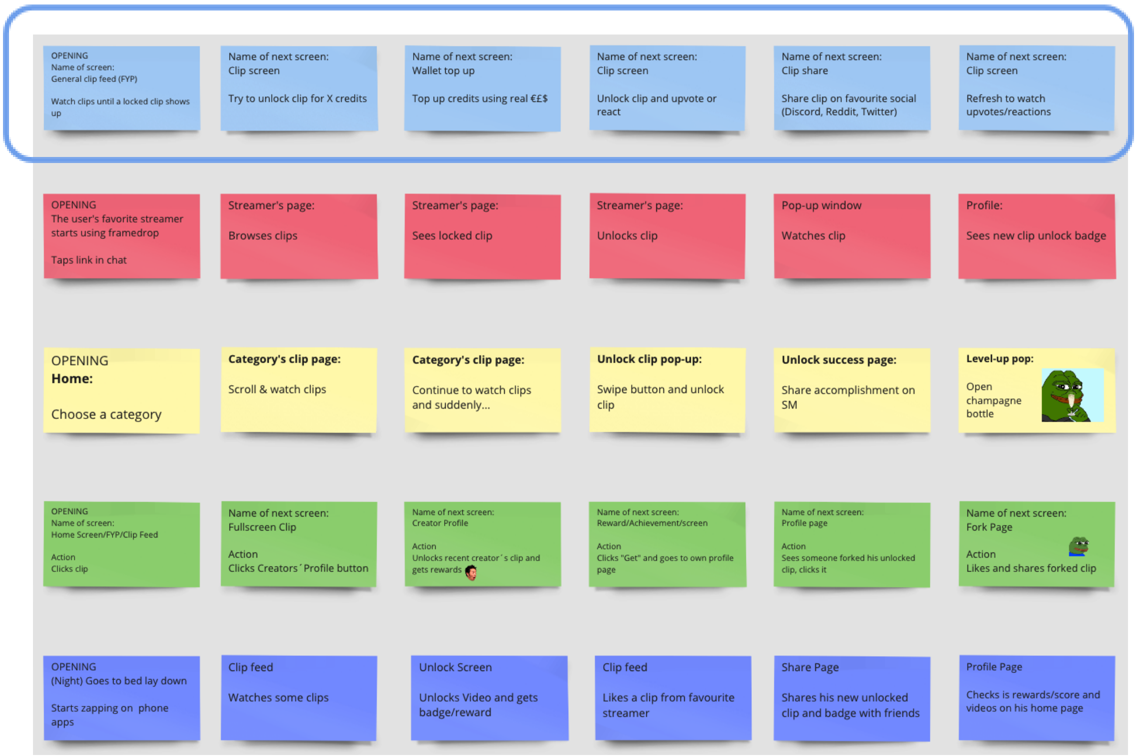
The next exercise was to analyze the sketches and individually vote for the best component or characteristic idea of the different solutions. The most voted ideas were discussed by the group and labeled. This exercise is important to mitigate interpretation problems.

The next part was an update to the book, called "Call out the hypotheses". It consisted of getting specific about the hypotheses that might meet the long-term goal and metrics, behind the winning solutions, which could be very helpful to focus the team on the storyboard, prototype and test [111]. The two most voted hypotheses considered to be the ones who might met the long-term goal and metrics were the following:

- "IF we implement gamification techniques THEN customers will feel rewarded and get hooked on the app";

- “IF we show "unlocked by", feature badges, and highlight level ups THEN customers will feel more eager to use the app and retain”;
- “If we make clip watching more interactive and community-driven, we'll have a much stronger value proposition than our competitors”.

To conclude the Product Design Sprint, the group used the approach inspired by AJ&Smart’s called “Storyboarding 2.0”, which represents an easier approach to start the storyboard. With this technique, we didn’t do the storyboard exercise because we felt that the conclusions gathered from this “pre-exercise” were enough to proceed to prototyping and we were running out of time. To perform this exercise, we needed to identify the prototype’s first 6 steps, starting with an opening scene or entry before the product. Then the following steps should represent screens and clicks to show who customers move through the prototype respecting the flow that was mapped in the first day. The storyboards are presented below, being the first the most voted by the group.



OPENING Name of screen: General clip feed (FYP) Watch clips until a locked clip shows up	Name of next screen: Clip screen Try to unlock clip for X credits	Name of next screen: Wallet top up Top up credits using real €£$	Name of next screen: Clip screen Unlock clip and upvote or react	Name of next screen: Clip share Share clip on favourite social (Discord, Reddit, Twitter)	Name of next screen: Clip screen Refresh to watch upvotes/reactions
OPENING The user's favorite streamer starts using framedrop Taps link in chat	Streamer's page: Browses clips	Streamer's page: Sees locked clip	Streamer's page: Unlocks clip	Pop-up window Watches clip	Profile: Sees new clip unlock badge
OPENING **Home:** Choose a category	**Category's clip page:** Scroll & watch clips	**Category's clip page:** Continue to watch clips and suddenly...	**Unlock clip pop-up:** Swipe button and unlock clip	**Unlock success page:** Share accomplishment on SM	**Level-up pop:** Open champagne bottle
OPENING Name of screen: Home Screen/FYP/Clip Feed Action Clicks clip	Name of next screen: Fullscreen Clip Action Clicks Creators´Profile button	Name of next screen: Creator Profile Action Unlocks recent creator´s clip and gets rewards	Name of next screen: Reward/Achievement/screen Action Clicks "Get" and goes to own profile page	Name of next screen: Profile page Action Sees someone forked his unlocked clip, clicks it	Name of next screen: Fork Page Action Likes and shares forked clip
OPENING (Night) Goes to bed lay down Starts zapping on phone apps	Clip feed Watches some clips	Unlock Screen Unlocks Video and gets badge/reward	Clip feed Likes a clip from favourite streamer	Share Page Shares his new unlocked clip and badge with friends	Profile Page Checks is rewards/score and videos on his home page

Figure 4. Storyboard 2.0 highlighting the most voted by the group

4.2. Conclusion

Concluding the Product Design Sprint, we felt that the exercise was a success and had the information needed to develop a prototype for the product. With the long-term goal and metrics in mind, and the sketches as a solution for the target moment, we could mitigate the error of designing a prototype that won't fit for the product.

5. Conclusion and Future Work

"In a nutshell, a Product Design Sprint is a workshop that allows businesses to reduce the risk associated with bringing a new product or feature to market, and to answer complex business questions within a very short time" [114]. The next phases will be focused on requisite gathering and priorities definition, so we can map all the different components of the application in time and create estimations for each of them.

Appendix F - User Story List

User Story	Functional Requirement	Priority	Done?
As a user, I should be presented with the menu icons, so I can interact with them.	us_cf1	Must Have	✓
As a user, I should be able to navigate between clips, so I can watch them.	us_cf2	Must Have	✓
As a user, I should be able to interact with clips, so I can pause and play.	us_cf3	Must Have	✓
As an authenticated user, I should be able to interact with the comment button, so I can view other people's comments.	us_cf4	Must Have	✓
As an authenticated user, I should be able to interact with the comment button, so I can comment on the clip.	us_cf5	Must Have	✓
As an authenticated user, I should be able to interact with a comment button, so I can like other people's comments.	us_cf6	Could Have	✗
As an authenticated user, I should be able to interact with the upvote button, so I can show my interest for that specific clip.	us_cf7	Must Have	✓
As an authenticated user, I should be able to interact with the share button, so I can share with the people that I want.	us_cf8	Must Have	✓
As an authenticated user, I should be able to interact with the fork button, so I can have a personalized clip copy done by me on my profile.	us_cf9	Will Not Have	✗
As a user, I should be able to interact with the clip's user avatar, so I can navigate to his profile.	us_cf10	Must Have	✓
As a user, I should be able to press the clip's claimer name, so I can check his	us_cf11	Must Have	✓

profile.			
As an authenticated user, I should be able to navigate to the notification center, so I can check who interacted with me.	us_cf12	Could Have	✗
As an authenticated user, I should be able to navigate to the "Following" tab, so I can only see clips from followed users.	us_cf13	Could Have	✗
As an authenticated user, I should be able to navigate to the "Feed" tab, so I can see all recommended clips from all users.	us_cf14	Could Have	✗
As an authenticated user, I should be able to navigate to the wallet menu, so I can check my wallet amount or top it up.	us_cf15	Must Have	✓
As a user, I should be able to navigate to the search menu, so I can search for users or videos.	us_cf16	Must Have	✓
As a user, I should be able to navigate to clip contest, so I can see clip classifications of a game that I like.	us_cf17	Must Have	✓
As an authenticated user, I should be able to navigate to my profile, so I can manage it.	us_cf18	Must Have	✓
As an authenticated user, I should be able to interact with an unclaimed clip, so I can try to claim it.	us_cf19	Must Have	✓
As an authenticated user, I should be able to see the auction time left and last bid made on an unsigned clip, so I can manage my bid.	us_cf20	Will Not Have	✗
As a non-authenticated user, I should be able to login or create an account when I try to interact with app functionalities.	us_cf21	Must Have	✓
As an authenticated user, I should be able to unlock a video, so I can own it.	us_cf22	Must Have	✓
As a user, I should be presented with a search bar, so I can search for users or	us_sm1	Must Have	✓

clips.			
As a user, I should be presented with popular clips, so I can explore it.	us_sm2	Will Not Have	✗
As a user, I should be presented with popular games, so I can explore it.	us_sm3	Will Not Have	✗
As a user, I should be presented with recommended clips, so I can explore it.	us_sm4	Will Not Have	✗
As a user, I should be presented with streamers on the rise, so I can explore them.	us_sm5	Will Not Have	✗
As a user, I should be presented with clips waiting to be unlocked, so I can try to claim them.	us_sm6	Will Not Have	✗
As a user, I should be able to search for a specific user or clip, so I can interact with them.	us_sm7	Must Have	✓
As a user, I should be able to navigate to clip feed, so I can view recommended clips.	us_sm8	Must Have	✓
As a user, I should be able to navigate to clip contest, so I can see clip classifications of a game that I like.	us_sm9	Must Have	✓
As an authenticated user, I should be able to navigate to my profile, so I can manage it.	us_sm10	Must Have	✓
As a non-authenticated user, I should be able to login or create an account when I try to interact with app functionalities.	us_sm11	Must Have	✓
As a user, I should be presented with a list of games when entering the clip contest, so I choose which I want to access.	us_cc1	Must Have	✓
As a user, I should be presented with a clip classification, so I can see who's winning.	us_cc2	Must Have	✓

As a user, I should be able to navigate back to the game list, so I can choose another game.	us_cc3	Must Have	✓
As a user, I should be able to interact with a specific clip, so I can watch it.	us_cc4	Must Have	✓
As a user, I should be able to switch between daily, weekly and monthly contest, so I can view classifications.	us_cc5	Must Have	✓
As an authenticated user, I should be able to display my clips only, so I can view my own clips.	us_cc6	Must Have	✓
As a user, I should be able to navigate to clip feed, so I can view recommended clips.	us_cc7	Must Have	✓
As a user, I should be able to navigate to the search menu, so I can search for users or videos.	us_cc8	Must Have	✓
As an authenticated user, I should be able to navigate to my profile, so I can manage it.	us_cc9	Must Have	✓
As a user, I should be able to see the contest time left when I enter a clip contest.	us_cc10	Must Have	✓
As a non-authenticated user, I should be able to login or create an account when I try to do with the app functionalities.	us_cc11	Must Have	✓
As an authenticated user, I should be able to click on the clip owner's name, so I can view his profile.	us_cc12	Must Have	✓
As an authenticated user, I should be able to click on the clip claimer's name, so I can view his profile.	us_cc13	Must Have	✓
As an authenticated user, I should be presented with my profile information, so I can manage it.	us_pm1	Must Have	✓

As an authenticated user, I should be able to navigate to the settings menu, so I can edit my experience on the application.	us_pm2	Will Not Have	✗
As an authenticated user, I should be able to press to navigate to my edit profile menu, so I can edit my profile information.	us_pm3	Will Not Have	✗
As an authenticated user, I should be able to navigate to the wallet menu, so I can top-up my wallet.	us_pm4	Must Have	✓
As an authenticated user, I should be able to press the Twitch icon, so I can navigate to my Twitch account.	us_pm5	Could Have	✓
As an authenticated user, I should be able to switch between "Streamed", "Claimed", "Forked", and "Upvotes" tabs, so I can see the content from each of them.	us_pm6	Must Have	✓
As an authenticated user, I should be able to interact with a clip, so I can watch it.	us_pm7	Must Have	✓
As an authenticated user, I should be able to navigate to clip feed, so I can view recommended clips.	us_pm8	Must Have	✓
As an authenticated user, I should be able to navigate to search menu, so I can search for users or videos.	us_pm9	Must Have	✓
As an authenticated user, I should be able to navigate to clip contest menu, so I can see clip classifications of a game that I like.	us_pm10	Must Have	✓
As an authenticated user, I should be able to check my current credits, so I can manage it.	us_wm1	Must Have	✓
As an authenticated user, I should be able to top-up my wallet, so I can make in-app purchases.	us_wm2	Must Have	✓
As an authenticated user, I should be able to navigate back to clip feed, so I can see recommended clips.	us_wm3	Must Have	✓

As an authenticated user, I should be able to navigate back to my profile, so I can manage it.	us_wm4	Must Have	✓
As a non-authenticated user, I should be able to authenticate via Twitch, so I can authenticate via Framedrop.	us_lm1	Must Have	✓
As a non-authenticated user, I should be able to authenticate via Framedrop, so I can login into the application.	us_lm2	Must Have	✓
As a non-authenticated user, I should login with Twitch with a streamer's code referral, so I can get more free credits	us_lm3	Must Have	✓
As a user, I should be able to see a splash screen with Framedrop logo, as I enter in the application.	us_lm4	Must Have	✓

Appendix G - Usability Test Script

Hello!

We are developing a mobile app for Framedrop, a company that creates highlights capture software for Twitch streams, to help content creators save time. These clips are then available on Framedrop's web app and can be purchased by fans/interested parties for the streamer/clip, unlocking the clip for other users to view. This application focuses on helping streamers grow as content creators and bring fans closer together.

As I mentioned, the clips that become available on the platform can be purchased, with the purchaser becoming the claimer of the clip, and becoming bound to the creator of the clip (streamer). Users who claim clips are automatically eligible to participate in a contest, where they can win tokens to claim new clips.

Users of the Framedrop application, can view clips that are unlocked, claim locked videos, upvote a clip (similar to the like function), comment and share.

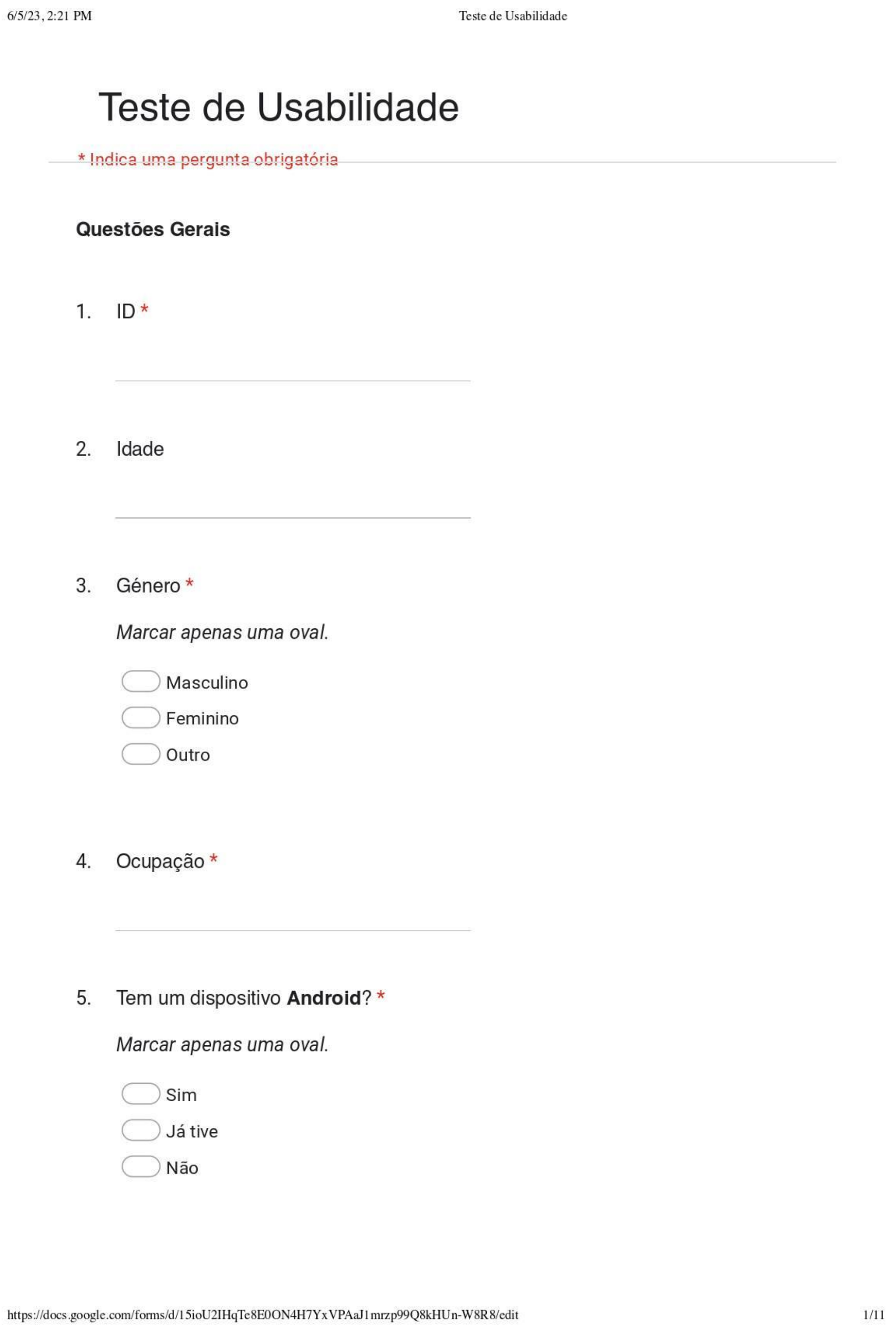
The mobile app follows on from the web app, giving users the same experience from their smartphone.

This session will be about 20 minutes long. Please feel free to be as candid as possible so that we can learn and improve this application. We want to make it clear up front that the purpose is to test the app and not you, meaning you will do no wrong here! Your sincerity is key to detecting problems/opportunities in early stages of development, avoiding critical issues in the future.

I would like to ask for your consent to record this session. The recording will be used internally, only to take results and avoid wasting precious information and time taking notes during the session.

We will start with some general questions before we begin testing the application.

Appendix H - Usability Test Form Questions



Teste de Usabilidade

* Indica uma pergunta obrigatória

Questões Gerais

1. ID *

2. Idade

3. Género *

Marcar apenas uma oval.

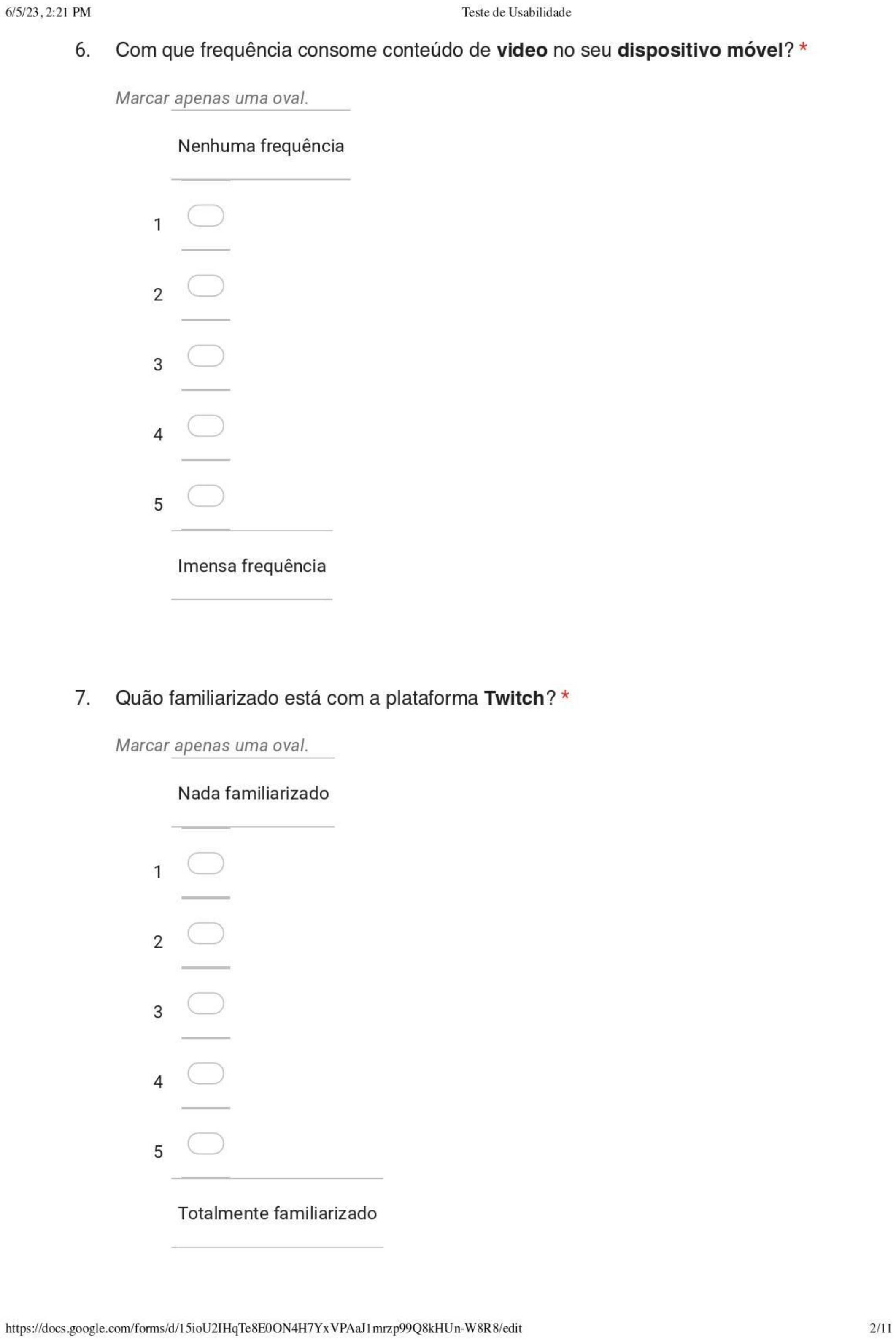
○ Masculino
○ Feminino
○ Outro

4. Ocupação *

5. Tem um dispositivo **Android**? *

Marcar apenas uma oval.

○ Sim
○ Já tive
○ Não



6. Com que frequência consome conteúdo de **video** no seu **dispositivo móvel**? *

Marcar apenas uma oval.

Nenhuma frequência

1 ○

2 ○

3 ○

4 ○

5 ○

Imensa frequência

7. Quão familiarizado está com a plataforma **Twitch**? *

Marcar apenas uma oval.

Nada familiarizado

1 ○

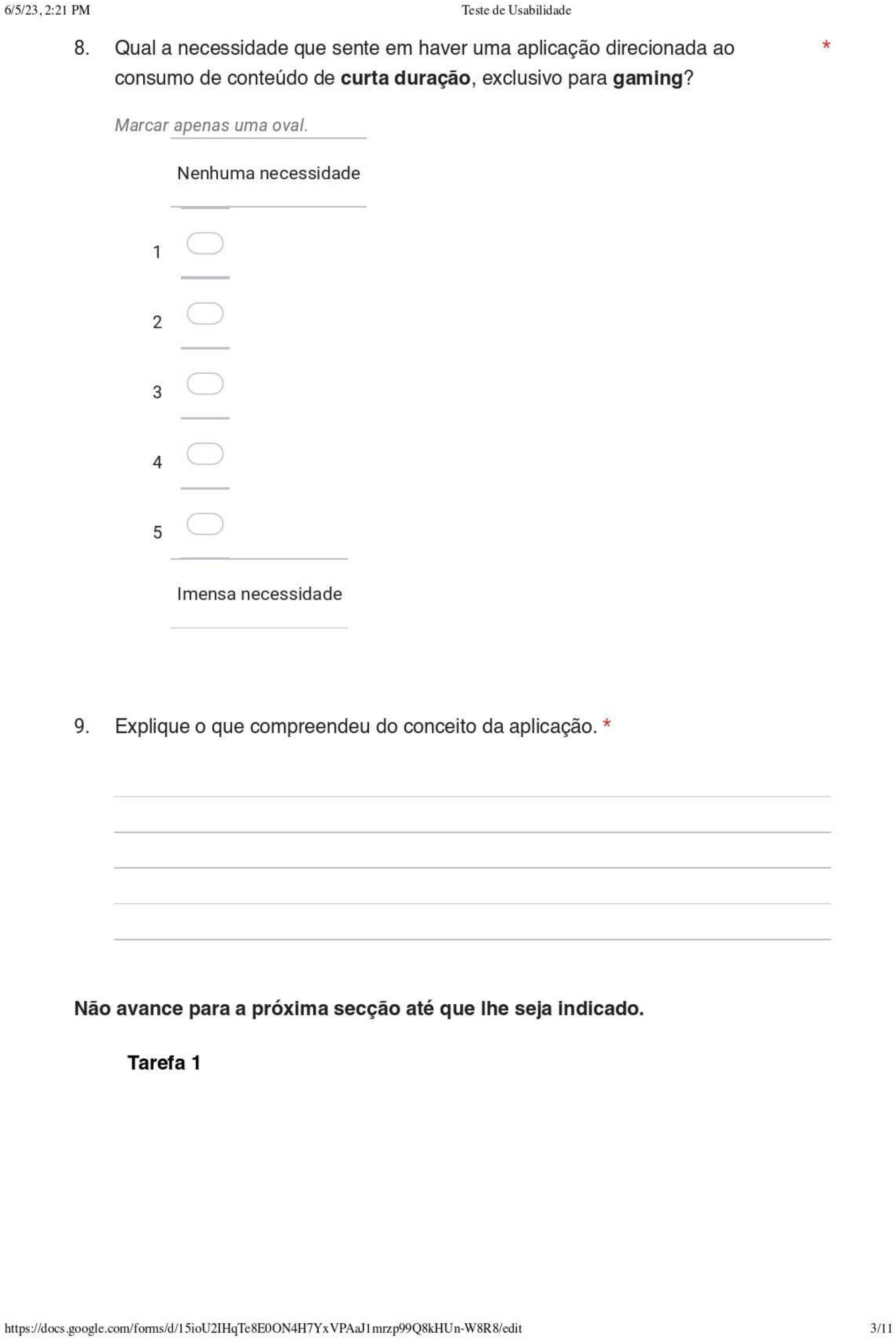
2 ○

3 ○

4 ○

5 ○

Totalmente familiarizado



8. Qual a necessidade que sente em haver uma aplicação direcionada ao consumo de conteúdo de **curta duração**, exclusivo para **gaming**? *

Marcar apenas uma oval.

Nenhuma necessidade

1 ○

2 ○

3 ○

4 ○

5 ○

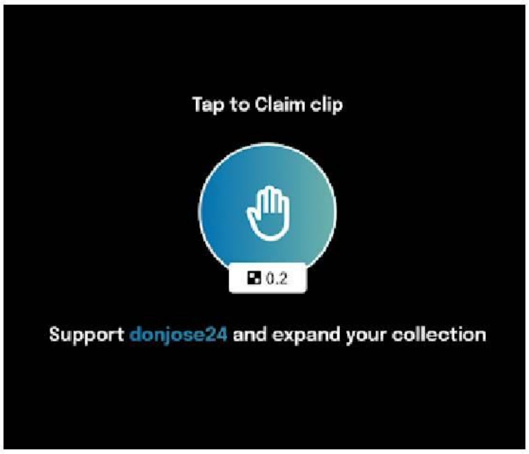
Imensa necessidade

9. Explique o que compreendeu do conceito da aplicação. *

Não avance para a próxima secção até que lhe seja indicado.

Tarefa 1

Exemplo de clip bloqueado



Tap to Claim clip

0.2

Support donjose24 and expand your collection

10. Quão complexo foi entender que não é possível dar upvote de um clip bloqueado?

Marcar apenas uma oval.

Muito complexo

1 ○

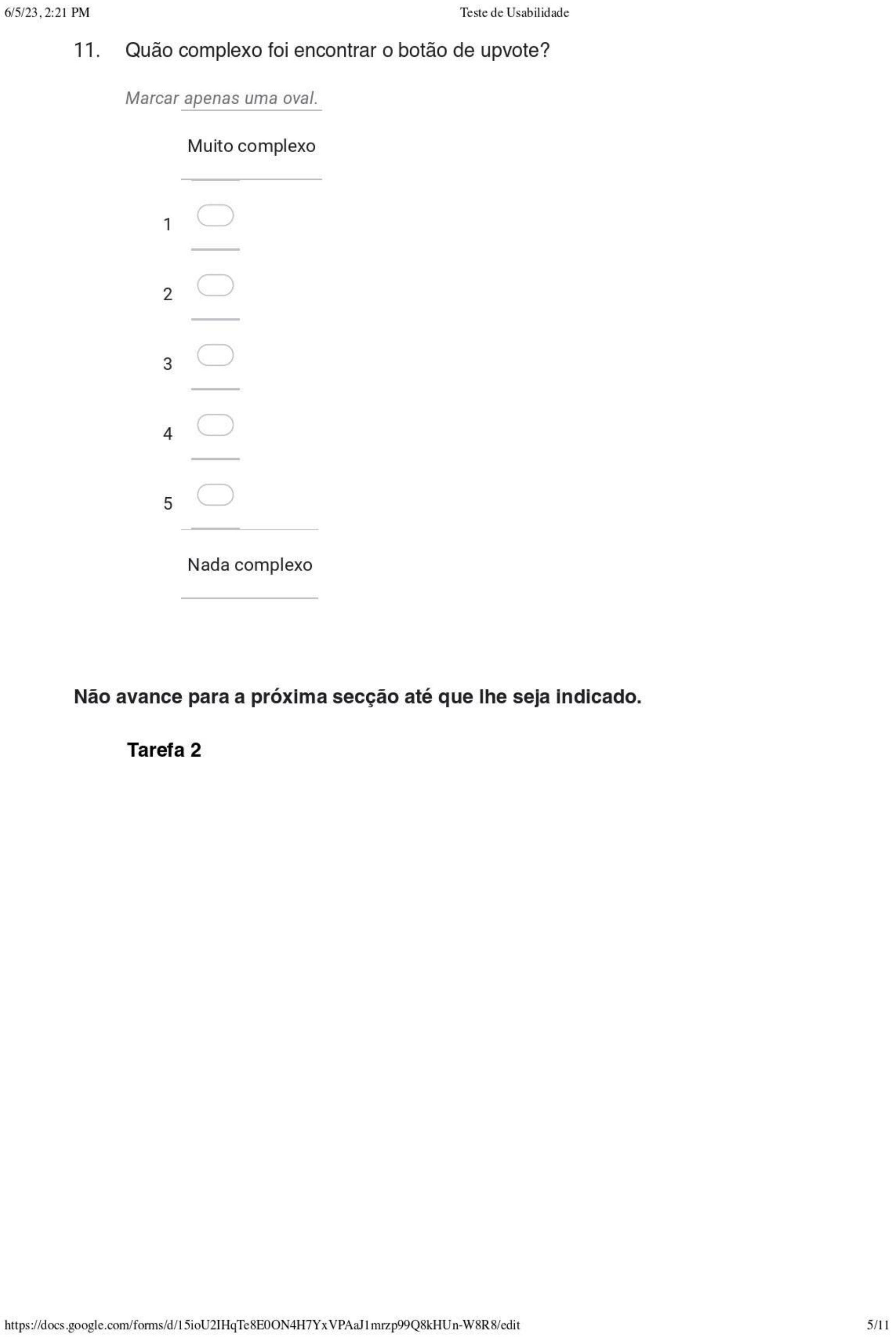
2 ○

3 ○

4 ○

5 ○

Nada complexo



11. Quão complexo foi encontrar o botão de upvote?

Marcar apenas uma oval.

Muito complexo

1 ○

2 ○

3 ○

4 ○

5 ○

Nada complexo

Não avance para a próxima secção até que lhe seja indicado.

Tarefa 2

12. Quão complexo foi identificar os jogos pelas fotografias apresentadas? *

Marcar apenas uma oval.

Muito complexo

1 ○

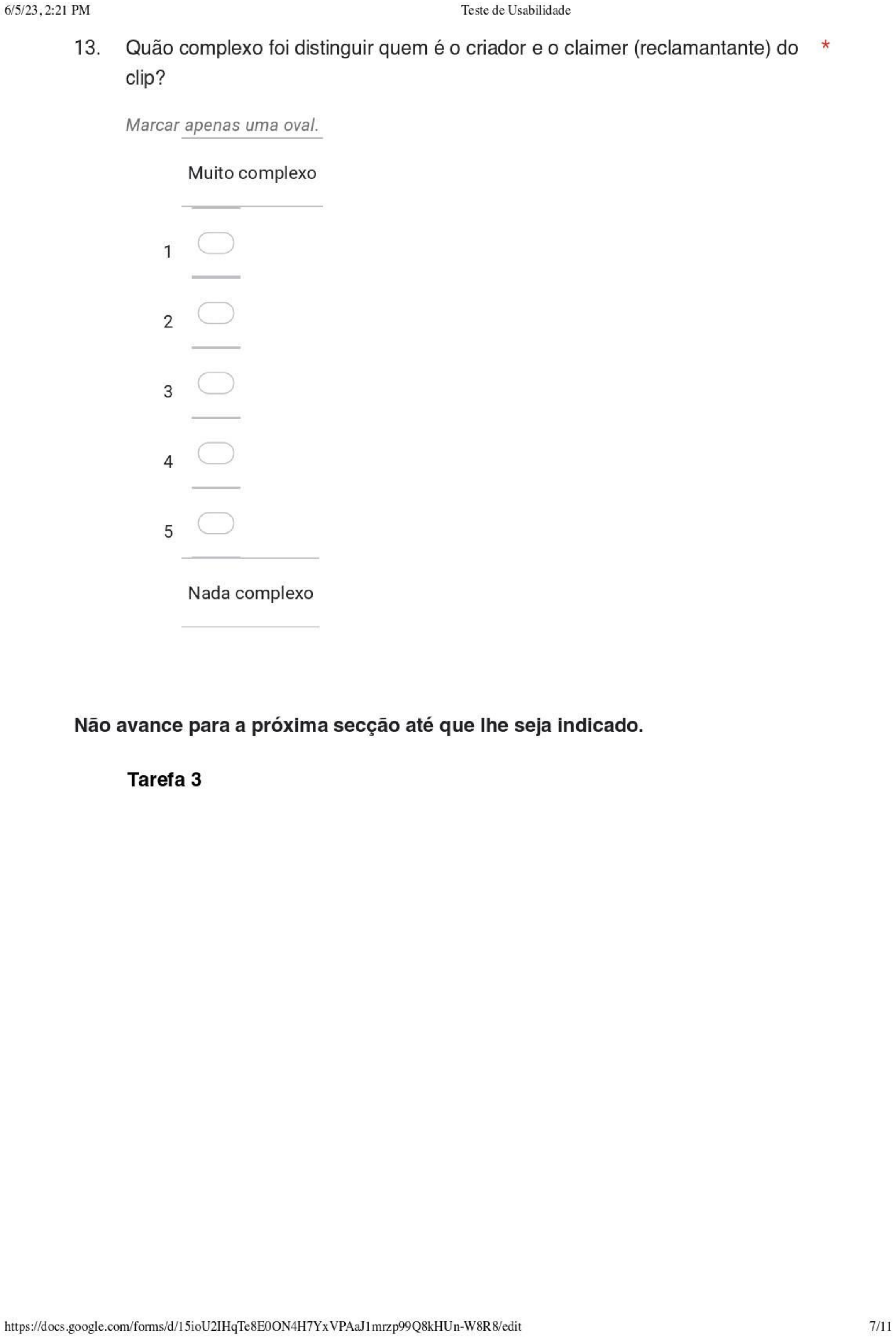
2 ○

3 ○

4 ○

5 ○

Nada complexo



13. Quão complexo foi distinguir quem é o criador e o claimer (reclamantante) do clip? *

Marcar apenas uma oval.

Muito complexo

1 ○

2 ○

3 ○

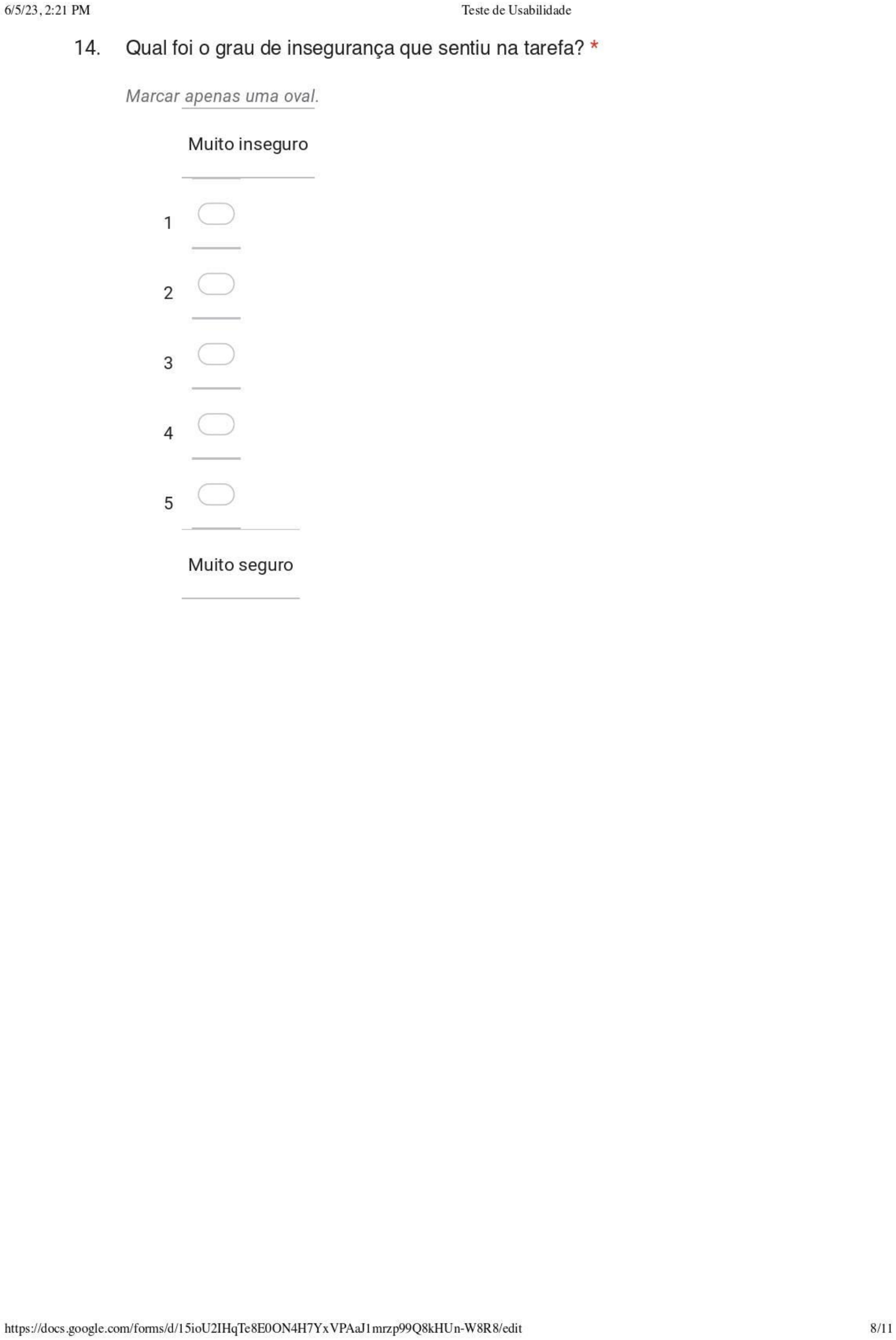
4 ○

5 ○

Nada complexo

Não avance para a próxima secção até que lhe seja indicado.

Tarefa 3



14. Qual foi o grau de insegurança que sentiu na tarefa? *

Marcar apenas uma oval.

Muito inseguro

1 ○

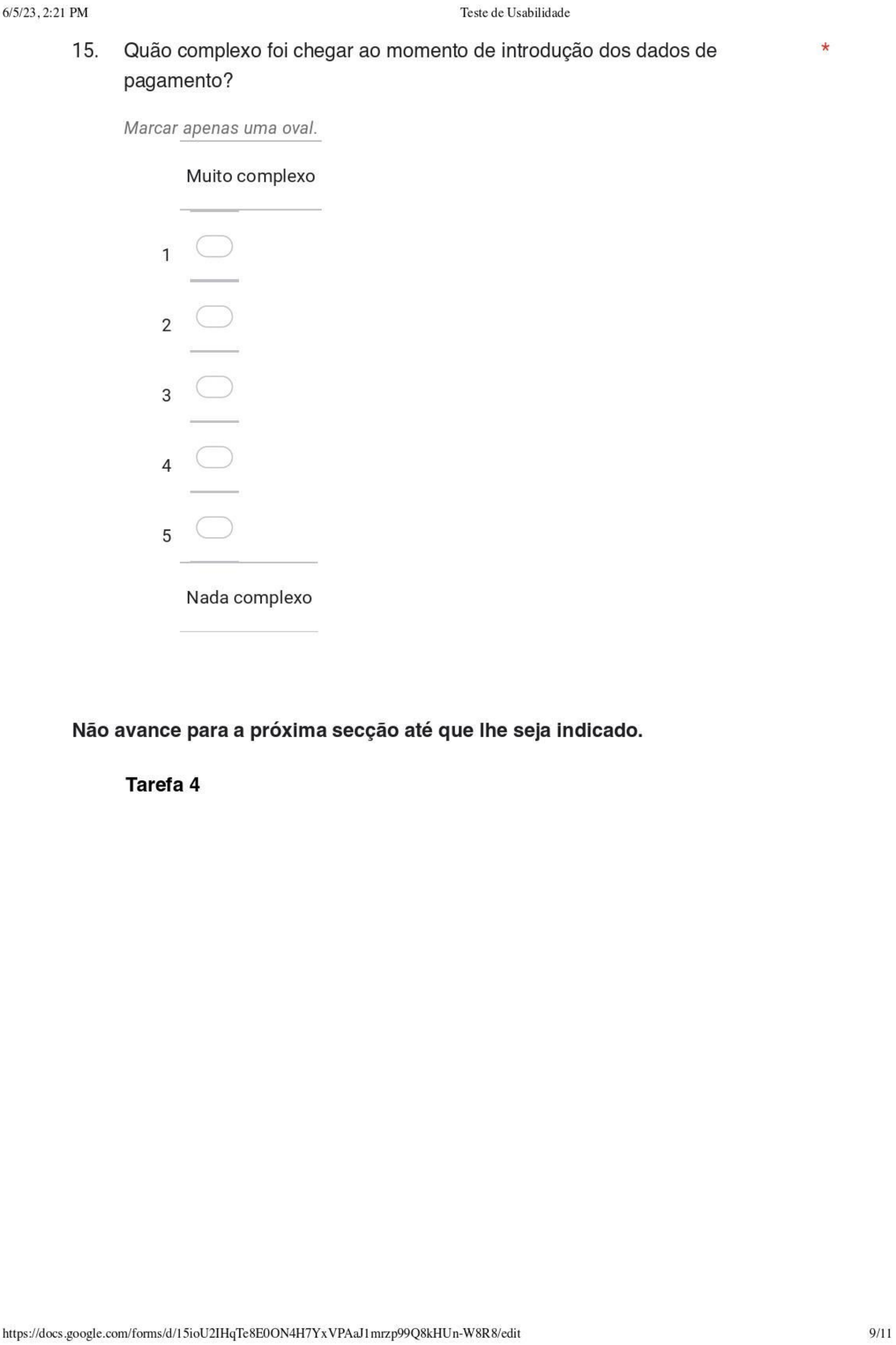
2 ○

3 ○

4 ○

5 ○

Muito seguro



15. Quão complexo foi chegar ao momento de introdução dos dados de pagamento? *

Marcar apenas uma oval.

Muito complexo

1 ○

2 ○

3 ○

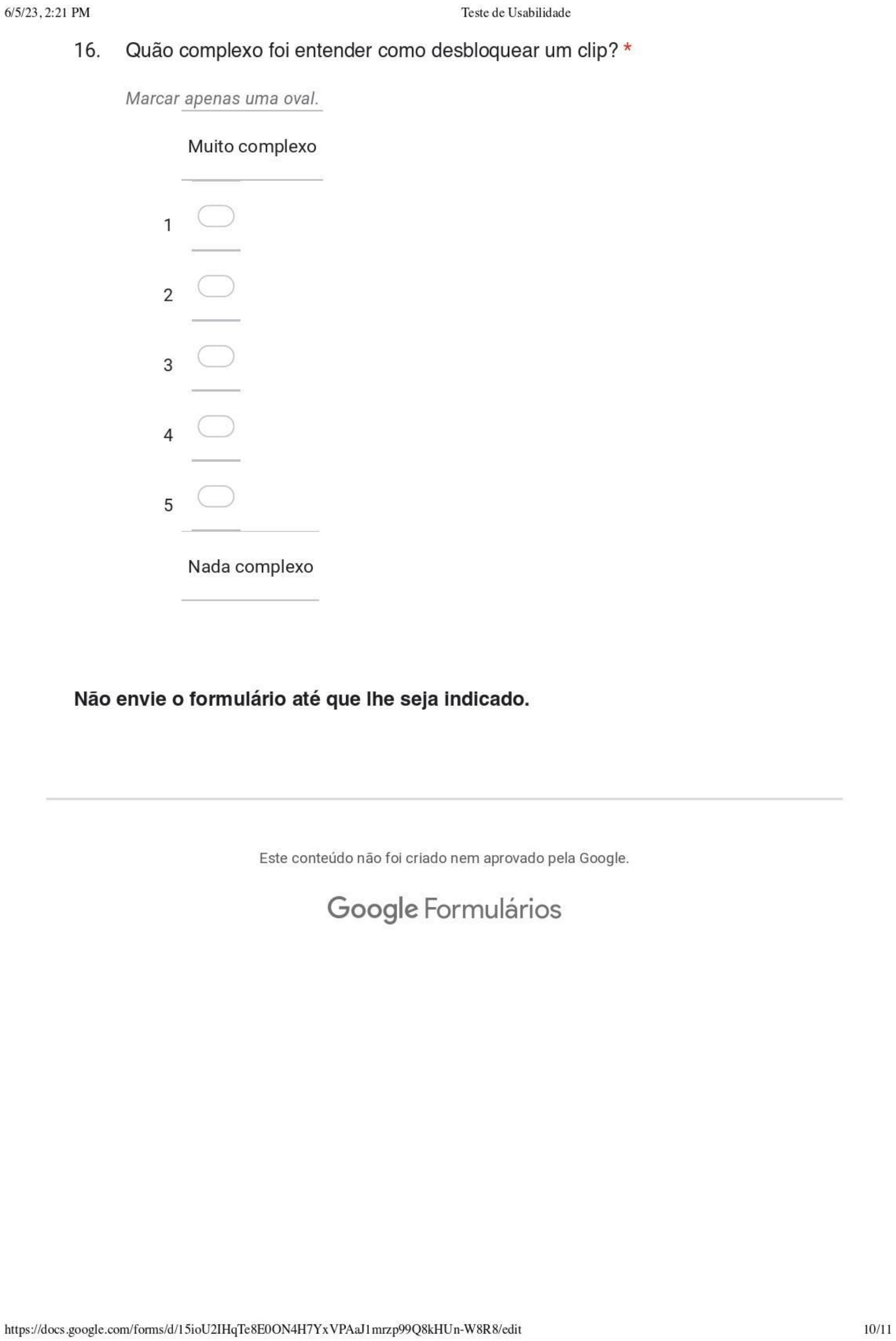
4 ○

5 ○

Nada complexo

Não avance para a próxima secção até que lhe seja indicado.

Tarefa 4



16. Quão complexo foi entender como desbloquear um clip? *

Marcar apenas uma oval.

Muito complexo

1 ○

2 ○

3 ○

4 ○

5 ○

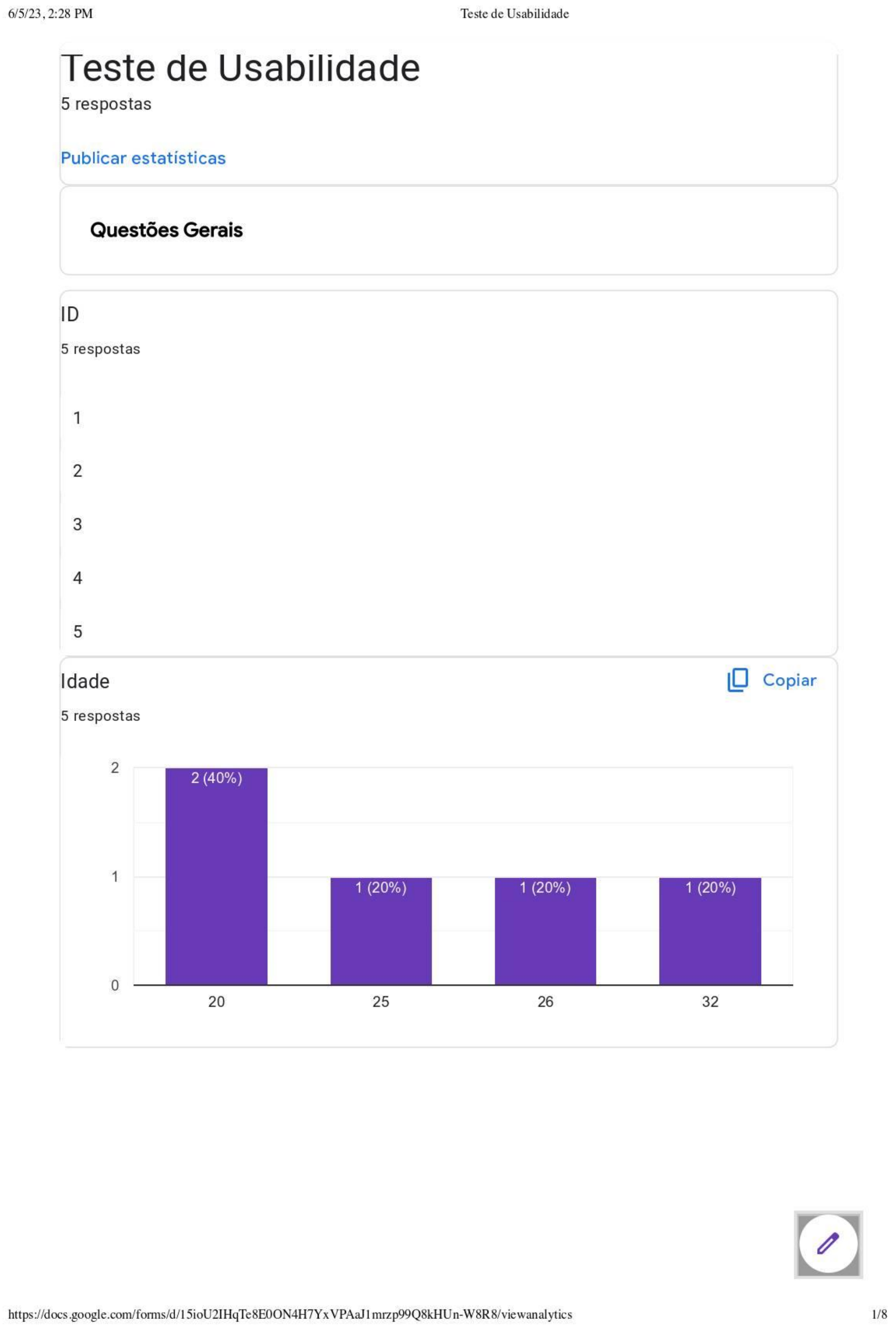
Nada complexo

Não envie o formulário até que lhe seja indicado.

Este conteúdo não foi criado nem aprovado pela Google.

Google Formulários

Appendix I - Usability Test Form Answers



6/5/23, 2:28 PM Teste de Usabilidade

Teste de Usabilidade

5 respostas

Publicar estatísticas

Questões Gerais

ID

5 respostas

1

2

3

4

5

Idade

Copiar

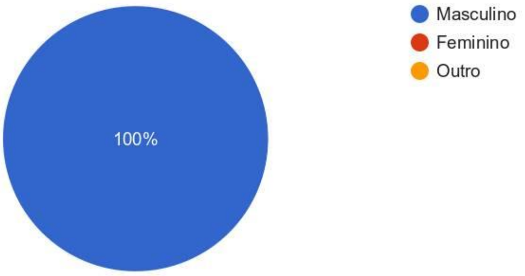
5 respostas

https://docs.google.com/forms/d/15ioU2IHqTe8E0ON4H7YxVPAaJ1mrzp99Q8kHUn-W8R8/viewanalytics 1/8

Género

Copiar

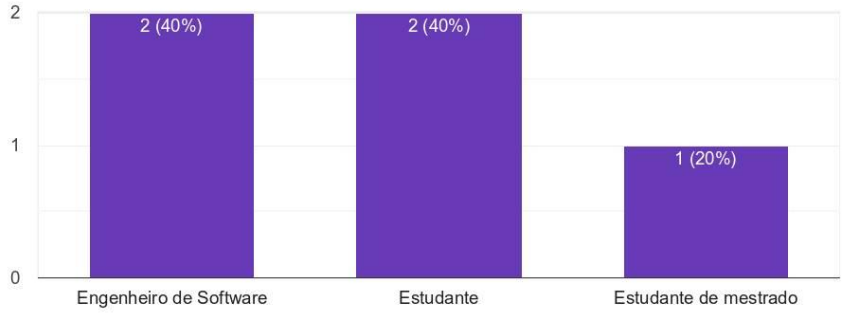
5 respostas



Ocupação

Copiar

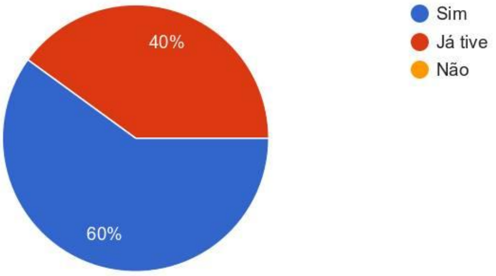
5 respostas



Tem um dispositivo **Android**?

Copiar

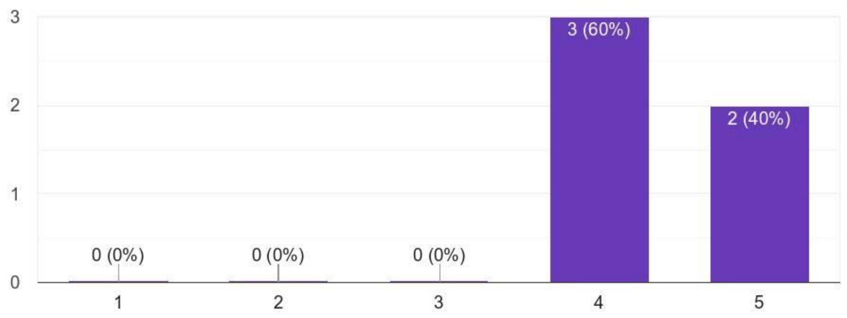
5 respostas



Com que frequência consome conteúdo de **video** no seu **dispositivo móvel**?

Copiar

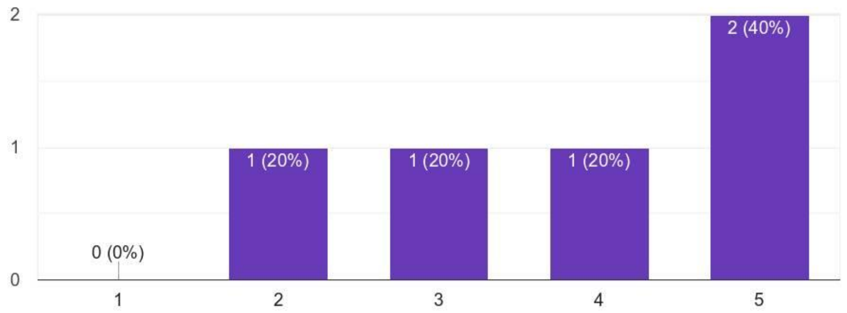
5 respostas



Quão familiarizado está com a plataforma **Twitch**?

Copiar

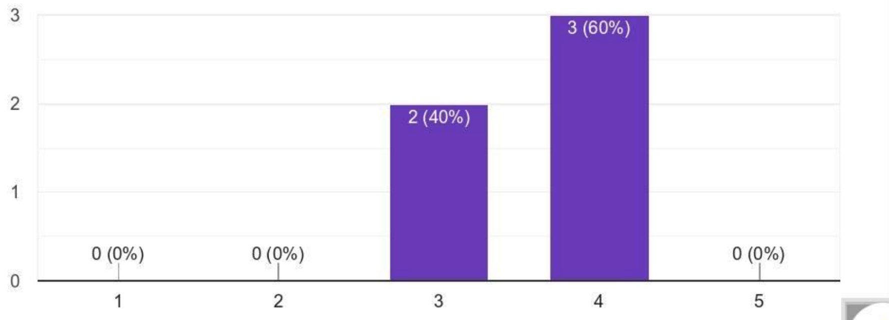
5 respostas



Qual a necessidade que sente em haver uma aplicação direcionada ao consumo de conteúdo de **curta duração**, exclusivo para **gaming**?

Copiar

5 respostas



Explique o que compreendeu do conceito da aplicação.

5 respostas

Os streamers que criam conteúdo, usam a plataforma, para selecionar o conteúdo mais interessantes. Partilham na plataforma. Outros utilizadores vêm esse conteúdo e podem fazer claim de video, seja o que isso for.

Market place de conteúdo (apenas comprado uma vez) e fica disponível para outros users. Outro conteúdo é adicionado pelos streamers.

Permite ver clips de streamer, highlights feitos pela plataforma. Os users podem (disse todas as interações). Podem entrar em clip contests.

Permite ao criador de conteúdo capturar os seus highlights e rentabiliza los. Permite aos viewers apoiar os criadores CE conteudo. Haver uma interação diferente da que existe na livestream

Tiktok de highlights de streams de jogos

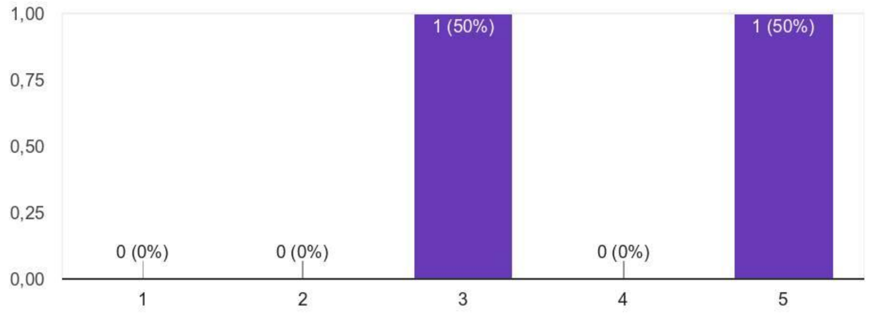
Não avance para a próxima secção até que lhe seja indicado.

Tarefa 1

Quão complexo foi entender que não é possível dar upvote de um clip bloqueado?

Copiar

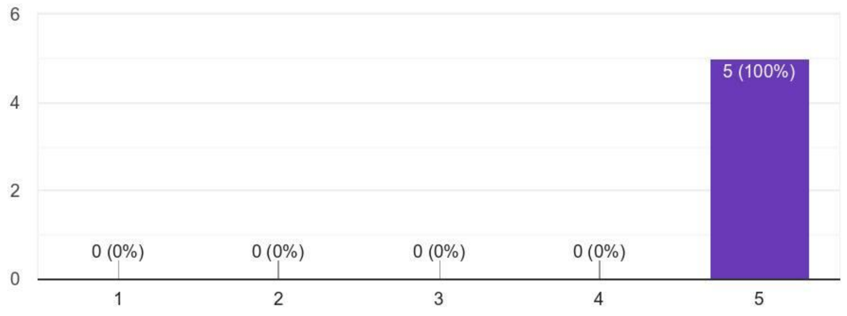
2 respostas



Quão complexo foi encontrar o botão de upvote?

Copiar

5 respostas



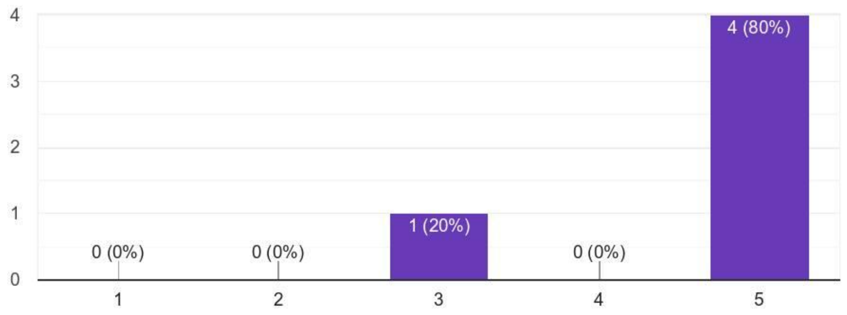
Não avance para a próxima secção até que lhe seja indicado.

Tarefa 2

Quão complexo foi identificar os jogos pelas fotografias apresentadas?

Copiar

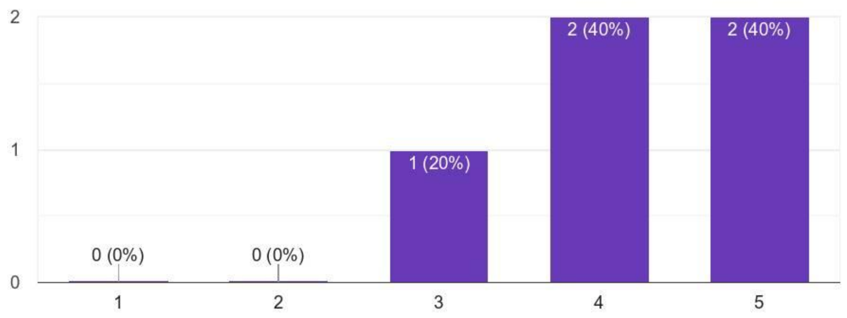
5 respostas



Quão complexo foi distinguir quem é o criador e o claimer (reclamantante) do clip?

Copiar

5 respostas



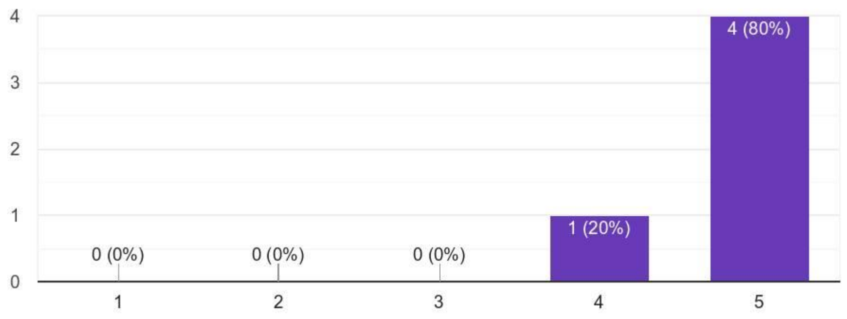
Não avance para a próxima secção até que lhe seja indicado.

Tarefa 3

Qual foi o grau de insegurança que sentiu na tarefa?

Copiar

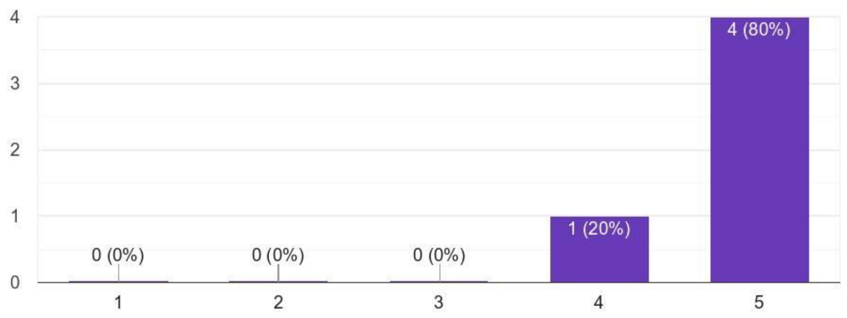
5 respostas



Quão complexo foi chegar ao momento de introdução dos dados de pagamento?

Copiar

5 respostas



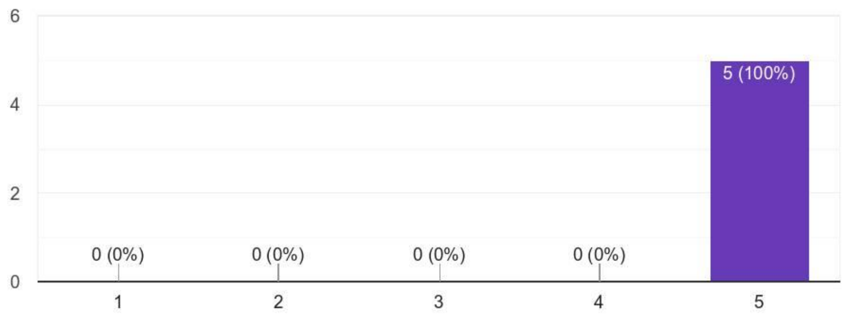
Não avance para a próxima secção até que lhe seja indicado.

Tarefa 4

Quão complexo foi entender como desbloquear um clip?

Copiar

5 respostas



Não envie o formulário até que lhe seja indicado.

Appendix J - Usability Test Results

Problems	Severity	Possible Solutions
Videos are slow to play and buffer several times	High	Change clips buffer size. Framedrop team changes video store service
Transition between videos not very fluid	High	Move some logic to another thread
Confirm that the user wants to unlock the clip, to avoid accidental touches	High	Create a confirmation pop-up or create a long-press animation
The app has a long loading time	Medium	Load only the necessary components at the beginning
When claiming a clip, the loading of the pull to refresh also appears	Low	Create a new state for the pull to refresh
When refreshing the page, only the video should be refreshed, not the whole page	Low	Update only the view that contains the list of clips
Icons (wallet, upvote) are not very explicit	Low	Create an onboarding page that explains buttons and their behaviors
Images have different sizes between devices	Low	Ask Framedrop to use images with the same sizes
Going to another screen and returning to the wallet gives error on paypal page	Low	Correct navigation between screens
The paypal menu could have more information (having only one button makes no sense)	Low	Add more content to the page
When watching a clip, the creator's avatar could be bigger	Low	Increase avatar size

Suggestions	Implementation Effort	Will it be included in the current release?
After opening the keyboard, you can hide it by touching anywhere on the screen	Low	Yes
Show mini clips of the user in the clip that is locked (similar to youtube videos, when the video ends)	High	No

Appendix K - Macrobenchmark Results

Google Pixel 3XL (2018)

App Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	553.0	571.0	615.0	451.5	466.5	508.7

Product Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	1 378.6	1 620.9	1 887.6	1 243.4	1 481.6	1 632.9

Scroll Clips	Base Code				Optimized App			
	P50	P90	P95	P99	P50	P90	P95	P99
Frame Duration	5.5	9.9	12.5	108.9	8.1	14.5	35.1	83.8
Frame Overrun	-14.1	-7.8	41.6	94.0	-7.9	12.8	48.1	70.0

Optimization			
	Min	Median	Max
App Startup Time	101.5 (18.35%)	104.5 (18.3%)	106.3 (17.28%)
Product Startup Time	135.2 (9.80%)	139.3 (8.59%)	254.7 (13.49%)

	P50	P90	P95	P99
Scroll Clips	0 (0%)	0 (0%)	0 (0%)	25.1 (23.05%)
	0 (0%)	0 (0%)	0 (0%)	24 (25.53%)

Samsung Galaxy S10E (2019)

App Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	539.8	583.8	927.2	485.0	554.8	649.8

Product Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	1 385.9	1 493.9	2 943.5	1 272.4	1 372.5	2 755.3

Scroll Clips	Base Code				Optimized App			
	P50	P90	P95	P99	P50	P90	P95	P99
Frame Duration	8.0	12.2	31.6	186.0	8.3	25.3	29.3	159.8
Frame Overrun	0.0	17.0	58.9	186.8	0.2	17.6	48.6	160.0

Optimization			
	Min	Median	Max
App Startup Time	54.8 (10.15%)	29 (4.97%)	277.4 (29.92%)
Product Startup Time	113.5 (8.19%)	121.4 (8.13%)	188.2 (6.39%)

	P50	P90	P95	P99
Scroll Clips	0 (0%)	0 (0%)	2.3 (7.28%)	26.2 (14.09%)
	0 (0%)	0 (0%)	10.3 (17.49%)	26.8 (14.35%)

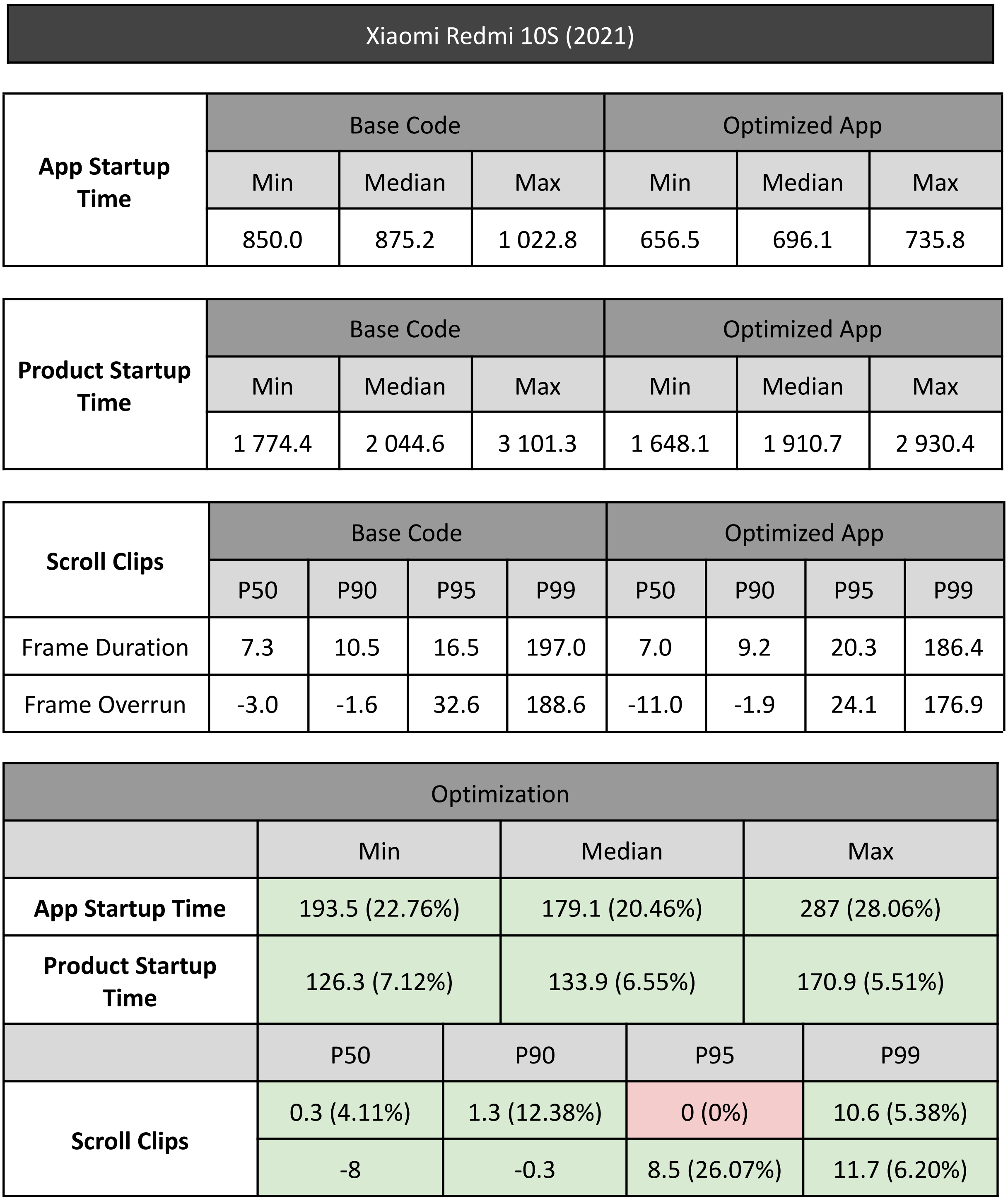
Xiaomi POCO M3 (2020)

App Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	1 458.7	1 508.5	2 170.7	1 278.7	1 359.2	1 743.3

Product Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	2 494.6	2 864.2	6 919.1	2 231.7	2 707.6	5 280.1

Scroll Clips	Base Code				Optimized App			
	P50	P90	P95	P99	P50	P90	P95	P99
Frame Duration	5.5	11.2	18.3	120.2	6.3	12.5	23.5	89.2
Frame Overrun	1.0	3.1	5.5	196.2	2.3	8.2	10.2	150.2

Optimization				
	Min	Median	Max	
App Startup Time	180 (12.33%)	149.3 (9.90%)	427.4 (19.69%)	
Product Startup Time	262.9 (10.54%)	156.6 (5.47%)	1 639 (23.69%)	
	P50	P90	P95	P99
Scroll Clips	0 (0%)	0 (0%)	0 (0%)	38 (25.80%)
	0 (0%)	0 (0%)	0 (0%)	46 (23.45%)



Xiaomi Redmi 10S (2021)

App Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	850.0	875.2	1 022.8	656.5	696.1	735.8

Product Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	1 774.4	2 044.6	3 101.3	1 648.1	1 910.7	2 930.4

Scroll Clips	Base Code				Optimized App			
	P50	P90	P95	P99	P50	P90	P95	P99
Frame Duration	7.3	10.5	16.5	197.0	7.0	9.2	20.3	186.4
Frame Overrun	-3.0	-1.6	32.6	188.6	-11.0	-1.9	24.1	176.9

Optimization				
	Min	Median	Max	
App Startup Time	193.5 (22.76%)	179.1 (20.46%)	287 (28.06%)	
Product Startup Time	126.3 (7.12%)	133.9 (6.55%)	170.9 (5.51%)	
	P50	P90	P95	P99
Scroll Clips	0.3 (4.11%)	1.3 (12.38%)	0 (0%)	10.6 (5.38%)
	-8	-0.3	8.5 (26.07%)	11.7 (6.20%)

Poco X4 Pro 5G (2022)

App Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	1 098.6	1 203.4	2 247.8	856.5	902.6	1 930.0

Product Startup Time	Base Code			Optimized App		
	Min	Median	Max	Min	Median	Max
	1 908.4	2 259.5	5 495.9	1 709.1	1 996.0	4 516.4

Scroll Clips	Base Code				Optimized App			
	P50	P90	P95	P99	P50	P90	P95	P99
Frame Duration	5.0	8.8	11.5	54.1	3.3	8.0	13.1	45.6
Frame Overrun	-3.6	-2.2	3.6	98.2	-3.6	-1.2	26.1	77.3

Optimization				
	Min	Median	Max	
App Startup Time	242.1 (22.04%)	300.8 (25.00%)	317.8 (14.14%)	
Product Startup Time	199.3 (10.44%)	263.5 (11.66%)	979.5 (17.82%)	
	P50	P90	P95	P99
Scroll Clips	1.7 (34%)	0.8 (9.09%)	0 (0%)	8.5 (15.71%)
	0 (%0)	0 (0%)	0 (0%)	20.9 (21.28%)