NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

SARA ALMEIDA ISIDRO RELVAS

BSc in Computer Science and Engineering

IMPROVING THE USER EXPERIENCE OF NON-LINEAR ACCESS TO VOD CONTENT

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
April, 2024

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF
COMPUTER SCIENCE

IMPROVING THE USER EXPERIENCE OF NON-LINEAR ACCESS TO VOD CONTENT

SARA ALMEIDA ISIDRO RELVAS

BSc in Computer Science and Engineering

Adviser: Teresa Romão
Associate Professor, NOVA University Lisbon

Co-adviser: Pedro Centieiro
Chief Operating Officer and Co-Founder, Magycal

Examination Committee

Chair: João Carlos Gomes Moura Pires
Associate Professor, NOVA University Lisbon

Rapporteur: Ana Paula Pereira Afonso
Assistant Professor, Faculty of Science of University of Lisbon

Member: Teresa Romão
Associate Professor, NOVA University Lisbon

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
April, 2024

Improving the User Experience of non-linear access to VOD content

Copyright © Sara Almeida Isidro Relvas, NOVA School of Science and Technology, NOVA University Lisbon.
The NOVA School of Science and Technology and the NOVA University Lisbon have the right, perpetual and without geographical boundaries, to file and publish this dissertation through printed copies reproduced on paper or on digital form, or by any other means known or that may be invented, and to disseminate through scientific repositories and admit its copying and distribution for non-commercial, educational or research purposes, as long as credit is given to the author and editor.

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v6.10.16) .

Acknowledgements

The journey of writing this dissertation was not an easy path, filled with challenges and personal doubts. However, with the constant support and guidance of my advisor Professor Teresa Romão and co-advisor Dr. Pedro Centieiro, this endeavor became a rewarding experience. For that, I'm extremely grateful.

A special thanks also to Magycal for collaborating with me on this thesis. Their expertise and guidance were crucial in achieving a final product that I am genuinely proud of.

To my "Mooshakamos", the peanut butter to my jelly, this academic journey would not be the same without you guys, and for that, I'm forever grateful.

Finally, I'd like to thank my favorite people, my family, who never stopped supporting me and always had my back no matter what. I love you dearly.

This is for you guys. I did it.

Abstract

The transition towards non-linear content consumption has seen a remarkable surge in popularity over the past few years. This shift is primarily due to the superior watching experience offered by Over-the-Top (OTT) media platforms compared to traditional linear television. OTT platforms provide users with access to an extensive array of content, available anywhere, anytime, that the users can see at their own pace. Despite the several advantages of OTT services, the current video players within these platforms still fall short in several functionalities that could significantly enhance the user experience while watching VOD content.

This dissertation focused on conceptualizing, designing, and developing new features to be integrated into current Android OTT video player interfaces that would enhance the user experience while watching VOD content and would be adaptable to any type of content. The proposed solutions include a customizable visual timeline that showcases all of the VOD key events, enabling users to jump to desired moments with just one tap. The user can also see all the video's main events sequentially like a highlight reel with the next event automatically starting after the current one concludes. Leveraging crowdsourcing, users can save custom events within the official timeline. Additionally, we proposed features to help users track their progress on the VOD. Upon reaching the final product, we conducted usability tests with users. These tests received remarkably positive feedback, as users enthusiastically embraced integrating these features into current OTT services, highlighting a highly positive experience with the implemented features and recognizing the improved user experience while consuming VOD content.

This dissertation had support from Magycal, a company renowned for its expertise in developing award-winning OTT platforms and second-screen applications.

Keywords: Over-the-top, Video On Demand, Human-Computer Interaction, Video Player, User Experience, User Interface, Mobile design, User Interaction

Resumo

A transição para o consumo não linear de conteúdo tem visto um aumento na popularidade nos últimos anos. Isto sucede-se devido à experiência superior de visualização oferecida por plataformas de mídia *Over-the-Top (OTT)* em comparação com a televisão linear tradicional. Estas plataformas oferecem acesso a conteúdo, disponível em qualquer lugar, a qualquer momento, que os utilizadores podem ver ao seu próprio ritmo. Apesar das vantagens dos serviços *OTT*, os atuais leitores de vídeo dentro dessas plataformas ainda carecem de várias funcionalidades que poderiam melhorar a experiência do utilizador ao assistir ao conteúdo.

Esta tese focou-se em conceber, projetar e desenvolver novas funcionalidades, a serem integradas nos atuais leitores de vídeo das plataformas *OTT* para dispositivos *Android*, com o objetivo de melhorar a experiência do utilizador ao assistir a conteúdo. As soluções propostas incluem uma *timeline* personalizável que apresenta todos os eventos-chave do video, permitindo aos utilizadores saltar para os momentos desejados facilmente. O utilizador também pode ver todos os eventos do vídeo sequencialmente com o próximo evento a começar após o atual terminar. Usando *crowdsourcing*, os utilizadores podem salvar eventos personalizados na *timeline* oficial. Além disso, propomos funcionalidades para ajudar os utilizadores a monitorizar o seu progresso no video. Após chegarmos ao produto final, realizámos testes de usabilidade com utilizadores. Estes testes receberam uma avaliação bastante positiva, com os utilizadores a mostrar entusiasmo com a integração destas funcionalidades nas atuais plataformas *OTT*, destacando uma experiência positiva e reconhecendo a experiência melhorada ao consumir conteúdo *VOD*.

Esta tese teve o apoio da Magycal, uma empresa renomada pela sua expertise em desenvolver plataformas *OTT* e aplicações *second-screen*.

Palavras-chave: Plataformas de Streaming, *Video On Demand*, Leitor de vídeo, Experiência do Utilizador, Interface do Utilizador, Interação Pessoa-Máquina, Design de aplicações moveis, Interação do utilizador

Contents

List of Figures

List of Listings

Acronyms

AVOD Advertising-based Video-on-Demand *(p. 7)*

HCI Human-Computer Interaction *(pp. vi, ix, 6, 9–11, 13, 15, 17)*

IDE Integrated Development Environment *(pp. vii, 35–37)*

JSON JavaScript Object Notation *(pp. vii, xii, 4, 18, 20, 21, 37, 40, 41, 56, 71)*

OTT Over-the-top *(pp. vi, ix, 1–9, 11, 18, 19, 29, 38, 71, 72)*

SDK Software Development Tool *(pp. vii, 35–37)*
SUS System Usability Scale *(pp. vii, 58, 64–67, 70, 72)*
SVOD Subscription Video-on-Demand *(pp. 7–9)*

TV Television *(pp. 1, 2)*
TVOD Transactional Video-on-Demand *(p. 7)*

UEQ User Experience Questionnaire *(pp. vii, x, xi, 59, 66–70, 72)*
UI User Interface *(pp. 4, 18, 20, 21, 38, 71)*
UX User Experience *(pp. vi, 6, 10–12, 17)*

VOD Video-on-Demand *(pp. vi, vii, ix–xi, 1–4, 7, 11, 17–20, 23, 24, 26–30, 37, 40–43, 45–49, 51, 52, 56–60, 62, 64, 71–73, 116, 117)*

1 Introduction

The introductory chapter of this dissertation serves as a presentation of the research conducted. In Section 1.1, the motivations for this thesis and the issues it aimed to address are explained. Section 1.2 provides information on the context in which this thesis was written and details the partnership associated with the project. Section 1.3 outlines the objectives of the thesis, describes the problems it aimed to solve, and proposes potential solutions. Section 1.4 discusses the contributions that this dissertation has made. Finally, Section 1.5 outlines the structure of this document.

1.1 Motivation and Problem Statement

Television (TV) has traditionally been a linear medium to watch content, meaning that viewers tune in to watch a certain program at a specific time. However, over the past few years, there has been a shift towards non-linear [1], on-demand forms of video content, where Over-the-top (OTT) media platforms like Netflix[1] and Video-on-Demand (VOD)[2] content come into play [2, 3].

This "cord-cutting" technology has gained popularity [4] in recent years due to several factors [5–7]. One reason is that these online platforms provide access to premium VOD content, and a superior experience to the user in a direct-to-consumer manner. Additionally, compared to traditional linear television where viewers are limited to the programs and channels that are available through their subscription or antenna, on-demand video platforms offer a wide variety of content [8], giving users more options and allowing them to discover new content that they may not have been able to access otherwise. Another advantage is the ability

[1] http://netflix.com

[2] VOD - content that is available in a platform for users to access and watch everywhere and anytime, instead of at a specific broadcast time.

to pause, rewind, and fast-forward through the media, which can be especially useful for sports events, where viewers may want to review a specific play or skip over breaks - with linear TV, viewers are at the mercy of the broadcaster and cannot control the pace at which they watch the content.

An additional factor that leads to the shift is the proliferation of connected devices and the convenience and flexibility that OTT platforms give us. In the past, if you wanted to watch TV, you had to be in front of a device at a certain time, and there had to be a physical connection, either through a traditional cable or an expensive satellite subscription. Now, with the widespread adoption of smartphones, tablets, and other internet-connected devices [9], it is possible to watch a variety of content ranging from series and fiction movies to sports events, with high-quality anytime, anywhere [10]. Also, with traditional TV if you miss a TV show you may not have the opportunity to watch it again, while in OTT platforms, the VOD content is available to watch anytime. This has made it much more convenient for people to watch on-demand video [11] and has contributed to the decline of linear TV watching [12].

The VOD market is witnessing significant growth, with projections indicating that it will reach a revenue of US$200.65 billion by 2025, according to Statista's VOD market evaluation [13] - Figure 1.1. This growth is reflected in the increasing downloads seen by major OTT media services, such as Netflix[3], Hulu[4], and Amazon Prime[5][14], as reported by Stream [15]. Younger demographics and individuals with busy schedules have already adopted these platforms as their primary source of video entertainment.

Despite the numerous advantages of non-linear content consumption through VOD platforms over traditional linear watching methods, users often encounter challenges when trying to customize their watching experience to specific moments or events within the content. For example, during a soccer match that can take up to two hours, a user might only want to watch the pivotal moments such as goals or fouls, or catch up on the second half they missed. Current video player interfaces used in OTT platforms lack the customization features necessary to cater to these preferences. This includes the absence of a way to easily navigate and watch specific moments within a VOD or to create and share personalized highlights with others. These limitations are significant concerns that this thesis addressed and tried to resolve by introducing new functionalities.

[3] https://www.netflix.com/
[4] https://www.hulu.com/
[5] https://www.primevideo.com/

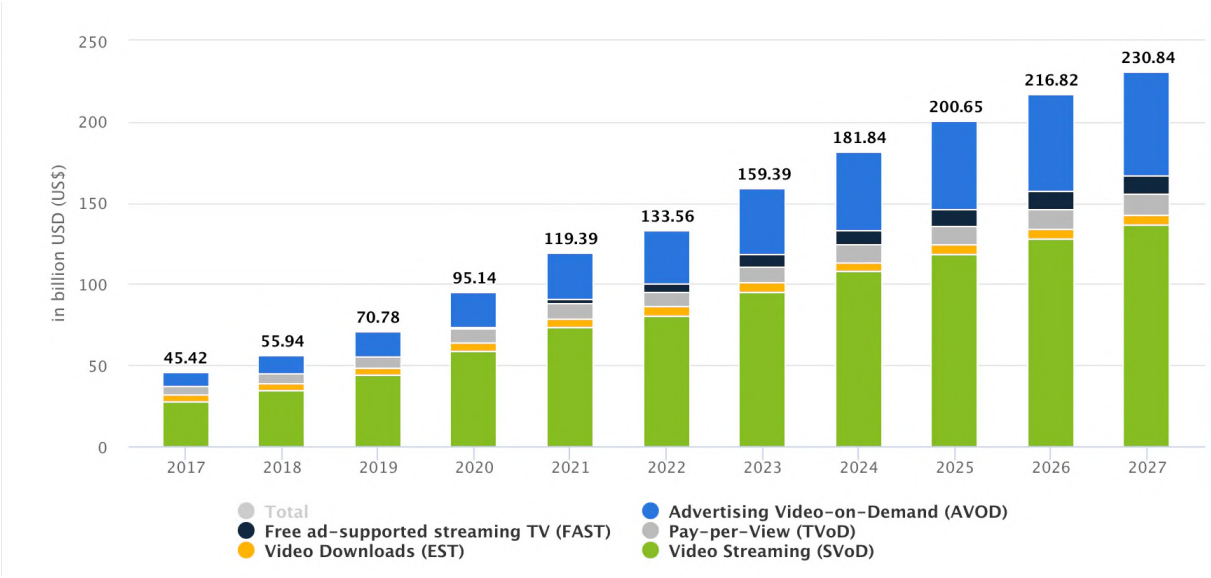

Figure 1.1: VOD market growth [13]

1.2 Context

This dissertation was developed as part of the Master's degree in Computer Science and Engineering at Nova School of Science and Technology, in partnership with the company Magycal[6]. Magycal, which was founded in 2012, specializes in creating OTT platforms and second-screen applications for various industries, including sports, movies, TV shows, and reality shows. In addition, it is known for incorporating innovative technology in its applications, such as the use of augmented reality and facial recognition.

The collaboration with Magycal proved invaluable, given the company's expertise in developing impactful OTT platforms such as Opto SIC[7], Panda+[8], and Sport TV[9]. Leveraging their extensive experience, Magycal offered highly beneficial insights throughout the development process. Furthermore, this partnership granted access to a variety of mobile devices, including tablets and smartphones facilitating real-time testing of the video player interface during its creation.

1.3 Goals and Solution

Despite ongoing modernizations, the existing video player interfaces in streaming services still fall short of providing a truly engaging and interactive user experience.

[6] http://magycal.com
[7] http://opto.sic.pt
[8] http://pandaplus.pt
[9] https://sporttv.pt/

This thesis outlines the creation of features to integrate into a video player interface designed to enhance the user-watching experience and user interaction with the VOD.

These features include a visual timeline that displays all of the video's main events, allowing users to navigate to crucial times with a single tap. This not only allows for easy content browsing but also encourages personalized interaction by allowing users to filter the event timeline based on specific parameters, allowing for a more tailored experience. To adapt to different videos, our proposed features adjust based on a JavaScript Object Notation (JSON) file, making it versatile for various content genres such as talent shows, reality shows, news, and more. The User Interface (UI) dynamically conforms to the provided JSON, ensuring a consistent and tailored experience for diverse events. Furthermore, we propose solutions that ensure ongoing enjoyment by allowing users to track their progress through the video, move back in time to a prior timestamp, and never lose their progress, allowing them to resume where they left off during the viewing experience. Taking personalization a step further, we also propose features that enable users to create personalized events within the video. Through the use of crowdsourcing, events marked by multiple users at a specific timestamp will integrate the video's official events timeline, being accessible to all users. This crowdsourcing mechanism operates by aggregating timestamps from multiple users, creating an average timestamp that best aligns with the correct moment in the video. This collaborative approach ensures accuracy and reliability in defining key events, enhancing the overall viewing experience for all users. It is important to understand that all of these features were introduced in a non-intrusive way that does not disrupt the content-watching experience.

1.4 Main Contributions

The main contributions of this project are:

- Identification of challenges in current OTT services related to the VOD content watching experience;
- Development of features to integrate into the video player interface of current streaming services, that solves those challenges and improves the user experience while watching VOD content;
- Provide knowledge, gathered with usability tests, on what features work or not and what improvements can be made.

1.5 Document Structure

The document is structured in six different chapters:

- **Chapter 1 - Introduction**: The first chapter provides an overview of the motivation behind the thesis, the context in which it was developed, the objectives it aimed to achieve, and the contributions it has made.
- **Chapter 2 - Background and Related Work**: This chapter presents a comprehensive literature review that covers the relevant studies and methodologies for the dissertation.
- **Chapter 3 - Conceptualization**: In this chapter, we explore how we conceptualized our solution. We outline the specific problems we encountered in current OTT video player interfaces and present the design iterations for each solution we developed, ensuring to incorporate expert feedback at every step. Following the development of the final design, we discuss the results of the usability test conducted.
- **Chapter 4 - Implementation**: This chapter delves into the details of the implementation process of each feature.
- **Chapter 5 - Evaluation**: In this chapter, we present and analyze the outcomes of the usability tests performed on the developed features.
- **Chapter 6 - Conclusions and Future Work**: The last chapter provides a comprehensive review of the dissertation, summarizing its conclusions and the scope of the work achieved. It also explores potential future improvements.

2

Background and Related Work

This chapter provides a detailed examination of relevant concepts to enhance comprehension of our work. First, we'll start by defining Over-the-top services, outlining their various subtypes, and discussing their rising popularity in Portugal. Afterwards, we'll delve into Human-Computer Interaction, Usability, Crowdsourcing in mobile applications, and User Experience, offering examples of related work in these areas.

2.1 Over-the-top (OTT) Platforms

Over-the-top services, as defined by the U.S. Federal Communication Commission, are video services that are broadcasted via the internet [16]. However, they go much beyond this simple explanation. In the realm of media and entertainment, OTT platforms refer to applications and services that provide audio, video, and other forms of media content over the internet, bypassing traditional distribution channels such as cable and satellite providers. These platforms typically use streaming technology to deliver content to users and can be accessed on a variety of devices, including smartphones, tablets, and smart TVs. Examples of these platforms include Opto SIC, Sport TV, Panda+, Disney+, YouTube, Netflix, HBO Max, Amazon Prime, Twitch, and many more. Some of these services offer a wide range of TV shows, movies, and other video content, either through a subscription model or on a pay-per-view basis, with some also providing live TV channels and sports programming [1, 2, 12, 17].

Several OTT services include extra features to improve the User Experience (UX). These options include live streaming, social networking, and customized recommendations. Through personalization, these platforms can take advantage of information about users watching patterns and preferences to propose video content in a way that is more relevant to them and more dynamic for them. As a

result, consumers have a more engaging experience and are more likely to stick around. A few other services also offer interactive elements, such as the ability to connect with friends and see what they are watching or to create and share playlists. Additionally, they can accommodate several languages and subtitles, making it simpler for consumers to access information in their preferred language. Some OTT platforms also allow the creation of original content that is only accessible on their platform, the downloading of content for offline watching, the live streaming of TV networks, sports competitions, and other content. According to user demand, they can also provide ad-free or ad-supported content, and they can monetize their service in a variety of ways, including through subscriptions, advertising, and pay-per-view [18–20].

In summary, OTT platforms offer users a more adaptable and customized way to access music, video, and other media content online. In response to the evolving needs and trends, these platforms are continuously advancing and transforming, incorporating new functionalities and services.

2.1.1 OTT Providers Video-on-Demand (VOD) Models

OTT streaming services use a variety of VOD business models, each of which has pros and cons [19–23].

The Subscription Video-on-Demand (SVOD) model, in which users pay a monthly or yearly price to access a wide variety of content. This particular business model is widely popular due to its ability to provide a consistent and steady revenue stream for the service. However, the downside lies in potential challenges to attract new customers, as they are required to pay a fee to access the content [17, 20, 23, 24].

The Advertising-based Video-on-Demand (AVOD) model, which depends on ad revenue to turn a profit, is another well-liked business strategy. Users are required to watch advertisements while streaming the content, which they can access for free. Since users do not have to pay to access the content, this approach has the major benefit of being able to draw in a sizable audience. Furthermore, because it doesn't necessitate a big investment in content, this strategy can be quite economical for the streaming service. The biggest drawback of this approach is that it might be challenging to monetize the audience because advertising revenue is frequently smaller than subscription revenue [18, 20, 22, 23].

The Transactional Video-on-Demand (TVOD) model, enables users to buy or rent specific pieces of content, including movies or TV series. Users just pay for the content they wish to see. Additionally, it enables the streaming service to

provide a variety of content since it can include both recent and classic films. The key drawback of this strategy is that it may not be as profitable as the subscription-based model because customers might not buy or rent enough content to produce a sizable profit [20, 22, 23].

Some OTTs use a combination of these business strategies, creating a hybrid model. For instance, a service might only allow subscribers to watch some content while making other content available for free with advertisements. The main advantage of this approach is that it can appeal to a wider audience and generate several revenue streams. In conclusion, these services can use a variety of business models to generate revenue, and each of them has its own advantages and disadvantages. The choice of model depends on the type of content offered, the target audience, and the company's overall business strategy [18, 19]. In Figure 2.1, it is possible to see and analyze what type of business models the most popular OTTs use.

Figure 2.1: Business Models of some OTT Providers [23]

2.1.2 Growth of OTT services in Portugal

In regard to this thesis is important to highlight the popularity of OTT video services in Portugal. SVOD platforms, services that provide customers with a selection of content for a monthly fee, have become more popular in Portugal with the advent of Netflix, NosPlay, and FOXPlay in 2015. The most noteworthy subscription streaming services to emerge in Portugal since then are Sport TV, HBO MAX, Disney+, Apple TV+, Amazon Prime, Filmin, SkyShowtime, and Opto SIC.

This kind of streaming service has recently been included in the monthly charge of some packaged deals by some communications companies, e.g Vodafone is including an HBO Max and Amazon Prime 3-month subscription in their cheapest deals[1], which allowed the percentage of Internet users who watched paid video on demand increased from 34% in 2020 to 42% in 2023 [25] - Figure 2.2.

As claimed by Marktest's[2] BStream study, one in four of Portuguese over the age of 15 subscribe to a streaming platform, with Netflix, Disney+, and HBO Max having the most subscribers. Amazon Prime Video has the highest percentage of male members (64%), Meo Filmes e Séries has the oldest subscribers, and among telecom carriers, Vodafone has the most SVOD service users (32%), while Nowo has the fewest (18%) [26].

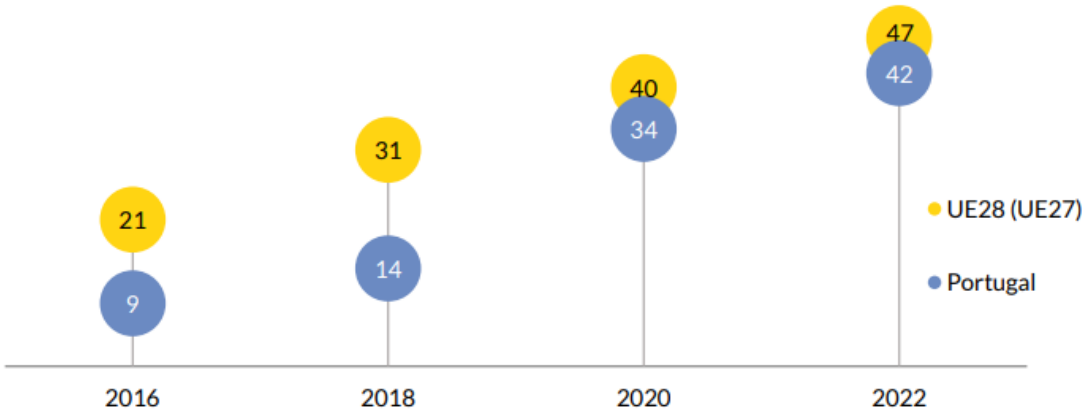

Figure 2.2: Proportion (%) of users watching paid video-streaming on demand on OTT [25]

2.2 Human-Computer Interaction (HCI)

In the digital age, human-computer interaction is a crucial concept. Every piece of software, hardware, and product we create involves human interaction, which requires deliberate and efficient design. Fundamentally, HCI is the study of how users interact with computers and how to create user-friendly, effective computer systems. It encompasses all aspects of interactive computing system design, evaluation, and implementation, always intending to improve human-computer interaction by taking into consideration the functionality, reliability, usability, and comfort of computer interfaces [27].

[1] https://www.vodafone.pt/pacotes.html
[2] https://www.marktest.com

Human-Computer Interaction, as its name suggests, focuses on the relationship between three main components: the user, the computer interface, and how they interact with one another - Figure 2.3. Designing an effective and usable product requires an in-depth understanding of the user, including their many sensory systems (sight, hearing, touch), the mental models they develop about their interactions, their national and cultural variations, and their learning styles. At its foundation, HCI seeks to strike a compromise between the ideal UX and the practical constraints of existing systems [28–30].

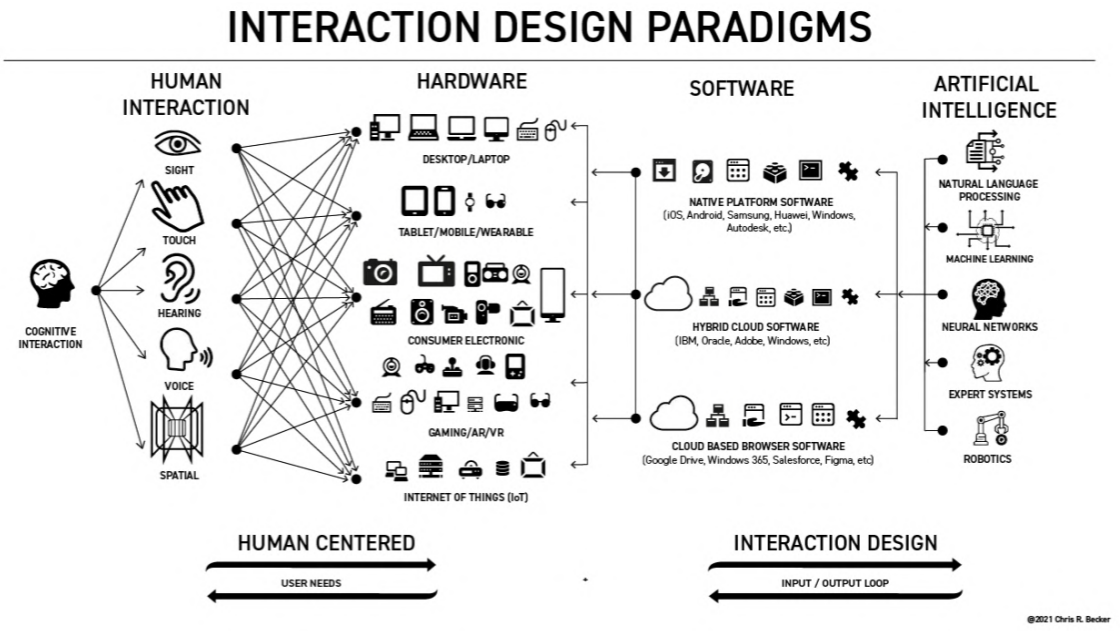

Figure 2.3: HCI Interaction Design Paradigms [31]

2.2.1 Usability and User Experience (UX)

Usability and User Experience are crucial ideas in HCI because they determine the success of a product in meeting users' needs and goals. Usability is the term that describes how simple it is for consumers to take advantage of a product to complete tasks and reach their objectives [32]. This comprises elements like the functionality, navigation, and user interface design of the product. Regardless of the user's level of technical expertise, a usable product is simple to operate and learn, making it available to a wide range of users [33, 34].

UX covers a wider variety of elements that affect how a product is used in general. It considers usability as well as additional factors including the product's visual appeal, utility, branding, and functionality. UX design aims to produce a product that not only satisfies the needs of the users but also improves their overall

experience. In the case of OTTs like Netflix, this essentially means how the user feels while browsing the catalog, navigating the menus, or watching the content [35]. In order to develop goods that are accessible, user-friendly, and appealing, usability and UX are both essential. Users are more likely to embrace and use a product effectively if it is useful and offers a positive User Experience, which increases user satisfaction and success in achieving goals [36].

As a result, usability and UX are essential HCI concepts since they guarantee that the requirements and expectations of consumers are met. In order to build components that are functional, accessible, and offer a positive user experience, designers must take into account both ideas.

In regards to this thesis, the focal point is to understand how human-computer interaction plays a critical role in mobile development and in the development of OTTs media players.

HCI is essential to mobile development since it focuses on designing interfaces that are intuitive and efficient for users to interact on small screens. It entails creating interfaces that are optimized for various input methods, including touch and speech, as well as various screen orientations and resolutions. For buttons and other interactive elements, this means using simple layouts, larger fonts, and larger touch targets. This concept is also crucial in the design of OTT platforms and video players, both of which require user-friendly interfaces for users to access and control VOD content. In OTTs, the goal of an interface is to simplify the process of finding and watching video content and provide a seamless and enjoyable watching experience. This experience requires a video player with an efficient and accessible-to-everyone design to make it easy for users to play, pause, fast-forward, and rewind video content, as well as adjust the volume and other settings.

2.2.2 Crowdsourcing in Mobile Applications

The concept of crowdsourcing, a term that combines "crowd" and "outsourcing" is the strategy of gathering a large volume of data from an audience/crowd [37]. While contributing to the enrichment of information, it also encourages a feeling of community. Introduced by Jeff Howe in 2006 [37], crowdsourcing became one of the most popular methods of gathering data and significantly impacted mobile application development, offering innovative features that provide a sense of community and improve user experiences.

An example of an mobile application that effectively leverages crowdsourcing

is Waze[3]. Waze is a navigation app that uses real-time data from its users to provide precise traffic information. Users contribute to the system by reporting their current location, speed, and any road conditions they encounter. This data is then shared with other users, creating a dynamic and responsive navigation system that can adapt to real-time changes. Figure 2.4 represents the flow of reporting a bad weather alert.

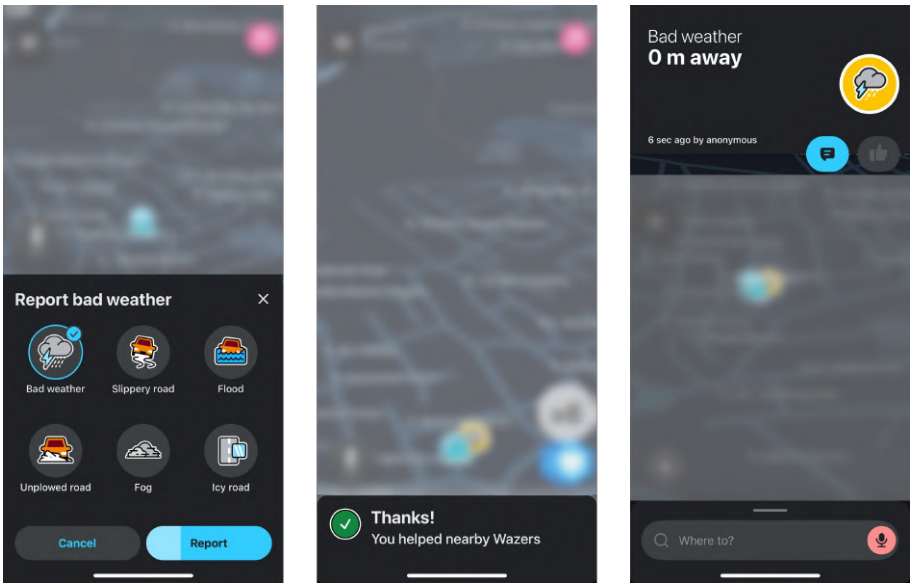

Figure 2.4: Action of reporting a bad weather alert on Waze

Google's reCAPTCHA[4] is another widely known example of crowdsourcing. This security service relies on crowdsourcing to differentiate between human users and bots. To achieve this, users are presented with challenges that are easy for humans to solve, but challenging for bots. The data collected from these challenges is then used to improve the accuracy of its algorithms, which in turn enhances the security of websites across the globe.

In conclusion, crowdsourcing has demonstrated its potential to significantly transform the mobile application development field. By incorporating users into the data gathering and verification processes, applications like Waze and Google reCAPTCHA have not only enhanced their services but also significantly contributed to a more secure and engaging user experience.

2.2.3 User Experience (UX) in Video Players

To improve the watching experience in video players, researchers are investigating new methods for creating a user-focused video player interface that can provide an

[3] https://www.waze.com/
[4] https://www.google.com/recaptcha/about/

immersive experience. This includes offering customization and filtering options and introducing new ways of browsing and navigating through videos. We have selected and presented some of these methods below.

Gyu Hong Kyung suggested a more user-centered video player interface that has the control buttons appear where the user's finger touches the display screen [38] - Figure 2.5. This study was successful thanks to an analysis of the currently available video players, the creation of a prototype, and a survey comparing the usability of the existing video players and the prototype introduced. The findings indicated that 98% of the users who participated in the usability tests preferred the proposed player, citing reasons such as "Users didn't have to stretch their fingers too far and they could manage the video player without worrying about an uncomfortable hold". Despite the interface suggestion being perceived as more convenient and user-friendly, we acknowledge that this solution could potentially be intrusive and disrupt the content-watching experience since it appears on top of the video player.

Figure 2.5: Interface proposed by Gyu [38]

Marco A. Hudelist et al. [39] introduced a new concept for video browsing on tablet devices. The issue with traditional video players on tablets is that they are not optimized for landscape orientation, which is the typical way users hold the device, making it difficult to interact with the video, especially with the video's timeline. The *ThumbBrowser* interface addresses this by providing new layouts and positioning for control buttons, such as a vertical seeker control resembling a timeline on the right side of the screen and a radial menu with additional playback features on the left side - Figure 2.6. A user study found that the *ThumbBrowser* was better at helping users locate target scenes with specified objects compared to a conventional standard player. The results showed that the *ThumbBrowser* was more effective, enjoyable, less frustrating, and just as easy to use as the standard player, while also requiring less mental and physical effort. However, one potential

criticism of the *ThumbBrowser* interface is its learnability. Given the unconventional layout and positioning of control buttons, users might find it challenging to adapt to the new interface. This could lead to a more challenging learning curve compared to more traditional video player interfaces.

Figure 2.6: Interface of the *ThumbBrowser* [39]

Klaus Schoeffmann et al. [40] proposed a video navigation method that uses context-sensitive swipe gestures to navigate forward and backward in videos replacing the conventional seekbar for timeline navigation. In this method, when the user touches the screen and gradually drags their finger to a direction, left or right, the current frame disappears and simultaneously the future frame or the past frame appears - Figure 2.7. The prototype's usability testing revealed that participants were able to effectively use the navigation features to find their desired scenes.

We acknowledge that a possible issue with this approach is that it relies on precise and consistent swipe gestures.

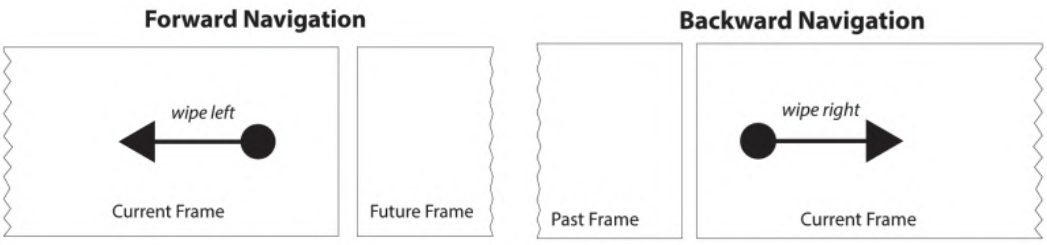

Figure 2.7: Multi-Touch Gestures Interaction [40]

Lukas Burgstaller et al. proposed a new video interface that tries to facilitate content navigation [41]. Unlike traditional video players where the user navigates

by dragging the timeline, this interface allows users to navigate through content by dragging the Scrubbing Wheel-like interface. This interface consists of a play/pause button in the center and a radial area for content navigation - Figure 2.8. Users can navigate the video by wiping clockwise or counterclockwise in the radial area, with fine-grained navigation performed by wiping in the outer circle, and coarse-grained navigation performed by wiping in the inner circle. The current playback position is displayed on a radial progress bar in the center. The results of the usability tests indicated that the new method of navigating videos is a promising substitute for conventional timeline navigation. Once again a potential critic of this approach is its learnability and reliance on precise and consistent swipe gestures.

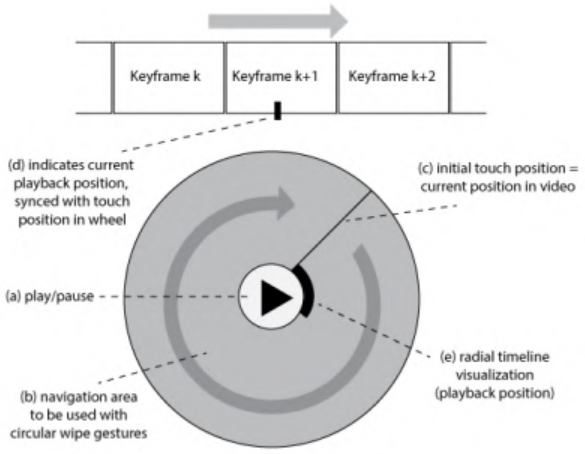

Figure 2.8: Interaction with the *Scrubbing Wheel* [41]

Previous researchers have suggested prefetching video segments to reduce seek latency, aiming to have users seek these prefetched segments, thereby achieving zero seek latency.

Axel Carlier et al. took a distinct approach by proposing to guide users towards prefetched segments by visually distinguishing them on the video timeline with a different color [42] - Figure 2.9. They hypothesize that users would prefer to seek segments with zero seek latency, leading to an improved overall video navigation experience. In addition to the prefetched segments, the video timeline also displays bookmarks which serve as annotations to important points in the video. A user study showed that this player interface resulted in up to 4 times more seeks to bookmarked segments and reduced seek latency by 40% compared to a commonly used video player interface today.

Adam Janin et al. designed a system called *Joke-o-mat HD* that allows users to

Figure 2.9: Proposed Video player with a timeline showing bookmarks and prefetched video segments [42]

navigate sitcoms by narrative themes, including scenes, punchlines, and dialog segments [43]. The system generates segmentation using expert annotators, automatic methods, or human-derived "found" data, such as fan-generated scripts and closed captions. The user interface includes filters for main actors and keywords, allowing users to search for specific moments in an episode and navigate directly to elements they remember from the past - Figure 2.10. With this interface, users would have more control over what they are watching. This proposed system is relevant to this thesis as we are also developing a feature in the video player where users can filter content by topics, enhancing customization and user experience.

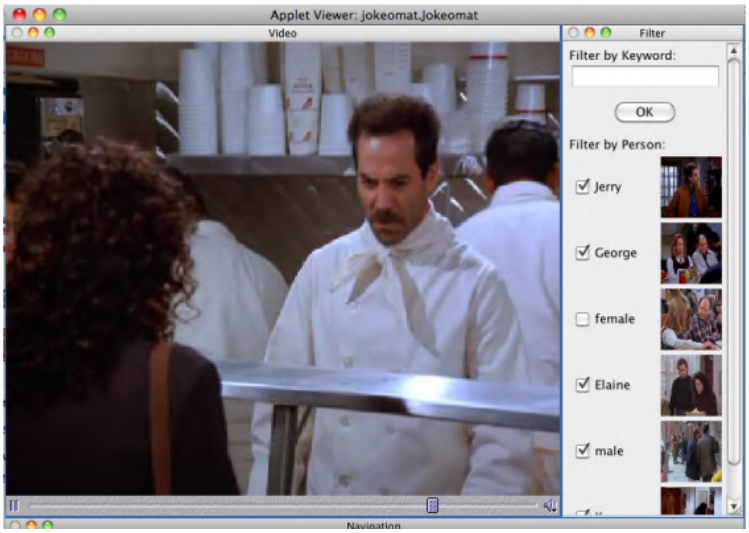

Figure 2.10: Proposed system with the Filter menu open [43]

Abir Al-Hajri et al. carried out a study to improve users' access and management

of their video-watching history by introducing two new visualizations called Video Timeline and Video Tiles [44] - Figure 2.11. These visualization methods enable users to view their video-watching history either in a timeline or grid format. The goal of the research was to create a user-friendly interface for managing video-watching history, including sharing clips and summarizing videos. To evaluate the effectiveness of the suggested visualizations, a prototype video viewer was designed. The usability tests indicated the efficiency and usability of the proposed visualizations in facilitating quick and successful navigation. A potential criticism of these visualization methods is their potential to overwhelm users with information. The Video Timeline and Video Tiles might present too much data at once, making it difficult for users to quickly identify and access specific content. This could lead to a less efficient navigation experience, as users might spend more time going through the visualizations than actually watching the content.

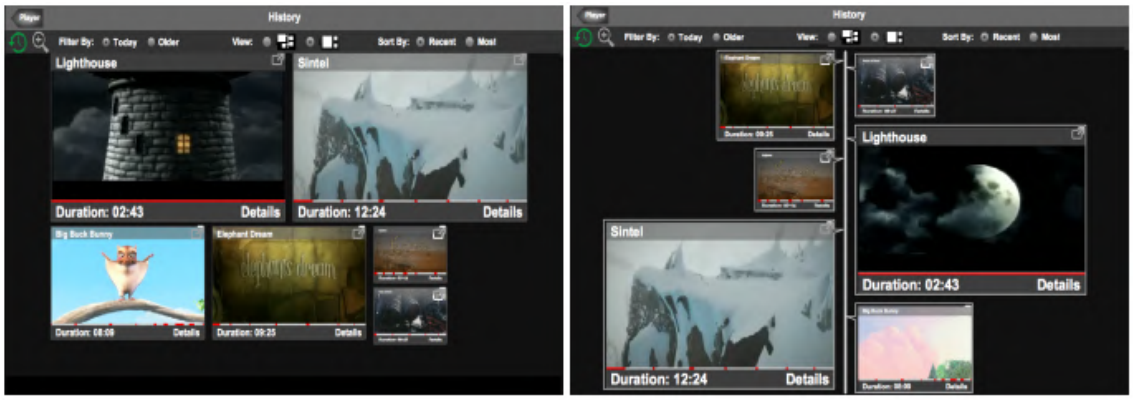

Figure 2.11: Proposed prototype with the user history visualized in one of two ways: as Video Tiles (left) or as a Video Timeline (right) [44]

In conclusion, this chapter has covered a large body of research that has significantly advanced User Experience (UX) in the realm of Video-on-Demand (VOD) content consumption. Although we have not incorporated these features into our work for this dissertation, these ideas have inspired us to develop new ways of engaging and interacting with content.

3

Conceptualization

This chapter introduces the concept, challenges, and innovative solutions aimed at improving the Video-on-Demand (VOD) content consumption experience. It begins by identifying specific problems encountered in the current VOD consumption experience, followed by outlining the solutions designed to address these issues, emphasizing the adaptability and scalability of these solutions to accommodate various types of VOD content. The chapter then explores the initial design, storyboards, the results of the first usability test, and the iterative design process that leads to the final design and prototype.

3.1 Concept

The concept of this thesis is to explore new features to integrate into current video player interfaces with the goal of improving the overall experience of consuming VOD content. Despite ongoing modernizations, the video player interfaces found in current streaming services continue to fall short of delivering a more engaging and interactive user experience. This thesis aims to improve the user experience while consuming VOD content by proposing a solution that is adaptable to any type of VOD content, ensuring a seamless and customizable watching experience. To allow for a dynamic User Interface (UI) adaptation based on the content type, the solution uses a JavaScript Object Notation (JSON) file, a highly flexible and scalable data format.

The initial step involved analyzing the most popular OTT video player interfaces, such as those used by Netflix, Disney+, Amazon Prime Video, and Sport TV, through direct use and discussions with advisors and Magycal. This analysis aimed to identify challenges or situations that could adversely affect the overall user experience.

The main challenge we encountered was the lack of a visual tool to identify the key events within a VOD. This limitation restricts users from focusing only on the crucial parts of a video, which increases the amount of time required to browse and find these critical moments. For example, in a cooking competition show, a user might be interested in watching only the important moments, such as the elimination of a contestant or the reveal of the winning dish, or even the recipe their favorite contestant did. In current video streaming platforms, the user would have to search in the seekbar for the moment he wanted to watch. To address this limitation, we propose introducing a visual timeline that shows all the video's main events, enabling users to jump to these crucial moments with a single tap. This feature simplifies the process of finding key moments in a VOD, which can be both challenging and time-consuming for users who prefer not to watch the entire video or just want to watch a certain key moment. To enhance the flexibility of viewing each event individually, we introduce a feature that enables the automatic and sequential playback of these events with just the push of a button. This allows users to experience all events in sequence, similar to a highlight reel of the video they are watching, while still offering the option to skip to the next event. If the user chooses not to skip, the next event will automatically start immediately after the current one concludes, providing a continuous viewing experience.

We also identified an issue related to the lack of options to filter the VOD based on specific topics. To address this, we introduce the concept of filters within the timeline, which allows users to filter the event's timeline based on their preferences, such as mood, characters, or specific content themes, always depending on the VOD type. After filtering the events, the users can see them one by one or sequentially and automatically.

Another issue we encountered was concerning progress tracking of the VOD. In current OTT's video players interfaces, there is no visual method to check the user's progress in the video while watching it. This issue often comes up when a user unintentionally clicks on the seekbar, causing them to jump to a different timestamp. Without a visual indicator or an easy way to return to the previous timestamp, users can become disoriented and unable to continue watching from their last timestamp. To address this challenge, we are proposing a solution that introduces a visual cue on the seekbar. This cue will show the furthest point the user has reached in their watching session, serving as a progress marker. Additionally, we propose a feature that allows users to undo a seek action, allowing them to return to their previous timestamp. These features allow users to track their progress throughout the video, move back in time to a prior timestamp, and never lose their progress thereby maintaining continuity and facilitating an easier

viewing experience.

We also took notice that currently there are no interactive elements on the video player that enable users to mark or highlight specific moments that stand out to them and share parts of the video with other users directly on the player. Our video player addresses these challenges by adding a feature that allows users to create and highlight personalized events within the video. Using crowdsourcing, events of the same type, identified by multiple users at a particular timestamp will be incorporated into the video's official events timeline, accessible to all users. This crowdsourcing mechanism aggregates timestamps from various users within a 30-second range, establishing an average timestamp that aligns with the correct moment in the video. This collaborative approach ensures accuracy and reliability in defining key events, enhancing the overall viewing experience for all users.

In summary, we propose a video player interface that aims to enhance the user's overall experience by incorporating all these features: a key events timeline with filters by events type, a way to better track the user's progress in the video, and a crowdsourcing mechanism to highlight events with others. These features not only simplify the content navigation process but also make the watching experience more personalized and interactive.

It's important to note that the features discussed in this chapter are the final iterations of our approach to addressing the challenges presented. Through a series of iterative design processes and usability testing, we identified several limitations with our initial approach and solutions. This process can be seen in section 3.2. The final solutions presented in this section are the result of these refinements.

To adapt the User Interface (UI) of our proposed features based on the video type/genre, we required a structured JSON file to be provided to our video player screen. This method enables us to dynamically modify the UI elements following the details specified in that JSON file, ensuring that the interface is specifically designed for each distinct video content type. If, for instance, the JSON file contains the VOD type parameter equal to "game" then the UI would adapt to a soccer match.

The structure required for the integration and adaptability of the player is illustrated by Listing 1.

In the Implementation chapter 4, we will explain each of the different parameters and present the final JSON used to implement and test our features.

```
{
  vod_type: String,
  vod_filters: [
    { name: String, icon: String },
    ...
  ],
  custom_filter: Boolean,
  custom_filter_types: Array.ofType(String),
  crowd_sourcing: Boolean,
  crowd_sourcing_types: [
    { name: String, icon: String, color: String },
    ...
  ],
  [vod_type value]_info: {
    % fields depending on the VOD type
  },
  vod_events: [
    {
      type: String,
      seconds_start: Long,
      seconds_end: Long,
      minute: Integer,
      icon: String,
      ...
    },
    ...
  ]
}
```

Listing 1: JSON structure required by our interface

3.2 Design Process

This section discusses the iterative and collaborative process used to create the design of the features investigated. It describes the initial vision with the presentation of storyboards representing each feature, execution of a usability test before the final iteration of the design, and subsequent iterations that resulted in the final proposed UI for each feature. After reaching the desired design, we evaluated it with another usability test of our Figma prototype, obtained very valuable feedback from experts in developing highly usable interfaces, and became ready to implement our features. Through this method, we hope to demonstrate the solution's progress, emphasizing the importance of experts feedback and testing

in improving the design to ensure that it effectively fulfills the demands of end users. After each design iteration, we held a discussion where we evaluated the designs and identified issues that needed improvement. All of the storyboards and designs were done using Figma[1], a web application for interface design.

3.2.1 Initial Design Iterations and Storyboards

The design process started with a collaborative brainstorming session involving Magycal advisor and the thesis advisor, which led to the formulation of several ideas and several different designs. These ideas were then conceptualized through Figma high-fidelity prototypes and storyboards, providing a visual representation of the proposed features and their integration into the user interface. After the investigation, we decided to approach the following ideas.

The first idea was to have a way to show the user the key events of the video involving showing them directly on the video player seekbar. This feature would allow users to visually identify key moments within the video content directly on the video player, enhancing their ability to navigate through the content more efficiently. We also wanted to have a direct way for the user to return to his previous timestamp after clicking on an event, a way for the user to keep track of his progress in the video.

The second main idea was to allow users to save moments locally. This feature would allow users to bookmark specific timestamps within the video, facilitating easy access to these moments at a later time with the help of a menu.

The final idea was to add filters to the video allowing the users to filter throughout the whole video by some topics. This feature would allow users to customize their viewing experience by selecting content based on a certain topic. The selected content then would be represented on the seekbar of the player by a different color, showing the desired content.

To represent those initial ideas, a few designs were made. These designs went through several iterations and each one will be represented by a subsection. In each subsection will be presented some storyboards that act as visual stories that show how users will use our solution, making it easy to see how the users will navigate and engage with the content.

3.2.1.1 First Iteration

In the first iteration of our design, and regarding the first idea, we designed a storyboard - Figure 3.1 -, that shows how the user would interact with the

[1] https://www.figma.com/

signalized events on the seekbar. The storyboard depicts a video player with a seekbar that has distinct markers for each key event. Each marker is represented by an icon that provides a clear visual cue for users to jump to these moments in the video. When a user clicks on a specific event marker, the video automatically jumps to that event. Additionally, users could go back to the timestamp they were at previously by using the "Previous Timestamp" button. However, after discussing the design with the advisors and experts, we realized that using only icons for each event would be confusing for users. For example, in a soccer match, it would be difficult to determine which team did what event or what player was responsible for the event. Furthermore, if the VOD had a lot of events, the seekbar would become crowded and difficult to understand, with events overlapping each other. Hence, we recognized the need for a better design solution.

Figure 3.1: First Iteration - Storyboard of interaction with key events and progress tracking

Concerning the feature to allow users to filter the content by topic, we represented that interaction on the storyboard of Figure 3.2. This storyboard illustrates a player interface that contains a filtering button that, when clicked, opens a menu listing various topics. Users can then select various topics to filter the VOD content accordingly, ensuring that they are presented with content that aligns with their

preferences. After filtering the content, the moments that fit the filter criteria will be displayed on the seekbar as white spaces. However, after reviewing the design, we understood that when filtering by a frequently mentioned topic, such as a character that appears in every scene, it can be difficult for users to identify the parts where that topic is most prominent. This is because the timeline can become cluttered, making it more challenging to pinpoint the most relevant moments.

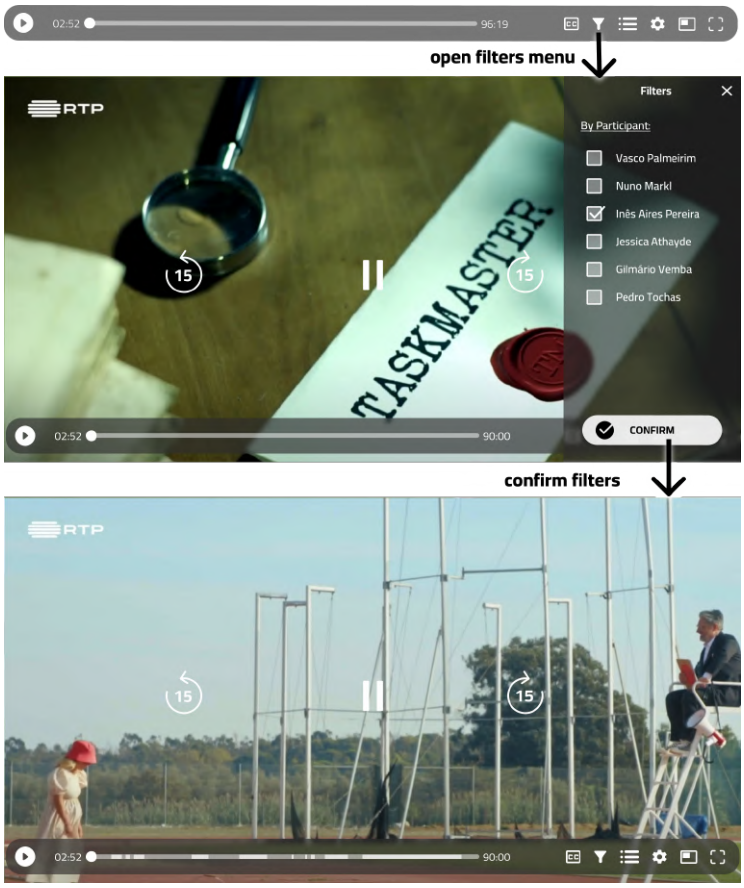

Figure 3.2: First Iteration - Storyboard of interaction with VOD filters

Regarding the idea of allowing users to save moments locally, we started by adding a menu item on the video player interface that when clicked, opens and displays all the key events of the VOD, along with any user-created ones. To create a custom event, the user can click on the "Create custom moment" button that would make a dialog appear where the user could enter the desired timestamp (by default, it uses the current timestamp of the player), and add a title related to that moment. Once the moment is saved, it appears on the menu with the video frame of the timestamp as the image, alongside the title and timestamp. The menu also allows the user to click and go directly to that moment of the video. Additionally, we added a "Previous Timestamp" button, that would appear when the user clicks an event, to allow the user to return to their previous timestamp,

without losing their progress on the video. After reviewing the design alongside the advisors and experts, we saw that there could be a better division between the video events and the user-created ones, because, with this design, if the video has a lot of events, the user would have to scroll to the bottom to see the ones that were created locally. Also, we identified that the create-moment dialog could be more efficient. This interaction can be seen in the storyboard of Figure 3.3.

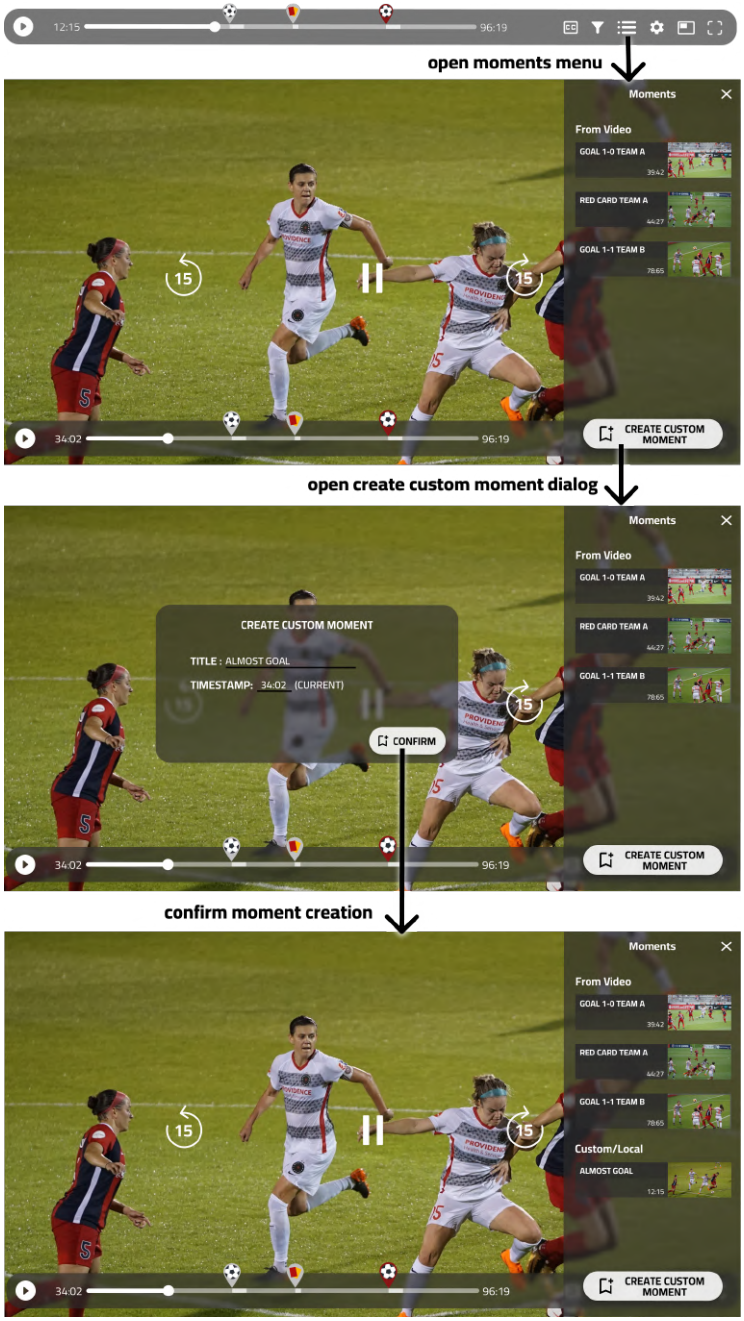

Figure 3.3: Storyboard that represents the feature of creating and going to a custom moment timestamp

3.2.1.2 Second Iteration

After realizing that there were some issues with the initial design, we decided to create a second iteration to address them. Specifically, we wanted to address the issues in each feature of the design by proposing a different design for each one.

Firstly regarding the timeline of events, our goal was to make the events in the seekbar more informative and recognizable to the user. To achieve this, we added a label below the icon and a color to each event. For example, on the storyboard - Figure 3.4 - that represented the interaction in a soccer match, the color of the event represented the team's color. This made it easier for users to identify the events and understand the information they were conveying. We also changed the progress tracking method, and now, users can undo a seek action or resume watching a video from the farthest point they previously saw using two new buttons on the bottom of the video player interface. This makes it easier for users to keep track of their progress while watching the VOD. This new interaction can be seen on storyboard 3.4.

Figure 3.4: Second Iteration - Storyboard of interaction with key events and progress tracking

We have also added a new feature that allows users to jump to the next event after the current event has finished. To achieve this, we have introduced a new layout that displays details of the upcoming event. Users can click on the "Up Next" layout to jump directly to the next event. This layout can be seen in Figure 3.5. Alternatively, they can wait for a countdown of 5 seconds after the current event ends to jump to the next event, or even cancel the action.

Figure 3.5: Second Iteration - "Up Next" layout.

In the context of video filtering, we have opted to focus on specific events within the video, rather than applying filters across the entire video content. Users will now have the ability to filter through various types of events and choose the ones they want to display on the seekbar. In the storyboard, which is shown in Figure 3.6, the new interaction is shown. The user opens the filter menu, which now includes various event types available within the video. Users have the option to either reset all filters, returning the seekbar to its original state with no events highlighted, or apply new filters and adjust the seekbar accordingly. This approach allows users to customize their viewing experience by selecting and displaying only the events they are interested in.

Figure 3.6: Second Iteration - Storyboard of interaction with VOD filters

Regarding the save moments locally feature, and to enhance the organization of the saved moments menu, we have categorized moments into two distinct sections: moments captured from the video and moments that users create themselves. Users can now easily locate a specific moment by its title using the search bar, which facilitates easy access to their saved content. To create custom timestamps, users can navigate to the "Custom Moments" section and select the "Create Custom Moment" button. Once users input the necessary details and confirm the creation, the new custom moment is seamlessly integrated into the "Custom Moments" section within the saved moments menu. This structured approach not only streamlines the process of saving and accessing moments but also ensures a more organized and user-friendly interface. The creation moment interaction is represented on Storyboard 3.7.

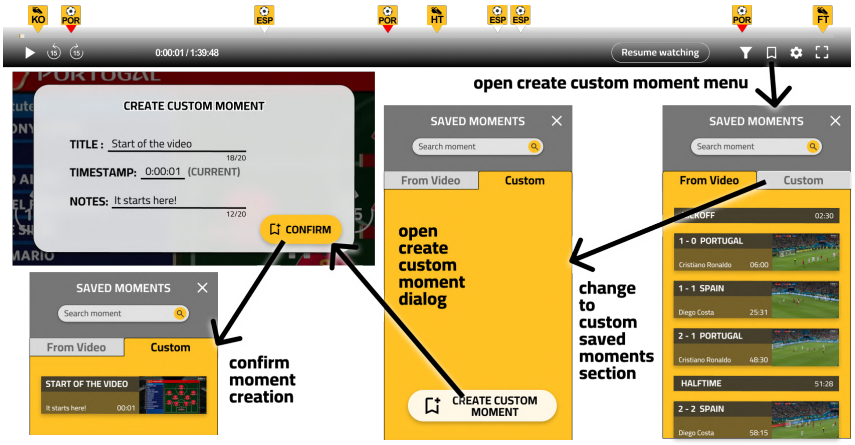

Figure 3.7: Second Iteration - Storyboard of the interaction of creating a custom moment

3.2.1.3 Design review and feedback from experts

After finishing the second iteration of the design, we held a discussion to gather feedback from Magycal experts and better understand whether the features met their expectations. The feedback received was mostly positive, but a few issues were identified. One of the primary issues was the possibility for events to overlap with each other in the VOD, especially when there are several events, leading to a cluttered and less distinguishable seekbar. Additionally, the design was criticized for seeming more like a web application than a mobile application, which could affect user perception and interaction. Regarding the progress tracking feature, some users encountered issues with the resume-watching and undo functionalities. Specifically, users misunderstood the undo feature, thinking it could reverse any action, including creating a custom moment or undoing a filter, rather than just

undoing a seek action. To address these concerns, we gathered all the feedback from both positive and negative aspects and developed a new solution to overcome these issues.

3.2.2 Final Iteration

Following the design review, we made a fundamental change in our designs and concepts. Magycal discussed the features with Sport TV, their award-winning sports OTT, and they expressed interest in implementing them into their mobile application video player. With this, we had to take a completely different approach, rethinking our concepts, leading to a very different design.

Initially, we had considered showing the VOD events directly on the player's timeline. However, in response to feedback, we opted for a distinct approach: introducing a separate, vertical timeline where the user could see every key event individually or in a sequential manner, similar to a highlights video. This timeline is designed to more effectively display all events in chronological order, with each event offering more detailed information. The filters remain exclusive to the timeline events, allowing users to filter the vertical timeline by different event topics. Users can then view these filtered events either individually or sequentially, with one event starting right after the previous one ends.

We also transitioned from allowing users to save moments locally to a more collaborative approach. Instead of individual users saving moments, we introduced a feature that leverages crowdsourcing to save moments on the timeline. This shift not only enhances the user experience by making the content more personalized but also fosters a sense of community among viewers by encouraging shared moments and experiences. We also changed our progress tracking system in order to simplify it for users. The "resume watching" button has been removed and replaced with a more straightforward method. Now, users can see their progress directly on the seekbar, where the portion of the video that they have watched is shown in a different color. This visual cue provides immediate feedback on their viewing progress and enhances the user experience. Additionally, we also made some changes to the undo feature. The button is still located at the bottom of the player, but it will only become active when the user performs a seek action. This aims to clarify that the undo feature is only applicable to actions such as seeking to a different timestamp, thus avoiding confusion. To make the interface cleaner and more user-friendly, we have rearranged the functionalities buttons into two main buttons. These buttons are located at the top of the video player, with one button dedicated to all timeline-related features, including the timeline itself and

filters, and the other button focusing on the save-moment crowdsourcing feature. This rearrangement reduces clutter and aligns with a more app-like look.

To better show these new interactions and ideas we created new storyboards that best illustrate each new interaction.

The initial storyboard, Figure 3.8, illustrates the interaction with the new vertical timeline. This new timeline section presents to the user more information about the video, such as images and labels. Each event has additional information, and in the example of the storyboard, a soccer match, an event can have the photo, name, and team of the soccer player, an icon representing the event type, and the event's timestamp. The user can access the timeline from the video player, scroll through the list of events, and tap to play the desired event. While watching an event, a layout appears at the top of the player, showing the user exactly what event he is seeing and more information about it. This design not only enhances the user's understanding and navigation through the content but also improves the viewing experience by providing more information about the VOD.

In the storyboard 3.9, we illustrate how users can apply filters to their timeline, allowing them to see only the events that match their specified criteria. To initiate this process, the user would tap the filter button located on the top left of the timeline section. After clicking the filter button, a variety of filtering parameters would become visible. For the example provided in the storyboard, which focuses on a soccer match, the user would select the desired parameters (e.g., a specific team, a particular type of event related to that team, and a player associated with the selected team and the selected event types). After making their selections and saving these preferences, the timeline would then display only the events that meet the user's specified criteria. Users have the option to tap on an individual event to watch it or tap the "Play Filtered Moments" button to play all the selected moments sequentially, enhancing the overall watching experience by allowing for more focused and personalized content consumption.

Figure 3.8: Final design - Storyboard of interaction with new vertical timeline and its events

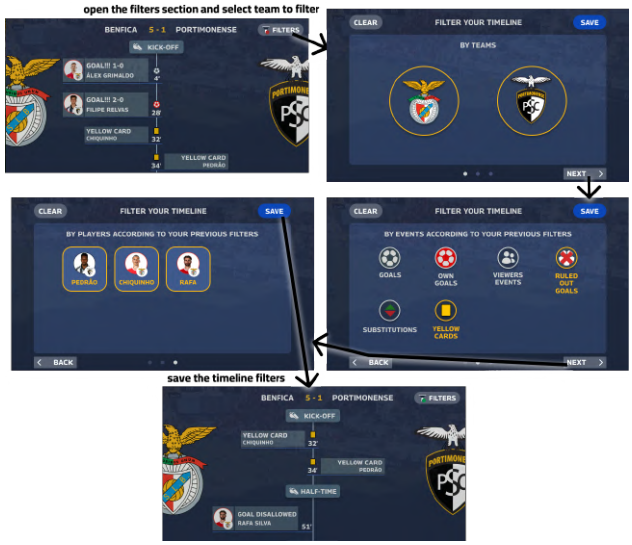

Figure 3.9: Final design - Storyboard regarding timeline filtering

On the storyboard 3.10 the interaction of watching the events sequentially is represented. By tapping the "Play Moments" button, all events begin to play automatically, with each event starting after the previous one ends. Users also have the option to skip to the next event by clicking the "Go to next event" label. This interaction streamlines the viewing process, allowing users to either enjoy the events in their entirety or navigate through them at their own pace.

Figure 3.10: Final design - Storyboard representing the action of watching events sequentially

To illustrate our new idea for the tracking process we created the storyboard 3.11. In the storyboard, we can see that after manually seeking to a different timestamp, users are provided with an activated Undo button. This button allows them to go back to their previous timestamp, providing a simple and intuitive way to revert to the previous timestamp. Users can also monitor their progress through a visual cue on the seekbar, which now indicates progress with a distinct color. This feature enhances the user experience by offering a clear visual representation of their progress, making it easier for them to track and navigate through the content.

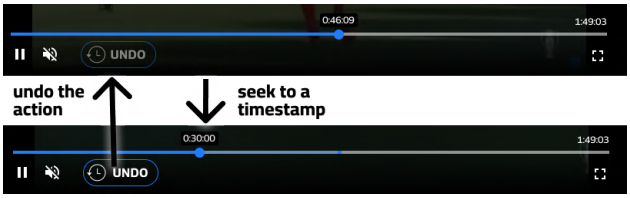

Figure 3.11: Final design - Storyboard representing the progress tracking

In regards to the new interactive crowdsourcing saving moments feature, we have created the storyboard 3.12, which outlines the process. To highlight a moment, the user would need to click on the "Save Moment" button at the top of the video player. A dialog would then appear, providing information about the crowdsourcing mechanism as well as a list of event types to choose from. Once the user has selected the event type and the related team, one of two things can happen. If a certain number of people have already highlighted that same moment with the same type and team around the same timestamp, the user would be able to see that moment on the official timeline. If the number of users highlighting the moment does not meet the threshold required to add the event to the timeline, the user will see a message thanking them for highlighting the event, but no event will be created at this time. Once the threshold is reached, the user will receive a notification, and the saved moment event will appear on the timeline.

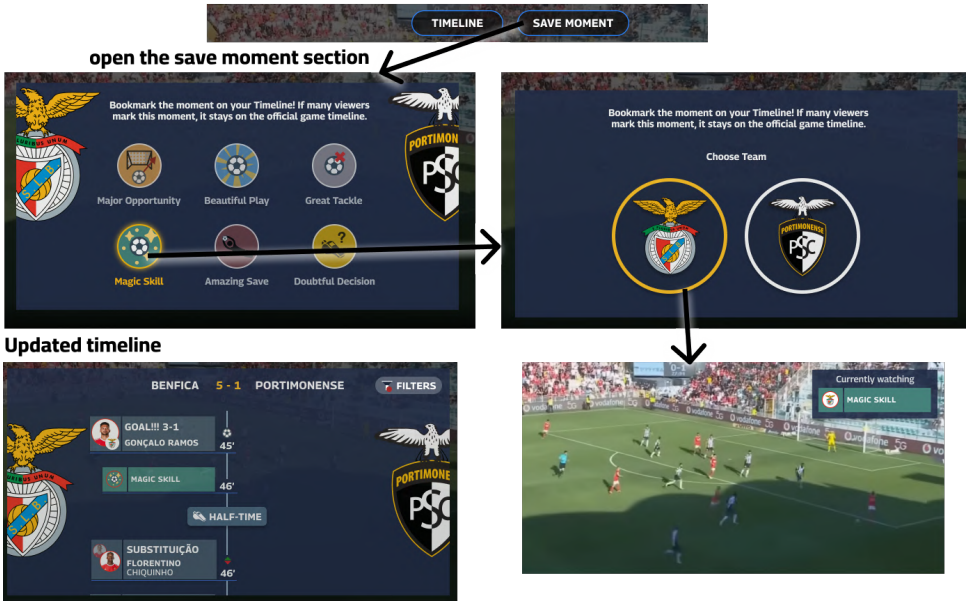

Figure 3.12: Final design - Storyboard of new interactive save moments action using crowdsourcing

3.2.2.1 Usability test of Final Iteration design and feedback

After we reached the final design it was crucial to have user feedback. With this in mind, we decided to do a small usability test only with the potential prototype users. The test consisted of simple tasks where the testers would have to navigate through events, filter them, and track their progress in the video while being observed and narrating their thoughts aloud. The script for these tasks is available in Annex I. Following each test, we collected feedback through Google Forms,

accessible in Annex II. The feedback was largely positive, though we identified several areas for improvement:

- The "Undo" action could be more intuitive if labeled as "Return to [timestamp value like 15:44]";
- The broom icon for clearing filters was unclear; it would be more effective to have "Clear All" and "Select All" buttons;
- Players filters should include the type of event each player participated in, along with the event's icon;
- Add pre-defined topic filters in the filters section instead of having the user always customize a filter.
- The "Save moment" button that opens the section to highlight a moment should be renamed to "Highlight Moment" for clarity.
- The seekbar's different color for progress tracking might confuse users, as it typically indicates prefetched content in other video player's interfaces.

After conducting several rounds of design revisions, taking into account feedback from experts, and conducting usability tests, we have reached a solution ready for development. This development is discussed in chapter 4.

4

Implementation

This chapter explores the technical aspects of our dissertation, focusing on the Android SDK and the used IDE for the project, the libraries used to implement our features, the integration with *THEOplayer*, and the development of each functionality.

4.1 Android Software Development Tool (SDK) and Android Studio Integrated Development Environment (IDE)

The Android Software Development Tool (SDK) is a collection of development tools required for Android app development[1]. This kit includes libraries, APIs for developers to interact with the Android system and hardware, a debugger, emulators, sample source code, documentation, and tutorials for the Android operating system. To acquire the Android Software Development Tool (SDK), we used the Android Studio Integrated Development Environment (IDE)[2]. This IDE, along with the SDK, offers a robust and comprehensive suite of tools for building, testing, and debugging Android applications. It features tools like a code editor with syntax highlighting and refactoring tools, a layout editor for efficient user interface design, a debugger, and an emulator capable of simulating a variety of Android devices, including tablets and TVs. Additionally, Android Studio supports integration with version control systems like GitHub and includes a built-in system for automating the compilation and testing of apps, known as the Gradle Build System.

[1] https://developer.android.com/tools
[2] https://developer.android.com/studio

For our implementation, we used the Android SDK with the Android Integrated Development Environment (IDE) as tools for developing our application because they serve as a support toolkit that provides a wide array of features to help on every phase of the development process. The integration with GitHub enabled us to keep track of our changes effectively and ensured the safeguard of our project.

4.1.1 Developing Android Applications with Java

We have chosen Java as the primary language for developing the interface of our video player, as it is widely used in Android app development. Java's object-oriented nature, along with its extensive range of frameworks and libraries, makes it an ideal choice for Android development. When writing Java code for Android, we need to create classes that extend the Android framework's components, such as Activities, Fragments, List Adapters, Services, Content Providers, and Broadcast Receivers. These components serve as the fundamental building blocks of an Android application, defining its behavior and user interface.

4.1.2 Android Development: Activities

In Android development, the fundamental component is the Activity [3], which represents a single screen with a user interface [45]. Activities are essential components of an app and are responsible for providing the user interface and handling user interactions. An application can consist of multiple Activities, with the Activity specified in the Android manifest file as the Main Activity serving as the application's entry point. The Android development paradigm revolves around the lifecycle of Activities, which consists of various states that an Activity transitions through during its lifetime. Each lifecycle stage has a corresponding callback that developers can override to manage resources, execute specific actions, or initiate processes, ensuring the application operates correctly under different conditions. The callbacks used on our application are *onCreate()*, where the activity is created but not yet visible to the user, *onResume()*, where the activity came from background and is now active, the *onPaused()* callback where the current activity is paused and the *onDestroy()* callback where the activity is destroyed and removed from memory. These lifecycles can be seen in figure 4.1.

[3] https://developer.android.com/reference/android/app/Activity

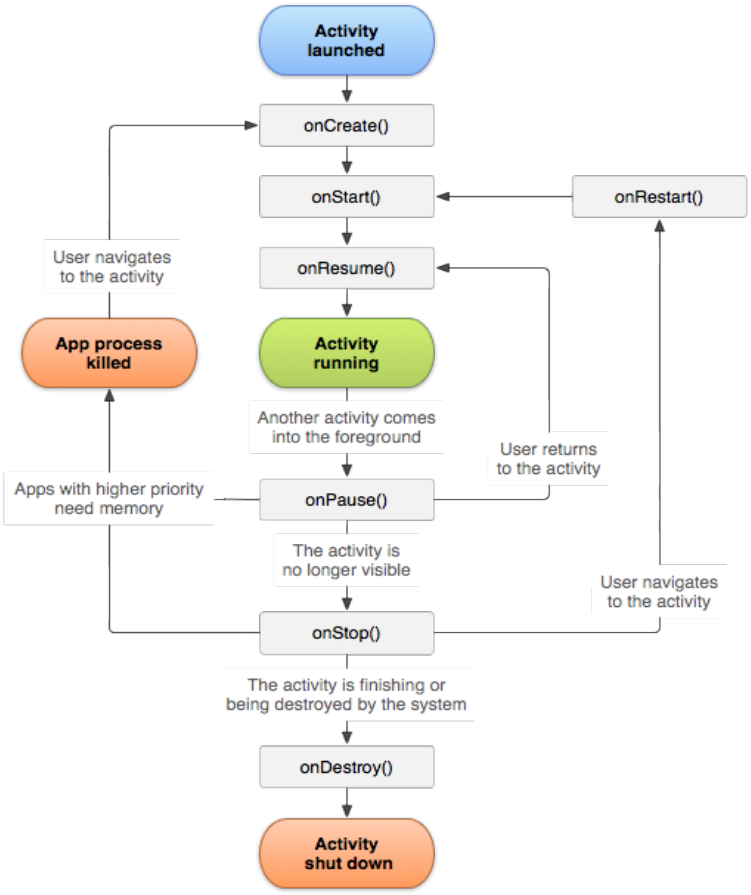

Figure 4.1: Activities Lifecycle [46]

In our implementation, we developed a single Activity named *VODActivity*, which encapsulates all the logic required for our features. This Activity is initiated in landscape mode and is solely dedicated to the video player and its interface. Regarding the lifecycle callbacks, on the *onCreate()* lifecycle listener, we load all the VOD information from a JSON file and convert it into our model using the GSON library. We then initialize the video player and its interface and begin to play the VOD. When the app is paused, we must also pause the video player and its listeners. *THEOplayer* provides functions such as *.onPause()* and *.onResume()* that we can call in respective callbacks that internally adjust the player's state accordingly. In the *onDestroy()* stage, we have to destroy the video player, using *THEOplayer*'s function *.onDestroy()*, and release it from memory.

4.1.3 User interface development using XML

XML is a markup language used in Android development to define the structure and appearance of our layouts. Each layout is constructed from a hierarchy of views organized within view groups, visually illustrating the arrangement of

elements on the screen. Views can encompass a wide range of UI elements, from simple buttons and text views to complex nested view groups that encapsulate multiple views [47]. This hierarchical organization helps in the development of dynamic and responsive user interfaces that adapt to various screen sizes and orientations.

Android offers several types of layouts for organizing UI elements on the screen, each having a specific purpose and capable of being nested within others to construct complex UI structures.

For our application, we used three distinct types of layouts. The *LinearLayout* arranges its children in a column or a row, making it ideal for static lists of items. The *ConstraintLayout* enables developers to define constraints and relationships between views, offering flexibility in layout design. The *RelativeLayout* supports the creation of layouts where elements are positioned relative to each other, useful for more complex UI arrangements.

To develop our feature's layouts, we created six XML files. The main XML file has the video player and its entire interface, including the timeline of events, filters, progress tracking, and the highlight moments layout. The remaining five XML files were specifically created for list items, as our lists are dynamically generated, needing a layout for each item.

4.2 Libraries

In the development of our project, we required external libraries, and in this section, we will detail each of these libraries.

4.2.1 *THEOplayer*

THEOplayer[4] served as the primary external library required for developing our features, given that our project focuses on video player interface capabilities, we needed a base video player to incorporate these features. *THEOplayer* was selected for the project due to Magycal's prior familiarity with it and its current usage on the OTT platform Sport TV[5]. Magycal has given authorization for the using of THEOplayer in this project, even though it is a premium player that incurs a fee for access.

THEOplayer is a cloud-hosted HTML5 video player that can deliver a single standardized format not requiring users to install third-party plugins on their

[4] https://www.theoplayer.com/
[5] https://play.google.com/store/apps/details?id=pt.sporttv.app

devices [48]. As a result, HTTP Live Streaming video and audio streams are easily played on all browsers across a broad spectrum of devices, including desktops, tablets, and televisions. This media player provides a variety of features such as 4K video playback, playlist creation, video markers creation, Chromecast, preview thumbnail frames, change playback speed, picture-in-picture, 360° view, virtual reality, subtitles, closed captions and many more [49]. It has its API with methods, properties, and events with easy integration and use[6].

To integrate this library, it was necessary to create a license for the player and add it to the Android manifest file. Also, we had to download the SDK file from the *THEOportal*[7] and insert it in the lib folder of our project. After adding in the needed dependencies in our Android project Gradle file, the player was ready to use. For styling *THEOplayer* components such as the seekbar color, the fullscreen button, and others, we used CSS. Additionally, since there's no player listener to detect the visibility of the player interface, we used JavaScript to facilitate communication between the main Java application and the player HTML interface. Through Javascript messages, we were able to determine the appropriate times to display our additional interface, ensuring it only appears when the video player's main interface is visible and is hidden as necessary. To integrate the *THEOplayer* on our XML layout, we created a *THEOplayerView*, called *theoPlayerView*, that automatically creates our video player in the layout. To modify the look of the video player we had to include an attribute *app:cssPaths* in the View that pointed to our CSS file, and to include the HTTP Interceptor for messages, we had to include in the View, the attribute *app:jsPaths* that pointed to our Javascript file. In our *VodActivity*, we obtained the *THEOplayer* instance by calling a function from *THEOplayer's* API on our *theoPlayerView*, specifically the *theoPlayerView.getPlayer()* function. This function provided us with the Player instance, which we could then use to setup the player, start/stop it, or even seek to a timestamp.

After obtaining the player instance, we initialized it by setting its source. For testing purposes, we used a soccer match video to test and implement our features. Once the source was set to our soccer match, we were able to monitor various player events, which were crucial for our implementation. These events included the *PlayerEventTypes.ERROR* event, where we would display an error message if a player error occurred; the *PlayerEventTypes.TIMEUPDATE* event, which is triggered when the current playback position changes as part of normal playback; and the *PlayerEventTypes.SEEKED* event, which is fired after the player seeks to a different timestamp. This last event could occur either through manual seeking with the

[6] https://docs.theoplayer.com/api-reference/web/theoplayer.md
[7] https://portal.theoplayer.com/login

seekbar or by seeking to a specific timestamp when seeing an key event. Identifying and distinguishing between these two methods of seeking presented a challenge. In the chapter 4 its discussed in detail the importance of these listener events. Additionally, we also used the function *theoPlayerView.addJavaScriptMessageListener()* to listen for JavaScript messages. If a message received equaled *vjs-user-active*, it indicated that the HTML player interface was visible, and we could display our interface. On the other hand, if the message was "vjs-user-inactive," we had to hide our interface.

To allow the VOD to be prefetched and have less latency, we changed a configuration in our *THEOplayer* called the PreloadType, allowing the player to immediately load metadata and media on source change.

4.2.2 *Gson*

Gson[8] is a Java library, developed by Google, that deserializes and converts JSON files into Java Objects and vice versa, allowing also the serialization of Java Objects into their JSON representation.

We use this library to convert the JSON file provided to the player to our Java Object Models. These models will be further explored in the Data Models section 4.3. To use this library, we initially added its dependency to our application's Gradle file. Subsequently, we created our models and annotated each field with *@SerializedName*. This annotation helps *Gson* in mapping each field of the JSON to the corresponding parameter in our model. To convert the JSON to our models, we called the function *Gson().fromJson(/jsonFilePath)*, which returns an object of the desired type, in our case, a *VODItem* object.

4.3 Data Models

4.3.1 JavaScript Object Notation (JSON)

As previously mentioned in section **??**, our goal was to ensure the video player interface is adaptable to any type of VOD. To achieve this, we needed a JSON file to be supplied to our interface, allowing it to adjust to the current content type. This JSON file needed a specific structure that included all the necessary

[8] https://github.com/google/gson

information to be provided to our interface. This information includes the VOD type, information specific to the VOD based on its type, VOD key events, filters applicable to those events, flags to manage the visibility of certain functionalities (such as custom filters and the crowdsourcing mechanism), and a list of custom filter types and crowdsourcing types if the corresponding flag is set to true.

To test our functionalities, we used a soccer match as an example. A JSON file was created to be provided to our interface, and this can be viewed in Annex III.

4.3.2 Java Data Models

We have developed multiple data models to organize, hold, and manage the information within our application making it easier to work with. These models were deserialized from JSON into Java objects using the *Gson* library.

We begin by discussing the *VODItem* data model. This model encapsulates the VOD, including the vod_type, which in our testing video is represented by the String "game", and a list of *VODEvent* objects representing the key events of the video to be displayed in the timeline. Furthermore, since the VOD type is "game", our application recognizes that this model also contains an [vod_type]Info object, in this case, a *GameInfo* object. This object holds all the needed information about the match, such as the teams involved and the final result, among other details. The model also includes details about the filters, which are represented by a list of *VODFilter* objects that define the predefined filters of the VOD. It also specifies a boolean value indicating whether the VOD includes custom filters, and if so, a list of custom filter parameters. Additionally, it contains a boolean to determine if the VOD uses the crowdsourcing mechanism, and if so, a list of *CrowdsourcingEventTypes*. The structure of this model is illustrated in Listing 2.

```
public class VODItem {

  @SerializedName("vod_type")
  private String vodType;
  @SerializedName("vod_events")
  private List<VODEvent> events;
  @SerializedName("game_info")
  private GameInfo gameInfo;
  @SerializedName("vod_filters")
  private List<VODFilter> filters;
  @SerializedName("custom_filter")
  private boolean hasCustomFilter;
  @SerializedName("custom_filter_types")
  private List<String> customFilterTypes;
  @SerializedName("crowdsourcing")
  private boolean hasCrowdsourcedEvents;
  @SerializedName("crowd_sourcing_types")
  private List<CrowdsourcingEventTypes> crowdsourcingEvents;

  ... Constructor and getter methods ...
}
```

Listing 2: VODItem Model

Concerning the *GameInfo* model, its structure is detailed in Listing 3. This model encapsulates all the important information of a VOD of type "game". It englobes all the necessary details for display within the interface, such as the abbreviated names of the teams, their logos, the colors of their jerseys to visually distinguish each team in an event, the final result of the match, and the full names of each team.

```
public class GameInfo {
    @SerializedName("team_home")
    private String teamHome;
    @SerializedName("team_away")
    private String teamAway;
    @SerializedName("team_home_logo")
    private String teamHomeLogo;
    @SerializedName("team_away_logo")
    private String teamAwayLogo;
    @SerializedName("team_home_jersey_color")
    private String teamHomeJerseyColor;
    @SerializedName("team_away_jersey_color")
    private String teamAwayJerseyColor;
    @SerializedName("final_result")
    private String finalResult;
    @SerializedName("team_home_full")
    private String teamHomeFull;
    @SerializedName("team_away_full")
    private String teamAwayFull;

    ... Constructor and getter methods ...
}
```

Listing 3: GameInfo Model

We also created a model called *VODEvent* to capture all the information about the key events in the video. This model, seen in listing 4, includes general information about each event, such as its type, the icon associated with it, the start and end time in seconds, and the exact minute in which it occurred. Additionally, it also contains more specific details depending on the type of video. For instance, in our case of testing (a soccer match), each event can be associated with a team and individual players, including their names and photos. If the VOD was a concert, then the model could have fields like the performer's name and music title. With the help of this model, we can easily retrieve all the necessary information to display in our interface.

```
public class VODEvent {

  @SerializedName("type")
  private String type;
  @SerializedName("icon")
  private String iconUrl;
  @SerializedName("seconds_start")
  private int secondsStart;
  @SerializedName("seconds_end")
  private int secondsEnd;
  @SerializedName("minute")
  private int minutes;
  @SerializedName("team")
  private String team;
  @SerializedName("player1_name")
  private String player1Name;
  @SerializedName("player1_photo")
  private String player1Photo;
  @SerializedName("player2_name")
  private String player2Name;
  @SerializedName("player2_photo")
  private String player2Photo;

  ... Constructor and getter methods ...
}
```

Listing 4: VODEvent Model

In order to display our filters properly on our interface, we created a model called *VODFilter*. This model contains the name and icon information required for proper display. We also created a similar model called *CrowdsourcingEventTypes* for crowdsourcing events, which additionally includes the color of the event. Both models are described in detail in listings 5 and 6.

```
public class VODFilter {

  @SerializedName("name")
  private String name;
  @SerializedName("icon")
  private String iconUrl;

  ... Constructor and getter methods ...

}
```

Listing 5: VODFilter Model

```
public class CrowdsourcingEventTypes {

  @SerializedName("name")
  private String name;
  @SerializedName("icon")
  private String iconUrl;
  @SerializedName("color")
  private String color;

  ... Constructor and getter methods ...

}
```

Listing 6: CrowdsourcingEventTypes Model

Finally, we needed a Java model to hold the information about a Person that is responsible for an event. This model can be seen in listing 7. Since our VOD type of testing was a soccer match, the Person can also be associated with a team.

```
public class Person {

    private String name;
    private boolean isSelected;
    private String photo;
    private List<VODEvent> events;
    private String team;

    public Person(String name, String team, String photo) {
        this.name = name;
        this.photo = photo;
        isSelected = false;
        events = new ArrayList<>();
        this.team = team;
    }
}
```

Listing 7: Person Model

By using these data models, we can efficiently organize, store, and manage our data, making it easier to integrate into the user interface.

4.4 Functionalities

In this section, the features implemented are discussed in more detail and illustrated through storyboards.

4.4.1 Video Player Screen

The video player screen is the only screen and entry point for our application and can be referenced in Figure 4.2. This screen contains the THEOplayer alongside its integrated HTML interface, which includes the seekbar and the pause and volume buttons. Our interface features two main buttons, each with a label and an icon: the first button is labeled "Timeline" and includes a timeline icon, guiding users to the VOD Key Events Timeline Section. The second button is labeled "Highlight Moment" and is accompanied by an icon, directing users to the Highlight Moment Section. These buttons are placed at the top of the video player, making them easily accessible without interrupting the viewer's experience. All our features, except for progress tracking, are incorporated within these two sections and each

time these sections are opened, the video player is paused to prevent it from continuing to play.

Figure 4.2: Video Player Screen

4.4.2 VOD key Events Timeline Section

When clicking the "Timeline" button on the top of the video player interface, the key events timeline section appears. In this section, we introduce general information about the VOD alongside a vertical timeline of key events. We also feature a filtering button to filter events on the timeline, a button to play these events sequentially, and a button to close the section and return to the video player.

In our example, seen in Figure 4.3, we showcase information about a soccer match. This includes the teams, their names, and logos, as well as the match's final result. The information is structured into two sections, with one side representing one team and the other side representing the opposing team. All the information displayed was obtained through our data models.

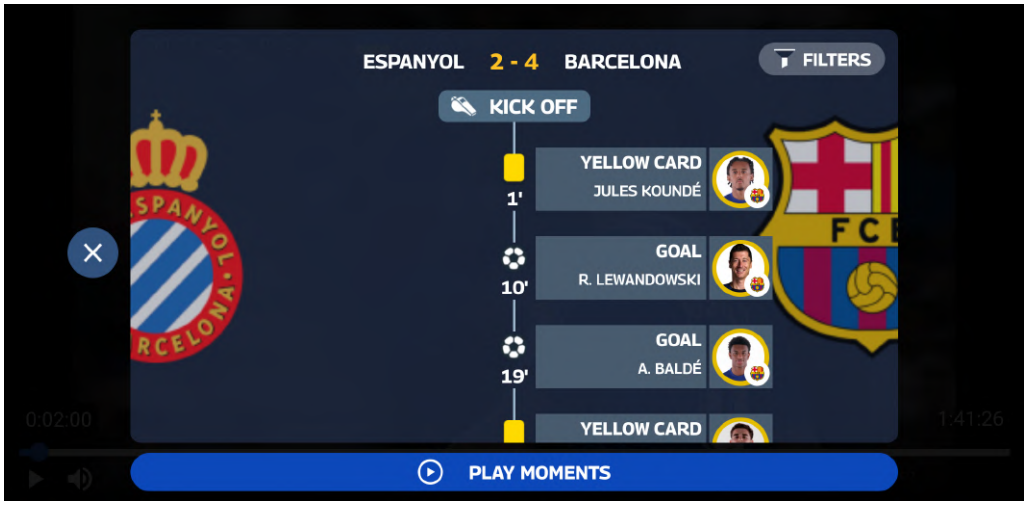

Figure 4.3: VOD key Events Timeline Section

4.4.2.1 Key Events Timeline

In our application, we have implemented a vertical *RecyclerView*, shown in Figure 4.4, that represents the timeline of key events. This timeline is in chronological order, with the earliest event appearing first, based on the *seconds_started* parameter in each event. This *RecyclerView* is a list that the users can scroll to find the desired event. Once finding the desired event, they can tap that event to start playing it.

To populate this *RecyclerView*, we developed a list adapter called *VODEventAdapter* that acts as a bridge between the data and the views presented. The adapter uses the model *VODItem* to extract the game events list and then, for each type of event, we determine how the data is visually represented in the *RecyclerView*. In our timeline, events can be positioned either in the center, on the left, or the right sides. In our testing scenario, each team is assigned a side, and their events are displayed on the side that corresponds to their team. A general or contextual event tends to be at the center of the timeline.

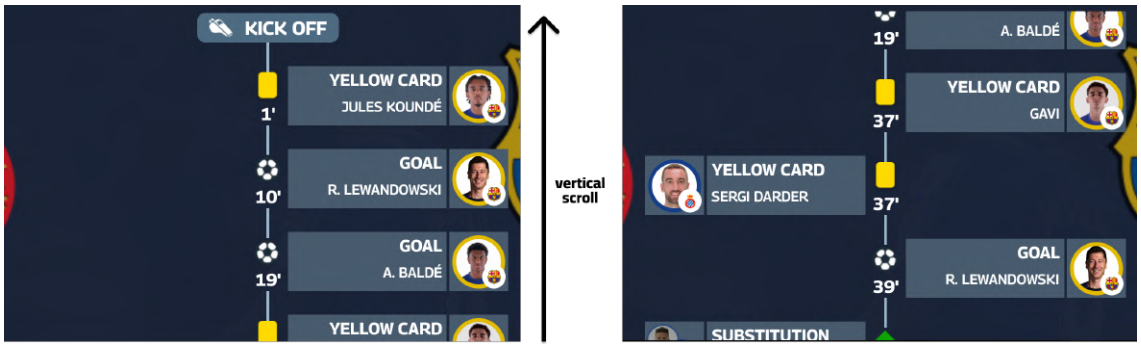

Figure 4.4: Vertical Events Timeline with the scrolling ability to show more events

The timeline can have two distinct event layouts, each corresponding to a specific type of event, and these layouts range from a simple design featuring an icon and text to a more complex layout that provides detailed information about the event. The event layout that includes more information also includes the minute in the match in which the event took place within the match. These different event layouts are illustrated in Figure 4.5.

Figure 4.5: Event Layouts: The left image showcases a simple design featuring just an icon and a label, while the right image displays a more detailed layout with additional information about the event.

To enhance the visibility and distinction of events generated through crowdsourcing, we assign a unique color to each type of crowdsourcing event. This color coding makes these events more recognizable when displayed in the timeline. For crowdsourcing events, we do not display the timestamp because it may not always be accurate. For example, the VOD we tested contained the soccer match half-time advertisements so, when saving a moment in the 70th minute of the match, it wouldn't align exactly with the 70th minute of the VOD but with the 85th minute of the video. Without a precise method to accurately align the event's timestamp with the VOD's time, we opted not to include the timestamp in these events. An example of a crowdsourcing event layout can be seen in Figure 4.6.

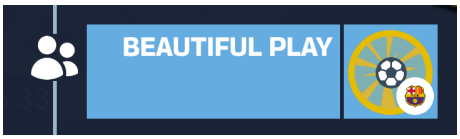

Figure 4.6: Crowdsourcing event layout with a different background color to distinguish between these and key events.

When the user taps on an event on the timeline, they are taken directly to the video player screen. After that, the video player's current time is synchronized with the event's start time, the playback is resumed, and the "Currently Watching" layout regarding that event becomes visible. This layout displays additional information about the event currently being watched, including the type of the event and an icon associated with it, the timestamp where the event occurred, and any additional information. Regarding our soccer match events, we also show information about the players responsible for the event.

When the event ends, the video player returns to the position the user was at before starting to watch the event. To achieve this, before synchronizing the video player with the event's start time, we save the current position of the video player. With the use of *THEOplayer PlayerEventTypes.TIMEUPDATE* listener, we are able to see if the video player already reached the event's end timestamp and if so, we redirect the user to the previous saved timestamp. The action of tapping on an event to watch it can be seen in Figure 4.7.

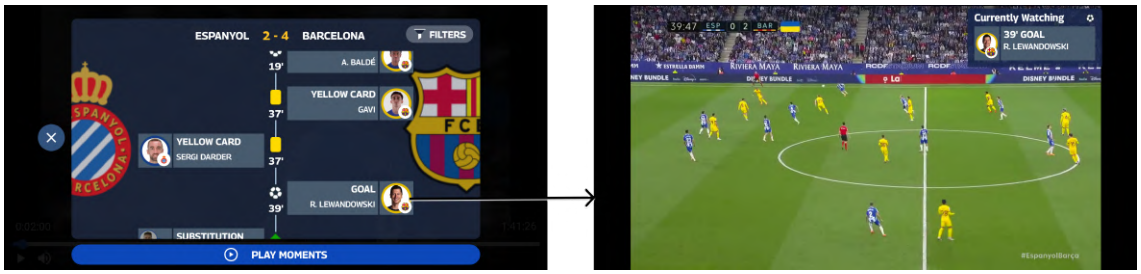

Figure 4.7: Action of tapping on an event to watch it.

If an event only contains a starting timestamp and does not have an ending timestamp, then it's considered a contextual event. In this scenario, the "Currently Watching" layout is not displayed, as these types of events only indicate a starting timestamp. An example of this type of event includes time events in our match, such as the "Kick Off", "Half Time", and "Full Time" events.

When the user taps the "Play Moments" button located at the bottom of the timeline section, all events displayed on the timeline begin to play automatically, with each event starting immediately after the previous one concludes. To implement this feature, we have a sequential list of events that the video player progresses through. If an event has a subsequent event, the "Go to next event" layout appears below the "Currently Watching" layout. This button enables users to directly navigate to the next event in the list. If there are no more events to be seen, the video player goes back to the saved previous timestamp the user was before starting to see the events, and the "Currently Watching" layout disappears. This action is illustrated in Figure 4.8.

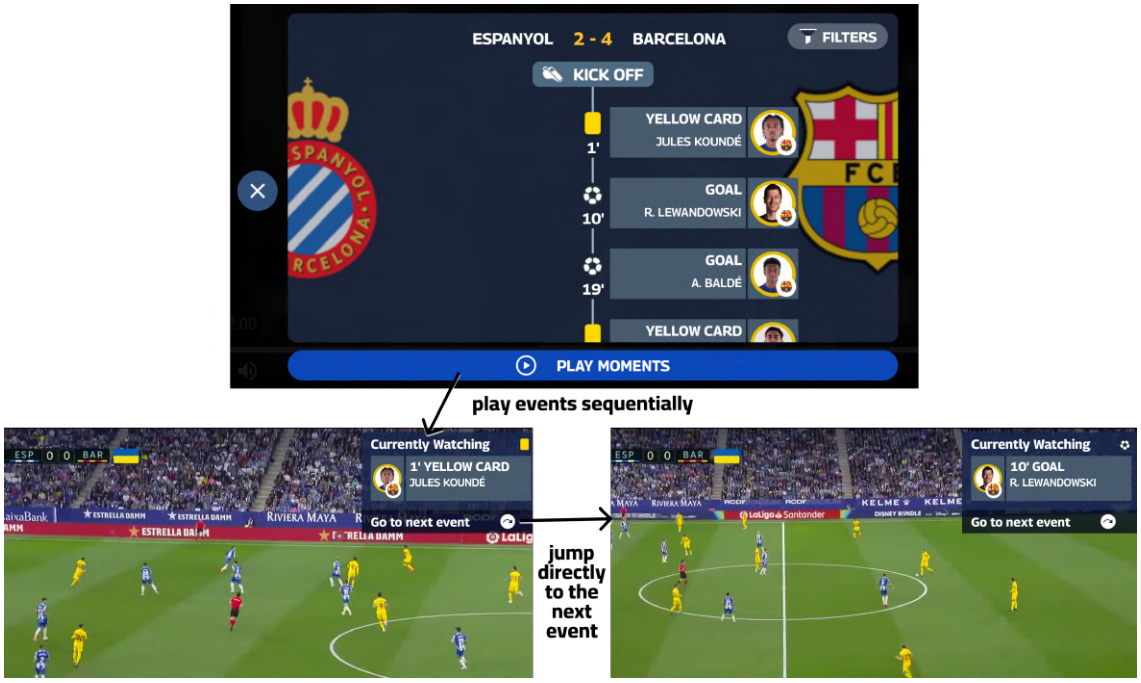

Figure 4.8: Action of watching events sequentially and jumping directly to the next event by tapping the "Go to next event" button.

If the user manually seeks to another timestamp while watching an event, we hide the "Currently Watching" layout, indicating that the user has stopped watching the events. To differentiate between manual seeks and those initiated by the video player during event playback, a boolean flag named isFromEvents is used. This flag is set to true whenever we begin watching an event or a series of events and is set to false each time the *THEOplayer PlayerEventTypes.SEEKED* listener is invoked. This distinction is crucial because both manual seeks and those triggered by the video player when navigating to an event's timestamp result in a *THEOplayer PlayerEventTypes.SEEKED* event. We can only cease event playback and hide the "Currently Watching" layout if the seek was manually performed by the user.

4.4.2.2 VOD Event timeline Filters

To tailor the timeline events more closely to user preferences, the application incorporates a filtering functionality. This feature is accessible via a button situated in the top right corner of the timeline section interface. Tapping this button reveals a filtering section, illustrated in Figure 4.9, where users can choose from predefined filters or create their own custom filter. These filter options were implemented using *ConstraintLayouts*, which allows for greater customization of the buttons. By default, all filters are deactivated, showing a white border around each of the

filters; upon selection of a filter item, their border turns yellow. When a user selects a predefined filter, the custom filter becomes inactive, as it can only be used independently. Upon choosing their filters, a save button becomes active, and after saving their selection, the timeline displays only the events that match the selected criteria. The filters are stored in variables, ensuring that when the user returns to the filtering section, all previously selected filters are displayed.

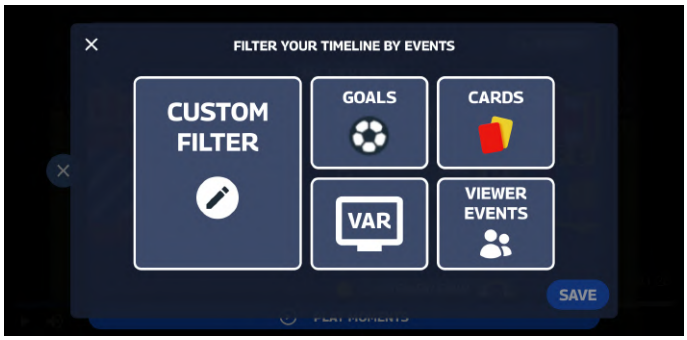

Figure 4.9: Event timeline filters section.

Concerning the predefined filters, these are retrieved from the *VODItem* object as a list of *VODFilter* objects and are represented in the filter section with each filter depicted by a rectangle with curved corners, potentially featuring an icon and a label. Users can select multiple predefined filters, and after saving, the timeline list will reflect these filters. If no events match the filters, a message is displayed. If the crowdsourcing mechanism is enabled for the VOD, a predefined filter is created for events generated through this method, referred to as "Viewer Events". The action of filtering the timeline using predefined filters is showed in figure 4.10.

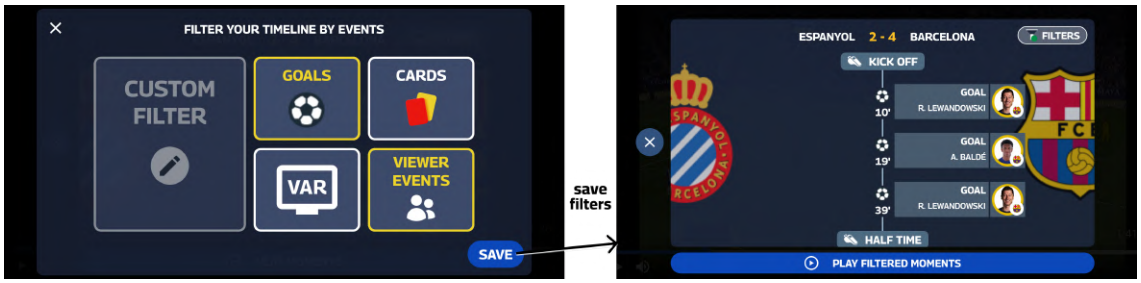

Figure 4.10: Action of filtering the timeline of the events using predefined filters.

Regarding the custom filter, it is more complex. Upon tapping the custom filter item, all the pre-created filters become disabled and a "next" button appears that when clicked, opens another section with more refined filtering options. Given that our *VODItem* object includes a list of three custom filter parameters, we offer

a three-step filtering process where each step depends on the choice made in the previous step. This ensures that some filtered events are always returned.

In our example illustrated in Figure 4.11, users can create a custom filter based on a specific team, a particular type of event related to that team, and a player associated with the selected team and event type. Users can only proceed to the next step after selecting a filtering parameter, at which point a "next" button becomes active. They can also return to a previous step using the "back" button present in each step. Additionally, we include "select all" and "clear" buttons to simplify parameter selection. Each filter section features a scrollable grid list with an adapter and these list items can be either simpler with only an icon and label or more complex, with a rectangular with curved corners layout, having more space for more detailed information.

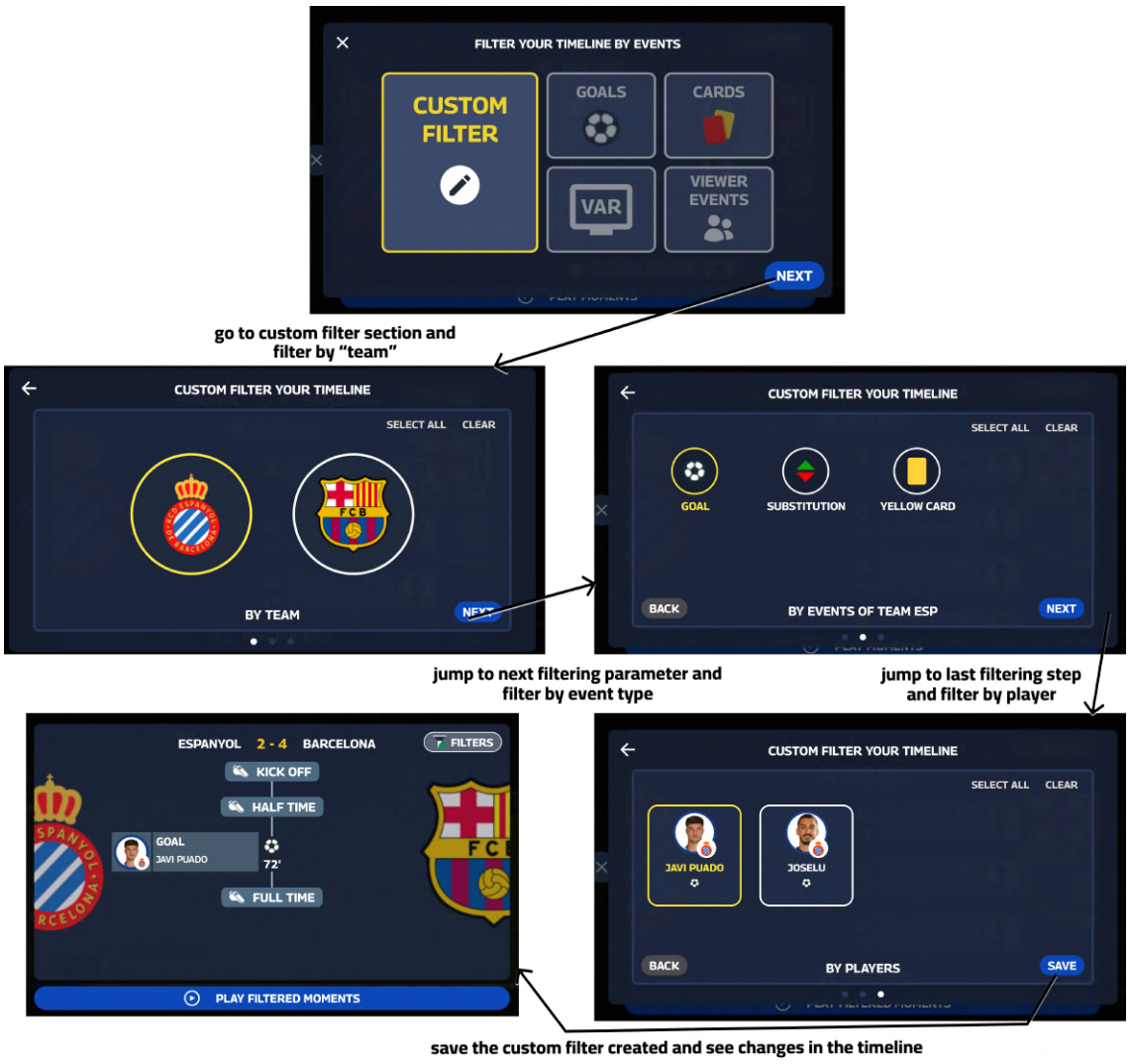

Figure 4.11: Action of filtering the timeline of the events using the custom filter.

After saving the events, the filter button layout on the timeline section changes to indicate that filters are active. This change is visible in Figure 4.12.

Figure 4.12: The filter button in its inactive state (left) and its active state (right), illustrating the visual distinction between the two states.

Following filtering, the "Play moments" button label changes to "Play filtered moments". This time, if the user taps the button, only the filtered events start to play sequentially.

4.4.3 Highlight Moment Section with the use of crowdsourcing

To enhance user interaction with the content they are watching, we developed a feature that enables users to add/highlight subjective events in the official timeline. To access the highlighting moment section, users must tap the "Highlight Moment" button located at the top of the video player interface. Upon tapping, a layout appears, providing information about the crowdsourcing mechanism as well as a list of event types for selection. This layout can be seen in Figure 4.13 and is populated with a list of *CrowdsourcingEventTypes* objects retrieved from our *VODItem* object. Each crowdsourcing event item has a title, icon, and color associated with it to distinguish between events in the timeline.

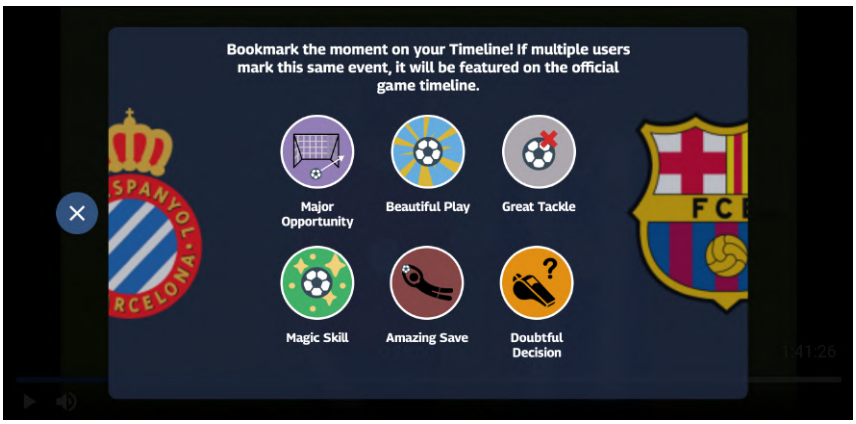

Figure 4.13: Crowdsourcing event highlighting main section with information about the highlighting method as well as a list of available events to choose from

Before highlighting the moment, the user must navigate to the timestamp they wish to highlight, as the timestamp saved for the event is the current timestamp of the video player.

After entering the Highlight Moment section and selecting the event type they wish to highlight, the interface can present another parameter for the user to choose. This second parameter can be optional but, in our testing scenario, we decided that the event is related to a team.

Once the user has chosen both the event type and the related team, one of two scenarios occurs. If enough people have already highlighted the same moment with the same type and team around the same timestamp, the event is added to the official timeline. If the number of users highlighting the moment does not meet the required threshold to add the event to the timeline, the user sees a message thanking them for highlighting the event, but no event is added to the timeline. Once the threshold is reached, the saved moment event is added to the official timeline and is visible to every user. In our testing, we selected a threshold of 20 users, which can be adjusted in the implementation. The action of highlighting a moment and the resulting timeline are illustrated in Figure 4.14

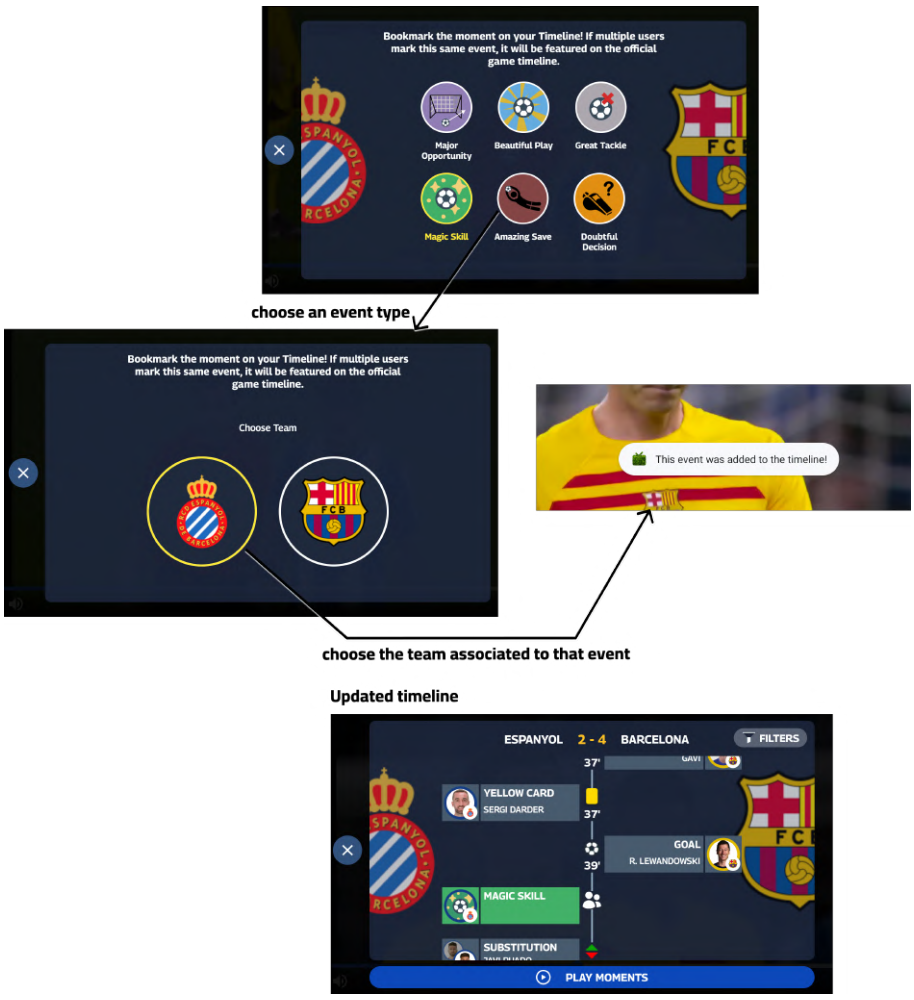

Figure 4.14: Action of highlighting a moment and updated timeline

When the threshold required to add the event to the timeline is reached, we need to determine the exact timestamp of that event. This is achieved by calculating the average of all timestamps that are within a 30-second window and share the same highlighting parameters. This approach helps us pinpoint the exact timestamp of an event by averaging the timestamps within proximity to each other. To calculate the timestamp where the event ends, we add 45 seconds to the starting timestamp. These events will also be presented in the JSON file provided to the video player.

4.4.4 VOD progress tracking

To allow the user to track their progress in the VOD, we implemented two new key features.

The initial feature we introduced involves adding a second thumb to the seekbar, which represents the farthest or maximum timestamp the user has viewed during the current session. This thumb layout is akin to the seekbar thumb in terms of shape and size but is distinguished by a different color. To consistently track the most distant timestamp, we monitor the *THEOplayer PlayerEventTypes.TIMEUPDATE* listener. Each time it's triggered, we evaluate whether our saved maximum timestamp variable should be updated with a newer, higher value. The position of the new seekbar thumb is refreshed whenever the user manually seeks to a timestamp greater than the previously recorded maximum timestamp.

Due to the inability to access the integrated HTML interface of the video player, which includes the seekbar main thumb's position on the screen, determining the correct position for displaying our new visual cue proved to be a significant challenge. If we could access the integrated seekbar thumb position, we could save the x and y coordinates of the thumb each time it reached a new maximum position and later display our new thumb in that same location. Without that access, we had to manually calculate the correct x-coordinate position to display our new cue, considering the saved maximum timestamp value, the total duration of the VOD, and the width of the video player. The resulting formula for this calculation was:

$$\frac{(\text{max timestamp}) * (\text{player width})}{(\text{player duration})}$$

Regarding the y-coordinate position to display our new thumb, we aligned the view to the bottom of the player view and applied a margin-bottom of 32 dp, and with this, it aligned perfectly with the seekbar y position. By using this approach,

we successfully addressed the challenge and precisely displayed the visual cue for the max timestamp on the seekbar.

We have also implemented a feature that allows users to return to their previous timestamp when they manually seek to a different position in the VOD. This feature can be useful in situations where the users accidentally tap the seekbar, seeking to another timestamp, and wish to return to their previous watching timestamp.

To achieve this functionality, we are always saving the video player's current position using the *THEOplayer PlayerEventTypes.TIMEUPDATE* listener. When manually seeking, the "Fast return to [timestamp]" layout is displayed. Tapping on that layout hides it and returns the video player to the previously saved timestamp.

This layout only is shown if the difference between the current/new and previous/old timestamps is at least 300 seconds, thereby preventing excessive requests for minor differences. This value can be changed depending on the VOD duration.

In Figure 4.15, the process of manually seeking to a timestamp, returning to the previous timestamp, and updating our new farthest timestamp thumb is illustrated. When the user manually seeks to a new timestamp, the "Fast return to [timestamp]" layout appears, indicating the availability of a previous timestamp to return to. The yellow thumb representing the farthest timestamp the user has reached also updates to the new position.

Figure 4.15: Action of highlighting a moment and resulting timeline.

5

Evaluation

This chapter presents the evaluation of the video player interface features we have implemented. The focus of this evaluation was to gather feedback from the users to verify the overall usability and user experience. To achieve these objectives, we conducted a usability test. A usability test is a structured method for evaluating an application by testing it on users. The focus of a usability test is to gather direct feedback from users, which is crucial in verifying the overall usability and user experience of an application.

5.1 Methodology

In the usability test, we used a test script (see Annex IV) to guide participants through the testing of all the features we developed. The script began with a brief introduction of the concept and a summary of the test, without revealing any features, to enable users to explore and discover them independently.

Participants were then presented with a series of specific tasks, including navigating through the VOD key events using the events timeline, interacting with timeline filters, using the crowdsourcing event highlighting mechanism, and engaging with the VOD progress tracking features. These tasks were done in a Huawei P40 Pro, and the VOD used for tests was a soccer match between Barcelona and Espanyol. This match had a total of 23 key events including goals, substitutions, and yellow card events.

After completing their tasks, participants were asked to fill out a Google Forms questionnaire (see Annex V). This questionnaire was divided into four sections. The first section collected demographic information from participants. The second section gathered feedback on each functionality and participants' comfort level while using them. The third section required participants to answer the System Usability Scale (SUS), a tool for measuring system usability [50]. The fourth and

final section involved completing the User Experience Questionnaire (UEQ), a tool for assessing the user experience of interactive products [51]. These four sections allowed us to collect valuable insights and information to enhance our features.

5.2 Participants' Background

The sample for the usability test included 17 participants, with 11 males (64.7%) and 6 females (35.3%). The age range of our participants was extensive, from 20 to 58, with a significant number of participants in their 20s. The average age of the participants was 29.7, and the standard deviation was 11.2, indicating the diversity in ages among our test participants.

Regarding occupation, the participants came from a variety of fields, including maritime police (5.9%), administrative roles (5.9%), and even a speech therapist (5.9%). Notably, there were several participants with backgrounds in software engineering and development (52.9%), as well as students (17.6%).

To better understand the experience of the participants with streaming platforms and Video-on-Demand (VOD) content, we inquired about their video consumption habits across different platforms. The responses varied among participants, with 10 individuals (58.8%) indicating they consume content on streaming platforms daily, 3 several times a week (17.6%), 2 once or twice a week (11.8%), 2 rarely (11.8%). These findings, as illustrated in Figure 5.1, show a significant level of experience of the participants with streaming platforms and VOD content consumption.

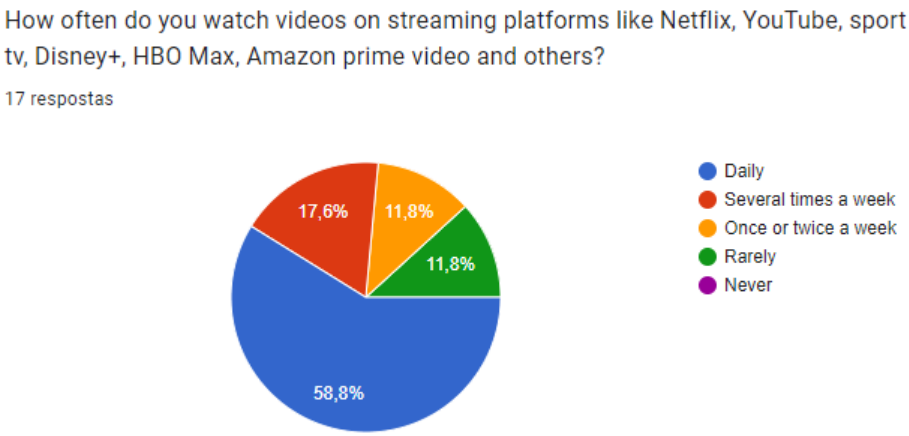

Figure 5.1: Overview of Video Consumption Habits of our participant sample

Finally, we asked participants to share experiences of usability issues they encountered in current streaming platforms. Participants reported challenges such as accidentally fast-forwarding and the difficulty of returning to the previous

timestamp, as well as the difficulty in locating specific events or timestamps within the video content, including notable moments like crashes or overtakes in an F1 race. These insights were valuable for better understanding the areas for improvement in current VOD streaming platforms.

5.3 Results - Individual Features

In this section, we discuss the participants' responses and feedback on feature-specific questions. The majority of the questions in this section were inquired on a Likert scale from 1 to 5, with 1 indicating complete disagreement and 5 indicating complete agreement with the statements presented [52]. Additionally, we included some open-ended questions for each feature to gather more detailed feedback.

5.3.1 VOD Key Events timeline

Interaction and navigation:

Concerning the VOD key events timeline interaction, every participant agreed that the timeline was intuitive. The mode of the results is the value 5, with 13 (76.5%) participants completely agreeing and 4 (23.5%) participants just agreeing that the timeline is intuitive. These results can be seen in annex VI.1.

The ease of finding and understanding where each key event occurs within the VOD also garnered very positive feedback (refer to annex VI.2), with 13 participants (76.5%) completely agreeing that it is easy to understand where each key event happens in the video, and 4 participants (23.5%) also agreeing with that same statement.

Regarding the ease of navigating through the event timeline, feedback was similarly positive (refer to annex VI.3), with 14 participants (82.4%) completely agreeing that the timeline is easy to navigate, and 3 participants (17.6%) agreeing also with that same statement.

Despite the overall positive feedback, there were areas for improvement identified. Two participants had difficulty locating the button that leads to the sequential view of events, indicating a need for clearer labeling or a more intuitive design. A suggestion was made to rename the button from "Play Moments" to "Play Key Events", which addresses the inconsistency in the terminology used across the interface.

Additionally, one participant suggested that after watching an event or completing a sequence of events, the interface should return to the timeline instead of

continuing in the video player. They also recommended the addition of a "go back to the timeline" button, which would be placed below the "Currently Watching" layout and appear on the last event in a sequence. This addition would enable users to easily navigate back to the timeline.

Filtering the events by parameters:

Regarding the intuitiveness of filtering the event timeline by various parameters, every participant agreed that the process was intuitive (refer to annex VI.4). The mode of the results is the value 5, with 10 participants (58.8%) completely agreeing and 7 participants (41.2%) just agreeing that the filtering process is intuitive.

Feedback on the appropriateness of the filtering options in the context of the video was also positive, with 15 participants (88.2%) completely agreeing that the filtering options were appropriate and 2 participants (11.8%) just agreeing with that same statement. Results can be observed in annex VI.5.

When asked if any other filtering parameters were missed or unnecessary, the unanimous response was negative, indicating sufficient and relevant options.

Despite the overall positive feedback, a few suggestions were made during the testing process. It was recommended that the custom filter should display the available parameters for filtering before the user enters the custom filter section, and that team names should be included alongside the logos to aid users who are not familiar with soccer matches. Furthermore, it was suggested that the "save" button for filters be replaced by an "apply" button.

5.3.2 Event Highlighting though crowdsourcing

In the context of highlighting custom events, feedback was very positive. The mode of the results is the value 5, with 11 participants (64.7%) completely agreeing that highlighting custom moments is intuitive, and 6 participants (35.3%) also agreeing with that statement. Results can be observed in annex VI.6.

Regarding the ease of saving custom moments in the event timeline, the feedback was also positive, with 15 participants (88.2%) completely agreeing that it is easy to use the save custom moments feature and 2 participants (11.8%) agreeing with that same statement. Refer to the annex VI.7 for the results.

The clarity of saved moments on the event timeline was also well-received, as seen in annex VI.8, with 16 participants (94.1%) completely agreeing that the highlighted moments are clearly displayed and distinguished from the other events in the timeline and just 1 participant (5.9%) agreeing with that same statement.

No issues were discovered on the tests.

5.3.3 VOD progress tracking

The features for progress tracking of the VOD gathered positive feedback but also revealed some usability issues during the usability tests.

The ability to track progress in the VOD was generally well-received, with 9 participants (52.9%) rating it as easy (refer to annex VI.9). However, some users did encounter issues related to these features, which will be discussed below.

Concerning the feature to jump back to the previous timestamp, a few participants faced challenges in locating the correct button and attempted to return to their previous timestamp using the seekbar. Despite this, as illustrated in Figure 5.2, we received a significant amount of positive feedback, with 9 participants (52.9%) rating the question a 5 and 5 participants (29.4%) rating the question a 4, with a total of 14 participants out of 17 agreeing that the feature was easy to use. Two users (11.8%) neither agreed nor disagreed with the statement, and one user (5.9%) disagreed.

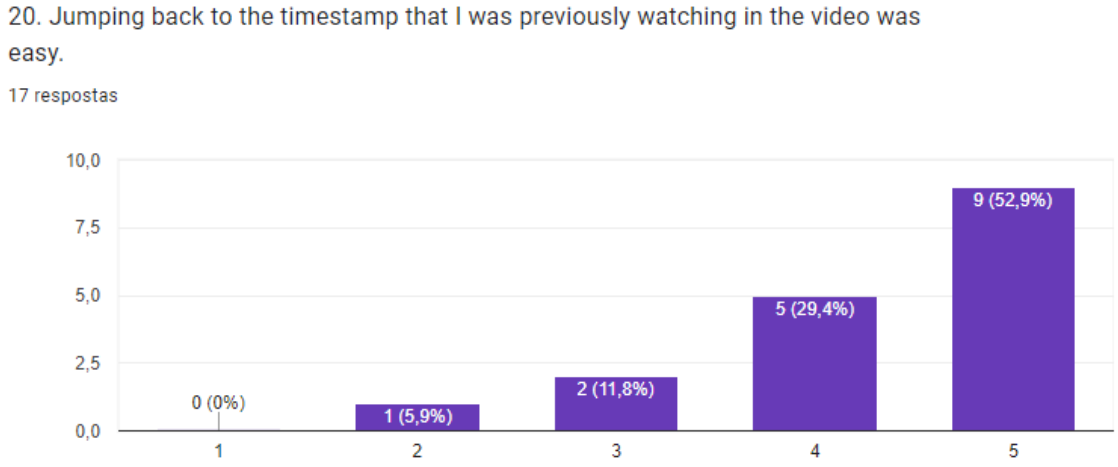

Figure 5.2: Jumping back to previous timestamps ease

Navigating to the farthest timestamp of the VOD was the feature participants found most challenging. Many did not recognize the yellow thumb as indicating the farthest timestamp. This difficulty may also stem from the fast-paced nature of our tests, where participants might not have focused on the integrated video player interface due to a lack of prior instruction on the features they were testing. We acknowledge that this feature is not immediately obvious, suggesting the need for a more distinctive design for our thumb, such as enlarging the thumb, altering its shape, or changing its color to better differentiate between the one we implemented and the video player one. Despite this, we received positive feedback from 6 participants (35.3%), who strongly agreed that the feature was easy to use.

Additionally, 5 participants (29.4%) agreed with this sentiment. Four participants (23.5%) neither agreed nor disagreed with the simplicity of the feature, while two participants (11.8%) disagreed, indicating that it was not simple for them.

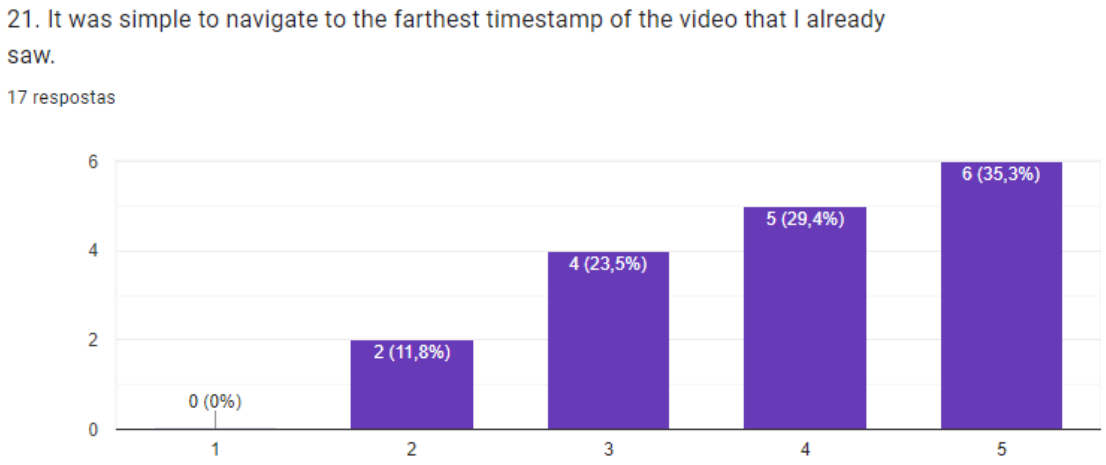

Figure 5.3: Navigating to farthest timestamp ease

The feedback indicates that while certain features may not be immediately apparent, they become more straightforward to use with time. Nevertheless, we are committed to enhancing and addressing all encountered issues to further improve the user experience.

5.3.4 Overall feedback

The majority of participants (64.7%) rated their experience with the video timeline and its features as excellent, with 5 participants (29.4%) rating it as good. Only 1 participant (5.9%) found the experience neutral. These results can be seen in annex VI.10.

A significant majority of participants also (94.1%) believed that these new features improve the overall user experience, with 16 participants (94.1%) strongly agreeing and only 1 participant (5.9%) agreeing (refer to annex VI.11).

There was a high level of interest in seeing these functionalities implemented on current streaming platforms, with the results in annex VI.12 showing that 16 participants (94.1%) expressed strong interest and 1 participant (5.9%) expressed some interest.

No participants found any functionality unnecessary, indicating that the features tested were perceived as relevant and beneficial to the user experience.

These responses collectively suggest that the participants had a positive overall experience with the video timeline and its features, believed that these new features enhance the user experience while watching VOD content, and expressed interest in their implementation on streaming platforms.

5.4 Results - System Usability Scale (SUS)

The System Usability Scale (SUS) is a widely used and standardized questionnaire designed for evaluating the usability of a system [50].

Consists of a total of 10 pre-defined statements about the tested system, with the odd-numbered items worded positively and the remaining even-numbered items worded negatively. It uses a Likert scale of rating with 1 representing "Strongly disagree" with the statement presented and 5 representing "Strongly agree".

Given that SUS is a customizable scale [53], minor adjustments were made to the pre-defined statements to more accurately reflect the characteristics of the system being tested. With these modifications in place, the participants were presented with the following items for rating:

1. I think that I would like to use these new video player functionalities frequently.
2. I found the new functionalities unnecessarily complex.
3. I thought the new functionalities were easy to use.
4. I think that I would need the support of a technical person to be able to use the new functionalities.
5. I found the various functionalities in this video player were well integrated.
6. I thought there was too much inconsistency in the video player.
7. I would imagine that most people would learn to use these new functionalities very quickly.
8. I found the new functionalities really hard to use.
9. I felt very confident using the video player with the new functionalities.
10. I needed to learn a lot of things before I could get going with these new functionalities of the video player.

This structure allows test participants to express their ratings on statements about the system, making it simple and effective for assessing usability.

To calculate the SUS score, the following steps were followed:

1. For the odd-numbered questions, subtract one from the user's score. For the even-numbered questions, subtract the user's score from five.
2. Sum all the scores obtained in the first step to obtain a value ranging from 0 to 40.
3. Multiply the result of step 2 by 2.5 to convert it into a SUS score, which ranges from 0 to 100.

To interpret the SUS score, it is important to consider its distribution across various systems and contexts. Studies with 500 different systems have revealed that the overall average SUS score is around 68, which suggests that a score of 68 represents the 50th percentile concerning usability ratings [54]. This implies that a score of 68 is considered average, while higher scores than 68 indicate above-average usability and lower scores than 68 show below-average usability [54].

This scale was initially developed to evaluate only the usability of a system, focusing on a single dimension. However, research has shown that SUS can also assess the learnability of a system [55], which refers to how easily users can learn to use the system effectively. Specifically, items 4 and 10 of SUS are associated with the learnability dimension, while the remaining eight questions are associated with the usability dimension.

In order to make the usability and learnability scores directly comparable with the overall SUS score, specific adjustments were applied to the scoring process. For usability, the summed score contributions of the relevant items are multiplied by 3.125. Similarly, for learnability, the scores of the two items specifically measuring learnability (items 4 and 10) are summed and then multiplied by 12.5 [55].

Figure 5.4 illustrates the individual scores of each participant on the System Usability Scale (SUS) including learnability and usability.

The SUS scores of the tested functionalities ranged from 62.5 to 100, with an average score of 90. This average score, when compared to the global average SUS score of 68, indicates that the participants found our video player interface to have significantly higher usability and learnability than the average interface.

The learnability scores, which assess how easily participants can learn to use the features integrated into the video player effectively, ranged from 75 to 100,

Participant	SUS Score	Learnability	Usability
1	92,5	100	90,625
2	92,5	100	90,625
3	97,5	100	96,875
4	62,5	75	59,375
5	95	100	93,75
6	82,5	87,5	81,25
7	92,5	87,5	93,75
8	92,5	87,5	93,75
9	92,5	100	90,625
10	100	100	100
11	97,5	100	96,875
12	87,5	100	84,375
13	92,5	100	90,625
14	82,5	75	84,375
15	85	100	81,25
16	95	87,5	96,875
17	90,0	100	87,5
Average	90	94,11764706	88,97058824
Standard deviation	8,660254038	8,967875675	9,447483276
Grade	A+		
Adjective Rating	Best Imaginable		

Figure 5.4: SUS score of participants including learnability and usability

with an average score of 94.11764706. This high average score indicates that the features were easy to learn.

Similarly, the usability scores, which measure the overall ease of use of the video player interface, ranged from 59.375 to 100, with an average score of 88.97058824. This average score suggests that the system is highly usable, with the participants finding it relatively easy to use the new interface and its features. However, one participant specifically rated our interface at 59.375 in terms of usability score. This particular score suggests that while the interface is generally considered highly usable, there may be areas for improvement, especially for this particular user.

The standard deviations for the SUS scores, learnability scores, and usability scores indicate relatively high variability in the scores, suggesting that the participant's perceptions of our video player interface features usability and learnability varied.

Overall, the results indicate that the system is highly usable and easy to learn, with an A+ grade and an adjective rating of "Best Imaginable" [56].

5.5 Results - User Experience Questionnaire (UEQ)

The User Experience Questionnaire (UEQ) is a popular tool for assessing the user experience of interactive products. It is recognized for its ease of application,

validity, and reliability, making it a valuable addition to data gathered from other evaluations, like SUS, by providing a subjective quality assessment.

It measures both usability dimensions (efficiency, perspicuity, dependability) as well as user experience dimensions (novelty, stimulation) [51]:

- Efficiency: Measures the effectiveness of users in achieving their goals while using the system.
- Perspicuity: Measures the ease with which users can become familiar with and learn to use the product.
- Dependability: Evaluates whether the users feel in control during interaction with the product and if the product is perceived as secure and predictable.
- Novelty: Refers to the degree of innovation and originality perceived in the system.
- Stimulation: Assesses the level of engagement and excitement generated by the system.

The UEQ also measures the Attractiveness dimension, which evaluates the aesthetic appeal of the product.

The questionnaire includes a total of 26 items, each represented by two terms with opposed meanings, rated on a 7-point scale. This structure enables participants to instantly express their feelings, impressions, and attitudes that arised while interacting with the product. After gathering the scores, they are converted to a scale from -3 to +3, with 0 signifying a neutral response. Opinions are categorized as neutral if they fall within the range of -0.8 to 0.8, negative if they are below -0.8, and positive if they surpass 0.8 [57].

To analyze our results we resorted to the Data Analysis tools provided in the User Experience Questionnaire's official website[1].

The overall results of the UEQ scales are illustrated in Figures 5.5 and 5.6. All scales had highly satisfactory results, with the Efficiency scale standing out as the highest, averaging 2.588, and the Novelty scale being the lowest, with an average of 2.015. The standard deviations of the resulting scales indicate consistency in the results. Additionally, the Mean and Median of the values are nearly identical, further demonstrating that the participants shared a similar perspective.

[1] https://www.ueq-online.org/

UEQ Scales (Mean, Standard Deviation and Median)			
Attractiveness	2,471	0,48	2,50
Perspicuity	2,279	0,55	2,25
Efficiency	2,588	0,46	2,75
Dependability	2,206	0,60	2,25
Stimulation	2,250	0,58	2,25
Novelty	2,015	0,65	2,25

Figure 5.5: Mean, Standard Deviation and Median of resulting UEQ Scales

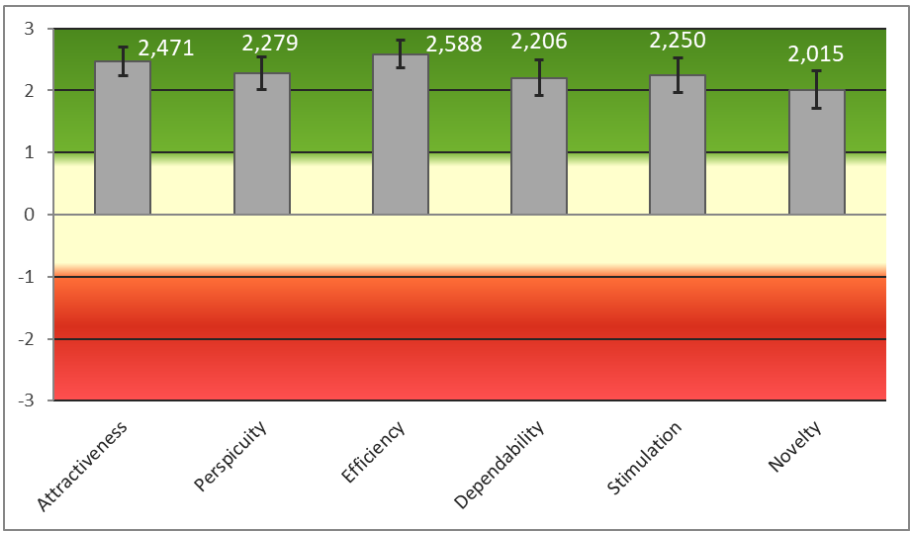

Figure 5.6: UEQ Scales

Figure 5.7 illustrates the distribution of responses for each item, providing a clear view of the consensus among answers. However, with the help of the heuristic provided in the Data Analysis tool on the UEQ website, we detected suspicious data. Three participants were found to have inconsistencies in their responses, with two inconsistencies affecting the Novelty scale and one affecting the Stimulation scale.

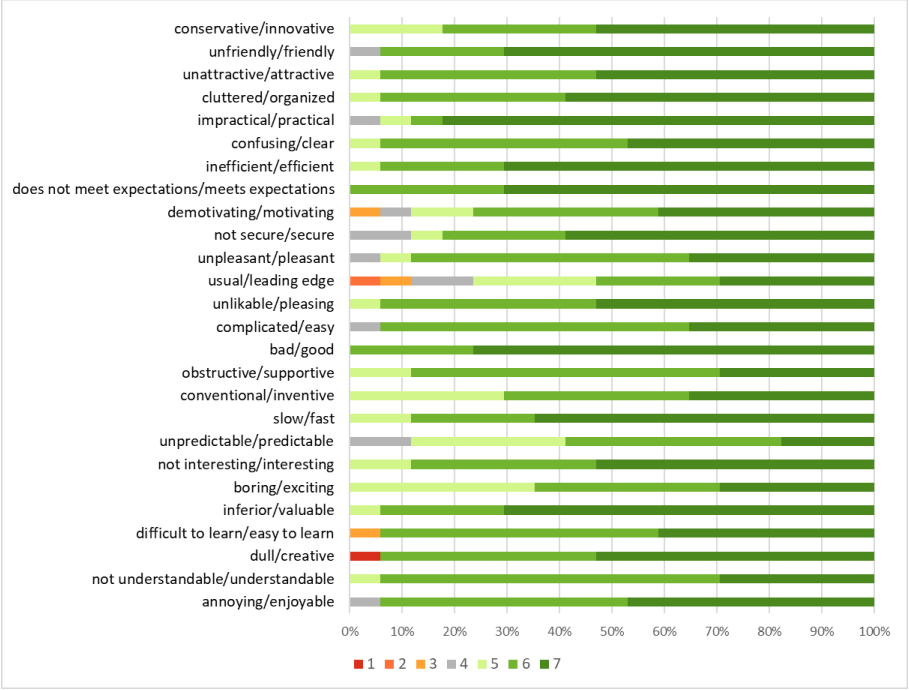

Figure 5.7: Distribution of Answers per item

According to the UEQ benchmark results diagram in Figure 5.8, our features were categorized as "Excellent", indicating they fall within the top 10% of the best results. This benchmark encompasses data from 468 distinct product evaluations and involves 21,175 individuals who have tested a system with the UEQ.

Scale	Mean	Comparisson to benchmark	Interpretation
Attractiveness	2,47	**Excellent**	In the range of the 10% best results
Perspicuity	2,28	**Excellent**	In the range of the 10% best results
Efficiency	2,59	**Excellent**	In the range of the 10% best results
Dependability	2,21	**Excellent**	In the range of the 10% best results
Stimulation	2,25	**Excellent**	In the range of the 10% best results
Novelty	2,01	**Excellent**	In the range of the 10% best results

Figure 5.8: UEQ benchmark results

In conclusion, the UEQ results showcase a very positive reception of our features. The data clearly indicates a strong consensus among the participants, with our features being highly regarded across various scales. These results not only validate the effectiveness of our features but also provide a solid foundation for future development and improvements.

5.6 Conclusion

In the evaluation chapter, we undertook a comprehensive evaluation of our video player interface to gather user feedback and identify areas for improvement.

Initially, we collected feedback from users regarding individual features, which revealed some issues that needed to be addressed. Potential solutions to these issues were also discussed, with a more comprehensive exploration of these solutions to be presented in Chapter 6.

Following this, we conducted a System Usability Scale (SUS) questionnaire, a quantitative evaluation method that measures the usability of a system. The results of our SUS were highly positive, with a score of 90, earning a grade of A+ and the adjective rating of "Best imaginable". This high score indicates that the participants found our video player interface to be highly usable and user-friendly.

Lastly, we conducted a User Experience Questionnaire (UEQ) survey, which focused on the participant's experiences when using our video player interface. The UEQ quantified users experiences across six dimensions, all of which gathered excellent results. This comprehensive feedback further validated the positive user experience and highlighted specific areas where our interface excels, as well as opportunities for further improvement.

In conclusion, the evaluation process gathered very positive feedback, highlighting the strengths of our video player interface features and their alignment with user expectations. However, it also revealed some issues that need to be addressed to further enhance the user experience.

6

Conclusions and Future Work

This chapter provides a comprehensive review of the dissertation, detailing its conclusions and the scope of the work accomplished. It concludes with a discussion in section 6.2 on potential future enhancements to further improve the video player interface.

6.1 Conclusions

In this dissertation, we proposed several features to be incorporated into the video player interface of current Android OTT platforms, aimed at improving the user experience of non-linear access to VOD content.

These features include a visual timeline that showcases all the video's key events, enabling users to jump to desired moments of the VOD with just one tap. Using the filters present in the timeline, the users can also customize the timeline to their desires. Another feature allows users to view the video's main events sequentially, akin to a highlight reel, with each event automatically playing after the previous one concludes, offering a seamless viewing experience. Further personalization is achieved with the "Highlight Moments" proposed feature that allows users to create personalized events within the video through crowdsourcing. Events with the same parameters marked by multiple users at a specific timestamp are integrated into the video's official events timeline, accessible to all users. Additionally, the interface helps users track their progress on the VOD, by allowing them to rewind to previous timestamps and resume from where they left off, ensuring continuous enjoyment.

The User Interface (UI) dynamically adjusts to different videos based on a JSON file, making it adaptable to a wide range of content genres, including talent shows, reality shows, news, and more.

During the research and design iterations of our solution, we systematically

gathered feedback from experts to ensure our features were the most user-friendly and efficient possible.

The proposed solutions are elaborated upon in Chapter 3, with practical implementation details discussed in Chapter 4.

Regarding the implemented features, we performed usability tests with users, which included a System Usability Scale (SUS) and a User Experience Questionnaire (UEQ). These tests resulted in a remarkably positive evaluation, as detailed in Chapter 5. Users enthusiastically embraced the integration of these features into current OTT services, highlighting a highly positive experience with the implemented features and recognizing the improved user experience while consuming VOD content.

6.2 Future Work

While our features received highly positive feedback during evaluation, there are opportunities for improvement.

One limitation we identified post-implementation of the highlight moments feature was regarding the calculation of the timestamp when adding a custom event to the timeline. Currently, we calculate the average timestamp of events with the same type around the same timestamp, but a more effective approach would be to use a weighted average timestamp. This method would assign weights to each timestamp based on the number of users who highlighted the event at that specific time or within a 20-second range. By giving more weight to timestamps with a higher number of highlights, we could get a more accurate timestamp to attribute to the saved event.

Another potential enhancement involves redesigning the timeline layout into two distinct formats. Firstly, a compact side menu would provide users with easy access to the timeline while watching the VOD, allowing for seamless interaction. Secondly, a detailed view option, accessible via a dedicated button in the menu, would contain our current layout with the filters and sequential event viewing functionality. This approach aims to make the timeline less intrusive, ensuring it does not dominate the entire video screen but instead serves as a convenient side menu.

Similarly, we could enhance the highlight events feature by transforming its layout into a side menu. This menu would include a scrollable list of all available events for users to choose from, along with parameters. Additionally, we could incorporate an information button to provide users with details about the

crowdsourcing mechanism. This adjustment would make the layout less intrusive and enable users to select the desired timestamp in the video player while saving a moment.

Another area of improvement concerns the progress tracking features, particularly focusing on the layout that represents the farthest timestamp the user has reached in the VOD. To make the layout more obvious and user-friendly, as mentioned in Chapter 5, a potential solution could involve redesigning the thumb to make it more distinctive. This could include enlarging the thumb, modifying its shape, or altering its color to clearly distinguish it from the video player's thumb.

We also believe it would be valuable to add the ability for users to locally save VOD events and moments. This feature would allow users to effortlessly bookmark specific timestamps within the video, facilitating easy access to these moments at a later time without having to wait for more users to select the events.

Finally, we will expand our testing to include a broader range of contexts, including news programs, concerts, talent shows, and more. This will allow us to assess the adaptability and effectiveness of our functionalities across various scenarios, ensuring continuous improvement and enhanced user experience.

Bibliography

[1] J. Cha. "Predictors of television and online video platform use: A coexistence model of old and new video platforms". In: *Telematics and Informatics* 30 (4 2013-11), pp. 296–310. ISSN: 0736-5853. DOI: 10.1016/J.TELE.2013.01.001 (cit. on pp. 1, 6).

[2] Y. N. K. Chen. "Competitions between OTT TV platforms and traditional television in Taiwan: A Niche analysis". In: *Telecommunications Policy* 43 (9 2019-10), p. 101793. ISSN: 0308-5961. DOI: 10.1016/J.TELPOL.2018.10.006 (cit. on pp. 1, 6).

[3] S. Parnami and T. Jain. "A Study on Increase in the Usage of OTT Streaming Services". In: *International Journal of Research in Engineering, Science and Management* 4.8 (2021-08), pp. 142–145. URL: https://journals.resaim.com/ijresm/article/view/1202 (cit. on p. 1).

[4] MarketsandMarkets. *Over-the-Top Services Market by Type, Monetization Model, Streaming Device, Vertical And Region - Global Forecast to 2024*. URL: https://www.reportlinker.com/p05838803/Over-the-Top-Services-Market-by-Type-Monetization-Model-Streaming-Device-Vertical-And-Region-Global-Forecast-to.html?utm_source=PRN (visited on 2023-01-01) (cit. on p. 1).

[5] M. Fudurić, E. C. Malthouse, and V. Viswanathan. "Keep it, shave it, cut it: A closer look into consumers' video viewing behavior". In: *Business Horizons* 61 (1 2018-01), pp. 85–93. ISSN: 0007-6813. DOI: 10.1016/J.BUSHOR.2017.09.008 (cit. on p. 1).

[6] A. Tefertiller. "Cable cord-cutting and streaming adoption: Advertising avoidance and technology acceptance in television innovation". In: *Telematics and Informatics* 51 (2020-08). ISSN: 07365853. DOI: 10.1016/j.tele.2020.101416 (cit. on p. 1).

[7] A. Tefertiller. "Media Substitution in Cable Cord-Cutting: The Adoption of Web-Streaming Television". In: *Journal of Broadcasting and Electronic Media* 62 (3 2018-07), pp. 390–407. ISSN: 15506878. DOI: 10.1080/08838151.2018.1451868. eprint: https://doi.org/10.1080/08838151.2018.1451868 (cit. on p. 1).

[8] A. Tefertiller and K. Sheehan. "TV in the Streaming Age: Motivations, Behaviors, and Satisfaction of Post-Network Television". In: *Journal of Broadcasting and Electronic Media* 63 (4 2019-10), pp. 595–616. ISSN: 15506878. DOI: 10.1080/08838151.2019.1698233 (cit. on p. 1).

[9] C. Annual and I. Report. *Cisco Annual Internet Report (2018–2023) White Paper*. 2018. URL: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf (cit. on p. 2).

[10] O. Budzinski, S. Gaenssle, and N. Lindstädt-Dreusicke. "The battle of YouTube, TV and Netflix: an empirical analysis of competition in audiovisual media markets". In: *SN Business Economics* 1 (9 2021-09). DOI: 10.1007/s43546-021-00122-0 (cit. on p. 2).

[11] M. A. Camilleri. "Understanding motivations to use online streaming services: integrating the technology acceptance model (TAM) and the uses and gratifications theory (UGT)". In: *Spanish Journal of Marketing - ESIC* (2020-12). ISSN: 2444-9709. DOI: 10.1108/SJME-04-2020-0074 (cit. on p. 2).

[12] S. Joshi, S. Sohag, T. Dewan, D. Chintan, S. Purohit, and G. Sumit. "Impact of Over the Top (OTT) Services on Telecom Service Providers". In: *Indian Journal of Science and Technology* 8 (2015-02), p. 145. DOI: 10.17485/ijst/2015/v8iS4/62238 (cit. on pp. 2, 6).

[13] Statista. *Video-on-Demand - Worldwide*. URL: https://www.statista.com/outlook/dmo/digital-media/video-on-demand/worldwide?currency=usd (visited on 2024-02-27) (cit. on pp. 2, 3).

[14] L. Barraclough. *Amazon Prime Video Available in More Than 200 Countries, Territories - Variety*. URL: https://variety.com/2016/digital/global/amazon-prime-video-now-available-in-more-than-200-countries-1201941818/ (visited on 2023-01-03) (cit. on p. 2).

[15] R. Benes. *Do People Stream More Licensed Video Content Than Original Content? - Insider Intelligence Trends, Forecasts Statistics*. 2019. URL: www.insiderintelligence.com/content/users-still-demand-licensed-content-from-ott-platforms (visited on 2023-01-01) (cit. on p. 2).

[16] B. Kotterink, I. Godlovitch, J. S. MARCUS, P. Nooren, J. Esmeijer, and A. Roosendaal. *Over-the-Top (OTTs) Players: Market Dynamics and Policy Challenges*. 2015. URL: https://www.europarl.europa.eu/RegData/etudes/STUD/2015/569979/IPOL_STU(2015)569979_EN.pdf (visited on 2023-01-12) (cit. on p. 6).

[17] K. Vieželytė. "Media Transformation in the Age of Netflix: User Interface and User Experience Trends of SVOD Platforms". Kaunas University of Technology, 2022. URL: https://epubl.ktu.edu/object/elaba:132872607/ (visited on 2023-01-01) (cit. on pp. 6, 7).

[18] M. Guevara. *Monetize your content with video ads*. 2021. URL: https://nunchee.com/blog/whats-the-best-ott-revenue-model-avod-svod-tvod (visited on 2023-01-19) (cit. on pp. 7, 8).

[19] "Industries-Media-and-Entertainment-OTT-Business-Models-Not-Everyone-Should-Be-Like-Netflix". In: *RedChalk Group* (2016). URL: https://www.redchalk.com/wp-content/uploads/2016/04/Industries-Media-and-Entertainment-OTT-Business-Models-Not-Everyone-Should-Be-Like-Netflix.pdf (visited on 2023-01-19) (cit. on pp. 7, 8).

[20] T. Z. Team. *OTT Business Models: A Full Breakdown on Making Revenue in OTT*. 2020-08. URL: https://www.zype.com/blog/types-of-ott-business-models (visited on 2023-01-19) (cit. on pp. 7, 8).

[21] E.-A. Park. "Business Strategies of Korean TV Players in the Age of Over-The-Top (OTT) Services". In: *The 43rd Research Conference on Communication, Information and Internet Policy Paper* (2015-03). URL: http://ssrn.com/abstract=2587540 (cit. on p. 7).

[22] E.-A. Park. "Prevalence of Business Models in Global OTT Video Services: A Cluster Analysis". In: *International Journal on Media Management* 21.3-4 (2019), pp. 177–192. DOI: 10.1080/14241277.2019.1695257 (cit. on pp. 7, 8).

[23] D. Pradhan. *SVOD, TVOD AVOD*. 2019-09. URL: https://medium.com/@i.durgesh.pradhan/svod-tvod-avod-9fd6648ff4d9 (visited on 2023-01-25) (cit. on pp. 7, 8).

[24] AppsFlyer. *SVOD (Subscription Video on Demand)*. URL: https://www.appsflyer.com/glossary/svod/ (visited on 2023-01-13) (cit. on p. 7).

[25] Anacom. *Servicos Over-The-Top 2022*. 2022. URL: https://www.anacom.pt/streaming/Relatorio_ServicosOverTheTop2023.pdf?contentId=1772078&field=ATTACHED_FILE (visited on 2023-01-12) (cit. on p. 9).

[26] S. News. *One in four Portuguese subscribe to a Streaming Platform*. URL: https://senalnews.com/en/research/one-in-four-portuguese-subscribe-to-a-streaming-platform (visited on 2023-01-12) (cit. on p. 9).

[27] G. Sinha, R. Shahi, and M. Shankar. "Human Computer Interaction". In: *3rd International Conference on Emerging Trends in Engineering and Technology* (2010), pp. 1–4. DOI: 10.1109/ICETET.2010.85 (cit. on p. 9).

[28] Adobe. *Human Computer Interaction (HCI)*. URL: xd.adobe.com/ideas/principles/human-computer-interaction (visited on 2023-01-28) (cit. on p. 10).

[29] N. Bevan. "International standards for HCI and usability". In: *International Journal of Human-Computer Studies* 55 (4 2001-10), pp. 533–552. ISSN: 1071-5819. DOI: 10.1006/IJHC.2001.0483 (cit. on p. 10).

[30] Simplilearn. *What is Human Computer Interaction? A Complete Guide to HCI | Simplilearn*. 2023. URL: https://www.simplilearn.com/what-is-human-computer-interaction-article (visited on 2023-01-28) (cit. on p. 10).

[31] C. R. Becker. URL: https://uxdesign.cc/a-comprehensive-list-of-human-computer-interactions-d72eaca2c0df (visited on 2023-01-30) (cit. on p. 10).

[32] *ISO - ISO 9241-11:2018 - Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts*. URL: https://www.iso.org/standard/63500.html (visited on 2023-01-30) (cit. on p. 10).

[33] N. Bevan, J. Carter, and S. Harker. "Iso 9241-11 revised: What have we learnt about usability since 1998?" In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9169 (2015), pp. 143–151. ISSN: 16113349. DOI: 10.1007/978-3-319-20901-2_13 (cit. on p. 10).

[34] D. Quiñones, C. Rusu, and V. Rusu. "A methodology to develop usability/user experience heuristics". In: *Computer Standards and Interfaces* 59 (2018-08), pp. 109–129. ISSN: 09205489. DOI: 10.1016/j.csi.2018.03.002 (cit. on p. 10).

[35] L. Punchoojit and N. Hongwarittorrn. "Usability Studies on Mobile User Interface Design Patterns: A Systematic Literature Review". In: *Advances in Human-Computer Interaction* (2017). DOI: 10.1155/2017/6787504. URL: https://doi.org/10.1155/2017/6787504 (cit. on p. 11).

[36] M. Hassenzahl, S. Diefenbach, and A. Göritz. "Needs, affect, and interactive products - Facets of user experience". In: *Interacting with Computers* 22 (5 2010), pp. 353–362. ISSN: 09535438. DOI: 10.1016/J.INTCOM.2010.04.002 (cit. on p. 11).

[37] J. Howe. "The Rise of Crowdsourcing". In: *Wired* 14 (2006) (cit. on p. 11).

[38] G. H. Kyung. "A suggestion for a smartphone video player interface". In: *Communications in Computer and Information Science* 529 (2015), pp. 27–31. ISSN: 18650929. DOI: 10.1007/978-3-319-21383-5_5 (cit. on p. 13).

[39] M. Hudelist and K. Schoeffmann. "An Evaluation of Video Browsing on Tablets with the ThumbBrowser". In: *International Conference on Multimedia Modeling* 10133 (2017-01), pp. 89–100. DOI: 10.1007/978-3-319-51814-5_8 (cit. on pp. 13, 14).

[40] K. Schoeffmann, K. Chromik, and L. Böszörmenyi. "Video navigation on tablets with multi-touch gestures". In: *2014 IEEE International Conference on Multimedia and Expo Workshops, ICMEW 2014* (2014-07), pp. 1–6. DOI: 10.1109/ICMEW.2014.6890560 (cit. on p. 14).

[41] K. Schoeffmann and L. Burgstaller. "Scrubbing Wheel: An Interaction Concept to Improve Video Content Navigation on Devices with Touchscreens". In: *2015 IEEE International Symposium on Multimedia (ISM)*. 2015-03, pp. 351–356. DOI: 10.1109/ISM.2015.20 (cit. on pp. 14, 15).

[42] A. Carlier, V. Charvillat, and W. T. Ooi. "A video timeline with bookmarks and prefetch state for faster video browsing". In: *MM 2015 - Proceedings of the 2015 ACM Multimedia Conference* (2015-10), pp. 967–970. DOI: 10.1145/2733373.2806376 (cit. on pp. 15, 16).

[43] A. Janin, L. Gottlieb, and G. Friedland. "Joke-o-Mat HD: Browsing Sitcoms with Human Derived Transcripts". In: *Proceedings of the 18th ACM International Conference on Multimedia*. MM '10. Firenze, Italy: Association for Computing Machinery, 2010, pp. 1591–1594. ISBN: 9781605589336. DOI: 10.1145/1873951.1874295 (cit. on p. 16).

[44] A. Al-Hajri, G. Miller, M. Fong, and S. Fels. "Visualization of personal history for video navigation". In: Association for Computing Machinery, 2014, pp. 1187–1196. ISBN: 9781450324731. DOI: 10.1145/2556288.2557106 (cit. on p. 17).

[45] A. Developer. *Android Development Fundamentals*. URL: https://developer.android.com/guide/components/fundamentals (visited on 2024-03-10) (cit. on p. 36).

[46] A. Developer. *Android Activity Lifecycle*. URL: https://developer.android.com/reference/android/app/Activity#ActivityLifecycle (visited on 2024-03-10) (cit. on p. 37).

[47] A. Developer. *Layouts in Views*. URL: https://developer.android.com/develop/ui/views/layout/declaring-layout (visited on 2024-03-10) (cit. on p. 38).

[48] THEOplayer. *Vualto chooses THEOplayer for HTML5 cross-platform streaming*. URL: https://www.theoplayer.com/blog/vualto-chooses-theoplayer-for-html5-cross-platform-streaming (visited on 2024-03-10) (cit. on p. 39).

[49] *THEOplayer Demo Overview Page*. URL: https://www.theoplayer.com/theoplayer-demo-overview (visited on 2023-02-01) (cit. on p. 39).

[50] J. Brooke. "SUS: A quick and dirty usability scale". In: *Usability Eval. Ind.* 189 (1995-11) (cit. on pp. 58, 64).

[51] J. T. Andreas Hinderks Martin Schrepp. *User Experience Questionnaire*. URL: https://www.ueq-online.org/ (cit. on pp. 59, 67).

[52] *Likert Scale*. URL: https://www.simplypsychology.org/likert-scale.html (visited on 2023-02-02) (cit. on p. 60).

[53] B. Klug. *An Overview of the System Usability Scale in Library Website and System Usability Testing*. URL: https://quod.lib.umich.edu/w/weave/12535642.0001.602?view=text;rgn=main (visited on 2024-03-20) (cit. on p. 64).

[54] J. Sauro. *Measuring Usability with the System Usability Scale (SUS)*. URL: https://measuringu.com/sus (visited on 2024-03-20) (cit. on p. 65).

[55] J. S. James R. Lewis. "The Factor Structure of the System Usability Scale". In: *Human Centered Design* 5619 (2009-12), p. 10. DOI: 10.1007/978-3-642-02806-9_12 (cit. on p. 65).

[56] J. Sauro. *5 Ways to Interpret a SUS Score*. URL: https://measuringu.com/interpret-sus-score/ (visited on 2024-03-20) (cit. on p. 66).

[57] M. Schrepp. *User Experience Questionnaire Handbook*. URL: https://www.ueq-online.org/Material/Handbook.pdf (visited on 2024-03-20) (cit. on p. 67).

I

First usability test script

Teste de Usabilidade – Protótipo Inicial

Dissertação "IMPROVING THE USER EXPERIENCE ON NON-LINEAR ACCESS TO VOD CONTENT" de Sara Relvas

O objetivo deste teste de usabilidade é avaliar a eficácia, eficiência e satisfação dos utilizadores ao utilizar plataformas de Streaming de vídeo, especificamente em relação a novas funcionalidades adicionadas ao leitor de vídeo. Estas novas funcionalidades incluem a possibilidade de ver a linha temporal de eventos do vídeo em detalhe, tendo uma ideia de onde acontece cada evento chave do vídeo, filtrar a linha temporal de eventos por vários parâmetros e guardar momentos personalizados na linha temporal de eventos. Também é possível anular a ação de avançar para uma *timestamp* ou mesmo avançar para a *timestamp* mais longe do vídeo em que já esteve, para continuar a visualização do vídeo a partir do ponto onde o deixou anteriormente. Estas funcionalidades irão adaptar-se a todo o tipo de conteúdo, desde séries, a jogos de futebol, filmes, noticiários, entre outros.

Este teste de usabilidade é destinado a utilizadores finais de plataforma de *streaming* de vídeo como Netflix, Youtube, SPORT TV. Estes utilizadores já estão familiarizados com a plataforma atual, mas precisarão de se adaptar às novas funcionalidades adicionadas.

Todas as novas funcionalidades foram projetadas para melhorar a experiência do utilizador e tornar o consumo de conteúdo mais personalizado e agradável. Por exemplo, os eventos importantes na *timeline* de eventos permitirão que os utilizadores saltem para as partes mais interessantes do vídeo, enquanto a filtragem da *timeline*, por diferentes parâmetros, permitirá que os utilizadores encontrem rapidamente o que procuram. Além disso, a criação de eventos personalizados permitirá ao utilizador criar eventos que fiquem na timeline oficial do vídeo, isto é, com o uso de *crowdsourcing*, se vários utilizadores marcarem um espaço de tempo com um evento específico, este acabará por ser integrado na timeline de eventos oficial do vídeo, estando disponível paraa todos os utilizadores.

Durante este teste, serão realizadas 8 tarefas simples sobre um cenário específico. Serás convidado a interagir com a plataforma e a realizar as tarefas propostas, enquanto um moderador observa e registra as interações do utilizador com o *player* de vídeo.

No final do teste, serás solicitado a fornecer feedback sobre a tua experiência de uso do *Player* de vídeo, incluindo sugestões de melhorias e pontos positivos e negativos da plataforma. Esses dados serão analisados, a fim de identificar possíveis problemas e pontos fortes do *Player* de vídeo. Com base nas informações registadas, iremos fazer ajustes e melhorias no *Player* de vídeo, de modo a oferecer uma experiência de visualização mais agradável e eficiente para os utilizadores da plataforma de *streaming*.

Tarefas

Cenário

Chegaste a casa do trabalho e perdeste um jogo de futebol do teu clube favorito, o Portimonense. Entras na plataforma de *streaming* de jogos de futebol e clicas para ver a repetição do jogo. São te apresentadas novas funcionalidades no *Player* de vídeo que decides usar para melhorar e personalizar a tua experiência de visualização do jogo.

1 – Vai para a timeline de eventos do jogo.

2 – Vai para o momento do terceiro golo do Benfica.

3 – Vai para os 20 minutos de vídeo e guarda o momento atual do vídeo como uma "Magic Skill" da equipa "Benfica". Como 20 outros utilizadores guardaram exatamente esse momento, este ficou guardado na *timeline* oficial do vídeo, disponível para todos os utilizadores da plataforma.

4 – Vai para a *timestamp* do vídeo mais longe em que já estiveste, ou seja, para continuares a ver o vídeo de onde ainda não viste.

5 – Anula a tua ação anterior

6 – Vê todos os momentos da *timeline* de eventos de maneira automática.

7 – Filtra a tua *timeline* de eventos por eventos de ambas as equipas, do tipo "*Ruled out goals*" e "*Yellow cards*" e por todos os jogadores que realizaram algum desses eventos.

8 - Vê todos os momentos filtrados da *timeline* de eventos de maneira automática.

9 – Retira todos os filtros anteriores e filtra novamente a *timeline* por eventos do Benfica, de todos os tipos e pelo jogador Gonçalo Ramos.

Muito Obrigada!

II

Usability Questionnaire for First Usability Test

Teste de Usabilidade de Leitor de Vídeo - Tese Sara Relvas

Este formulário serve de apoio aos testes de usabilidade do protótipo inicial do leitor de vídeo da dissertação "Improving the User Experience of non-linear access to VOD content" da estudante Sara Relvas. Este formulário é anónimo e apenas será usado para fins de análise para a possível melhoria do produto, bem como possível identificação de problemas de utilização.

* Indica uma pergunta obrigatória

1. Idade *

2. Género *

Marcar apenas uma oval.

- ◯ Masculino
- ◯ Feminino
- ◯ Não binário
- ◯ Outro
- ◯ Prefiro não dizer

3. Costuma utilizar plataformas de Streaming, como por exemplo Netflix, SportTV ou Disney+? *

Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

4. Se respondeu sim à pergunta anterior, qual é a regularidade com que utiliza plataformas de Streaming?

Marcar apenas uma oval.

- ◯ Raramente
- ◯ Uma vez por semana
- ◯ Entre 1 a 3 vezes por semana
- ◯ Mais de 3 vezes por semana
- ◯ Todos os dias

Em relação à utilização do player de vídeo apresentado

5. Quais as funcionalidades que lhe interessaram mais? *

Marcar tudo o que for aplicável.

- ☐ Timeline Vertical de Eventos.
- ☐ Criação de eventos personalizados na timeline através de crowdsourcing.
- ☐ Filtragem de conteúdo por diferentes parâmetros.
- ☐ Possibilidade de avançar para o ponto mais longe do vídeo em que esteve.
- ☐ Possibilidade de anular a ação anterior quando passa para uma nova timestamp e assim retornar à timestamp anterior.
- ☐ Nenhuma

6. Estaria interessado em ver estas funcionalidades presente nas plataformas de Streaming? *

Marcar apenas uma oval.

- ◯ Sim
- ◯ Não

7. Achou alguma funcionalidade desnecessária? Se sim, qual e porquê?

8. Notou falta de alguma funcionalidade? Se sim, quais?

9. Pontos positivos da utilização do player de vídeo

10. Pontos negativos da utilização do player de vídeo

11. Sugestões e possíveis melhorias

Muito obrigada pelo seu tempo!

Este conteúdo não foi criado nem aprovado pela Google.

Google Formulários

III

JSON File used for testing

```
{
    "vod_type": "soccer_game",
    "vod_filters": [
        {
            "name": "goal",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        },
        {
            "name": "card",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        },
        {
            "name": "VAR",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        }
    ],
    "custom_filter": true,
    "custom_filter_types": ["team, event, player"],
    "crowd_sourcing": true,
    "crowd_sourcing_types": [
        {
            "name": "Major Opportunity",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        },
        {
            "name": "Beautiful Play",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        },
        {
            "name": "Great Tackle",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        },
        {
            "name": "Magic Skill",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        },
        {
            "name": "Amazing Save",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        },
        {
            "name": "Doubtful Decision",
            "icon": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        }
    ],
    "vod_info": {
        "team_home": "ESP",
        "team_away": "BAR",
        "team_home_logo": "https://api.sofascore.app/api/v1/team/2814/image",
        "team_away_logo": "https://api.sofascore.app/api/v1/team/2817/image",
        "team_home_jersey_color": "#173E87",
        "team_away_jersey_color": "#E4BC00",
        "final_result": "2 - 4",
        "team_home_full": "ESPANYOL",
        "team_away_full": "BARCELONA"
    },
```

```
"vod_events": [
    {
        "type": "kick_off",
        "seconds_start": 100,
        "seconds_end": -500,
        "minute": 0,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "yellow card",
        "team": "BAR",
        "player1_name": "Jules Koundé",
        "player1_photo": "https://api.sofascore.app/api/v1/player/827212/image",
        "seconds_start": 170,
        "seconds_end": 287,
        "minute": 1,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "goal",
        "team": "BAR",
        "player1_name": "R. Lewandowski",
        "player1_photo": "https://api.sofascore.app/api/v1/player/41789/image",
        "seconds_start": 706,
        "seconds_end": 792,
        "minute": 10,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "goal",
        "team": "BAR",
        "player1_name": "A. Baldé",
        "player1_photo": "https://api.sofascore.app/api/v1/player/997035/image",
        "seconds_start": 1269,
        "seconds_end": 1360,
        "minute": 19,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "yellow card",
        "team": "BAR",
        "player1_name": "Gavi",
        "player1_photo": "https://i.postimg.cc/cLF9JGbZ/1bf058fb42cb87ad15ed7125b3c39264.png",
        "seconds_start": 2365,
        "seconds_end": 2423,
        "minute": 37,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "yellow card",
        "team": "ESP",
        "player1_name": "Sergi Darder",
        "player1_photo": "https://api.sofascore.app/api/v1/player/110783/image",
        "seconds_start": 2365,
        "seconds_end": 2423,
        "minute": 37,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
```

```
{
    "type": "goal",
    "team": "BAR",
    "player1_name": "R. Lewandowski",
    "player1_photo": "https://api.sofascore.app/api/v1/player/41789/image",
    "seconds_start": 2489,
    "seconds_end": 2562,
    "minute": 39,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "substitution",
    "team": "ESP",
    "player1_name": "Javi Puado",
    "player1_photo": "https://api.sofascore.app/api/v1/player/891511/image",
    "player2_name": "Sergi Gómez",
    "player2_photo": "https://api.sofascore.app/api/v1/player/125625/image",
    "seconds_start": 3202,
    "seconds_end": 3217,
    "minute": 46,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "substitution",
    "team": "ESP",
    "player1_name": "F. Calero",
    "player1_photo": "https://api.sofascore.app/api/v1/player/857205/image",
    "player2_name": "Denis Suárez",
    "player2_photo": "https://api.sofascore.app/api/v1/player/138383/image",
    "seconds_start": 3202,
    "seconds_end": 3217,
    "minute": 46,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "half_time",
    "seconds_start": 3240,
    "seconds_end": -500,
    "minute": 46,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "goal",
    "team": "BAR",
    "player1_name": "Jules Koundé",
    "player1_photo": "https://api.sofascore.app/api/v1/player/827212/image",
    "seconds_start": 3685,
    "seconds_end": 3710,
    "minute": 52,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "substitution",
    "team": "ESP",
    "player1_name": "Adrià Pedrosa",
    "player1_photo": "https://api.sofascore.app/api/v1/player/928672/image",
    "player2_name": "Martin Braithwaite",
    "player2_photo": "https://i.postimg.cc/rmrQg0FF/582fca5afe9a90eb2fd0299218be2cb1.png",
    "seconds_start": 3826,
    "seconds_end": 3842,
    "minute": 55,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
```

```
{
    "type": "substitution",
    "team": "BAR",
    "player1_name": "O. Dembélé",
    "player1_photo": "https://api.sofascore.app/api/v1/player/818244/image",
    "player2_name": "Raphinha",
    "player2_photo": "https://api.sofascore.app/api/v1/player/831005/image",
    "seconds_start": 4245,
    "seconds_end": 4274,
    "minute": 62,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "substitution",
    "team": "BAR",
    "player1_name": "Jordi Alba",
    "player1_photo": "https://api.sofascore.app/api/v1/player/35199/image",
    "player2_name": "R. Araújo",
    "player2_photo": "https://api.sofascore.app/api/v1/player/925097/image",
    "seconds_start": 4245,
    "seconds_end": 4274,
    "minute": 62,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "goal",
    "team": "ESP",
    "player1_name": "Javi Puado",
    "player1_photo": "https://api.sofascore.app/api/v1/player/891511/image",
    "seconds_start": 4872,
    "seconds_end": 4961,
    "minute": 72,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "substitution",
    "team": "BAR",
    "player1_name": "M. Alonso",
    "player1_photo": "https://api.sofascore.app/api/v1/player/69408/image",
    "player2_name": "A. Christensen",
    "player2_photo": "https://api.sofascore.app/api/v1/player/186795/image",
    "seconds_start": 4970,
    "seconds_end": 4970,
    "minute": 74,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
{
    "type": "substitution",
    "team": "BAR",
    "player1_name": "Ansu Fati",
    "player1_photo": "https://api.sofascore.app/api/v1/player/962883/image",
    "player2_name": "Gavi",
    "player2_photo": "https://i.postimg.cc/cLF9JGbZ/1bf058fb42cb87ad15ed7125b3c39264.png",
    "seconds_start": 4970,
    "seconds_end": 4970,
    "minute": 74,
    "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
},
```

```
    {
        "type": "yellow card",
        "team": "BAR",
        "player1_name": "Jordi Alba",
        "player1_photo": "https://api.sofascore.app/api/v1/player/35199/image",
        "player2_name": "",
        "player2_photo": "",
        "seconds_start": 5405,
        "seconds_end": 5434,
        "minute": 82,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "substitution",
        "team": "ESP",
        "player1_name": "Luca Koleosho",
        "player1_photo": "https://api.sofascore.app/api/v1/player/1216282/image",
        "player2_name": "Brian Oliván",
        "player2_photo": "https://api.sofascore.app/api/v1/player/351500/image",
        "seconds_start": 5559,
        "seconds_end": 5591,
        "minute": 84,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "substitution",
        "team": "ESP",
        "player1_name": "Edu Expósito",
        "player1_photo": "https://api.sofascore.app/api/v1/player/877262/image",
        "player2_name": "N. Melamed",
        "player2_photo": "https://api.sofascore.app/api/v1/player/989878/image",
        "seconds_start": 5705,
        "seconds_end": 5743,
        "minute": 86,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "substitution",
        "team": "BAR",
        "player1_name": "Franck Kessié",
        "player1_photo": "https://api.sofascore.app/api/v1/player/359226/image",
        "player2_name": "Pedri",
        "player2_photo": "https://api.sofascore.app/api/v1/player/992587/image",
        "seconds_start": 5864,
        "seconds_end": 5890,
        "minute": 89,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },
    {
        "type": "goal",
        "team": "ESP",
        "player1_name": "Joselu",
        "player1_photo": "https://api.sofascore.app/api/v1/player/69418/image",
        "seconds_start": 6046,
        "seconds_end": 6065,
        "minute": 92,
        "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
    },

        {
            "id": "23",
            "type": "full_time",
            "seconds_start": 6075,
            "seconds_end": -500,
            "minute": 92,
            "icon:": "https://i.postimg.cc/3wzv3VtV/1bf058fb42cb87ad15ed7125b3c39264.png"
        }
    ]
}
```

IV

Final Usability test script

Improving the User Experience on non-linear access to VOD Content - Final prototype usability test

The purpose of this usability test is to evaluate the effectiveness, efficiency, and user satisfaction when using video streaming platforms. Specifically, the focus is on new features added to the video player. These features were developed within the context of a master's dissertation developed in the Computer Science Department of NOVA School of Science and Technology.

These features are designed to cater to a wide range of content, from TV series and football games to movies and news broadcasts. The usability test targets end-users of video streaming platforms such as Netflix, Youtube, and sport tv. While these users are already familiar with the current platform, they will need to adapt to these new functionalities.

This study is not going to evaluate any capabilities or knowledge of the test volunteers, but instead, their experience with the application. Take your time to complete the assigned tasks and share your genuine feedback on your experience. Should you encounter difficulties or need clarification, don't hesitate to ask questions. While completing the tasks, think aloud, and share your thoughts, impressions, and any concerns that come to mind. This will provide valuable insights into your decision-making process and overall user experience.

Throughout the test, you will engage in twelve straightforward tasks within a specific scenario. A moderator will observe and record your interactions with the video player. At the conclusion of the test, you will be invited to share feedback on your experience with the video player. This includes providing suggestions for improvements and the identification of both positive and negative aspects of the platform. The collected data will undergo analysis to pinpoint potential issues and strengths of the video player. Subsequently, adjustments and enhancements will be implemented, aiming to deliver a more enjoyable and efficient viewing experience for platform users.

Any feedback collected is confidential, and the analysis and results of this study will maintain that anonymity. By proceeding, you confirm that you have read and understood the purpose of this study, the procedures involved, and that you consent to participate in the test.

Tasks

Scenario:

You arrive home from work and realize you missed a soccer match of your favorite club, Barcelona. You log into the soccer game streaming platform and click to watch the replay. New features on the video player are presented to you, which you decide to utilize to enhance and customize your viewing experience of the game.

1 - Navigate to the match events timeline.

2 - Navigate to the moment of Barcelona's fourth goal and watch it.

3 - Automatically view all key events that happened in this match. After you have seen the third event, go to the task 4.

4 - Go to the 45-minute mark in the video and save the current video moment as a "Magic Skill" for the "Barcelona" team. Since 20 other users have saved this exact moment, it is now stored in the official video timeline, accessible to all platform users.

5 – Go to the previous timestamp you were.

6 - Filter your timeline by Card events.

7 - Automatically view all moments filtered in the events timeline. After you have seen the third event, pass to the task 8.

8 – See the moment that you and 19 other users saved.

9 – Navigate to the farthest timestamp in the video where you were.

10 - Filter the timeline for events related to Barcelona, of type substitution and by the player Gavi.

11 – Watch all the goals and viewer events of the match. Viewer events are the events that were created by crowdsourcing.

12 – Pause the video.

Thank you.

V

Usability Questionnaire for Final Usability Test

Enhanced Video Streaming Platform Usability Test

We appreciate your participation in this study, which aims to understand how our new functionalities impact your user experience. Your feedback will be instrumental in improving our platform's usability and functionality.

Please take your time to complete the tasks and provide your genuine feedback. Remember, your answers are anonymous and will be used solely for research purposes.
We value your honesty and openness in sharing your thoughts, impressions, and any concerns that come to mind. Your feedback will help us identify potential issues and strengths of the platform, ultimately leading to improvements for a more enjoyable and efficient viewing experience.
Thank you for your time.

* Indica uma pergunta obrigatória

1. Age *

2. Sex *

Marcar apenas uma oval.

- ◯ Male
- ◯ Female
- ◯ Other

3. What is your occupation? *

4. How often do you watch videos on streaming platforms like Netflix, YouTube, sport tv, Disney+, HBO Max, Amazon prime video and others? *

   Marcar apenas uma oval.

   - ◯ Daily
   - ◯ Several times a week
   - ◯ Once or twice a week
   - ◯ Rarely
   - ◯ Never

5. Do you use any specific features or functionalities regularly?

6. Have you ever encountered any usability issues or difficulties while using these platforms?

Usability and Functionality Feedback

This section is dedicated to gathering your insights on the usability and functionality of our application. Your answers will provide valuable information on how intuitive and efficient our new functionalities are.

7. I think that the timeline is intuitive. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

8. I found that it was easy to understand where each key event happens in the video. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

9. I thought that the events timeline was easy to navigate. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

10. Did you encounter any issues while viewing the key events timeline? If yes, please describe them.

11. Filtering the event timeline by various parameters was intuitive. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

12. The filtering options were appropriate in the context of the video. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

13. Were any other parameters for filtering that you missed? Was any of them unnecessary?

14. Did you encounter any issues while filtering the event timeline? If yes, please describe them.

15. Creating custom moments in the event timeline was intuitive. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

16. It was easy to save custom moments in the event timeline. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

17. The saved moments appeared clearly on the event timeline. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

18. Did you encounter any issues while creating custom moments? If yes, please describe them.

19. Tracking my progress in the video was easy. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

20. Jumping back to the timestamp that I previously watching in the video was easy. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

21. It was simple to navigate to the farthest timestamp of the video that I already saw. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

22. Did you encounter any issues while tracking your video progress? If yes, please describe them.

23. Overall, how would you rate your experience with the video timeline and its features? *

Marcar apenas uma oval.

- ◯ Very poor
- ◯ Poor
- ◯ Neutral
- ◯ Good
- ◯ Excellent

24. I believe these new features improve the overall user experience on the platform. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly agree

25. I would be interested in seeing these functionalities implemented on Streaming platforms. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

26. Did you find any functionality unnecessary? If so, which one and why?

27. Did you notice any missing functionalities? If so, what are they?

28. Is there anything else you would like to tell us about your experience?

System Usability Scale Questions

This section includes questions designed to measure the usability of our application through the System Usability Scale (SUS). Your feedback will help us understand how intuitive and easy to use our application is. Please rate each statement on a scale of 1-5, where 1 means "Strongly Disagree" and 5 means "Strongly Agree".

29. I think that I would like to use this new video player functionalities frequently. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

30. I found the new functionalities unnecessarily complex. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

31. I thought the new functionalities were easy to use. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

32. I think that I would need the support of a technical person to be able to use the new functionalities. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

33. I found the various functions in this video player were well integrated. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

34. I thought there was too much inconsistency in the video player. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

35. I would imagine that most people would learn to use these new functionalities very quickly. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

36. I found the new functionalities really hard to use. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

37. I felt very confident using the video player with the new functionalities. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

38. I needed to learn a lot of things before I could get going with this new functionalities of the video player. *

Marcar apenas uma oval.

	1	2	3	4	5	
Stro	◯	◯	◯	◯	◯	Strongly Agree

User Experience Questionnaire

Please make your evaluation now.

For the assessment of the product, please fill out the following questionnaire. The questionnaire consists of pairs of contrasting attributes that may apply to the product. The circles between the attributes represent gradations between the opposites. You can express your agreement with the attributes by ticking the circle that most closely reflects your impression.

Example:

attractive	○	⊗	○	○	○	○	○ unattractive

This response would mean that you rate the application as more attractive than unattractive.

Please decide spontaneously. Don't think too long about your decision to make sure that you convey your original impression.

Sometimes you may not be completely sure about your agreement with a particular attribute or you may find that the attribute does not apply completely to the particular product. Nevertheless, please tick a circle in every line.

It is your personal opinion that counts. Please remember: there is no wrong or right answer!

39. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
ann	◯	◯	◯	◯	◯	◯	◯	enjoyable

40. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
not	◯	◯	◯	◯	◯	◯	◯	understandable

41. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
crea	◯	◯	◯	◯	◯	◯	◯	dull

42. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
easy	◯	◯	◯	◯	◯	◯	◯	difficult to learn

43. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
valu	◯	◯	◯	◯	◯	◯	◯	inferior

44. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
bori	◯	◯	◯	◯	◯	◯	◯	exciting

45. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
not i	◯	◯	◯	◯	◯	◯	◯	interesting

46. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
unpı	◯	◯	◯	◯	◯	◯	◯	predictable

47. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
fast	◯	◯	◯	◯	◯	◯	◯	slow

48. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
inve	◯	◯	◯	◯	◯	◯	◯	conventional

49. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
obst	◯	◯	◯	◯	◯	◯	◯	supportive

50. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
goo	◯	◯	◯	◯	◯	◯	◯	bad

51. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
com	◯	◯	◯	◯	◯	◯	◯	easy

52. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
unli	◯	◯	◯	◯	◯	◯	◯	pleasing

53. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
usu	◯	◯	◯	◯	◯	◯	◯	leading edge

54. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
unp	◯	◯	◯	◯	◯	◯	◯	pleasant

55. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
secu	◯	◯	◯	◯	◯	◯	◯	not secure

56. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
mot	◯	◯	◯	◯	◯	◯	◯	demotivating

57. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
mee	◯	◯	◯	◯	◯	◯	◯	does not meet expectations

58. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
inefi	◯	◯	◯	◯	◯	◯	◯	efficient

59. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
clea	◯	◯	◯	◯	◯	◯	◯	confusing

60. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
impı	◯	◯	◯	◯	◯	◯	◯	practical

61. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
orga	◯	◯	◯	◯	◯	◯	◯	cluttered

62. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
attra	◯	◯	◯	◯	◯	◯	◯	unattractive

63. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
frier	◯	◯	◯	◯	◯	◯	◯	unfriendly

64. *

Marcar apenas uma oval.

	1	2	3	4	5	6	7	
con:	◯	◯	◯	◯	◯	◯	◯	innovative

Thank you for your time.

Este conteúdo não foi criado nem aprovado pela Google.

Google Formulários

VI

Usability Test Questionnaire Results

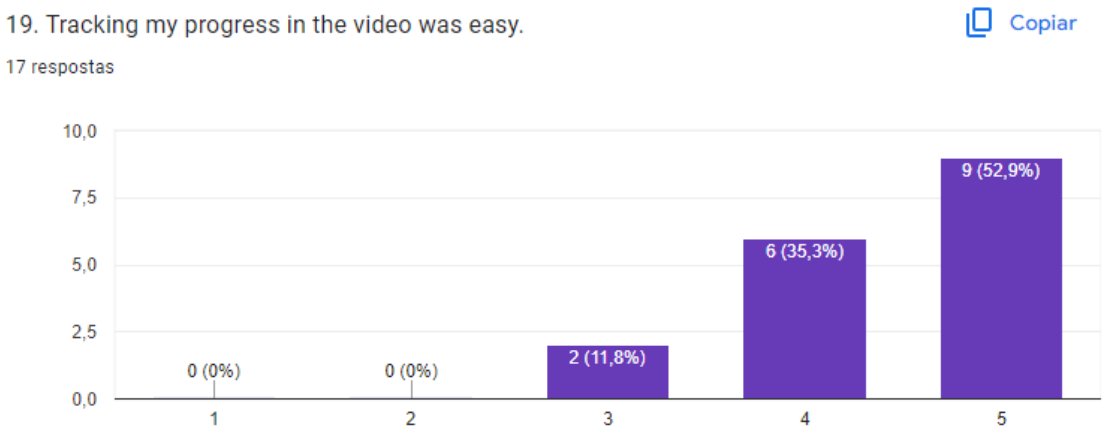

Figure VI.9: Tracking progress ease

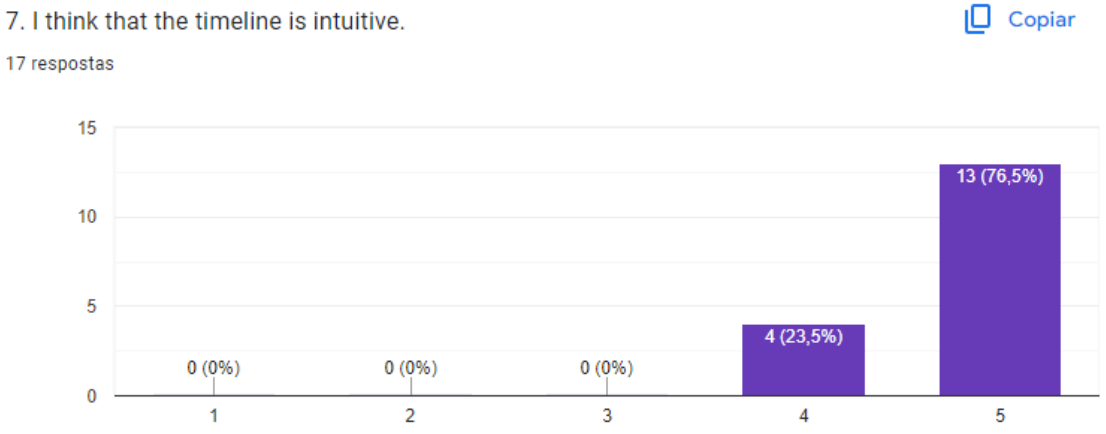

Figure VI.1: Intuitiveness of the VOD events timeline

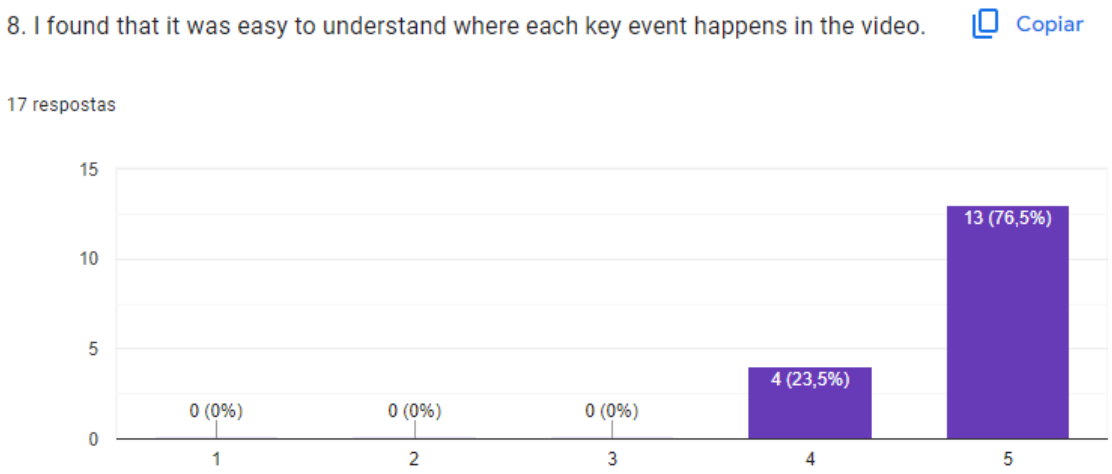

Figure VI.2: Key event clarity on timeline

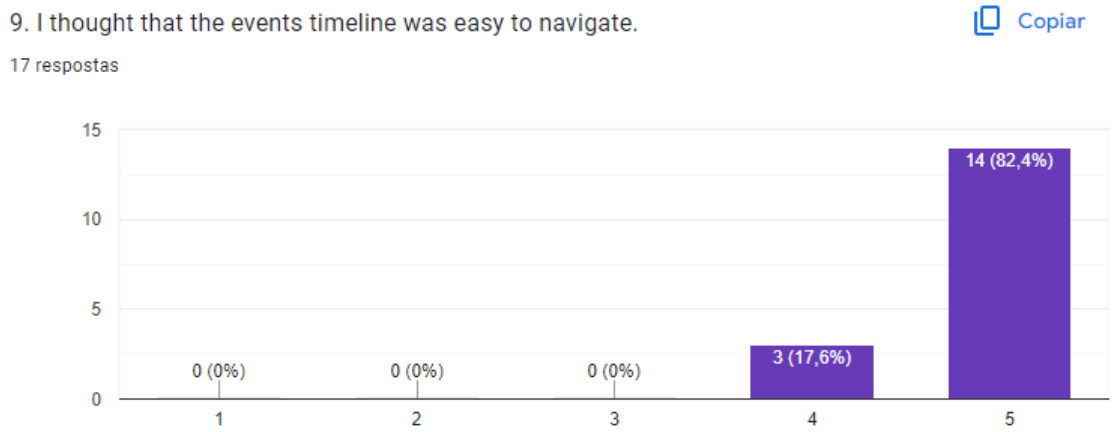

Figure VI.3: VOD events timeline navigation ease

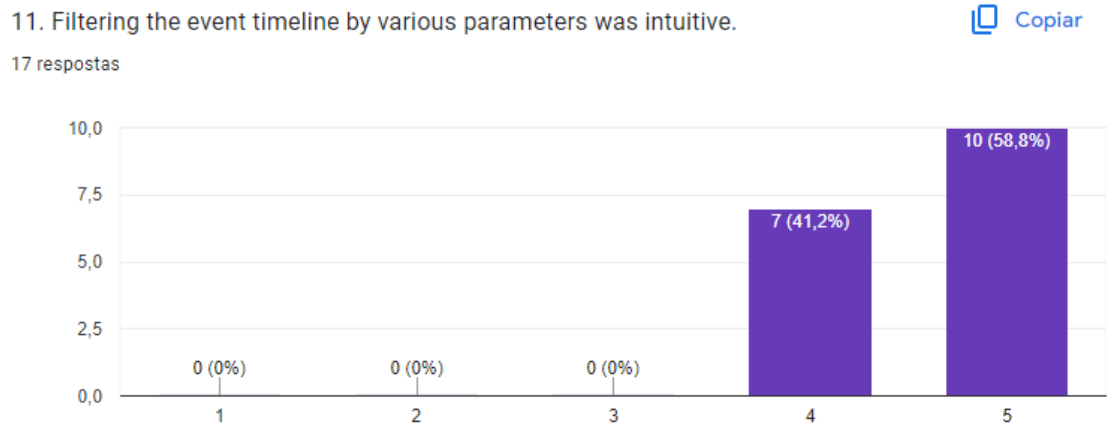

Figure VI.4: Filtering intuitiveness

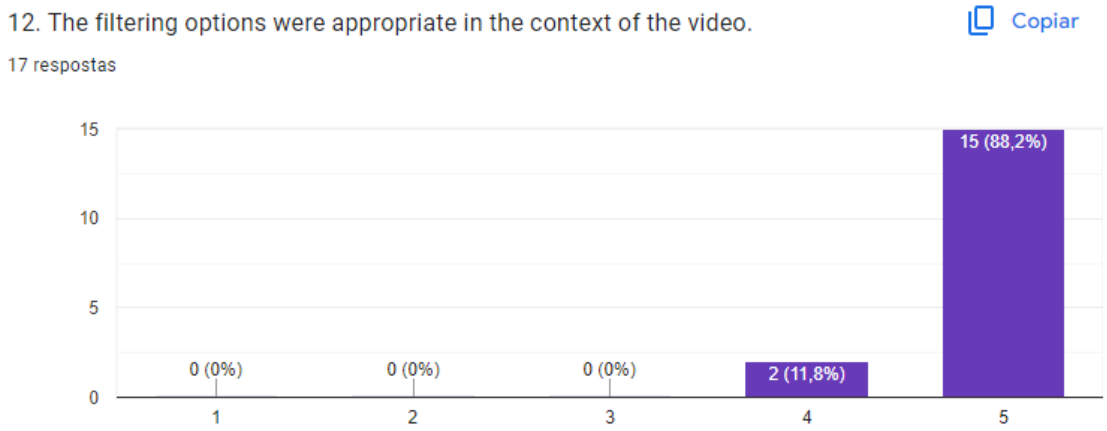

Figure VI.5: Feedback on filtering parameters

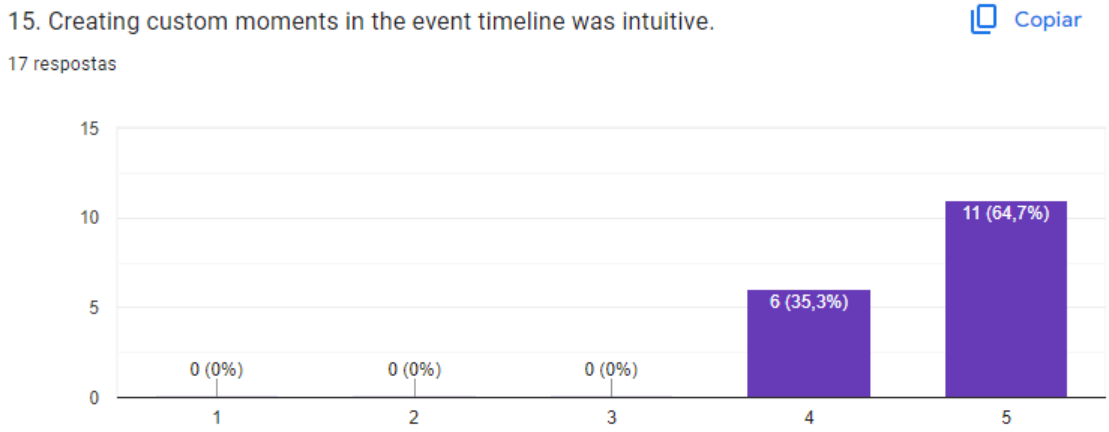

Figure VI.6: Creating custom moments intuitiveness

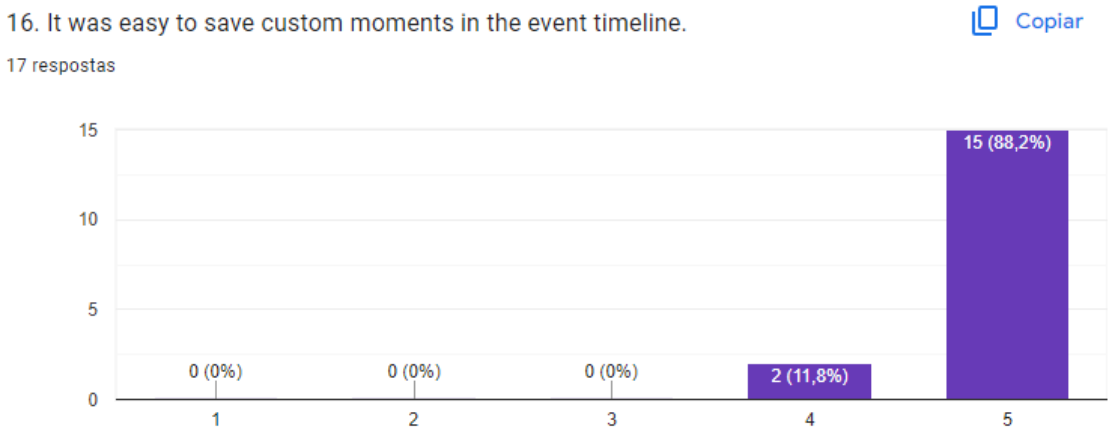

Figure VI.7: Saving custom moments ease

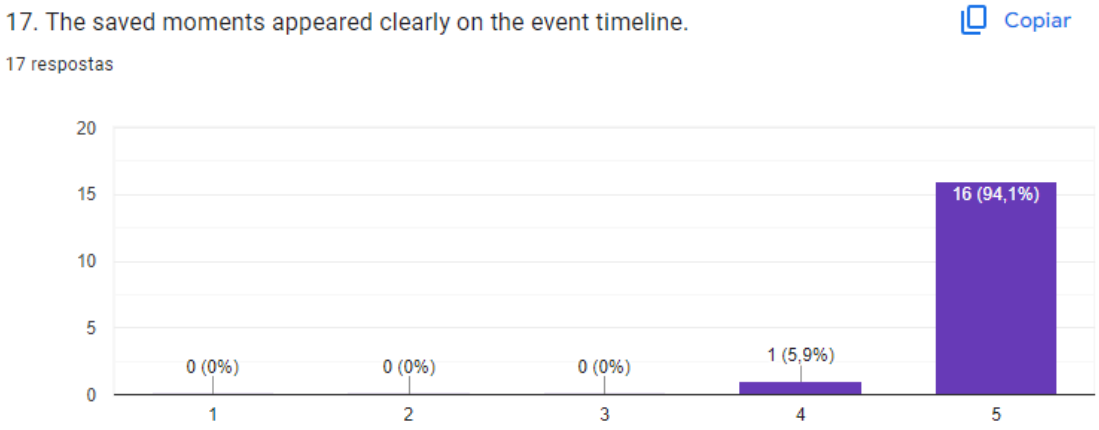

Figure VI.8: Saved moments clarity on timeline

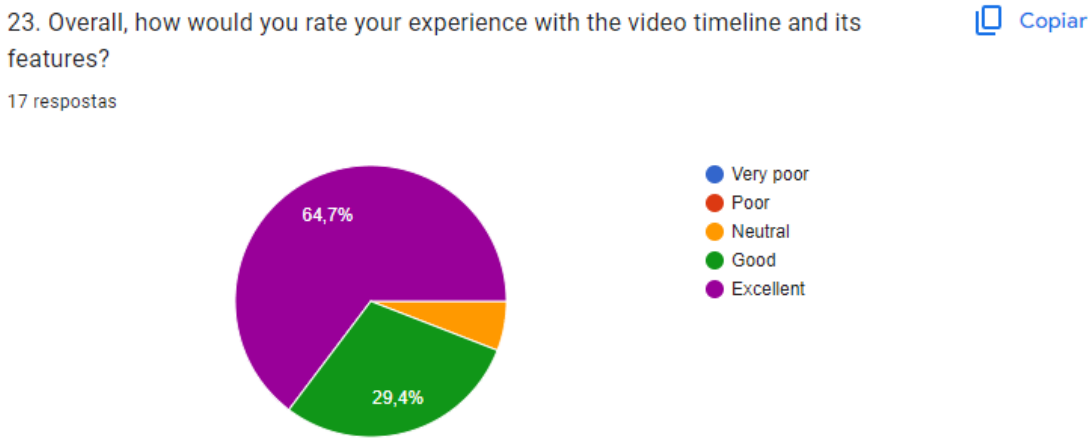

Figure VI.10: Overall user experience rating

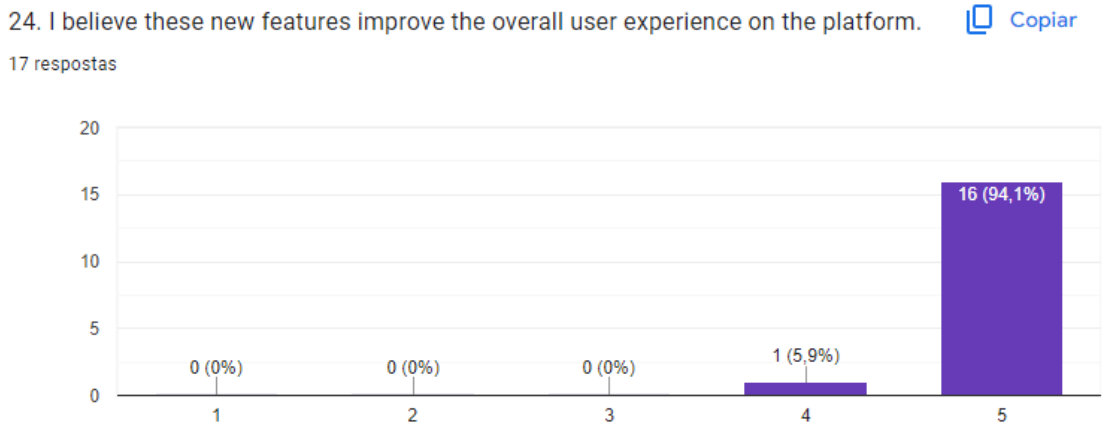

Figure VI.11: Feature improvement on streaming platforms

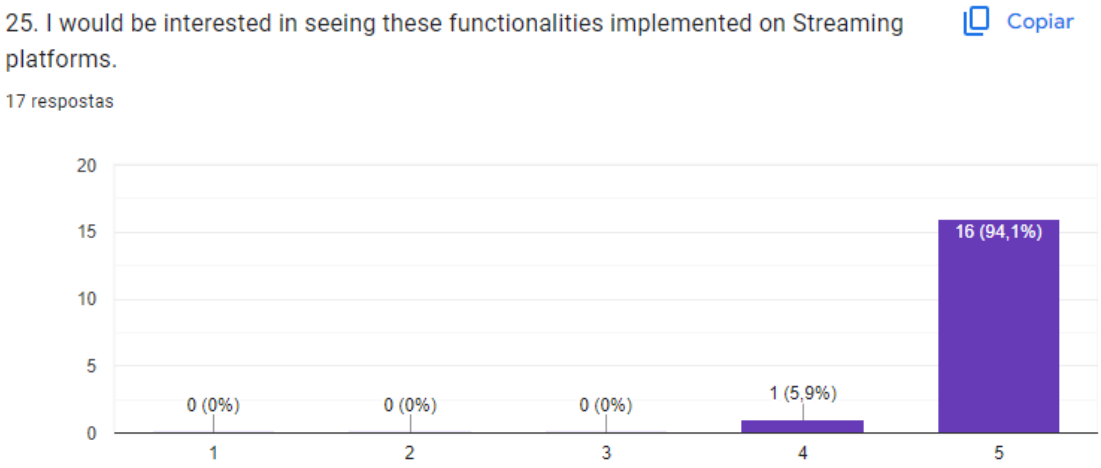

Figure VI.12: Interest in feature implementation on streaming platforms