



UiT The Arctic University of Norway

Faculty of Science and Technology
Department of Computer Science

Dárkon

Architecting of a Soccer Performance Analytics Software Toolkit

Sebastian Lyng Johansen

INF-3990 Master's Thesis in Computer Science - May 2023

This thesis document was typeset using the *UiT Thesis L^AT_EX Template*.

© 2023 – <http://github.com/egraff/uit-thesis>

“640K ought to be enough for anybody.”
–Bill Gates, 1981

Abstract

Sports today are more competitive than ever, with an eye for extreme details. Small margins can differentiate between succeeding or failing. That is why it is essential for a team to pay attention to all these margins and use them to their advantage. Technology today plays a huge part in sports. With video cameras and sensors following the players' every move, it is vital to use this technology to gain an advantage over the opponent.

This thesis presents Dárkon, a video analysis system to help teams with their video analysis needs. Dárkon works together with a tagging system developed simultaneously to provide filtering based on field position, type of event, and outcome of event to find a video of a specific event or playlist of events. Along with this, Dárkon will attempt to collect videos from different sources, to bring everything into one coherent system.

The videos of a single event, sequence, or playlists are generated on the fly from the metadata for the entire video to keep the storage to a minimum, while still being able to only play specific parts of a video from an entire match.

Contents

Abstract	iii
List of Figures	ix
List of Tables	xi
List of Abbreviations	xiii
1 Introduction	1
1.1 Problem Definition	2
1.2 Methodology	2
1.3 Scope and Limitations	3
1.4 Research Context	4
1.5 Outline	6
2 Background & Related Work	7
2.1 Football Analysis	7
2.1.1 Statistics	8
2.1.2 Videos	9
2.2 Existing Systems	9
2.2.1 ProZone	10
2.2.2 Hudl	10
2.2.3 Nacsport	12
2.2.4 Footovision	12
2.2.5 Summary of Existing Systems	14
2.3 Eliteserien Highlights	15
2.4 HTTP Live Streaming (HLS)	15
2.5 FFmpeg	17
2.6 Summary	18
3 Requirement Specification	19
3.1 Functional Requirements	19
3.1.1 Group/Team	19
3.1.2 Large Video Uploads	20

3.1.3	Integration with Tag-Based System	20
3.1.4	Import of Metadata	20
3.1.5	Tag-Based Analysis	20
3.1.6	Video Viewing	20
3.1.7	Statistics	21
3.1.8	Thumbnail Generation	21
3.1.9	External Sources	21
3.2	Non-Functional Requirements	21
3.2.1	Privacy and Compliance	21
3.2.2	Availability	22
3.2.3	Usability	22
3.2.4	Scalable	22
3.2.5	Security	22
3.2.6	Maintainable	22
3.3	Summary	23
4	Design & Implementation	25
4.1	System Architecture	25
4.2	Frontend	27
4.3	Backend	27
4.4	Database	28
4.4.1	Database Layout	29
4.5	Create Profile	31
4.6	Create Team	31
4.7	Edit Profile	33
4.8	Handling of Uploaded Videos	34
4.9	Video Metadata	37
4.9.1	Metadata Format	37
4.9.2	Uploading of Metadata	41
4.10	Streaming of Uploaded Videos	42
4.10.1	Entire Video	42
4.10.2	Event Based	43
4.10.3	Sequence	45
4.10.4	Playlist	47
4.11	Statistics	48
4.12	Third-Party Systems	50
4.13	Summary	54
5	Experiments	55
5.1	Setup	55
5.2	Re-Encode Video	55
5.3	Thumbnail Generation	57
5.4	Sport and League Agnostic	59
5.4.1	FA Community Shield	59

5.4.2	Handball Match	59
5.4.3	Summary of Sport and League Agnostic	60
5.5	Database Location	61
5.6	Evaluation	62
5.6.1	Players	63
5.6.2	Coach	65
5.7	Summary	67
6	Discussion	69
6.1	File System vs. MongoDB GridFS	69
6.2	Parsing HLS Manifest	71
6.3	Thumbnail Generation	71
6.4	Deployment	72
6.4.1	Azure	72
6.4.2	Local	73
6.4.3	Combination	73
6.4.4	Summary of Deployment	74
6.5	Hudl Integration	74
6.6	Scalable	74
6.7	Usability	74
6.8	Availability in Dárkon	75
6.9	Security	76
6.9.1	Authentication	76
6.9.2	Access Control	76
6.10	Privacy	77
6.11	Comparing Dárkon to Existing Systems	78
6.12	Summary	79
7	Concluding Remarks	81
7.1	Thesis Conclusion	82
7.2	Future Work	83
7.2.1	Integration with Tagging System	83
7.2.2	Live Streaming	83
7.2.3	Statistics	83
7.2.4	Overlay	84
7.2.5	Legal Aspect of Video Sources	84
7.2.6	Machine Learning	84
7.2.7	Combine and Export Videos	84

List of Figures

2.1	Screenshot of the ProZone software [22].	10
2.2	Screenshot of the Hudl software [25].	11
2.3	Screenshot of the Nacsport software [26].	13
2.4	Screenshot of the Footovision software [27].	14
2.5	Illustration of how HLS works [28, p. 224].	16
2.6	Example of an HLS manifest file with two second segments. .	17
4.1	System architecture of Dárkon.	26
4.2	Collections in the database.	29
4.3	Example of a user in the users-collection.	29
4.4	Example of a team in the teams-collection.	30
4.5	Example of a tag in the tags-collection.	30
4.6	Example of an uploaded video in the uploadedVideos-collection.	31
4.7	Form when creating a user.	32
4.8	Page when creating or joining a team.	33
4.9	Settings for a team.	33
4.10	Page when editing a user.	34
4.11	Preview of when uploading a video.	35
4.12	Implementation of how the backend handles upload, storage and encoding of a video.	36
4.13	Example illustration of the first iteration of general purpose tagging.	37
4.14	Example of the more detailed tagging from the first iteration.	38
4.15	Structure of tagging data in Rust.	40
4.16	Example of a tagged event from a match.	40
4.17	Option for uploading a metadata file.	41
4.18	Page showing the entire uploaded match.	42
4.19	Page showing all single events.	43
4.20	Filters for a single event.	43
4.21	Implementation of finding the start- and end-time of a se- quence.	44
4.22	Implementation of generating a manifest file.	45
4.23	Page showing sequences from a match.	46
4.24	Filters available for a sequence.	46

4.25	Page showing playlists from a match.	47
4.26	Filters available for playlists.	48
4.27	Example of a generated manifest file for a playlist.	49
4.28	Page showing statistics from a match.	49
4.29	Page showing the integration with Eliteserien Highlights. . .	50
4.30	Example response from Forzasys with metadata from Eliteserien.	53
4.31	Stored information about an event from Eliteserien Highlights.	54
5.1	Command to copy codec with FFmpeg.	56
5.2	Command to re-encode with FFmpeg into two second HLS segments.	56
5.3	Point plot of the difference in seconds between re-encoding and copying the codec.	56
5.4	Upload of tags when no thumbnails exist.	57
5.5	Upload of tags when thumbnails already exist.	58
5.6	Tags when filtering dribbles [39].	59
5.7	Tags when filtering bad shots [40].	60
5.8	Query for tags-collection used for latency experiment.	61
5.9	Latency difference between a database in Norway and USA. .	62
5.10	Q1: How important is it for you to get statistics from a match or training session? Q2: How important is it for you to be able to go back and watch videos from a match or training session?	63
5.11	Q3: How easy is it to find relevant video clips using Dárkon? Q4: How likely are you to continue using Dárkon if still available in the future?	64
5.12	Q5: Which of these features are the most important for you? (Pick two)	64
5.13	Q1: How happy are you with the current way of analysing a match after playing? Q2: How happy are you with using Dárkon for analysis after a match? Q3: How easy is it to use Dárkon for your needs?	65
5.14	Q4: How likely are you to continue using Dárkon if still available in the future? Q5: How helpful would it be to get video clips from the first half in the half-time break?	66
5.15	Q6: Which of these features are the most important for you? (Pick two)	67
6.1	Mobile view of Dárkon.	75

List of Tables

- 4.1 Description of the first iteration of metadata. 38
- 4.2 More detailed description of each tag. 38

List of Abbreviations

AI Artificial Intelligence

API Application Programming Interface

CDN Content Delivery Network

CSG Cyber Security Group

FIFA Fédération Internationale de Football Association

GB Gigabytes

GDPR General Data Protection Regulation

GPS Global Positioning System

HLS HTTP Live Streaming

HTTP Hypertext Transfer Protocol

JSON JavaScript Object Notation

KB Kilobytes

MB Megabytes

ML Machine Learning

NFF Norwegian Football Federation

PC Personal Computer

POC Proof of Concept

- RAM** Random Access Memory
- REST** Representational State Transfer
- RTMP** Real-Time Messaging Protocol
- SLA** Service-Level Agreement
- SSD** Solid State Drive
- TIL** Tromsø Idrettslag
- UIT** University of Tromsø
- URL** Uniform Resource Locator
- VM** Virtual Machine



Introduction

Football, also called soccer, is one of the most popular sports in the entire world, if not *the* most popular. Millions of fans all around the world are cheering for their favorite team to win the different tournaments. The competition is more fierce than ever, and there is an enormous amount of money involved in winning the biggest trophies.

To have a chance at winning the biggest trophies and succeeding, it is crucial to take every possible opportunity to get an upper hand on the competition. With this, the technology in football is only becoming increasingly popular, and with the huge amount of resources the biggest teams have, they often have a whole dedicated apartment just for analyzing their opponent and their own matches.

Analysis in football is widely adopted among the elite clubs today, but not as much in the lower leagues where the resources are not the same. The smaller clubs cannot afford employing people dedicated to analyze the game nor have data as easily available as the elite clubs.

Many large enterprise solutions today store a vast amount of data about multiple clubs ranging from the personal information about a player profile to all the teams' analysis data. The tagging of the matches is also often outsourced to countries like the Philippines, where hundreds of people are tagging matches from various leagues all around the world [1, p. 1-7]. While the systems are often regarded as highly accurate, there is still the possibility of inaccuracy,

making the data less valuable. Other systems also provide unique cameras, which they operate in conjunction with, which may provide vendor lock-in to the system.

Dárkon is a video analysis system that works with third-party tagging systems developed alongside Dárkon, and tries to minimize this gap between the clubs with many resources and those with fewer resources. Requiring no external cameras or wearables while still being able to filter videos based on different parameters like field position, type of events, and outcome of events.

1.1 Problem Definition

Video analysis in football today is mostly of help to the elite clubs with many resources. Today's software is mostly enterprise solutions that charge money that the clubs in lower divisions do not have the budget for. This thesis will try to develop a Proof of Concept (POC) system with disruptive technology to narrow this gap and potentially help these smaller clubs get started on analysis while also providing an alternative to the already existing systems.

Therefore, the problem definition is defined as:

This thesis will design, implement, and evaluate a system prototype for managing annotated clips of a match for visualization. It will thus investigate scalable and privacy-preserving backend services that support this, as well as frontend services providing ease of use for coaches and athletes. A particular focus will be on giving athletes and coaches flexibility in finding specific match clips to analyze further.

1.2 Methodology

The ACM Task Force on the Core of Computer Science presents their final report [2], three paradigms to divide the discipline of Computer Science into.

Theory is the first paradigm. This paradigm consists of four steps in development and is rooted in mathematics.

1. Definition: Characterize objects of study

2. Theorem: Hypothesise possible relationships among them
3. Proof: Determine if the relationships are true
4. Interpret results

Abstraction is the second paradigm. This paradigm is rooted in the experimental scientific method. This paradigm also consists of four stages.

1. Form a hypothesis.
2. Construct a model and make a prediction
3. Design an experiment and collect data
4. Analyze results

Design is the last paradigm. This paradigm is rooted in engineering. This consists of four steps to solve a problem with the construction of a device or system.

1. State requirements
2. State specifications
3. Design and implement the system
4. Test the system

With these paradigms lined out, Dárkon falls into the design paradigm where a POC system will be designed, implemented, and tested according to the requirements set out.

1.3 Scope and Limitations

It is necessary to make some assumptions concerning the problem definition in this thesis. Dárkon will develop a POC system to assist clubs with analysis regarding a video already tagged with metadata. The thesis will mainly focus on the video aspect of analysis and provide some statistics based on the same tagged data used for the video analysis.

One assumption is, therefore, that the video provided to Dárkon is correctly

annotated with metadata in a specific form. The coaches are generating this metadata in another system built in collaboration with Dárkon. This metadata should therefore be understood by both systems.

Dárkon is mainly developed with football in mind but might be expanded later to make it more sports-agnostic.

1.4 Research Context

This thesis is written in collaboration with Cyber Security Group (CSG)¹ at University of Tromsø (UiT). CSG is a research group that seeks to provide new knowledge and innovative distributed system technologies in Computer Science.

CSG conducts research and innovation that spans multiple disciplines and faculties. With a focus on the intersection of computer science and other fields, their ultimate goal is to create new knowledge, research tools, and innovative technologies in this converging space. They tackle real-world challenges that combine academic principles with practical engagement and innovation.

CSG conducts experimental research in computer science systems, particularly in the construction and design of scalable, efficient, fault-tolerant, privacy-preserving, and secure distributed systems. They deploy these systems in real-world settings to gather user insights and feedback. CSG encourage active user involvement in their research, and their students interact and work with specific user partners in applied domains of interest. They also follow an open publishing and distribution policy for their software artifacts for others to keep building on.

The approach of CSG is usually to research problems using an experimental systems approach, where they construct and evaluate a prototype middleware system to solve concrete problems. However, if the problem lends to it, they may use a more theoretical approach.

CSG's research primarily focuses on building robust and efficient software infrastructures, trusted data management and storage, secure networking, multimedia support, and privacy-preserving algorithmic analysis.

CSG has a wide range of experience developing systems where Vortex [3], Balava [4], and TACOMA [5] are only some examples. Vortex is a multiprocessor

1. <https://site.uit.no/arcsecc/>

operating system that is event-driven. Vortex uses an efficient event architecture to balance load across processors automatically, and can sustain up to 80% higher throughput running a web server compared to Linux on the same hardware [3]. Balava is a system for managing computation with confidential data that spans multiple clouds, both private and public [4]. Balava used the TACOMA system for extending servers with new functionality in their Suorgi component [4]. TACOMA is a project that investigates software support for mobile agents through multiple prototypes.

One of the main partners of CSG is The Corpore Sano Centre². They seek to conduct interdisciplinary research and innovation at the intersection of computer science, health informatics, medicine, sport science, psychology, and law.

They focus on using novel computer science technologies in these fields. Within this context, CSG investigates structuring techniques for future-generation large-scale information access applications. This research involves partitioning an application into cooperating modules, optimizing interaction among them, deploying and interacting with users, providing integrity, security, auditing, and ensuring fault tolerance and privacy.

CSG is also known for collaboration in the scientific context of sports science. This includes systems like Muithu [6, 7], Bagadus [8, 9], and DAVVI [10, 11]. Muithu is a system that integrates real-time coach notations with related video sequences. It uses a smartphone interface, where the coach can annotate an event during a match with some metadata, like who the player was and what happened. Then the event would be synced up with the video of the match, and the specific sequence related to the annotated tag would be available during half-time, or after the match.

Bagadus is a system that integrates a sensor system, a soccer analytics annotations system, and a video processing system using an array of cameras. When it was introduced in 2013, the cameras could not capture the entire field. Bagadus used an array of cameras to stitch them together, capturing the entire field in one image. While also providing digital zoom and automatically tracking a player by mapping the sensor data from the ZXY system³ to image pixels.

DAVVI is a multimedia entertainment platform that can deliver multi-quality video in a torrent-similar way. They use event extraction tools and process metadata from information on the internet to be able to give a search field

2. <https://site.uit.no/corporesano/>

3. <https://tracab.com/>

where users can query for events using keywords. The result of the query will give a video playlist of events matching the query.

Another collaborator on this thesis is Hamna IL and their head coach Martin Rypdal, which is also a professor in the department of mathematics and statistics at UiT. Hamna IL is a local team in Tromsø which plays in the fourth division in Norway. Martin is responsible for the tagging system that provides the metadata for a video to Dárkon. Hamna IL has already started using the system for their video analyzing needs. They have also provided valuable feedback on how they want the system to work, and what is essential for them. Therefore, the system is adapted to their needs, but is also flexible to adapt more features and iterate on the existing functionality for improvements.

1.5 Outline

The rest of this thesis is structured as follows:

Chapter 2 provides background and relevant related work in the field of video analysis in football.

Chapter 3 provides the related functional and non-functional requirements for Dárkon.

Chapter 4 describes the design and implementation of Dárkon.

Chapter 5 explains the experimental setup, how it was done, and also covers the results with regards to the functional and non-functional requirements of Dárkon.

Chapter 6 discusses some relevant topics with regards to decisions made while developing Dárkon.

Chapter 7 concludes the thesis while also covering future work.

/2

Background & Related Work

The previous chapter introduced Dárkon and the problem definition of this thesis. This chapter will explain what analysis in football is, and how it is used today. It will also introduce existing commercial systems, and present some background into technologies that is vital to how Dárkon works.

2.1 Football Analysis

As football is one of the world's most popular sports, there is also a tremendous amount of money involved. Winning and competing in the most competitive leagues, like the UEFA Champions League, comes with a vast amount of money. However, to compete for the big titles, the teams must outperform the others, making the need to get every advantage possible [12].

By analyzing their own team, and the opponent, it is possible to gain an advantage in observing specific patterns of where to attack the opponent, and where the team might improve. One example is by analyzing passing networks; it is possible to develop a unique style of play [13]. FC Barcelona is a prime example of having a unique style of play. Their famous Tiki-Taka has helped them win numerous competitions in the past.

Another example of analysis in football, and how important it could be, is the successful history of Liverpool Football Club in the latest years. In 2011, Michael Edwards joined Liverpool from Tottenham Hotspur as a director of performance and analysis [1, p. 243-244]. Edwards had a background in Informatics from Sheffield University, and had previously worked for ProZone. He was from the company sent out to be an analyst for Portsmouth under Harry Redknapp [1, p. 244].

Edwards was hired to bring the data analysis up to speed in Liverpool as they were behind their competition in this area [1, p. 245]. Edwards started to meet with data analysts and data providers that claimed they could help the club, but it was the company Decision Technology that caught his eye [1, p. 245]. However, Liverpool could not buy the entire company, and instead hired their data scientist Ian Graham, a PhD graduate from the University of Cambridge [14].

After Edwards and Graham joined Liverpool, they started working towards creating a data department that could help the club take more data-driven decisions. The latest years, their data-driven approach has been regarded as highly successful, with the signings of Roberto Firmino, Mohamed Salah, Alisson Becker, and Virgil van Dijk. However, one of their greatest signings is the manager Jürgen Klopp, which in 2019 said: *They're the reason I'm here* [15], hinting at if it were not for Edwards and Graham, the club would not have hired him as the manager.

It is difficult to know how much Liverpool would have achieved without Michael Edwards, Ian Graham, and their data-driven decisions, or if the success is because of the talent of Jürgen Klopp. It is, however, a testament to the job of Edwards and Graham, the success of Liverpool, and the feeling in and around the fans after they both resigned in 2022. [1, p. 262-263].

2.1.1 Statistics

One widely adopted way of analyzing football is through statistical measurements. This helps to gain insights into a team and player's performance, and could further help identify trends and patterns to help further improve individual players or the entire team. One of the leading providers of statistics in football today is Opta Sports¹. They provide data for several leagues worldwide, both for the consumer and the teams. Opta is a company specializing in data-driven technology for sports with the help of Artificial Intelligence (AI), helping teams make decisions about the analysis of matches, and monitor

1. <https://statsperform.com/opta/>

athletes and recruitment.

Statistical analysis has become increasingly popular in football. In the latest world cup in Qatar, Fédération Internationale de Football Association (FIFA) provided a detailed statistical report of each game [16]. By utilizing statistical measurements, a team may be able to pinpoint weaknesses of the opposition or weaknesses in their team. They may then be able to develop a training plan specific to this weakness to help them improve. One example is by analyzing the data; they might see that they struggle to make the final pass in the opposition half, and may then modify the training plan to help focus on improving this.

Overall, statistical analysis is a crucial tool to be competitive in football today. For the coaches to gain this valuable knowledge, they may adopt the plan and help the team succeed.

2.1.2 Videos

Another common method to analyze a match of football is through the use of video.

Video analysis is hugely adopted in the world of football today. Elite clubs spend a considerable amount of money on systems and personnel to help them analyze all the video clips of an opponent and their team to gain an advantage over the opponents.

Elite clubs like Liverpool have huge personnel working with analysis. One of their jobs is to analyze the video of an upcoming opponent, trying to find weaknesses and where to attack them [17].

As analyzing videos may be time-consuming, elite clubs often have separate departments that do separate tasks regarding video analysis. In Liverpool, departments exist for Post-Match, Individual player, and Opposition analysis [18]. This shows how much time and money clubs on the highest level invest in analysis.

2.2 Existing Systems

This section will look at some existing systems today and what they offer.

2.2.1 ProZone

ProZone is one of the actors in the context of data analysis that sparked the whole analytics boom in the world of football [19]. They developed a video based computerized tracking system to capture players' movement patterns. This helped with quantifying the work ratio of the players during a game, and also made it possible to look at the characteristics of the players' movements [20, 21].

ProZone also developed a second product called ProZone MatchViewer. This system allowed events with the ball to be recorded and associated with frames within the video. It had options to specify passing, receiving, shooting, goal-keeping, and crossing statistics for all the players [21].

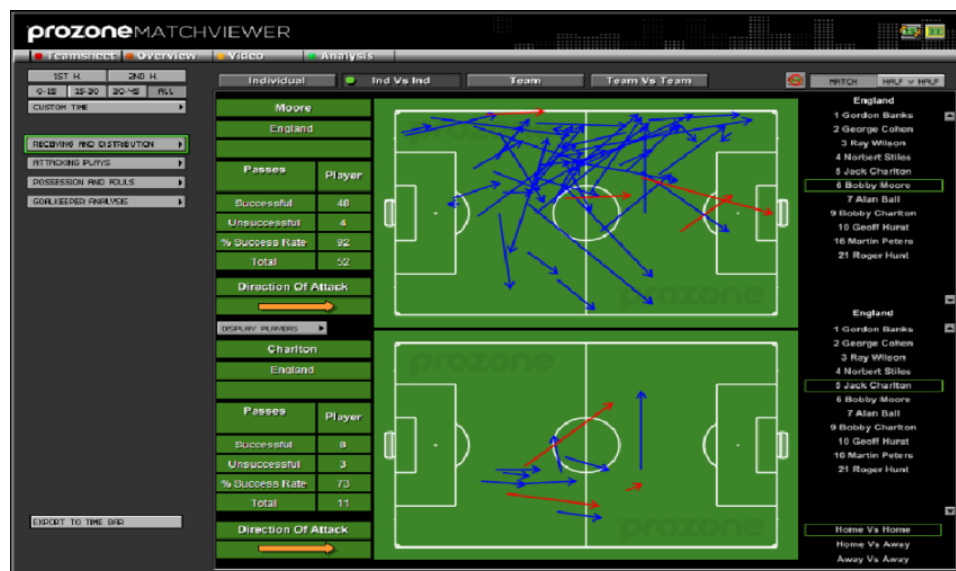


Figure 2.1: Screenshot of the ProZone software [22].

ProZone is known under a different name today. In 2015, they were acquired by Statsperform, the same company that owns Opta Sports, which was discussed in section 2.1.1.

2.2.2 Hudl

Hudl² is another tool that provides video analysis, and is widely adopted in football clubs [23]. This is also the tool currently used by the local team Tromsø Idrettslag (TIL), which currently plays in the highest division, Eliteserien, in

2. <https://www.hudl.com/>

Norway.

Hudl provides a wide range of opportunities for doing analysis, but this comes with a price tag that some teams may find expensive. They have different packages depending on whether the team is a youth, high school, or club, with the lowest starting price of 400\$ per year, ranging up to 3,300\$ per year [24].

Hudl comes with a package deal combined with the software and a camera that is meant to be used with the software. This camera has an integrated AI tracking system which follows the action, requiring no cameraperson.

It would be reasonable to assume that Hudl stores its data in the cloud, given the number of users and data they store from teams worldwide. This information could, however, not be confirmed by their website, nor from personal contact with Lea Daranyi from Sales Rep at Hudl on Friday, 31.March 2023.

From the personal contact it became known that Hudl has a tagging service with a giant team of analysts that can do tagging on demand. This means that if a team does not have the time or resources, the analyst team from Hudl can do it.

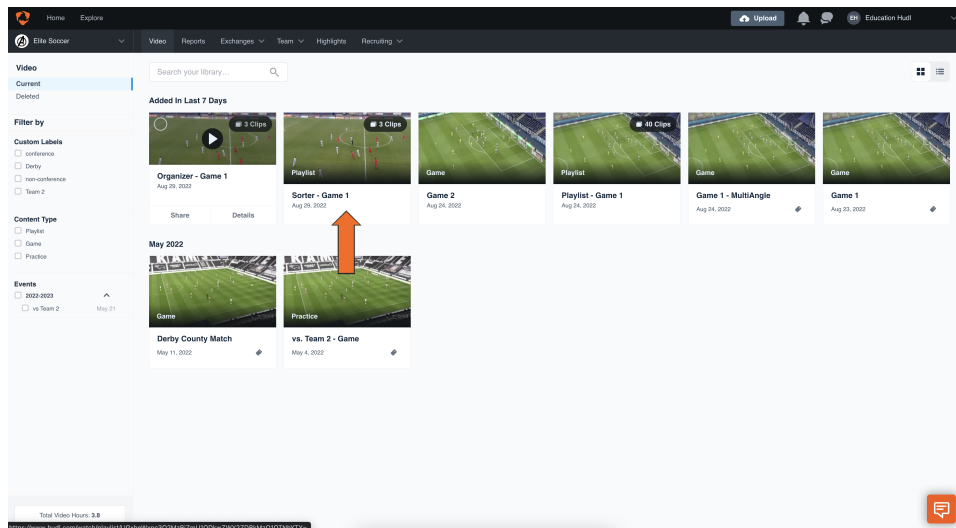


Figure 2.2: Screenshot of the Hudl software [25].

2.2.3 Nacsport

Nacsport³ is one of the providers of video analysis software used by multiple English Premier League teams as of January 2023. This includes Liverpool [17], one of the best teams in English football, both historically and in recent years.

Nacsport is a full-blown tool with different subscription levels giving flexibility for the analysis. Each subscription level also comes at a different price point, so picking the one that suits the team and the budget best is possible.

To take advantage of Nacsport, one would have to obtain a license from them. This license is only available for one Personal Computer (PC) at a time. This means that when using a different PC, they would either need to buy a new license, or transfer the current license to the new PC.

The system is downloaded to a PC, allowing access to annotate a match in real-time, and hindsight. This also allows storing data locally and working offline, annotating a match with specific events.

It is presumed that the data is stored in the cloud when it is not stored locally on the PC with the license. However, this could not be confirmed since they did not provide any information regarding this, nor did they reply with an answer to questions provided to them.

Nacsport does not support all operating systems for offline use, as there is no desktop app for Linux. It is only compatible with Windows and macOS, which could be a potential problem for Linux users who intend to use the software.

2.2.4 Footovision

Footovision⁴ develops advanced AI data collection, and helps with analysis in football. They have several partners, including La Liga, FC Barcelona, and Norwegian Football Federation (NFF).

Footovision collects tracking data, ball, and non-ball events from any video feed. They also provide analysis dashboards and performance metrics that can be fully customized for football clubs.

All the tagging is done internally by Footovision, meaning the coach can focus

3. <https://www.nacsport.com/>

4. <https://www.footovision.com/>



Figure 2.3: Screenshot of the Nacsport software [26].

on other tasks. However, this also means that the control is taken from the coach and the club, meaning they might interpret measurements differently than internally in Footovision. Footovision has some events for which each person can define their own parameters, like, *What is a sprint?* or *What is a fast attack?*.

Information about Footovision was not easily accessible on their website, and there is little to no information available elsewhere. Therefore, personal contact through their customer support was necessary to gain more insight into their system.

Pierre Miralles in Footovision said on Monday, 3. March 2023, they store events regarding a player both with, and without the ball. This includes a player's position during the match. All of this is stored in a French cloud.

These events are tagged in a mix of automatic and manual work by their employees, reducing the need for work by the coaches using Footovision. However, this also means they have predefined events that they are tagging, and there is little room for customization regarding missing or unnecessary tagging of events.

For requesting the data, they provide their customers with an Application Programming Interface (API) and a web page where all the game, team, and player analysis can be done directly.



Figure 2.4: Screenshot of the Footovision software [27].

2.2.5 Summary of Existing Systems

The existing systems provide little to no information on how things are done, and how the data is stored in their system. The different systems also store a lot of data, which could be misused if unauthorized people access this data. Regarding how some of them store a massive amount of data, how non-transparent they are about how they store it is worrying. Two out of the three companies we contacted either did not respond, nor could answer to how, and where the data is stored.

Another argument is the quality of the annotations. While some provide an option for a team to tag a video themselves, a lot of this work could be done by the company providing the software automatically, and manually by their employees. While it could free up work for employees at the club, it could also lower the quality of the tagged events. Some events could be subjective to the eye, and some may also miss events that could impact the analysis and give a false sense of information. A lot of the existing commercial companies also outsource the tagging to other countries like the Philippines [1, p. 1-7]. Even though these systems claim to be highly accurate with the tagging data, it could still be some inaccuracies that could lead to patterns in the play not being analyzed, making the tool less effective than it could be. Automatic tagging by the enterprise company may also make it harder to analyze a training session, as they are dependent on the company for tagging.

2.3 Eliteserien Highlights

One of the third-party systems that Dárkon is integrating with is Eliteserien Highlights. Eliteserien Highlights is a deployment of Forzify⁵, which is a product of Forzasys⁶.

Eliteserien Highlights⁷ makes it possible to show annotated clips from the broadcast stream of matches from Eliteserien in Norway. These matches are annotated by a team that watch the matches, and it is done in multiple stages. On the first level, a single person monitors multiple matches, and annotate these matches concurrently. The second stage will fine-tune the events with descriptions, thumbnails, and the start/stop frame for the video.

2.4 HTTP Live Streaming (HLS)

HLS is an adaptive streaming protocol designed by Apple [28]. HLS works by segmenting the video into smaller chunks of content (usually 5-10 seconds, depending on the content). These chunks are stored in segment files that have the extension .ts.

As well as segmenting the video, an index file that reference these segments is created. This index file, also called manifest file, is a regular text file as a .m3u8 extension, and is used by the client to request segments from the server.

Figure 2.5 shows that HLS has three significant components - the media preparation, the web server, and the client player.

The media preparation server is responsible for accepting audio and video input, encoding the input, and segmenting them into individual chunks of files. These are the segment files which is referenced from the manifest file.

The system can also create a hierarchical set of index files, with a primary index file referencing other index files based on the network quality between the client and the server. These index files can then reference segments encoded at different qualities. This makes it possible to utilize HLS for adaptive bitrate streaming [29].

The web server stores these chunks, and is responsible for hosting and delivering

5. <https://forzasys.com/Forzify.html>

6. <https://forzasys.com/>

7. <https://highlights.eliteserien.no/>

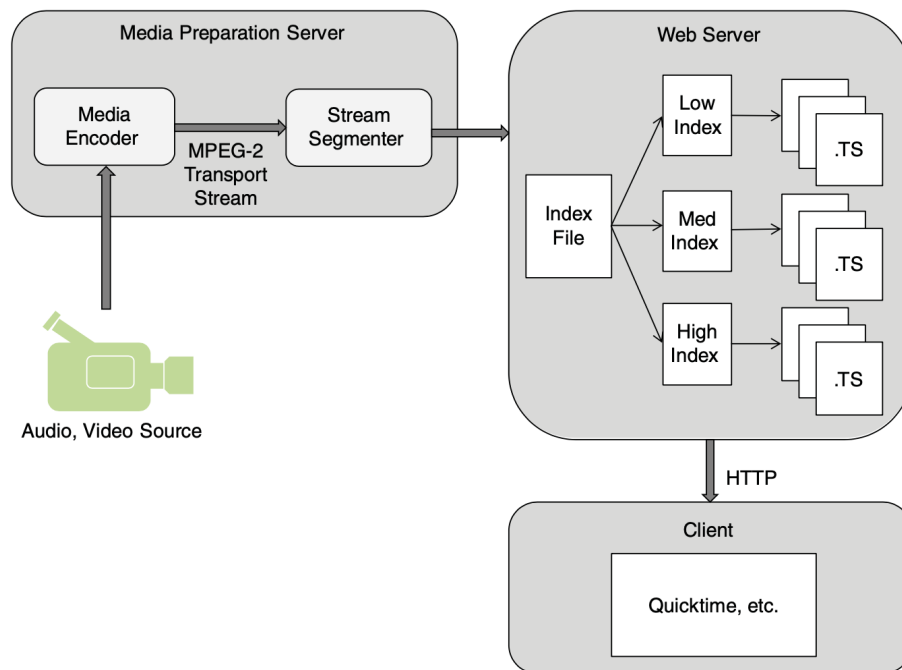


Figure 2.5: Illustration of how HLS works [28, p. 224].

the chunks when requested by the client.

The client player is responsible for playing the chunks the web server returns. The client requests the index files, and the specific chunks it wants to play from the web server. Suppose there exist a hierarchical set of index files with chunks of different qualities depending on the bandwidth between the client and the server. In that case, the client is responsible for requesting the chunk with the most appropriate quality.

While HLS allows for adaptive streaming quality, it also makes generating a playlist of segments on the fly possible. By providing an index file with the segments wanted, the client will request the provided segments in the index file, making it possible to fetch and play only a portion of the video, while not increasing the footprint on the disk by having duplicate data. Allowing for generating playlists of segments on the fly is a core requirement in Dárkon, and Dárkon uses HLS to make this possible.

```
#EXTM3U
#EXT-X-VERSION:3
#EXT-X-TARGETDURATION:3
#EXT-X-MEDIA-SEQUENCE:0
#EXTINF:2.000000,
output0.ts
#EXTINF:2.000000,
output1.ts
#EXTINF:2.000000,
output2.ts
#EXTINF:2.000000,
output3.ts
#EXTINF:2.000000,
output4.ts
#EXTINF:2.000000,
output5.ts
#EXTINF:2.000000,
output6.ts
#EXTINF:2.000000,
output7.ts
#EXTINF:2.000000,
output8.ts
#EXTINF:2.000000,
output9.ts
#EXTINF:2.000000,
output10.ts
#EXT-X-ENDLIST
```

Figure 2.6: Example of an HLS manifest file with two second segments.

2.5 FFmpeg

FFmpeg is an open-source multimedia framework that allows for encoding, decoding, transcoding, and streaming of videos. FFmpeg is cross-platform, meaning it works across Linux, macOS, and Windows, and also runs on various processor architectures⁸.

FFmpeg combines several different tools into different programs, including an industry-leading command line tool written in C to make converting multimedia files between different formats easier.

Dárkon utilizes FFmpeg to encode the uploaded videos into HLS, while also generating thumbnails for the entire video and the specific events. This gives a visual representation of each event with the thumbnail, and also make it possible to utilize HLS for the core features of Dárkon.

8. <https://ffmpeg.org/>

2.6 Summary

In this chapter, we have described existing systems that already exist, and how they work. We have also described how HLS works, and what FFmpeg is, as these components are vital to how Dárkon works. We can observe that the existing systems are non-transparent in the way they are handling the data for all the teams. Most of these systems either does the tagging internally in the company, or outsource the tagging to countries like the Philippines [1, p. 1-7], which helps teams worry less about annotating a match. However, this could also give less accurate data for their team if the tagging is not done properly. In the next chapter, we will present the core requirements of Dárkon, both functional and non-functional.

/3

Requirement Specification

From the previous chapter, we observed that most tagging systems are non-transparent in their handling of data, which could be a privacy and trust issue. In this chapter, we will devise and present the requirement specification of Dárkon. Requirements of a system is outlining what it should do, which services it provides, and the constraints on the operation of the system. [30, p .83]. We will specify the functional and non-functional requirements of Dárkon with the problem definition in mind.

3.1 Functional Requirements

Functional requirements describe what the system should do. This depends on the type of software being used, and the expectations the users have [30, p .85].

3.1.1 Group/Team

Dárkon should be able to group members into a cohort that share videos. Users with a coach role is the only ones that should be able to create a team. The coaches of the team should be able to invite other users to join the team. As the coaches and players transfer to a different club, a coach using Dárkon should be able to remove the users from the team.

3.1.2 Large Video Uploads

Dárkon should be able to handle uploads and storage of relatively large video files. An entire football match is minimum 90 minutes, which means the videos that are uploaded are usually at least 90 minutes long. Sometimes they are longer if the half-time break is included in the video. This usually corresponds to a video file that takes up multiple Gigabytes (GB) of storage, and Dárkon should be able to handle this.

3.1.3 Integration with Tag-Based System

Dárkon should be able to easily integrate with a third-party system that generates metadata related to a video. The events from the metadata should be in a predefined format agreed upon between Dárkon and the third-party tagging system, so Dárkon is able to perform analysis on the video based on the tagging data.

3.1.4 Import of Metadata

Dárkon should be able to handle upload and storage of a metadata file. This should be on a predefined format agreed upon between Dárkon and the system that generates these tags. Dárkon should then read this metadata, create thumbnails for each of the events, store it persistently, and be able to use it for analysis.

3.1.5 Tag-Based Analysis

Dárkon should be able to easily provide clips of a video based on the metadata. This includes filtering events based on players, shots, fouls, dribbles, position on field, result of an event, and more. It should be possible for a player to easily retrieve a video with all their events from a match or training session.

3.1.6 Video Viewing

After uploading a video of an entire match, it should be possible to watch the entire match. It should also be possible to watch a single event, sequence of events, or a playlist of discontinued events as a single video given that metadata to the video is provided.

3.1.7 Statistics

Dárkon should be able to provide statistics for matches based on the metadata for a video. It should be able to present this information, making it easy for a player to find out how many passes, goals, and assists they got during a match.

3.1.8 Thumbnail Generation

A thumbnail should exist for the entire video, as well as for the single events, sequences, and playlists of videos. The thumbnails should be automatically generated when the video and metadata is uploaded.

3.1.9 External Sources

Dárkon should be able to integrate external sources into the system. Eliteserien Highlights is an external source that should be integrated into the system to optionally retrieve highlights of already annotated clips. This will provide clips from the broadcast of Eliteserien matches, and a team should be able to optionally subscribe to their videos given that they are playing in Eliteserien. Other external sources should be easy to integrate into Dárkon given that they have an open API.

3.2 Non-Functional Requirements

Non-functional requirements are requirements not directly concerned with the specific services delivered to the users of the system. Non-functional requirements like security, availability, and performance usually apply to the system as a whole, and are often more critical than individual functional requirements [30, p .87].

3.2.1 Privacy and Compliance

Data about a team and users should be kept to a minimum. Dárkon should only store necessary information to make it work according to the other requirement specifications. It is also important that Dárkon is transparent in how the collected data is stored and processed, and that it is compliant with General Data Protection Regulation (GDPR). What GDPR is, and how Dárkon is in compliance with it will be discussed in section 6.10.

3.2.2 Availability

Dárkon needs to be available for use at all times. It could be crucial to help with analysing a game before a match, or even do analysis in the half-time break of the match. Dárkon aims for availability above three nines, or 99.9%. How Dárkon deals with this will be discussed more in section 6.8.

3.2.3 Usability

Dárkon is intended to be used by both the coaches and players of a team, and should thus be easy to use and understand. While offering extensive options for analysis, it should still be easily understood by the team. How Dárkon tries to achieve usability will be further discussed in section 6.7.

3.2.4 Scalable

The system needs to be scalable in regards to the number of users, and the amount of storage. When the number of users of the system increases, both the number of requests, and the storage will likely increase, and the system should be able to handle this. Dárkon will be handling videos which are both high in storage capacity, and expensive to process. Thus, the system should be able to scale up both storage needs, and the processing needs when the number of users increase.

3.2.5 Security

As the videos uploaded to the system is intended for internal use only, the system should be secure in regards to authentication and access control. How Dárkon deals with this will be further discussed in section 6.9.

3.2.6 Maintainable

Dárkon should be a maintainable system that makes it easy for new developers to start working on, and maintain. It is especially important for the specification of the metadata for a match. It should also be easy to integrate videos from a new third-party system to have a single coherent system given that they have an API that allows for it.

3.3 Summary

This chapter has introduced the functional and non-functional requirements of Dárkon. As we observe from the requirements, Dárkon is intended to be a scaleable video system, which comes with challenging functionality. The next chapter will describe how Dárkon is designed, and also provides some implementation details into the system, and how it deals with the challenges.

/4

Design & Implementation

The previous chapter outlined the non-functional and functional requirements of Dárkon. This chapter will describe the design and implementation of Dárkon in the context of those requirements, and the problem definition from section 1.1. It will also present an architecture to how Dárkon is built up.

4.1 System Architecture

Dárkon is implemented with an architecture consisting of four different components. The four different components is the frontend (1), backend (2), data storage (3), and external sources (4). Figure 4.1 shows how the different components are organized.

The frontend (1) is responsible for requesting data from the backend (2), and displaying this to authorized users. This is the component that the users of the system are interacting with.

The backend (2) has several responsibilities in the system. As a coach has access to functionality a player does not, the backend (2) is split into what a coach has access to (2.2), and what a player has access to (2.1). Analysis (2.3) is the third layer in the backend, and this is where the core functionality of Dárkon is implemented. This layer communicates with the external sources (4), and provides analysis based on the data collected. As well as communicating with

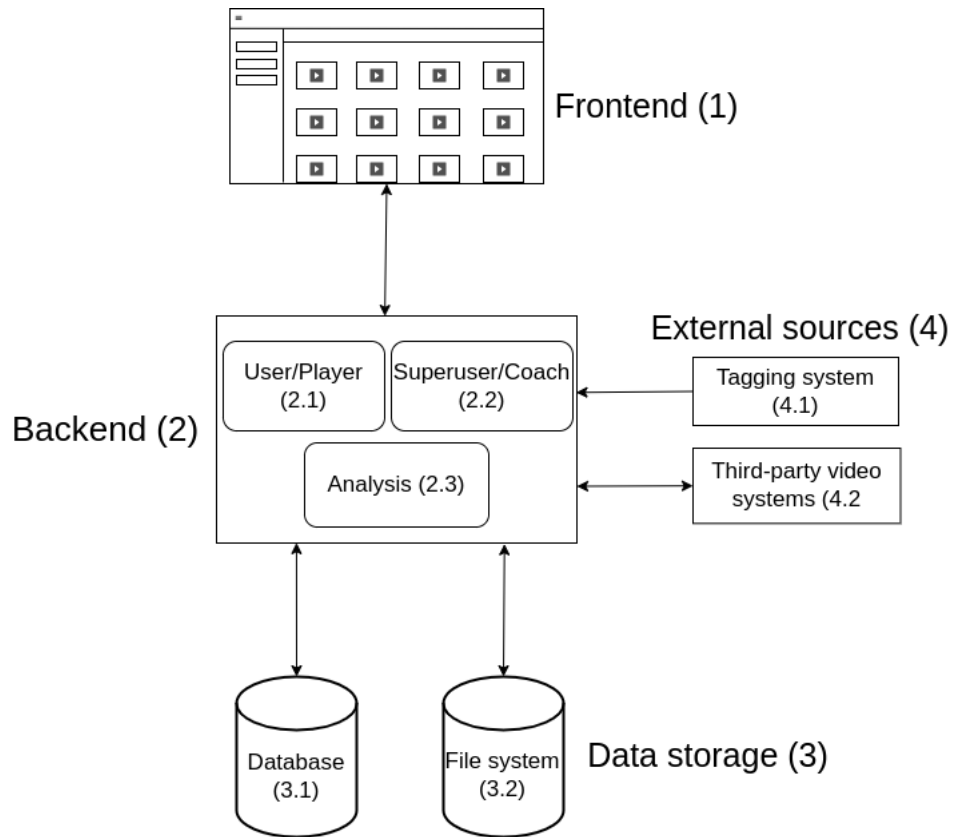


Figure 4.1: System architecture of Dárkon.

the external sources (4), the backend (2) is also responsible for communicating with the data storage (3), for retrieving and storing data gathered from the frontend (1), and from the external sources (4).

The data storage (3) is split into two different layers. The first layer is the database (3.1). This layer stores all the information Dárkon needs, except for the videos and thumbnails. Storing the videos and thumbnails is the responsibility of the file system (3.2).

The external sources are split in two. The first includes the tagging system (4.1) that generates metadata for Dárkon. The other part is the external video sources (4.2) integrated into Dárkon. This includes Eliteserien Highlights which was introduced in section 2.3.

4.2 Frontend

The frontend is the user interface of Dárkon. The main functionality of the frontend is to communicate with the backend, and provide a visualization of the content provided by the backend. The interface should be intuitive on both PC and mobile devices.

As a user should be able to access the system from a PC, a mobile application is not a suitable option for Dárkon. The decision was, therefore, to create a web application. Since the system should also be accessible from a phone, the web application should have a user interface that fits on a smaller screen. This should make it accessible on all devices that have an internet connection.

Through the device interacting with the frontend, it should be possible to view the videos uploaded to Dárkon, along with being able to filter it based on the metadata. It should also be possible to view statistics from matches through the frontend.

For choosing a programming language and framework the frontend, the most important thing was familiarity for fast development, combined with rich library support. For familiarity it was between Vue.js¹ and React². However, with library support in mind, React was chosen as the framework. For the programming language, TypeScript was chosen for type safety.

4.3 Backend

The backend of Dárkon has the responsibility of processing requests from the frontend using an API defined by the backend. When choosing a technology for implementing the API, two popular options are Representational State Transfer (REST) and GraphQL. Important characteristics includes performance and familiarity. For familiarity reasons, REST is preferred over GraphQL. Also, since Dárkon's core functionality is to stream videos, there is a lot of information flow of potentially large files. This favours REST over GraphQL [31]. Therefore, it was chosen to implement a REST API.

Another potential responsibility is fetching third-party data from external sources like Eliteserien Highlights. This is, however, an optional responsibility, and not necessary for clubs that are not competing in Eliteserien. The primary job of the backend is to store large videos, and also be able to generate playlists

1. <https://vuejs.org/>
2. <https://reactjs.org/>

and sequences from a video on the fly based on user queries.

Familiarity, security, and performance was important characteristics when deciding on a programming language for the backend. Because of those characteristics, the language chosen for the backend is Rust. Security and performance in Rust are some of the strong selling points [32]. This made it an excellent choice. Rust is built on an ownership model, and is a compiled language with close to C performance. Its safety lies in the chosen ownership model instead of manual memory management, or garbage collection. It enforces strict rules to avoid memory leakage while removing the overhead of having a garbage collector at runtime. Rust has a growing ecosystem of libraries called crates. These can be installed to reuse code, making the development process quicker. Rust has also been voted for the most beloved programming language seven years in a row as of December 2022 [33].

4.4 Database

The database of Dárkon stores information about a user, the team they are a part of, and metadata about the videos available. The videos are stored on the file system rather than the database. The reason for this will be discussed in section 6.1.

When choosing a database, the first question is whether it should be relational or non-relational. Because tags for a video are a significant factor in Dárkon, we need to be flexible in storing metadata about what happens in a video. Since the data also is collected from a third-party, and developed alongside Dárkon, it should be easy to adapt to changes in the structure. Because of this, we chose a non-relational database.

When choosing a non-relational database, it was important to have rich developer support, and familiarity for fast development. Popular NoSQL databases includes Amazon DynamoDB [34], Apache Cassandra [35], Redis [36], and MongoDB³. Dárkon will be hosted on Azure⁴, which is a cloud computing platform from Microsoft. They offer a wide range of services, and Azure Cosmos DB with an API for MongoDB⁵ is one of them.

One of Dárkon's requirements is to have availability over three nines, and with

3. <https://www.mongodb.com/>

4. <https://azure.microsoft.com/en-us>

5. <https://learn.microsoft.com/en-us/azure/cosmos-db/mongodb/introduction>

Azure's database service for MongoDB, they offer an availability of 99.95% [37]. With these reasons in mind, MongoDB was chosen as the preferred database.

4.4.1 Database Layout

MongoDB uses collections to separate documents in the database. Dárkon uses five different collections to store information needed by the system. This includes information about a video, user, and team. These collections are listed in figure 4.2.

- users
- teams
- tags
- uploadedVideos
- forzasysVideos

Figure 4.2: Collections in the database.

```
{
  "_id": "ObjectId('6389de80f9228ecc56ea331d')",
  "firstName": "Coach 1",
  "lastName": "",
  "email": "coach@team.com",
  "password": "$argon2i$v=19$m=4096,t=3,p=1$XGfpPaYor8X710+W2eMcE3R+oDBhMe
↪ IJCy1vyVdGCg4$A2c4CiX1ttVLV/Yf6VIKh1xtGaJ0Z0s06V0pH1aDvhc",
  "userRole": "Coach",
  "eliteserienPlayerId": null,
  "teamId": "ObjectId('638dbe4fbb486b5f24278b55')"
```

Figure 4.3: Example of a user in the users-collection.

```
{
  "_id": "ObjectId('638dbe4fbb486b5f24278b55')",
  "name": "Team",
  "players": [],
  "coaches": [
    "6389de80f9228ecc56ea331d"
  ],
  "joinTokens": [
    {
      "id": "96078760-db75-49ad-9299-8fd6f1034e31",
      "exp": "2023-03-02T21:31:35.1Z"
    }
  ],
  "eliteserienTeamId": 10
```

```
}

```

Figure 4.4: Example of a team in the teams-collection.

```
{
  "_id": "ObjectId('6419742f03f16a5280881462')",
  "videoId": "ObjectId('644268409e832ceb4b7e23dc')",
  "startPlayer": {
    "team": "our",
    "player": "Player 1"
  },
  "endPlayer": {
    "team": "our",
    "player": "Player 2"
  },
  "event": "Pass",
  "eventResult": "TeamMate",
  "startCoordinates": {
    "x": 27.5,
    "y": 30.899999618530273
  },
  "endCoordinates": {
    "x": 6.900000095367432,
    "y": 29.100000381469727
  },
  "videoUrl": "http://localhost:5000/api/filestorage/sequence/video/640886_
↪ c010d6a764648726e7?from=120000&to=121799",
  "thumbnailUrl": "http://localhost:5000/api/filestorage/video/640886c010d_
↪ 6a764648726e7/120000.png",
  "ballWin": false,
  "breakThrough": false,
  "dangerousMistake": false,
  "description": "Pass by Player 1",
  "startTime": 120000,
  "endTime": 121799,
  "playingDirection": "right"
}
```

Figure 4.5: Example of a tag in the tags-collection.

```
{
  "_id": "ObjectId('644268409e832ceb4b7e23dc')",
  "user": "ObjectId('6389de80f9228ecc56ea331d')",
  "teamId": "ObjectId('638dbe4fbb486b5f24278b55')",
  "filename": "Liverpool - Man City",
  "thumbnailUrl": "http://localhost:5000/api/filestorage/video/644268409e8_
↪ 32ceb4b7e23dc/thumbnail.png",
  "description": "Liverpool - Man City",
  "videoUrl": "http://localhost:5000/api/filestorage/video/644268409e832ce_
↪ b4b7e23dc/output.m3u8"
}
```

```
}
```

Figure 4.6: Example of an uploaded video in the uploadedVideos-collection.

Figure 4.3, 4.4, 4.5, and 4.6 are examples of documents in their respective collections. The last collection is generated from the metadata from the response of Eliteserien Highlights, and an example of a document from that collection is illustrated in figure 4.31.

4.5 Create Profile

For a team to use Dárkon, and start analyzing their matches, there is a setup phase to create users for all the players and coaches, and link them to the same team.

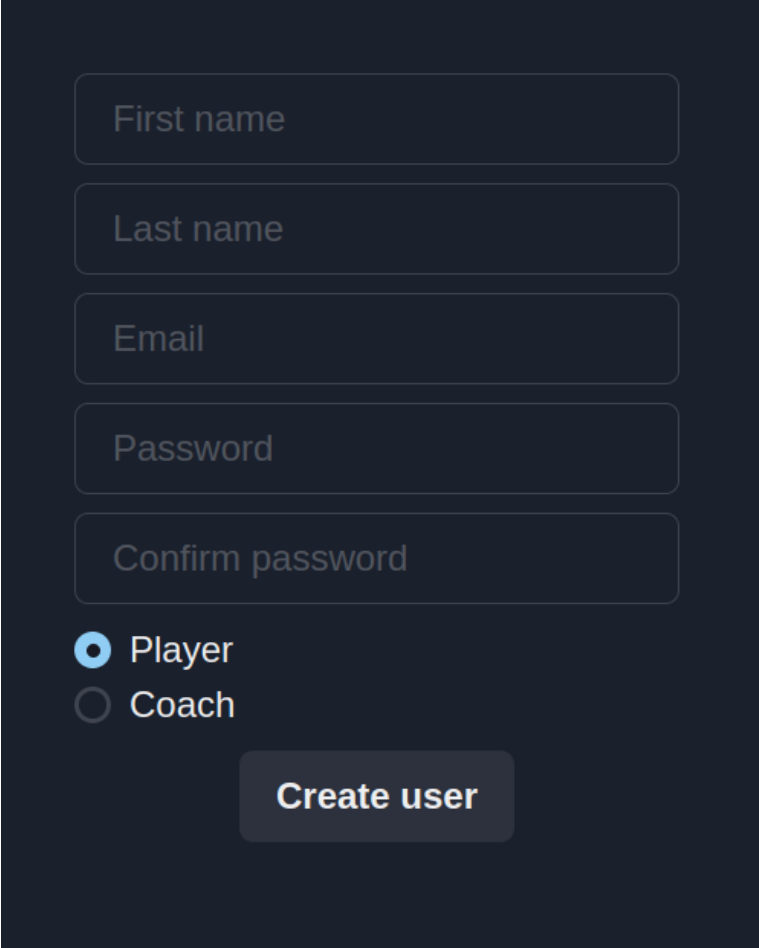
Before being able to create a team, the coach needs to create a user. This is done through the page shown in figure 4.7.

When creating a user through the form shown in figure 4.7, they need to provide some personal information. This includes an email, password, and whether they are a player or a coach. This information is sent to the backend using the REST API. The backend will then hash the password using Argon2 as the hashing algorithm. Why Argon2 was chosen as the hashing algorithm for passwords is discussed in section 6.9. This information is then stored persistently in the users-collection of the database.

4.6 Create Team

One of the functional requirements for Dárkon is that it should be possible to create a group where all the videos are shared with members of the group. This is possible through the feature of creating a team. After a coach has created a user on the system, the coach must create a team through the form shown in figure 4.8 before being able to upload videos. Here the coach enters a name for the team. When creating a team, the backend generates a unique id for the team that all players, coaches, and videos will be associated with. This makes it possible to only share videos with everyone that is a member of the team.

When the coach has created the team, the players and other coaches can get



The image shows a user creation form on a dark background. It consists of five vertically stacked input fields with rounded corners and light gray text: 'First name', 'Last name', 'Email', 'Password', and 'Confirm password'. Below these fields are two radio button options: 'Player' (which is selected, indicated by a blue dot) and 'Coach'. At the bottom center of the form is a dark gray button with the text 'Create user' in white.

Figure 4.7: Form when creating a user.

invited to join the team through a token. All coaches that are part of a team is able to generate a token on the page shown in figure 4.9. The coach can then send this token to the players and other coaches, and they are able to join the team by navigating to the same page as shown in figure 4.8. This page will however look different for the players, as they do not have the option to create a team.

The token is only available for 60 hours. This means that, if the players and other coaches does not use it within this time frame, the coach will need to generate a new token.

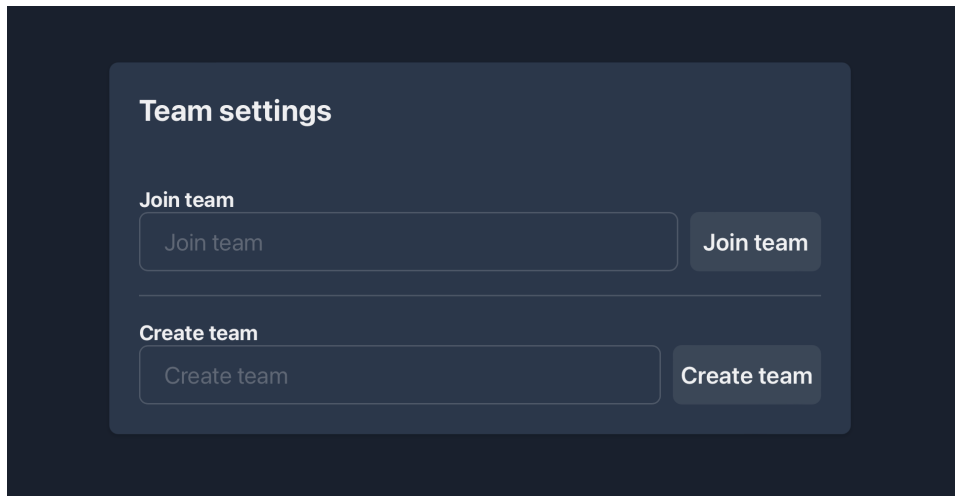


Figure 4.8: Page when creating or joining a team.

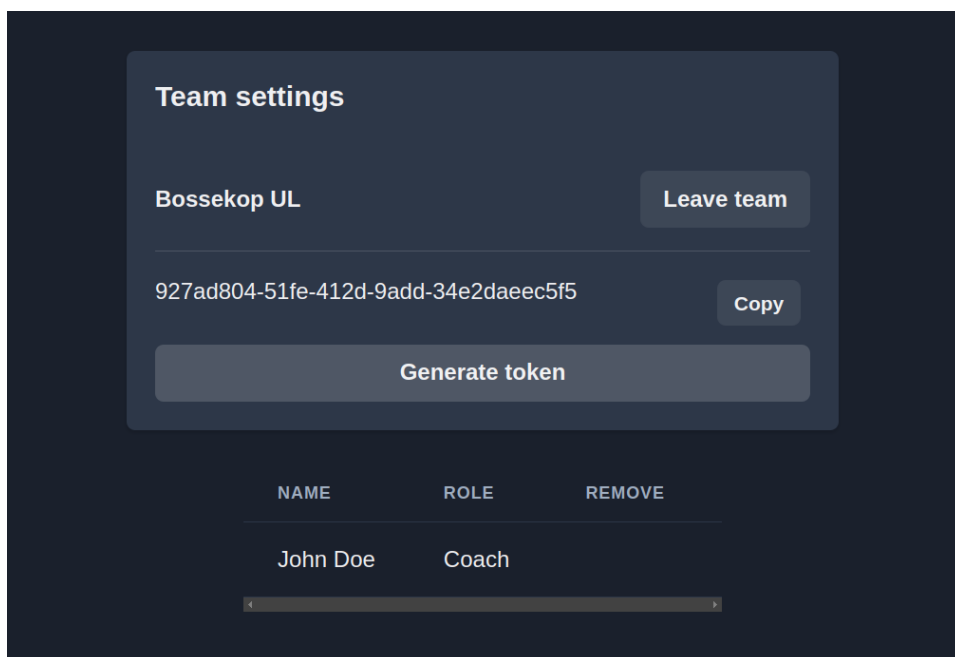
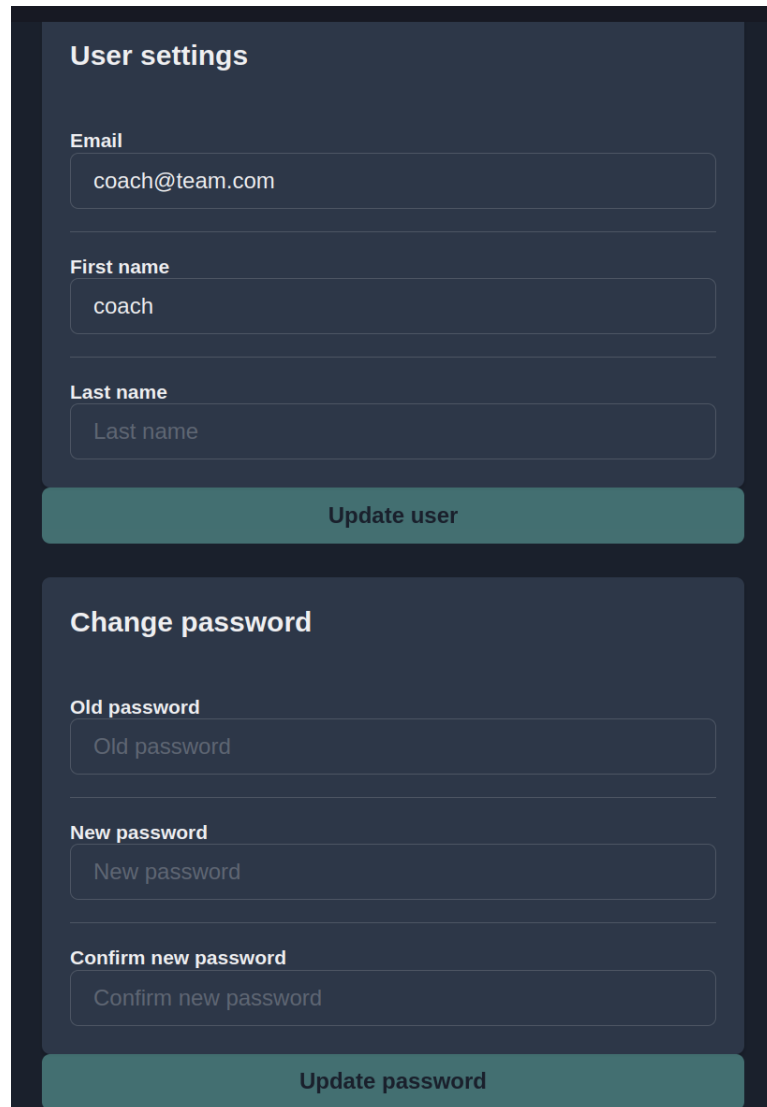


Figure 4.9: Settings for a team.

4.7 Edit Profile

After a profile is created on Dárkon, it is also possible to change the personal information that they provided when creating the account. This includes the email, name, and password.



The image displays two stacked form cards on a dark background. The top card is titled "User settings" and contains three input fields: "Email" with the value "coach@team.com", "First name" with the value "coach", and "Last name" with the placeholder "Last name". Below these fields is a teal button labeled "Update user". The bottom card is titled "Change password" and contains three input fields: "Old password" with the placeholder "Old password", "New password" with the placeholder "New password", and "Confirm new password" with the placeholder "Confirm new password". Below these fields is a teal button labeled "Update password".

Figure 4.10: Page when editing a user.

Figure 4.10 shows the page when a user is changing their information. The card on the top is for changing their email or name, while the bottom card is for changing their password.

4.8 Handling of Uploaded Videos

Before a team can start analysing their match, the video must be uploaded to Dárkon. As discussed in section 3.1.2, the size of the videos uploaded to Dárkon

will usually be multiple GB. Handling upload and storage of videos of this size is one of Dárkon's core requirements, and this section will explain how the system handles this.

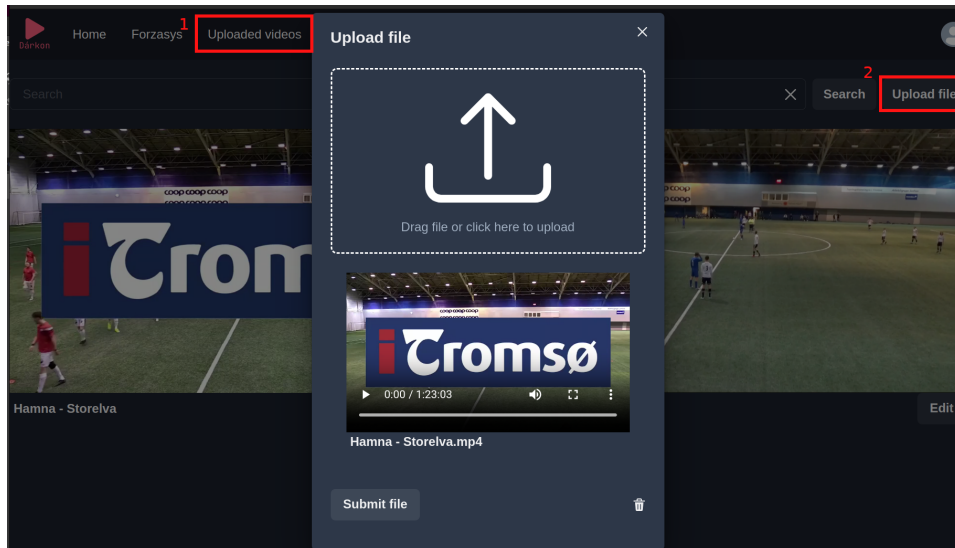


Figure 4.11: Preview of when uploading a video.

Figure 4.11 shows the preview of what a user observes when uploading a video. To upload a video, the user navigates to the page by clicking on rectangle number one in the navigation bar from figure 4.11. Clicking on rectangle number two will bring up the pop-up for uploading a video.

To upload a video, one could either click or drag a video into the striped box, and then click the button to submit it.

The client will then send the video as a Hypertext Transfer Protocol (HTTP) POST request to the REST API defined by the backend. The video will be sent with a content type of *multipart/form-data*. For the server to handle large video uploads, it streams the incoming data. The backend incrementally writes the video to disk with a temporary name. If the video were instead stored in memory, the size of the video would be dependent on the amount of Random Access Memory (RAM) available.

When the entire video is stored persistently, the video is sent to FFmpeg for an encoding job. This job encodes the uploaded video into HLS by copying the codec for faster encoding. The reason for copying the codec instead of re-encoding the video is because re-encoding is time-consuming. This will be discussed further in section 5.2 and 6.4.

```

pub async fn save_video(
    &self,
    id: ObjectId,
    mut field: Field,
) -> Result<PathBuf, anyhow::Error> {
    let tmp_path = self.videos_path.join(format!("{}", id.to_hex()));
    let _path = tmp_path.clone();
    let mut f = web::block(|| ftd::fs::File::create(_path)).await??;
    while let Some(chunk) = field.next().await {
        let data = chunk?;
        let Ok(Ok(file)) = web::block(move || {
            f.write_all(&data).map(|_| f)
        }).await else {
            tokio::fs::remove_file(tmp_path).await?;
            return Err(anyhow::anyhow!("Couldn't write bytes into video"));
        };
        f = file;
    }

    let dir_path = self.videos_path.join(id.to_hex());
    tokio::fs::create_dir(&dir_path).await?;
    let output = self.get_storage_path(id, DarkonDatabase::VIDEO_FILENAME);
    ffmpeg::hls_encode_video_copy_codec(&tmp_path, &output).await?;
    tokio::fs::remove_file(tmp_path).await?;

    Ok(output)
}

```

Figure 4.12: Implementation of how the backend handles upload, storage and encoding of a video.

Figure 4.12 shows the implementation in Rust of streaming the video, saving it under a temporary name, before eventually encoding the video using FFmpeg.

Initially, the video was stored in a single file using MP4 as a video format container. However, one of the core requirements of Dárkon is to be able to watch small sequences, or multiple discontinued events from a match or training session. Having the entire video in a single MP4 container would make this requirement difficult to fulfill. By using FFmpeg to split the video into multiple smaller segments, and playing them over the network using HLS as the streaming protocol, makes this possible. As it is the clients responsibility to request the correct segments when using HLS, the backend could send back a manifest file including the requested segments, and the client will then only play the segments returned by the server.

As well as encoding the video to HLS, a thumbnail is generated from the video to be displayed to the user as visual feedback before clicking on the video. A

thumbnail is generated from the first frame in the fifth second in the video. The fifth second is chosen based on the research on automatic vs. static generation of thumbnails [38]. This is, however, dependent on the video being more than five seconds long. If the video is shorter than five seconds, it will use the first frame of the video as a thumbnail.

4.9 Video Metadata

To be able to start analyzing the uploaded videos, Dárkon needs some metadata about each video. This metadata is generated and exported to a JavaScript Object Notation (JSON)-file through another system. While developing Dárkon, there is a separate thesis that deals with the generation of tags. However, the metadata that Dárkon currently uses is generated through a system developed by the head coach of Hamna IL, Martin Rypdal.

4.9.1 Metadata Format

There has been an active collaboration between Dárkon and Martin's system to agree on a metadata format that both systems understand. Starting out, Martin's system generated events in two different files, where each line in one file correlated with the same line in the other file.

This required parsing two text files and mapping them together, having more room for error if one of the files was not configured correctly.

a	b	c	d	e	f	g
{22.11,	{82.5,	24.6},	22.18,	{79.3,	12.3},	"P1", "Pass", "P2"}
{22.18,	{79.3,	12.3},	22.24,	{79.4,	34.4},	"P2", "Pass", "P3"}
{22.24,	{79.4,	34.5},	22.31,	{64.6,	53.6},	"P3", "Pass", "P4"}
{22.31,	{64.7,	53.8},	22.35,	{57.2,	49.5},	"P4", "Pass", "P5"}
{22.35,	{57.2,	49.5},	22.44,	{83.3,	27.2},	"P5", "Pass", "P1"}
{22.44,	{83.3,	27.2},	22.44,	{72.9,	10.4},	"P1", "Pass", "P2"}
{22.44,	{72.9,	10.4},	22.51,	{56.3,	10.0},	"P2", "Føring", "P2"}
{22.51,	{56.3,	10.0},	22.54,	{51.5,	22.4},	"P2", "Pass", "P6"}
{22.54,	{51.5,	22.5},	22.59,	{59.3,	44.6},	"P6", "Pass", "P3"}
{22.59,	{58.8,	43.7},	22.67,	{44.9,	53.2},	"P3", "Pass", "P4"}

Figure 4.13: Example illustration of the first iteration of general purpose tagging.

Figure 4.13 shows an example of 10 events and how the metadata was generated from Martin's system. The players' names are substituted with Px for privacy. Table 4.1 displays the meaning of each field in the metadata file.

Figure 4.14 shows an example of the more detailed information about the 10

Tag	Meaning
a	Time in seconds for the start of the event
b	X and Y coordinate for the start of the event
c	Time in seconds for the end of the event
d	X and Y coordinate for the end of the event
e	Player performing the action
f	Type of action
g	Result of action

Table 4.1: Description of the first iteration of metadata.

```

a b c d e
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}
{0, 0, 0, 0, 0}

```

Figure 4.14: Example of the more detailed tagging from the first iteration.

Tag	Meaning
a	Player won the ball
b	Player made a dangerous mistake
c	Breakthrough ball
d	Goal
e	Corner

Table 4.2: More detailed description of each tag.

same events as in figure 4.13, and table 4.2 shows the description of the different fields.

When starting to develop Dárkon, it quickly became noticeable that the structure of the metadata was difficult to work with. Having two different files required both of them to be uploaded to Dárkon. In addition to this, the custom structure of the metadata did not allow for easy parsing compared to a standardized structure like JSON. By collaborating with Martin, an agreement of a structure that would work for both systems was made. As the metadata did not use a standardized format, Martin agreed to switch to JSON to make it easier for

Dárkon to parse the data. Since JSON is a standardized format, it reduced the need for manual parsing by the backend. By using a popular crate in Rust, `serde`⁶, it is possible to automatically deserialize a JSON object into an object in Rust.

```
use serde::{Deserialize, Serialize};

#[derive(Serialize, Deserialize)]
pub enum TagsEventResult {
    #[serde(rename = "motspiller")]
    Opponent,
    #[serde(rename = "medspiller")]
    TeamMate,
    #[serde(rename = "goal")]
    Goal,
    #[serde(rename = "setPieceFor")]
    FoulWon,
    #[serde(rename = "setPieceAgainst")]
    FoulLost,
}

#[derive(Serialize, Deserialize)]
pub struct Tag {
    #[serde(rename = "timeId")]
    pub event_id: i32,
    pub event_type: TagsEvent,
    pub start_player: TagsPlayer,
    pub end_player: TagsPlayer,
    pub result: TagsEventResult,
    pub start_coordinates: Coordinates,
    pub end_coordinates: Coordinates,
    pub start_time: f64,
    pub end_time: f64,
    pub ball_win: bool,
    pub break_through: bool,
    pub dangerous_mistake: bool,
    pub playing_direction: PlayingDirection,
}

#[derive(Serialize, Deserialize)]
pub struct Coordinates {
    pub x: f32,
    pub y: f32,
}

#[derive(Serialize, Deserialize)]
pub enum PlayingDirection {
    LeftToRight,
    RightToLeft,
}
```

6. <https://crates.io/crates/serde>

```

#[derive(Serialize, Deserialize)]
pub struct TagsPlayer {
    pub team: String,
    pub player: String,
}

#[derive(Serialize, Deserialize)]
pub enum TagsEvent {
    Pass,
    #[serde(rename = "Føring")]
    Dribble,
    #[serde(rename = "Avslutning")]
    Shot,
    #[serde(rename = "Innlegg")]
    Cross,
}

```

Figure 4.15: Structure of tagging data in Rust.

```

[
  {
    "timeId": 1,
    "eventType": "Pass",
    "startPlayer": {
      "team": "our",
      "player": "Player 1"
    },
    "endPlayer": {
      "team": "our",
      "player": "Player 2"
    },
    "result": "medspiller",
    "startCoordinates": {
      "x": 82.5,
      "y": 24.6
    },
    "endCoordinates": {
      "x": 79.30000000000001,
      "y": 12.4
    },
    "startTime": 22.12,
    "endTime": 22.19,
    "playingDirection": "RightToLeft",
    "ballWin": false,
    "breakThrough": false,
    "dangerousMistake": false
  }
]

```

Figure 4.16: Example of a tagged event from a match.

Figure 4.16 shows the metadata after switching to JSON. This shows a pass from one player to another. Since JSON is a structure that allows for nested objects, it also allowed for grouping everything into one file. This made it easier for Dárkon to work with, as only one file needs to be uploaded. As mentioned, it also made it easier to deserialize into Rust types as structs and enums, which can be observed in figure 4.15.

4.9.2 Uploading of Metadata

When the coach is finished tagging the match with specific events on the agreed standard, the metadata file is exported from the third-party system, and uploaded to Dárkon by clicking on the rectangle shown in figure 4.17.

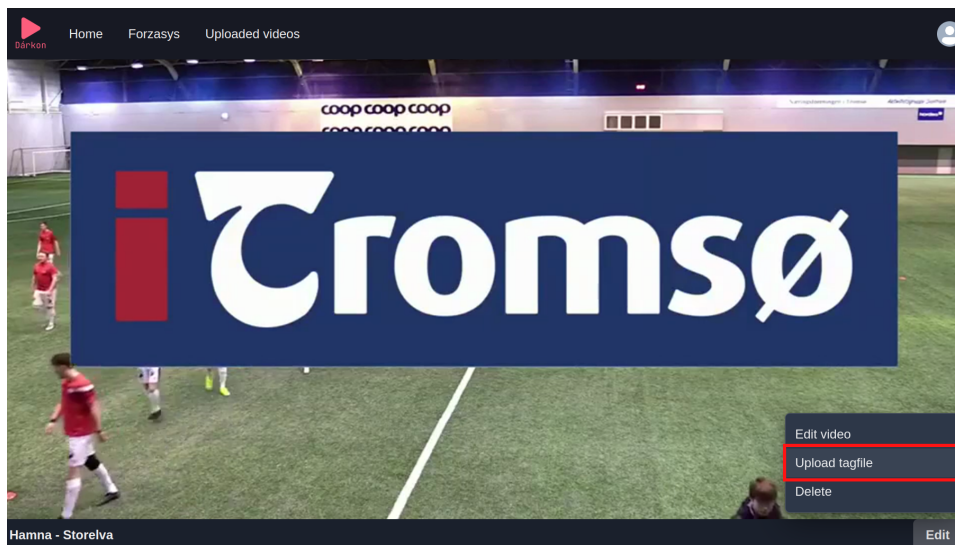


Figure 4.17: Option for uploading a metadata file.

The metadata file is sent to the server. It then stores all the objects inside the metadata file persistently in the tags-collection in the database as events for the corresponding video. This gives options for filtering specific sections from the video file, making it possible to retrieve only portions of the video instead of the entire video.

Along with storing the metadata, a thumbnail for each event is created using the start-time of the event as the offset. These thumbnails are shown in a carousel browsing video clips from the match. The carousel layout can be observed in figure 4.19.

4.10 Streaming of Uploaded Videos

By utilizing the metadata and HLS encoding, Dárkon can stream the video in multiple ways to the user. While being able to stream the entire video, it can also stream only portions of the video, filtered by the user based on the metadata uploaded along with the video. This section will present how Dárkon achieves this.

4.10.1 Entire Video

When streaming the entire video, the main manifest file which is created when encoding the video to HLS, is sent from the server to the client. The client will then request the segments in the manifest file from the server, and play them through a video player.



Figure 4.18: Page showing the entire uploaded match.

Figure 4.18 shows the page for streaming the entire video. The right side of the figure, shows a list of events for the specific video, allowing the user to jump directly to the specified event using the start-time from the metadata.

Since the metadata also contains information about specific events, the user is able to filter based on some of these events, making it easier to find exactly what they are looking for.

4.10.2 Event Based

Along with streaming the entire video, Dárkon can play just a single event, avoiding loading the entire video.

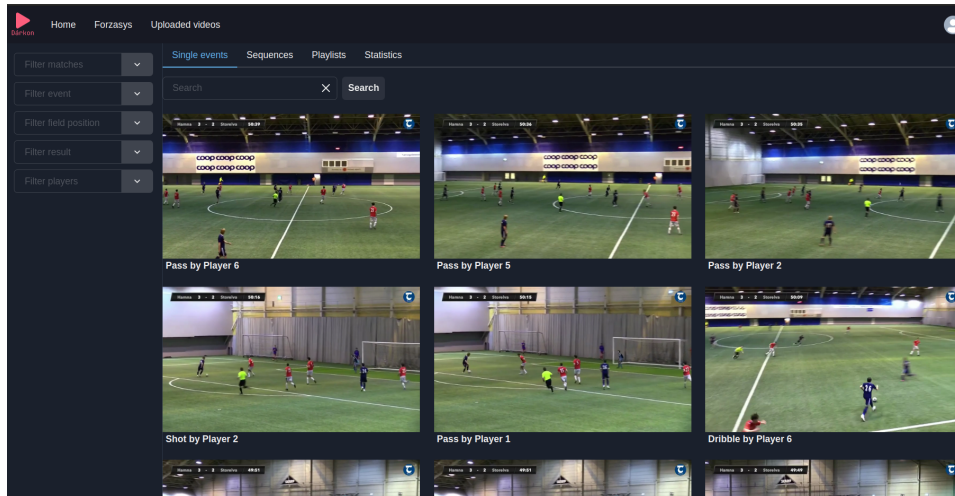


Figure 4.19: Page showing all single events.

Figure 4.19 shows the carousel for all the events by themselves, allowing the user to click on a thumbnail to only play the specific event.

A user can filter out certain events based on predefined parameters to make it easier to find a specific event. The available filters are, which match it was, which type of event they are looking for, where on the field it happened, and the event's outcome. These events may be filtered together so one could find a shot from the midfield from a particular match that ended in a goal.



Figure 4.20: Filters for a single event.

Figure 4.20 shows the available filters when filtering on single events, except

for the filter of which match it was, since that filter depends on which videos are uploaded to Dárkon. For the filter based on players, the names of the players are replaced with *Player 1-8* for privacy.

```
pub fn get_sequence(
    from_ms: u32,
    to_ms: u32,
    playlist: &MediaPlaylist,
) -> Vec<ExtInf> {
    let from_sec = (from_ms / 1000) as f32;
    let to_sec = (to_ms / 1000) as f32;

    let mut duration_start = 0.0;
    let mut duration_end = 0.0;

    let start_time_offset = 2.0;
    let end_time_offset = 2.0;

    playlist
        .segments
        .iter()
        .filter_map(|segment| {
            duration_start += segment.duration;

            if (duration_start + start_time_offset > from_sec)
                && (duration_end - end_time_offset < to_sec)
            {
                duration_end += segment.duration;
                return Some(ExtInf {
                    duration: segment.duration,
                    location: segment.uri.clone(),
                    output_num: segment
                        .uri
                        .chars()
                        .filter(|c| c.is_ascii_digit())
                        .collect:::<String>()
                        .parse:::<u32>()
                        .ok()?,
                });
            }

            duration_end += segment.duration;

            None
        })
        .collect()
}
```

Figure 4.21: Implementation of finding the start- and end-time of a sequence.

When a client requests to watch a single event, it sends the start- and end-time as query parameters to the backend. The server will then read the entire

main manifest from start to end, adding the duration of all the segments together until it finds the first segment that happens *before* the user-specified start-time.

The server then takes the following segments *including* the segment that ends after the end-time specified by the client. The implementation of this is observed in figure 4.21.

A temporary manifest file is then constructed with the segments containing the timestamp by the user. It is returned to the client, who can then play the segments containing the specific event. The implementation of the manifest file construction is observable in figure 4.22.

```
pub fn generate_m3u8(
    target_duration: f32,
    segments: Vec<ExtInf>,
    id: ObjectId,
) -> String {
    let vec: Vec<_> = segments
        .into_iter()
        .map(|segment| {
            format!(
                "#EXTINF:{}\n{}\n",
                segment.duration,
                get_api_path(segment.location, id),
            )
        })
        .collect();
    let mut m3u8 = String::new();
    m3u8.push_str("#EXTM3U\n");
    m3u8.push_str(&format!("#EXT-X-TARGETDURATION:{}\n", target_duration));
    m3u8.push_str("#EXT-X-VERSION:3\n");
    m3u8.push_str(&vec.join(""));
    m3u8.push_str("#EXT-X-ENDLIST");
    m3u8
}
```

Figure 4.22: Implementation of generating a manifest file.

4.10.3 Sequence

When analyzing a football match, it is not always enough to only watch a single event lasting a few seconds. It may be interesting to watch a sequence of events. For instance, a coach may be interested in finding all the sequences where the team keeps possession of the ball for at least 10 events. Adding to this, the coach may only want the sequences ending in a shot to analyze if there is something in common with these sequences.

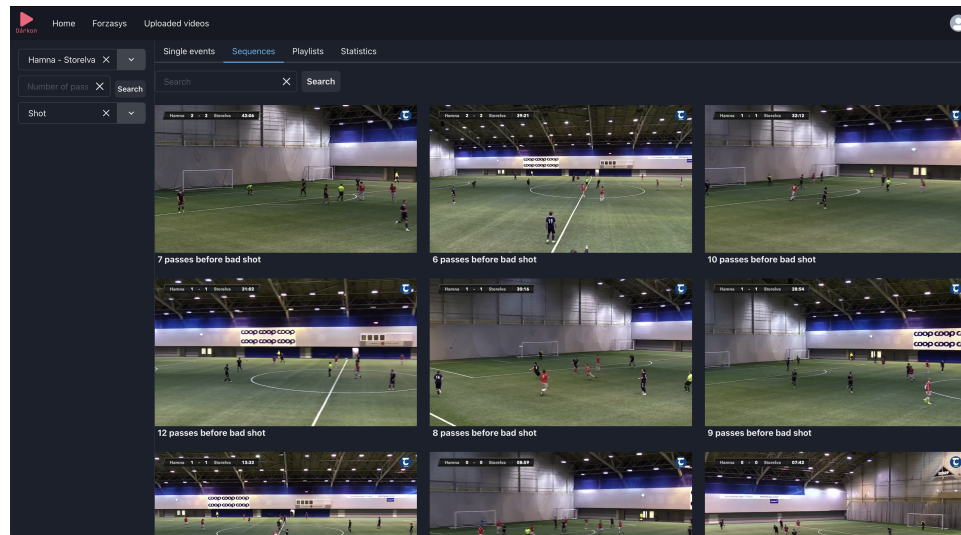


Figure 4.23: Page showing sequences from a match.

In Dárkon, this is possible by utilizing the metadata for the video and the segmentation of the video using HLS. As the client is responsible for asking for the segments to play, the server may generate a manifest file including the necessary segments for the client to utilize.

To get a sequence where the team has possession for more than x number of events, the events for a match need to be sorted in descending order based on the offset in the video. All the events are then iterated through in the order from last to first. All events are assumed to be part of a sequence until the opponent wins the ball. If the sequence of events reaches the threshold specified, it is committed as a definite sequence. However, it will continue adding the following events to the sequence until the ball is lost.

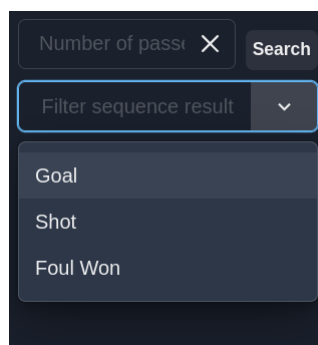


Figure 4.24: Filters available for a sequence.

Figure 4.24 shows the available filters for a sequence. Sequence filters are

different from filtering a single event. This is because sequences are naturally finished when the team loses the ball. Therefore it does not make sense to filter the result of a sequence to when a player loses the ball.

The client is returned the start- and end-time after calculating the sequences, displaying the thumbnail from the first event in the sequence in the carousel shown in figure 4.23.

When the user wants to view the sequence, the start- and end-time is sent to the server, which will generate the manifest file in the same way as for single events. This can be observed in figure 4.22.

4.10.4 Playlist

A playlist is a video of discontinued sequences played as one coherent video. In Dárkon, it is possible to get all involvements from a player for a single match, and combine them all into a single video.

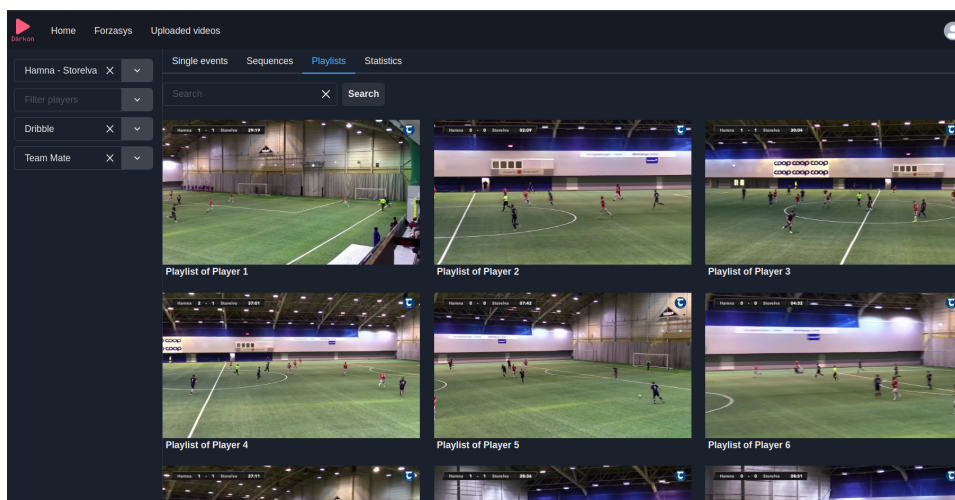


Figure 4.25: Page showing playlists from a match.

Figure 4.25 shows a carousel of thumbnails for the different playlists available. The playlists in the carousel are grouped per player, and it is possible to filter the events based on the event type, result, and player.

Figure 4.26 shows the filters available for a playlist. These are almost the same as for a single event. The only difference is the exclusion of filtering on the field position, which is not implemented at the current moment. As in figure 4.19, the real names of the players are replaced with *Player 1-8* for their privacy.

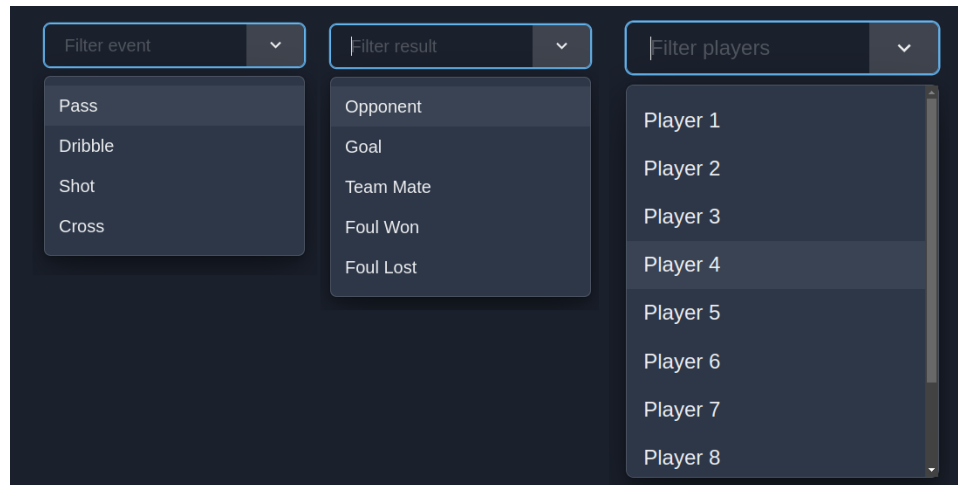


Figure 4.26: Filters available for playlists.

The server uses the filters specified by the client to generate a manifest file, including the segments encapsulating the events which should be included in the playlist. The segments may be out of order when constructing the manifest file for a playlist, since it is one coherent video of a player's events throughout a game. For a client to properly play a video of discontinued segments, an EXTINF-tag: EXT-X-DISCONTINUITY must be placed between the discontinued segments.

Upon generating the manifest file for a playlist, the server uses the segments and inserts the specified tag if the segments are out of order.

Figure 4.27 illustrates an example of a generated manifest file for a playlist with the discontinuity tag placed between segments that are out of order. Giving the client the information about discontinuity, it can play the segments as one coherent video.

4.11 Statistics

While developing Dárkon, there was a proposal from the coach of Hamna IL to include a tab of statistics from a match, utilizing the already stored metadata. This would summarize a player's performance in a match with statistics.

Figure 4.28 demonstrates some statistics from a match based on the same metadata used to find specific video clips from the match.

```

#EXTM3U
#EXT-X-TARGETDURATION:3
#EXT-X-VERSION:3
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output801.ts
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output802.ts
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output803.ts
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output804.ts
#EXT-X-DISCONTINUITY
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output925.ts
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output926.ts
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output927.ts
#EXT-X-DISCONTINUITY
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output1065.ts
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output1066.ts
#EXTINF:2
https://localhost:8080/api/filestorage/video/xxx/output1067.ts
#EXT-X-DISCONTINUITY
#EXT-X-ENDLIST

```

Figure 4.27: Example of a generated manifest file for a playlist.

PLAYER	INVOLVEMENTS	PASSES	SUCCESSFUL PASSES	DRIBBLES	SUCCESSFUL DRIBBLES	GOALS	ASSISTS	BALL WON
Player 1	25	17	16	5	5	0	0	2
Player 2	30	22	19	4	3	0	0	6
Player 3	27	23	19	3	1	0	0	3
Player 4	26	22	20	4	3	0	0	3
Player 5	60	52	45	6	6	0	0	9
Player 6	13	9	6	3	3	0	0	0
Player 7	9	9	9	0	0	0	0	1
Player 8	7	7	6	0	0	0	0	0
Player 9	17	10	8	4	4	0	0	0

Figure 4.28: Page showing statistics from a match.

One of the fields specifies the number of assists in a game. This is not something that is stored in the metadata directly, but rather something that is calculated

based on the order of events when navigating to the page for statistics.

By iterating through the events in decreasing order based on the offset in the video, we can calculate whether a player got an assist. The previous events are examined when reaching an event that specifies a goal, and if it finds a pass to the player that scored, the player that made the pass got an assist.

4.12 Third-Party Systems

Another objective of Dárkon is to provide integration to third-party video sources that allow it. Integration with Eliteserien Highlights is already working, and integration with other third-party sources is being investigated. Another source that Dárkon tried to integrate with is Hudl. However, the decision was not to integrate with Hudl for now. The reason is explained in section 6.5.

To integrate with Eliteserien Highlights, a user may click on rectangle one to navigate to the page with Eliteserien Highlights clips. Upon the first navigation to the page, a team id is requested. To integrate with Forzasys for videos in Eliteserien, the team id that they use in their system is needed for the backend to fetch the correct videos from their API. When the team id is provided, the backend will immediately query their API for videos dating back til 1.January 2022, storing information about the events locally in the database. It also starts a background job that fetches new videos once a week.

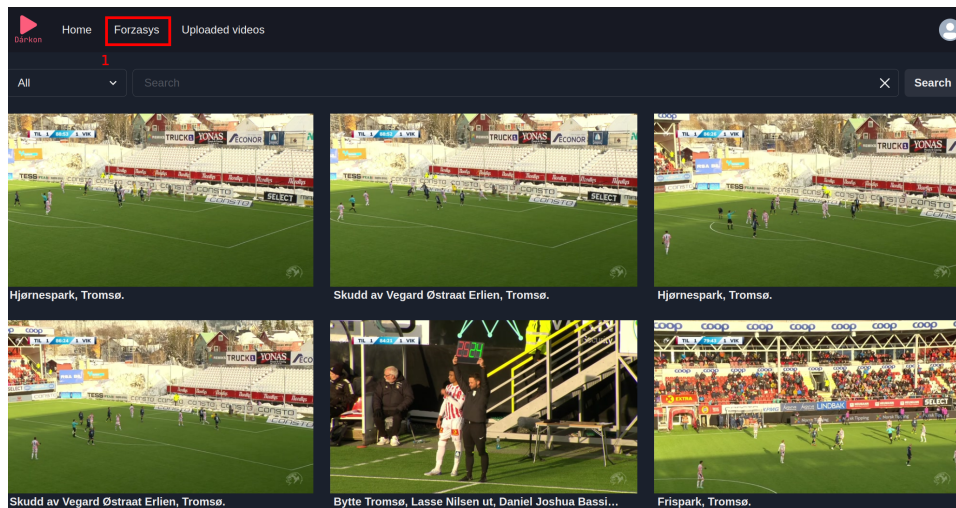


Figure 4.29: Page showing the integration with Eliteserien Highlights.

Figure 4.29 shows a carousel of videos from Eliteserien Highlights, which is

integrated into Dárkon. This shows the videos from TIL which is the local team in Tromsø.

```
{
  "playlists": [
    {
      "creator": {
        "id": "xxx",
        "name": "tags-video",
        "profile_image": null
      },
      "date": "2021-09-18T19:50:28.131089Z",
      "description": "Skudd av Player, Team.",
      "duration_ms": 25000,
      "events": [
        {
          "content_source": "Mediabank",
          "description": "Skudd av Player, Team.",
          "from_timestamp": 7161000,
          "recording_timestamp": "2021-09-18T19:49:20.000000Z",
          "tags": [
            {
              "action": "shot",
              "player": {
                "id": 0,
                "type": "player",
                "value": "Player"
              },
              "shot type": {
                "type": "shot type",
                "value": "standard"
              },
              "team": {
                "id": 0,
                "type": "team",
                "value": "Team"
              }
            }
          ],
          "to_timestamp": 7186000,
          "video_asset_id": 0,
          "video_asset_name":
↪ "https://xxx.cloudfront.net/mediabank/thumb/eliteserien/xxx"
        }
      ],
      "frontend_url": "https://highlights.eliteserien.no/playlist/xxx",
      "game": {
        "attendance": 0,
        "date": "2021-09-18",
        "end_of_1st_half": "2021-09-18T18:46:07.000000Z",
        "end_of_1st_overtime": null,
        "end_of_2nd_half": "2021-09-18T19:50:03.000000Z",
        "end_of_2nd_overtime": null,

```

```

    "everysport_id": null,
    "extra_referees": null,
    "finished_time": "2021-09-18T19:50:03.000000Z",
    "home_team": {
      "coach": null,
      "id": 0,
      "import_id": 0,
      "import_source": "FIKS",
      "logo_url":
↪ "https://cdn.forzasys.com/team-logos/tippeligaen/Team.png",
      "name": "Team",
      "short_name": "Team"
    },
    "home_team_goals": 3,
    "id": 0,
    "import_id": "xxx",
    "import_source": "FIKS",
    "is_available_publicly": true,
    "officially_closed": false,
    "phase": "finished",
    "public_release_timestamp": "2021-09-18T20:30:00.000000Z",
    "referee_external_id": null,
    "referee_name": null,
    "round": 23,
    "stadium_name": null,
    "start_of_1st_half": "2021-09-18T18:01:04.000000Z",
    "start_of_1st_overtime": null,
    "start_of_2nd_half": "2021-09-18T19:01:43.000000Z",
    "start_of_2nd_overtime": null,
    "start_of_penalties": null,
    "start_time": "2021-09-18T18:00:00.000000Z",
    "stop_periods": [],
    "tournament": {
      "id": 13,
      "league_name": "eliteserien",
      "name": "2021"
    },
    "tournament_id": 13,
    "tournament_name": "Eliteserien",
    "visiting_team": {
      "coach": null,
      "id": 0,
      "import_id": 15,
      "import_source": "FIKS",
      "logo_url": "https://cdn.forzasys.com/team-logos/forzify-backend ]
↪ -eliteserien/Team2.png",
      "name": "Team2",
      "short_name": "Team2"
    },
    "visiting_team_goals": 0
  },
  "hd_thumbnail_override_url": null,
  "hd_thumbnail_url":
↪ "https://xxx.cloudfront.net/mediabank/thumb/eliteserien/xxx/xxx.jpg",

```



```

    "id": "xxx",
    "is_accessible": true,
    "is_live": false,
    "is_placeholder": false,
    "is_private": false,
    "minified_frontend_url":
↪ "http://highlights.eliteserien.no/webview/playlist/xxx",
    "rating": 1,
    "thumbnail_override_url": null,
    "thumbnail_url":
↪ "https://xxx.cloudfront.net/mediabank/thumb/eliteserien/xxx/xxx.jpg",
    "video_url": "https://api.forzasys.com/eliteserien/playlist.m3u8/xxx
↪ :7161000:7186000/Manifest.m3u8",
    "view_count": 0
  }
]
}

```

Figure 4.30: Example response from Forzasys with metadata from Eliteserien.

Figure 4.30 illustrates an example response the backend gets from the API of Eliteserien Highlights. The response contains a lot of extra information that is not needed for Dárkon. Therefore, the response is parsed, trimmed down, and stored locally in the database. Dárkon does not download the video files, but instead uses the Uniform Resource Locator (URL) from the API of Eliteserien Highlights to play the videos directly from them. This means that if their service is down, watching the videos through Dárkon will be impossible. However, this minimizes the need for storage, as Dárkon is already storing videos uploaded by the users.

Instead of parsing and storing the metadata locally, a request could be made to their API for new information each time a user requests it. This would remove the need for a separate collection in Dárkon's database. However, this would provide unnecessary data processing, as much of the information is not needed for Dárkon. Storing the metadata locally also limits requests to their API to only when a user is watching the specific video.

Figure 4.31 shows the same event after removing the fields that are not relevant to Dárkon. This is an example of a collection in the `forzasysVideos`-collection in Dárkon's database.

```
{
  "_id": "xxx",
  "forzasysId": "xxx",
  "description": "Skudd av Player, Team",
  "round": 23,
  "date": "2021-09-18T19:50:28.131089Z",
  "teamId": 0,
  "action": {
    "type": "shot",
    "content": {
      "player": {
        "id": 0,
        "type": "player",
        "value": "Player"
      }
    }
  },
  "videoUrl": "https://api.forzasys.com/eliteserien/playlist.m3u8/xxx:7161_
↳ 000:7186000/Manifest.m3u8",
  "thumbnailUrl":
↳ "https://xxx.cloudfront.net/mediabank/thumb/eliteserien/xxx/xxx.jpg"
}
```

Figure 4.31: Stored information about an event from Eliteserien Highlights.

4.13 Summary

This chapter has described the design and implementation of Dárkon with the requirements from chapter 3 in mind. The main features of the system includes a way to watch a single event, sequence, and playlist extracted from the entire uploaded video. The next chapter will provide some experiments of the system to observe how it has been improved over time.

/5

Experiments

The previous chapter described the design and implementation of Dárkon in the context of the requirements outlined in chapter 3. This chapter will provide experiments for the system, and also present an evaluation of the Dárkon provided by the coach and players of Hamna IL.

5.1 Setup

The following experiments are conducted on an HP EliteDesk 800 G6 Desktop Mini PC running Ubuntu 22.04.2 LTS x86_64. The machine's CPU is a 10th generation Intel i7-10700 @ 4.800GHz with eight cores, each containing two threads, giving a total of 16 threads. The GPU configured is an Intel CometLake-S GT2 [UHD Graphics 630]. The machine also has 16GB of DDR4 RAM.

5.2 Re-Encode Video

When uploading a video to Dárkon, the backend copies the codec and encodes it into HLS. This means that it does not re-encode the video. Not re-encoding the videos may lead to problems like the video not being playable since it has an encoding that Dárkon does not support. However, re-encoding the entire video is also very time-consuming.

The video for this experiment lasts 1 hour and 23 minutes. The file size is 1.1GB. Both encoding jobs in this experiment are done using FFmpeg, with the commands from figure 5.1, and 5.2.

```
ffmpeg -i <input> -acodec copy -vcodec copy -preset ultrafast
↪ -flags -global_header -f hls -hls_time 2 -hls_list_size 0
↪ <output>
```

Figure 5.1: Command to copy codec with FFmpeg.

```
ffmpeg -i <input> -start_number 0 -hls_time 2 -hls_list_size 0 -f
↪ hls -force_key_frames:v 'expr:gte(t,n_forced*2)' <output>
```

Figure 5.2: Command to re-encode with FFmpeg into two second HLS segments.

In figure 5.1 and 5.2, <input> is replaced with the path to the video to be encoded, and <output> is replaced with the path to where the encoded video will be stored. Each of these commands was run five times each.

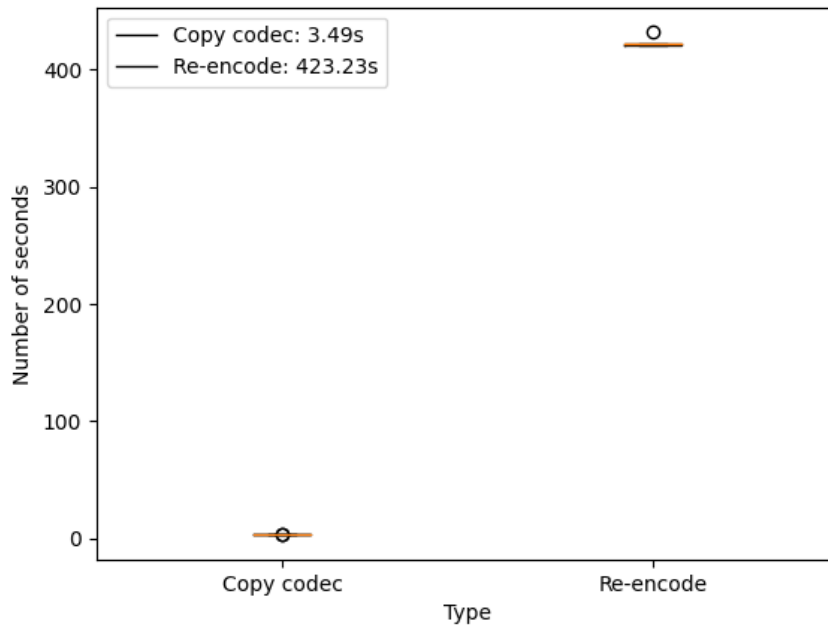


Figure 5.3: Point plot of the difference in seconds between re-encoding and copying the codec.

From figure 5.3, we can observe the time in seconds between re-encoding the video vs. copying the codec. The y-axis describes the time in seconds, while the

x-axis describes the encoding and re-encoding of the video. This experiment shows that the time to re-encode the video took about 423 seconds, while copying the codec took about 3 seconds. This means that re-encoding the given video is two orders of magnitude slower than copying the codec.

5.3 Thumbnail Generation

When uploading the metadata from the tagging system that provides them, one of Dárkon's responsibilities is to generate a thumbnail for each event. This provides a visual representation of the events when they are displayed in a grid. Since the metadata could contain several hundred events, and a thumbnail must be created for each, it could be a time-consuming process. In an earlier version of Dárkon, the thumbnails were generated sequentially, one event at a time. However, this took an unnecessary amount of time, since the thumbnails could be created in parallel. By creating multiple threads, each generating one thumbnail, we are able to speed up the process of creating thumbnails.

In this example, it is the metadata for the video from section 5.2. There were, in total, 449 events in the metadata, which means 449 thumbnails were created. This experiment was run five times each.

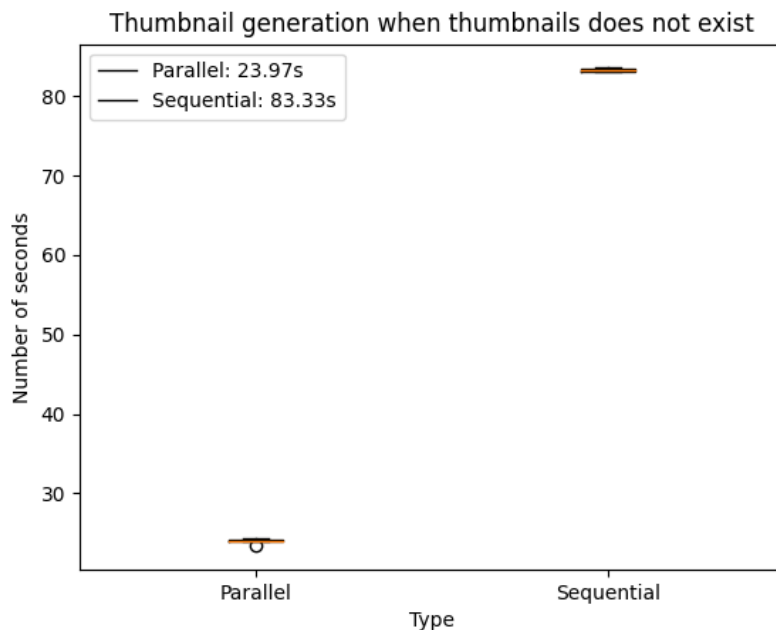


Figure 5.4: Upload of tags when no thumbnails exist.

Figure 5.4 illustrates the difference between the first implementation of generating thumbnails sequentially, and generating them in parallel when the events are uploaded to a corresponding video. The sequential implementation used 83 seconds on average, while the parallel version used 24 seconds.

When generating thumbnails for the events, the thumbnails are stored with the timestamp of the event as the filename. When resubmitting metadata for a video, it will check whether a thumbnail for the specific timestamp already exists. If it already exists, it will not overwrite the thumbnail. This should make it faster if the metadata is uploaded a second time, because it does not need to start a new process with FFmpeg.

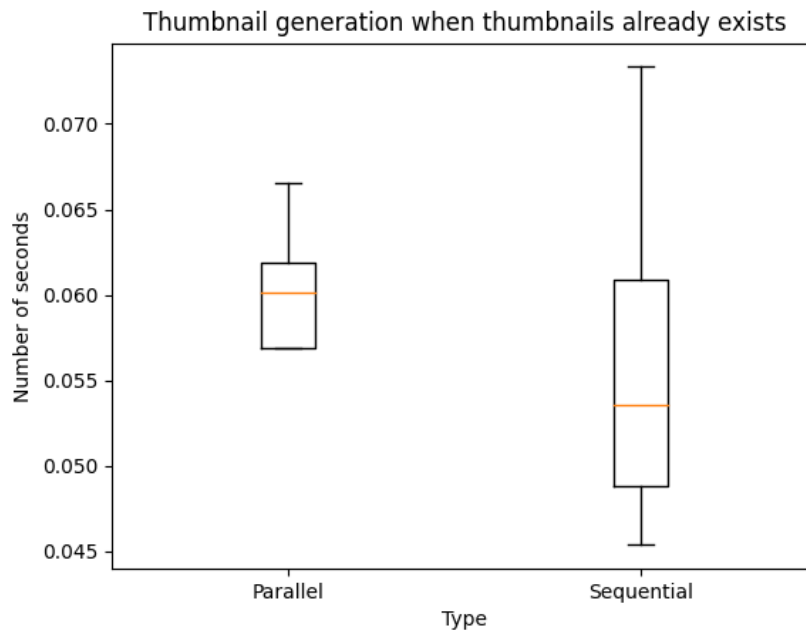


Figure 5.5: Upload of tags when thumbnails already exist.

Figure 5.5 illustrates that both the parallel version, and the sequential version is almost equally fast when the thumbnails already exist. On average, the sequential version is a bit faster, but fluctuates a bit more than the parallel version. This demonstrates that it is the thumbnail creation that is time-consuming when uploading the metadata. Since it takes under a second to recreate the tags in this case, there is a negligible difference between the parallel and sequential solution when the thumbnails already exist.

5.4 Sport and League Agnostic

While Dárkon is mostly developed with football in mind, it should already be agnostic across some sports, and also across different leagues and levels of teams. To test out this hypothesis, the FA Community Shield final from 30. July 2022, between Liverpool and Manchester City was chosen to test a different league and level. To test a different sport, a handball match between Norway and Hungary from the Women's World Cup in 2019 was chosen.

5.4.1 FA Community Shield

To try out Dárkon on the FA Community Shield final between Liverpool and Manchester City, a four minute clip from the match was extracted and annotated as Dárkon expects.

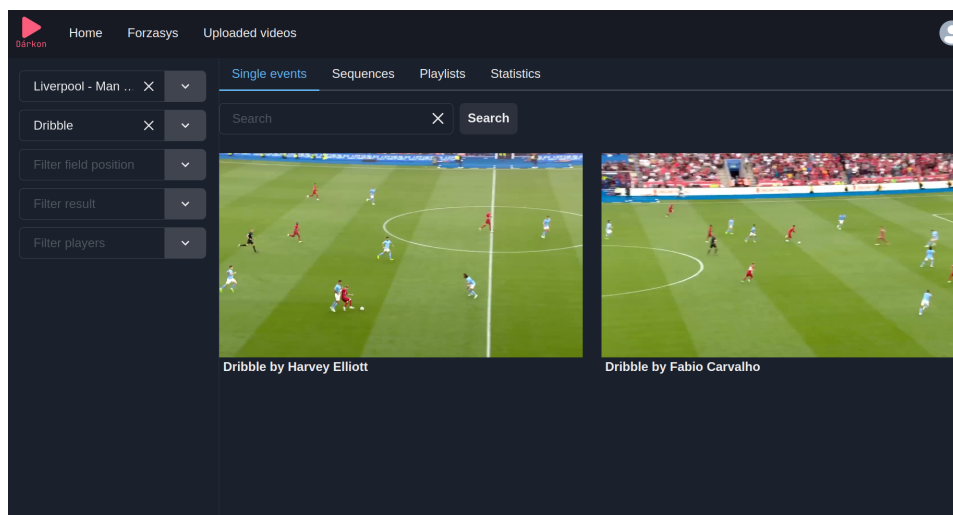


Figure 5.6: Tags when filtering dribbles [39].

Figure 5.6 shows the available clips from the Liverpool match after filtering on dribbles. Here we observe that Harvey Elliott and Fabio Carvalho had one dribble each during the four minutes that was annotated. By clicking on the clips, they are available to watch.

5.4.2 Handball Match

By evaluating if Dárkon is able to be sports-agnostic at this stage, a four minute clip from a handball match from the Women's World Cup in 2019 was used. This four minute clip was annotated the same way as every other video from a

football match.

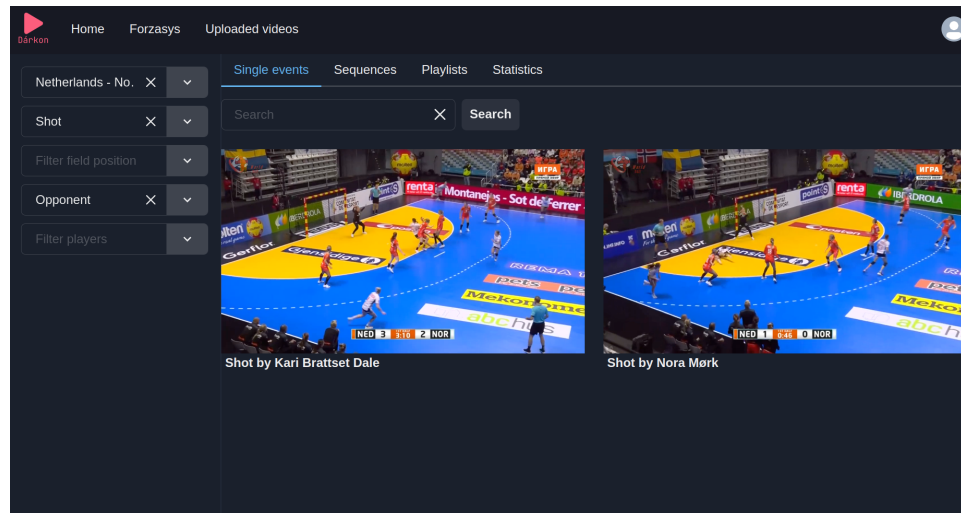


Figure 5.7: Tags when filtering bad shots [40].

Figure 5.7 illustrates that it is possible to filter based on, for instance, bad shots. During the span of the first four minutes, there was two shots that did not end in a goal.

This demonstrates that Dárkon is already able to annotate a handball match, making it available for players and coaches to analyze. One thing that was noticeable when tagging the handball match, is that filtering on field position is not possible at the moment. This is because the field is a bit different from a football field, both in size, and how it is sectioned. In a football match, it is for instance possible to filter events based on being inside the opponents box. This does not work on a handball field where it is illegal to enter the opponents box.

5.4.3 Summary of Sport and League Agnostic

Annotating the FA Community Shield final, and a handball match from the Women's World Cup demonstrates that Dárkon is working for both different levels of the game, and different sports. This illustrates that Dárkon is already level and sports-agnostic to some extent, even though it is developed mainly with football in mind. While all the functionalities like filtering based on the field position does not work for handball yet, this is something that could be expanded in the future.

5.5 Database Location

Dárkon is currently hosted on Azure by using a Virtual Machine (VM), and a database service. The VM is connecting to the database through a connection string, and is communicating with it over the network.

Starting out, the VM was located in Norway, but the database was located in USA because of the default settings on Azure. As this means that the devices are physically further apart than if both were in Norway, the latency for communication between the VM and the database became noticeable. After switching the location of the database to Norway, the latency got significantly reduced, and the overall experience got better because of the performance improvement.

Two databases were setup to quantify the improvement after switching the physical location of the database. One in Norway, and the other in USA. For the experiment, the tags-collection was queried five times each with the query from figure 5.8. Both collections in the databases were identical, each having 449 documents.

```
db.tags.find(  
  {  
    videoId: ObjectId("640886c010d6a764648726e7"),  
    event: "Pass",  
  },  
  {},  
  { sort: { startTime: -1 } }  
);
```

Figure 5.8: Query for tags-collection used for latency experiment.

Figure 5.9 shows the difference in latency between Norway and USA for setting up a connection with the database, and getting the results back from a query. It is a noticeable difference where it takes over three seconds for the result of the query when the database is located in USA. However, it takes under a second when the database is located in Norway. This experiment was conducted with a single query, and since the system is continuously communicating with the database, the latency becomes quite noticeable when using Dárkon. This is because multiple queries are made when navigating through the system on different pages.

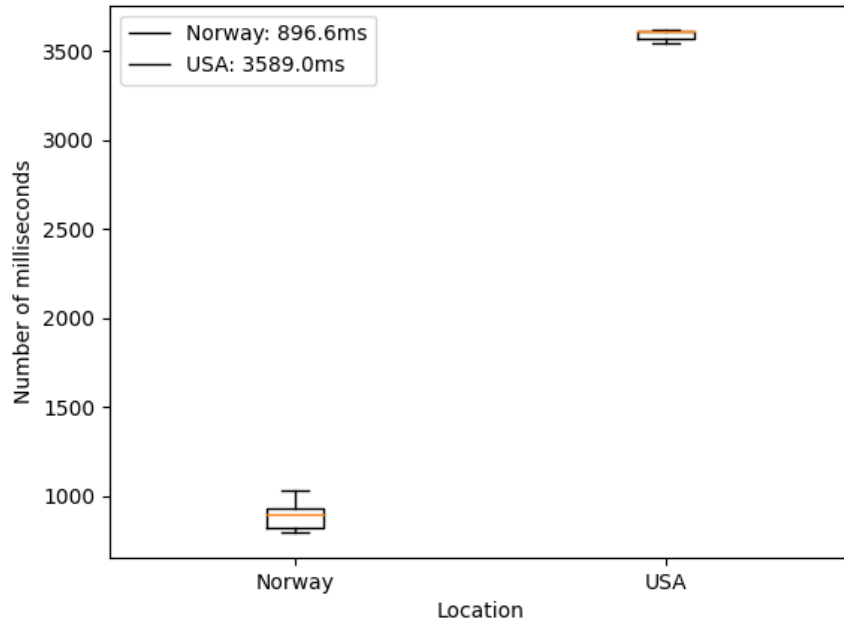


Figure 5.9: Latency difference between a database in Norway and USA.

5.6 Evaluation

As part of the experiments, a user evaluation was sent out to Hamna IL which has been using the system for the previous months. This has helped with what to focus on developing, and this section will present the evaluation from both the head coach, and the players on how the system is working for them.

To conduct the evaluation, multiple questions were sent out to the coach and the players. Most of the questions for the players and the coach are the same, and all but one questions are using the Likert Scale [41]. The questions are answered with a value from one to five, where three is a neutral answer. Lower than three means they disagree, and higher means they agree.

Hamna IL has previously not used any video analysis system, meaning that this user evaluation will focus on how satisfied they are with Dárkon, and what is most important for them. This could help with deciding what features are worth focusing most on for the future work, and to see if they are satisfied with Dárkon thus far.

5.6.1 Players

The players were sent five questions, where four of them are using the Likert Scale, and the last question is them answering what is most important for them in Dárkon. The results of the questions consists of answers from 11 players that are currently playing for Hamna IL.

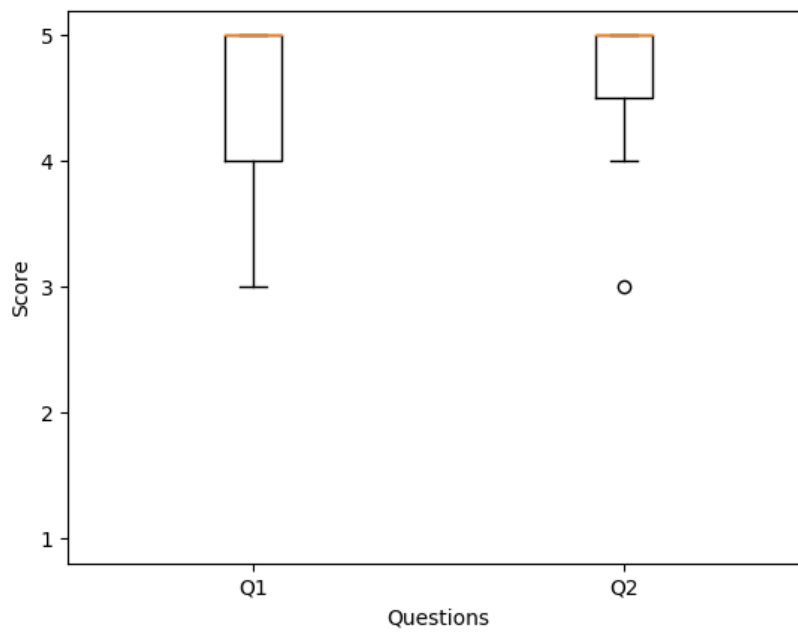


Figure 5.10: Q1: How important is it for you to get statistics from a match or training session?
Q2: How important is it for you to be able to go back and watch videos from a match or training session?

Figure 5.10 illustrates that the players think it is important to be able to find statistics and videos from a match or training session. Hamna IL has previously not had a software analysis tool to help with video analysis. Given how important it is to find statistics and videos from a match of training session, there exists a market for Dárkon. The next questions will reveal whether they are content or not with using Dárkon.

From figure 5.11, we can see that the players think it is relatively easy to find relevant video clips using the system after a match or training session. Along with this, we observe that the majority of the players will still continue to use Dárkon if still available in the future.

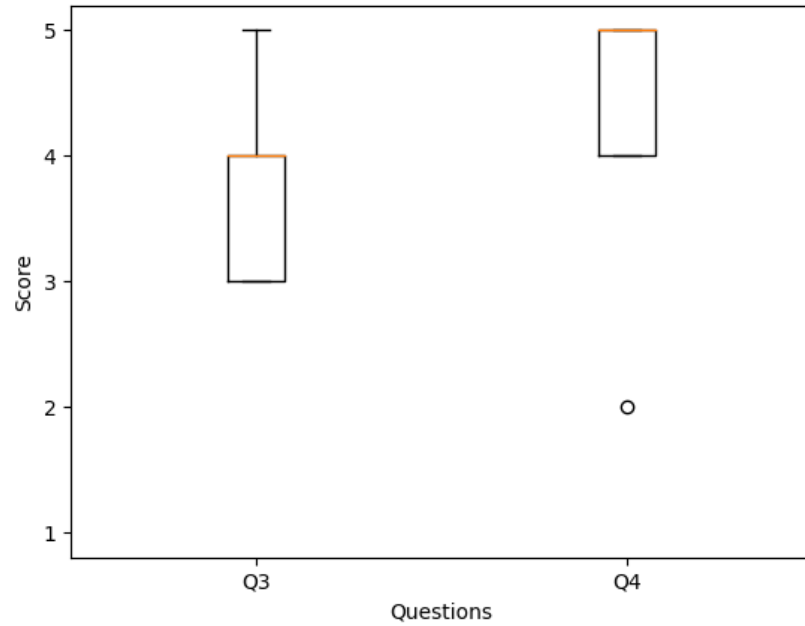


Figure 5.11: Q3: How easy is it to find relevant video clips using Dárkon?
Q4: How likely are you to continue using Dárkon if still available in the future?

The last question to the players was to find out which feature they think is the most important. This could help decide what to focus on in the future.

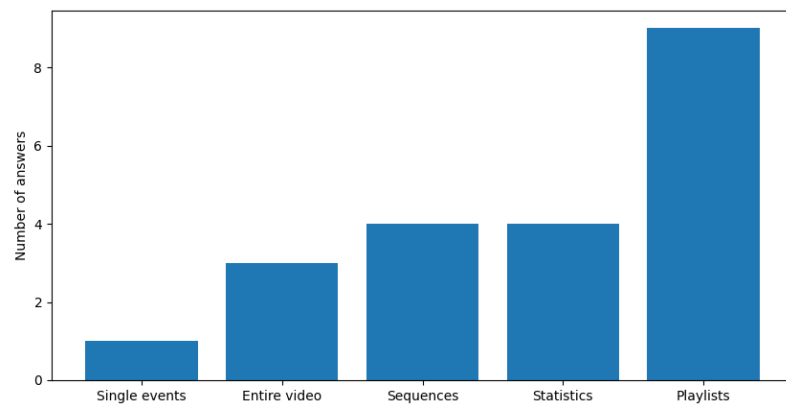


Figure 5.12: Q5: Which of these features are the most important for you? (Pick two)

Figure 5.12 shows that the players think the most important features are, being able to get statistics, or being able to see a playlists of all their involvements in a match.

5.6.2 Coach

Hamna IL currently has three coaches, where two of them are assistant coaches. Currently, it is only the head coach which has started using Dárkon. This means that the evaluation from the coaches will only include answers from the head coach.

The head coach of Hamna IL was sent six questions, where five of them are using the Likert Scale. The last question is the same as the one the players received, which could be used to find out which features are the most important to continue developing further.

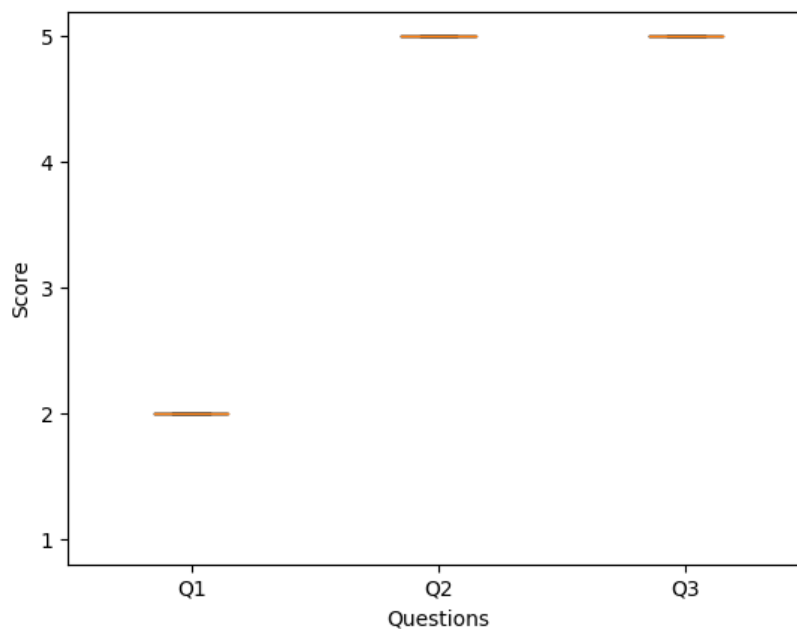


Figure 5.13: **Q1:** How happy are you with the current way of analysing a match after playing?
Q2: How happy are you with using Dárkon for analysis after a match?
Q3: How easy is it to use Dárkon for your needs?

Figure 5.13 shows the result from three questions. The first question asks how happy the coach is with the current way of doing analysis after a match, and

question two shows how happy he is conducting the analysis using Dárkon. These questions and answers implies that Dárkon is helping with doing analysis after a game, compared to how it was done previously when they did not have a system for helping them.

Question three also shows that the coach think it is easy to use Dárkon. This is related to the non-functional requirement about usability from section 3.2.

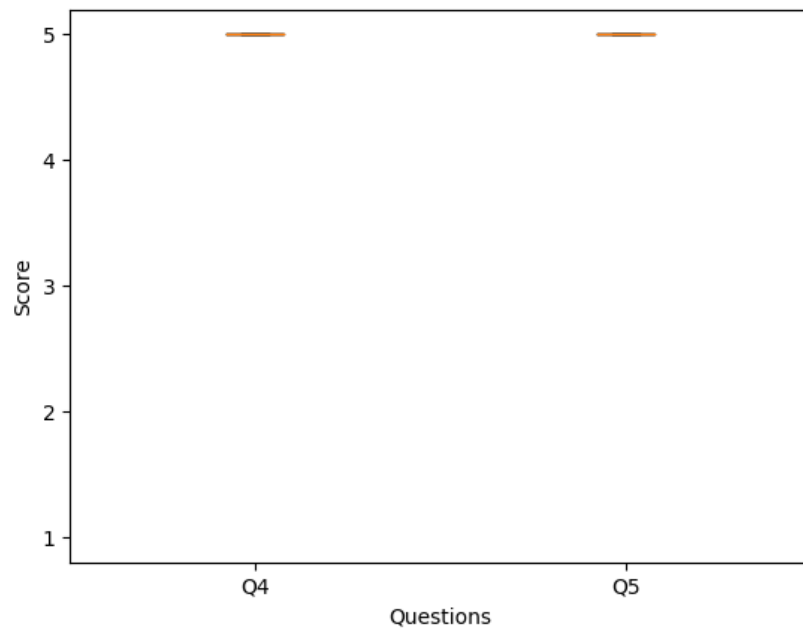


Figure 5.14: Q4: How likely are you to continue using Dárkon if still available in the future?

Q5: How helpful would it be to get video clips from the first half in the half-time break?

Figure 5.14 shows the result of two questions, where the first question shows that he is very likely to continue using the system if it is still available in the future.

Question five asks how helpful it would be to have video clips from the first half already available in the half-time break. The result of this question shows that this would be very helpful. This shows that having live streaming into the system where some initial analysis could be done by an assistant coach would be very beneficial in the future.

Figure 5.15 shows that the head coach think the most important features are

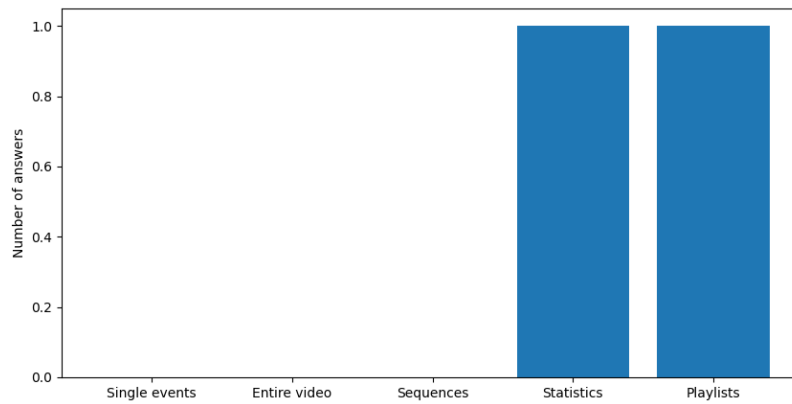


Figure 5.15: Q6: Which of these features are the most important for you? (Pick two)

playlists of the players, and statistics from a match. This demonstrates that the coach and the players agree on which features are the most important, meaning that these are features that could be improved even more in the future.

5.7 Summary

This chapter has provided some experiments of Dárkon while also conducting a user evaluation of how Hamna IL is liking the system. The user evaluation demonstrates that there is a space for Dárkon. The experiments reveals that the system is already working for different leagues and sports. It also demonstrates how important it is for the devices communicating together over the network to be physically close to each other. The next chapter will discuss some different choices made while developing the system in the context of the requirements outlined in chapter 3.

/6

Discussion

In the previous chapter, experiments about the systems were conducted. This chapter will provide discussions regarding the thesis in the context of the requirements from chapter 3, and the experiments from the previous chapter.

6.1 File System vs. MongoDB GridFS

One of Dárkon's main objectives is to be able to handle the upload of large video files. By default, MongoDB has a limit of 16 Megabytes (MB) per document and is therefore not a possible solution for storing videos.

However, MongoDB has a specification called GridFS¹ that supports storing documents exceeding 16MB. GridFS stores files in chunks of 255 Kilobytes (KB), and allows for querying for a file and letting the driver reassemble the chunks as needed for you. The advantage of this solution is that it is possible to access specific chunks of a file without having to operate on the file as a whole. MongoDB will also make the file and metadata in sync.

Another solution for storing the videos, is storing them on the file system instead of in a database. The advantage of this is that it does not clutter the

1. <https://www.mongodb.com/docs/manual/core/gridfs/>

cache of the database with chunks of files. It is also generally faster to store large files on the file system compared to inside a database [42].

Another advantage of not storing the videos inside GridFS, is the implication it will have if Dárkon in the future switch to a different database. In that case, it is necessary to migrate the files stored in GridFS to another storage system. Also, since Dárkon is currently deployed in the cloud, and the plan is to use a service to re-encode the uploaded videos, and serve them from a Content Delivery Network (CDN), it would require more effort to migrate away from GridFS than the local file system of the VM Dárkon is deployed on.

The main objectives of Dárkon were that it should be able to handle the upload and storage of large videos while also being able to scale well. Therefore, the file system was chosen over GridFS. While it means that we must keep the metadata stored in the database in sync with the file stored on the file system, we gain more flexibility in case of scaling and using a CDN in the future for streaming the videos and thumbnails. The database will then be much smaller, and could more easily be replicated, while we can still utilize a CDN to serve the multimedia content closer to the client requesting the data.

As Dárkon is currently deployed on Azure, the plan is to migrate towards using Azure Media Services². This is a service from Azure that can automatically re-encode videos into fixed length segments, and also provide multiple resolutions of the video. Azure Media Services is also using Azure Blob Storage³ under the hood [43]. This is Microsoft's object storage solution for the cloud [44].

Object storage is a concept that was introduced in the early 1990's [45]. The files in an object storage is saved in a flat data environment, and is usually the preferred storage option for static storage that is written once, and read potentially many times [46]. As these are characteristics that applies to Dárkon, object storage is a great option for storing the uploaded videos.

As the videos are currently stored on the file system on the deployed VM, Dárkon is not utilizing the advantages of an object storage. However, as discussed, the plan for Dárkon is to migrate from the file system to use Azure Media Services in the future.

2. <https://azure.microsoft.com/en-us/products/media-services>

3. <https://azure.microsoft.com/en-us/products/storage/blobs>

6.2 Parsing HLS Manifest

Since Dárkon encodes the video by copying the codec, it cannot guarantee fixed segment lengths for the video. This means that it is not possible to calculate which segments contain a specific time frame without parsing the entire manifest file.

However, section 5.2 demonstrated that re-encoding a video is a slow process. Therefore, the decision was that parsing the manifest file each time is an acceptable solution for a POC system. This makes it possible to work with any number of seconds per segment. While it will probably include more noise in the video generated, since it cannot guarantee small segments, Dárkon is still fully functional without re-encoding the video into fixed segment lengths. However, calculating the segments is a possible solution in the future if Dárkon can guarantee fixed segment lengths.

6.3 Thumbnail Generation

A thumbnail from the video is generated with FFmpeg when uploading a video to Dárkon. By doing this, we can show a frame from the video without loading the entire video. A thumbnail for a video should generally be descriptive of the video content. However, deciding which frame best suits the thumbnail is complex. The quality of a thumbnail is also subjective.

As stated earlier, a thumbnail is generated for each event when uploading the corresponding metadata for the video. Each tag has a timestamp for when in the video the event is. FFmpeg can then use this timestamp to generate a thumbnail for the specific event. The generated thumbnail is linked to each event, and is also used for playlists and sequences. By reusing the thumbnail from the first event for a sequence and playlist, we can visualize the video without increasing disk storage by creating a separate thumbnail for the sequence and playlist.

When generating a thumbnail for the entire video, Dárkon finds the first frame in the fifth second. This depends on the video being over five seconds, and should be the case for most videos, given that it will mainly be uploaded videos of an entire football match. However, if the video is shorter than five seconds, the first frame in the video is chosen as a fallback.

One optimization that could be done with generating a thumbnail, is to apply Machine Learning (ML) to generate an optimal thumbnail for the video [38, 47]. The user study from [38] showed that the automatic selection of thumbnails

outperformed the static generation in most cases. This proves that there is room for improvement in choosing the optimal thumbnail for a specific video since Dárkon currently uses static generation of thumbnails.

6.4 Deployment

There are different options for the deployment of Dárkon. Local deployment, where a server in the teams' office is running, is one option. Another option is to utilize the cloud, giving less maintenance and no upfront costs of investing in hardware. Another possibility is deploying some services in the cloud, and others locally. This section will discuss and compare the different options.

6.4.1 Azure

Deploying Dárkon in the cloud is one of the options, and this is how Dárkon is deployed today. Currently, the cloud provider is Azure, but other options like Google Cloud and Amazon AWS are also popular cloud providers [48].

The system is currently deployed on a VM with Ubuntu 20.04 LTS x86_64. The CPU is an Intel Xeon Platinum 8370C @ 2.793GHz with four cores. It also has 16GB of RAM. The videos uploaded are stored locally on the VM's disk, which has a Solid State Drive (SSD) of 256GB. The database used by the system is a separate Azure Cosmos DB for MongoDB service.

Deploying in the cloud gives many resources, and one of them is Azures Media Services. Azure Media Services can automatically re-encode the video into multiple resolutions, and then have it available for the client to play using a CDN. This way, we could free up the load of the server. As seen in section 5.2, re-encoding a video is a time-consuming process, meaning the server will be under heavy load during the entire re-encoding.

However, utilizing all the cloud capabilities early in the development, we risk vendor lock-in to the cloud provider, meaning that it could be challenging, and require significant work to switch to another cloud provider in the future. Another benefit to local encoding of the video is that we can fine-tune the parameters for the encoding job, and have a video up and running quickly by copying the codec of the video. However, this could lead to problems, as we need control over which codec the video has, and may therefore be unable to play the video without manually re-encoding it before uploading. Another potential problem is if a video is uploaded with 4K resolution. When the video is encoded to HLS, only the original resolution is kept. This means that Dárkon

does not currently use all the advantages of HLS, where the client is able to ask for a lower resolution. If the client does not have the network speed to handle a 4K stream, they will experience buffering of the video.

While there are several advantages to deploying in the cloud, there are also downsides. One is that it could be expensive. Having deployed Dárkon to Azure in the latest months, the monthly cost is about \$315 using most of the default configurations when setting up the database and VM.

6.4.2 Local

Another deployment option could be to deploy on a local server located in the club's offices. Compared to a cloud deployment, this would require more maintenance work, and could cause issues in the case of disk corruption. While cloud providers often offer a backup solution for the data periodically, this ease of use is not there with local deployment.

While deploying Dárkon in the cloud gives many advantages regarding maintenance and automatic backup, it could be more expensive. Dárkon allows deploying the system locally on a PC, meaning a team would only need to pay for the hardware they want to run on. This deployment option could be much cheaper if they opt for a PC with reasonably priced hardware.

Deploying Dárkon locally means the server could be under heavy load when uploading videos. This is also the case for how Dárkon is deployed today. However, this could be fixed by utilizing some cloud services like Azure Media Services, while running everything else locally.

6.4.3 Combination

A third option could be deploying some services locally, and others in the cloud. As discussed, there could be drawbacks both when deployed locally, and in the cloud. Cloud deployment could quickly get very expensive when using multiple services, and deploying Dárkon locally makes it harder to scale, especially when uploading videos.

This is where one could use some services from the cloud, like Azure Media Services, to re-encode videos, and provide access to them through a CDN, while the rest is running on a local server. It could also be possible to setup a database service to have all the data backed up, while only running the server locally.

6.4.4 Summary of Deployment

As discussed, multiple deployment options of Dárkon give the teams flexibility. If they want something cheap that works, they could deploy locally, and if less maintenance for the deployment is wanted, they could utilize cloud services. They could also opt for a combination of the two, deploying some services in the cloud, and others locally.

6.5 Hudl Integration

While providing an optional integration with Eliteserien Highlights, a solution to provide integration with Hudl was investigated. A prototype implementation with integration for Hudl was developed, and working. However, it was later removed from Dárkon as the integration could not be automatic due to no open API's from Hudl. Therefore, integration with Hudl is not feasible at the moment, but is possible in the future if they open their API for it.

6.6 Scalable

As of now, Dárkon is hosted on a VM, and deployed through Azure for Hamna IL to use. As the number of users starts to scale, and the storage size grows, the videos and re-encoding job could be improved by utilizing Azure Media Services. This would allow for scaling the storage, and leaving the server free to handle other requests as the videos would be uploaded to another service for re-encoding and storing.

6.7 Usability

To make Dárkon easy to use for both the coaches and the players, an interface for PC and mobile was developed. As we live in a world where everyone potentially has access to a phone 24/7, the system should still be accessible and usable without a PC.

Figure 6.1 shows some pages from Dárkon in a mobile view. These pages are respectively the same as figure 4.18 and 4.19. Since the phone does not have as much screen real estate, a different view had to be composed to make it fit to the size of a phone. This should make Dárkon more usable, as one is not dependent on having a PC to use the system.

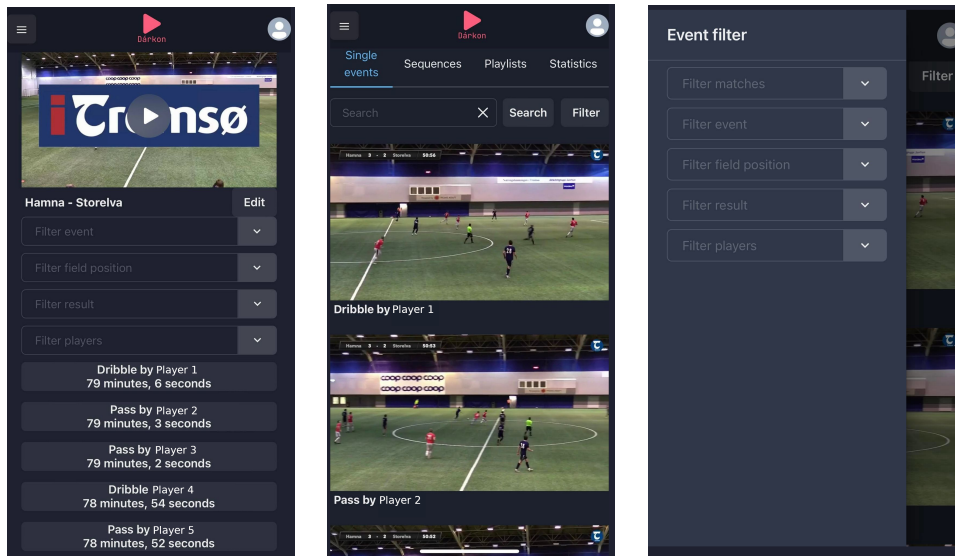


Figure 6.1: Mobile view of Dárkon.

While being actively worked on, Hamna IL has consistently been giving feedback on how to make the system more usable regarding the presentation of clips. The evaluation from section 5.6 also illustrated that both the coach and players find it easy to use Dárkon.

6.8 Availability in Dárkon

Availability is a property that states how often the system is available [49, p. 509]. It is also one of the non-functional properties of Dárkon.

Since Dárkon is hosted on Azure, the availability of the system depends on the availability of the resources from Azure. By looking at Azures Service-Level Agreement (SLA) [37] from April 2023, we can observe that they claim 99.99% availability for the Azure Cosmos DB for MongoDB, and 99.95% for a VM. Multiplying these percentages gives an SLA with a total of 99.94% availability for Dárkon.

Along with availability in the sense that the system is operable, it should also try to have analysis of a game quickly available after the game is finished. This is dependent on the video footage of the game being uploaded quickly, along with how fast the coaches are with annotating the game in hindsight.

Along with this, given that the video has a codec that is supported by Dárkon,

it should be faster to upload the video from a match compared to if we would have re-encoded the entire video. This was demonstrated in section 5.2.

By utilizing the tagging system that is developed alongside Dárkon, it started out by being very time-consuming for the coach to annotate a match. However, after weekly meetings and discussions on how to improve this, the amount of time to annotate a single match has been significantly reduced. Applying ML along with technology to capture the voice for annotations has quickly speed up this process. This leads to the video with annotation being faster available to Dárkon, and the analysis being available in a shorter amount of time.

The discussions and development on how to tag a match is constantly improving. Currently, the time-consuming factor is having the players' coordinates on every single event. The next step in attempting to improve the tagging speed will be to experiment with equipping the players with a Global Positioning System (GPS) tracker, or avoid tagging the coordinates in the first iteration. Avoiding tagging the coordinates would allow for the analysis to be faster available after a match, leaving out some details. However, tagging the coordinates in hindsight after the match would still be possible, automatically improving the analysis once re-uploaded to Dárkon.

6.9 Security

6.9.1 Authentication

The system will collect information about team members and the teams' videos. This information should be kept private. Therefore, some care when designing the system concerning security was necessary. Today, Dárkon uses authentication with email and password. To make the password secure, Argon2 was chosen for the hashing algorithm. This is a memory-hard function designed to be used for passwords [50]. It also won the Password Hashing Competition in 2015, an open competition run by a group of cryptographers to agree on a recommendation for a standard hashing algorithm [51] for passwords.

6.9.2 Access Control

Access Control is the process of ensuring that the users are whom they claim to be, and to only give access to those who need it [52, p. 19]. One part of access control in Dárkon is user authentication. The second part is using the user-provided information on whether a user is a player or a coach. This information is provided when signing up. The players and coaches differ in what they are

allowed to do. For instance, only a coach can create a team, invite users to join, and remove users from the team's profile in Dárkon.

6.10 Privacy

In section 2.2 we observed that existing systems today are secretive in how they are storing and processing customer data. Asking three of the existing commercial system providers on how they process the data, only one of them could provide an answer. This is a concern regarding privacy, as they are not transparent in the way they are handling the data.

Dárkon on the other hand aims to be fully transparent in the way it handles data, and also be compliant with GDPR⁴. As of now, the videos Dárkon uses are behind a subscription to a local newspaper in Tromsø. The head coach of Hamna IL, Martin, has a subscription where he gets the videos, and use them for tagging. Then he uploads the videos along with the metadata to Dárkon. Before Dárkon is deployed in a large scale, we need to examine the legal aspect of this.

GDPR is a privacy and security regulation that all systems that process data from EU residents must comply with. Since Dárkon is processing data, it should be in regulation with GDPR for the seven data protection principles. The following will explain the seven principles and how Dárkon is adhering to them.

Lawfulness, fairness, and transparency

The data will only be used to make the analysis easier for a club on a day-to-day basis, providing filtering options, and easy-to-access clips of a match or training session.

Purpose limitation

This regulation states that the system must process the data for legitimate, explicitly specified purposes. Dárkon will only process data to help make analysis easier for the team.

Data minimization

Data minimization states that the system should only collect as much data as

4. <https://gdpr.eu>

necessary for specified purposes. Dárkon does not collect information that is not needed for either analysis or visualization on the frontend.

Accuracy

This states that personal data must be kept up to date. Dárkon allows the users to change the user information they provided when creating a user. This includes the email, name, and password.

Storage limitation

The system may only store personal identifying data for as long as necessary for the purpose specified. Dárkon will only store the data for as long as the user still has an account.

Integrity and confidentiality

This regulation says that the processing must ensure security, integrity, and confidentiality by, for example, using encryption. How Dárkon deals with this is already discussed in section 6.9.

Accountability

Accountability states that the data controller must always be able to demonstrate compliance with GDPR for all of these principles. This means that Dárkon should be able to prove compliance with the principles at all times.

6.11 Comparing Dárkon to Existing Systems

In chapter 2, some existing systems were introduced. Comparing these to Dárkon, there are some differences as well as some similarities.

The similarities are in that all of these systems make it easier for teams to do analysis based on data, and help the teams improve. Some existing systems are sports-agnostic, and some are only developed with football in mind. Footovision is specific for football, while Hudl, ProZone, and Nacsport are sports-agnostic. While Dárkon is mainly developed with football in mind, section 5.4 demonstrated that the system is already somewhat agnostic for ball sports with similar events like football. The system will also continue to be developed to be even more sports-agnostic.

Regarding data collection, Nacsport did not respond to inquiries about how

and where they stored their data. Hudl responded, but could not provide the information. However, Footovision could provide the information. Dárkon is fully transparent in storing the data, as discussed in section 6.4.

For getting started with video analysis, Dárkon offers flexibility regarding how a team wants to use it. Compared to the other systems, Dárkon also offers a way for the team to have total control of their data with the system deployed on a server that they have complete control over, making them only pay for the hardware running the system.

When analyzing the videos, most existing systems offer tagging of the matches for the club, leaving the club with no control of the tagging. The enterprise companies will usually outsource the tagging to other countries. The ones tagging the matches are usually doing it for several matches a week, and are watching hundreds to thousands hours of football every single season [1, p. 3]. Dárkon is being developed simultaneously as a tagging system. This will make the system independent of the video, meaning the videos could come from any camera. It will also make it possible to use for training sessions, as they could film the training sessions and do the tagging after. How long it takes before the analysis is ready after a match or training session depends on the time it takes to annotate the video. Dárkon is able to show videos and statistics as soon as the video and metadata are uploaded to the system.

6.12 Summary

This chapter discusses some decisions made during the development of Dárkon in the context of the requirements described in chapter 3, and the experiments from chapter 5. Finally, it compares Dárkon to some of the other existing commercial systems previously introduced in chapter 2. The next chapter will conclude the thesis, and also align some planned future work.



Concluding Remarks

This thesis presents Dárkon, a POC video analysis system that works in conjunction with other tagging systems to help teams with their video analysis. While Dárkon is developed as a POC system, it is already being used by Hamna IL which has helped with collaboration and feedback in the process. Hamna IL will also continue to use the system in the future, as the work of Dárkon will continue.

To recap the starting point of the thesis, the problem definition was outlined as:

This thesis will design, implement, and evaluate a system prototype for managing annotated clips of a match for visualization. It will thus investigate scalable and privacy-preserving backend services that support this, as well as frontend services providing ease of use for coaches and athletes. A particular focus will be on giving athletes and coaches flexibility in finding specific match clips to analyze further.

To evaluate the thesis, Dárkon was created. Dárkon is a software analytics toolkit that aims to help teams getting started with video analysis.

Chapter 3 started out by specifying the requirements of the system, both functional and non-functional. The non-functional requirements states requirements that are not directly concerned with the services delivered to the users

of the system [30, p .87]. The functional requirements states what the system should be able to do [30, p .85].

Chapter 4 presents the design, implementation, and architecture of Dárkon in the context of the requirements from chapter 3.

In Chapter 5 we present experiments for the system. As demonstrated in the experiment from section 5.2, re-encoding a video is a time-consuming process that could potentially harm the scalability of the system. As the system is currently deployed on a single VM in the cloud, the server will be under heavy load during the entire re-encoding.

Chapter 6 presents a discussion regarding choices made while developing Dárkon. Why Dárkon currently copies the codec instead of re-encoding is discussed with regards to the result of the experiment from section 5.2. It also discusses the choice of storing the videos locally on the file system rather than using MongoDB GridFS.

As presented in this thesis, Dárkon is already in use by Hamna IL, which is a local club in Tromsø, playing in the fourth division in Norway. They have already used the system for several months, while giving feedback to help Dárkon develop and fit to their needs. The evaluation from section 5.6 shows that Dárkon is getting positive feedback from the head coach and the players.

7.1 Thesis Conclusion

Based on the work presented in this thesis, we can conclude with the following:

Dárkon is a system prototype that is able to manage annotated clips of a match for visualization. We have investigated privacy-preserving and scalable backend services that supports this.

Dárkon is currently deployed on a single VM and is processing the videos locally, and storing them on the file system. Thus, it is not as scalable as it could be. However, as discussed in section 6.4, there are plans to migrate over to a cloud service for the video storage and encoding to make it more scalable.

We have also investigated frontend services for ease of use for the coaches and athletes. Dárkon provides an interface that is available for both PC and mobile phones. Section 5.6 presented evaluation from Hamna IL, where both the coach and players found it easy to use.

7.2 Future Work

Since Dárkon is developed as a POC system over a limited time, there are room for improvements. There are already plans to continue developing the system, making it more robust, and turn it into a system that also could be valuable for other teams than Hamna IL. This section will introduce some of the planned future work for Dárkon.

7.2.1 Integration with Tagging System

As the tagging system is developed concurrently with Dárkon, the integration between the two systems is not satisfactory at the moment. This will improve in the future, when the systems have defined API's to communicate over.

7.2.2 Live Streaming

Currently, there is only a solution to upload the video to Dárkon, and no possibility for live streaming a feed over a protocol like Real-Time Messaging Protocol (RTMP). This is something that should be worked on for a future version of Dárkon, to get closer to real-time analysis. This will help with making decisions during a match, like doing a substitution during the half-time break. As figure 5.14 illustrated, the head coach of Hamna IL would find it very helpful to have video clips available in the half-time break. Live streaming the video to Dárkon would make this possible.

7.2.3 Statistics

As described in chapter 2, collecting videos for analysis could be valuable for a club, and most elite clubs today have huge departments dedicated for this. However, statistics for the game could also help with looking at things from a different perspective than videos. By using the same tags for collecting videos, a large amount of statistics from the games could be presented in a player-and team-dashboard. While some of this is already implemented with a simple page for showing statistics from a single match, it can be further improved with graphs and heat-maps of a match, and also show passing networks between the players. As described in section 2.1, analyzing passing networks could help a team develop a unique style of play. This is something that has already been wished for in a future version of Dárkon, and is something that should be implemented in the future.

7.2.4 Overlay

To further help with video analysis, it could be helpful to be able to draw on top of the videos to visualize the context. This is something that should be investigated in the future. It would help the coaches to visualize their ideas to the players, making it easier for them to understand how the coach want them to position themselves, play the ball, and where to run into spaces to mark a threat for their opponent.

7.2.5 Legal Aspect of Video Sources

Dárkon is currently using the videos from a local newspaper which is behind a subscription that the head coach of Hamna IL is paying for. Before Dárkon is deployed in a larger scale, we need to explore the legal aspect of using their videos in Dárkon. While we are currently using the broadcast streams from the newspaper, we are also currently investigating using our own cameras.

7.2.6 Machine Learning

Currently, the tags are generated through a third-party system where a lot of the tagging is done manually. This will be integrated into Dárkon in the future. Along with this, manual tagging is time-consuming, and this could be improved with the help of ML to automatically generate some tags. There has already been put in some preliminary work into this, and it will continue to be worked on in the future to free up time for the coaches.

7.2.7 Combine and Export Videos

It is currently not possible to export and download videos from Dárkon. This along with combining multiple events as you like into a custom video is something that is highly requested from Hamna IL. This would allow teams to easily make a presentation to either prepare for the next match or analyze the previous match.

Bibliography

- [1] Rory Smith. *Expected goals the story of how data conquered football and changed the Game Forever*. Mudlark, 2022.
- [2] D. E. Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner, Paul R. Young, and Peter J. Denning. “Computing as a Discipline.” In: *Commun. ACM* 32.1 (Jan. 1989), pp. 9–23. ISSN: 0001-0782. DOI: 10.1145/63238.63239. URL: <https://doi.org/10.1145/63238.63239>.
- [3] Robbert van Renesse, Aage Kvalnes, Dag Johansen, and Audun Arnesen. *Vortex. An event-driven multiprocessor operating system supporting performance isolation*. eng. Universitetet i Tromsø, 2003.
- [4] A. Nordal, A. Kvalnes, J. Hurley, and D. Johansen. “Balava: Federating Private and Public Clouds.” eng. In: *2011 IEEE World Congress on Services*. IEEE, 2011, pp. 569–577. ISBN: 1457708795.
- [5] Dag Johansen, Kåre J. Lauvset, Robbert van Renesse, Fred B. Schneider, Nils P. Sudmann, and Kjetil Jacobsen. “A TACOMA Retrospective.” In: *Softw. Pract. Exper.* 32.6 (May 2002), pp. 605–619. ISSN: 0038-0644. DOI: 10.1002/spe.451. URL: <https://doi.org/10.1002/spe.451>.
- [6] Dag Johansen, Magnus Stenhaus, Roger B. A. Hansen, Agnar Christensen, and Per-Mathias Høgmo. “Muithu: Smaller footprint, potentially larger imprint.” In: *Seventh International Conference on Digital Information Management (ICDIM 2012)*. 2012, pp. 205–214. DOI: 10.1109/ICDIM.2012.6360105.
- [7] Magnus Stenhaus, Yang Yang, Cathal Gurrin, and Dag Johansen. “Muithu: A Touch-Based Annotation Interface for Activity Logging in the Norwegian Premier League.” In: *Proceedings of the 20th Anniversary International Conference on MultiMedia Modeling - Volume 8326*. MMM 2014. Dublin, Ireland: Springer-Verlag, 2014, pp. 365–368. ISBN: 9783319041162. DOI: 10.1007/978-3-319-04117-9_37. URL: https://doi.org/10.1007/978-3-319-04117-9_37.
- [8] Pål Halvorsen, Simen Sægrov, Asgeir Mortensen, David Kristensen, Alexander Eichhorn, Magnus Stenhaus, Stian Dahl, Håkon Stensland, Vamsidhar Gaddam, Carsten Griwodz, and Dag Johansen. “Bagadus: An integrated system for arena sports analytics - A soccer case study.” In: Feb. 2013, pp. 48–59. DOI: 10.1145/2483977.2483982.

- [9] Håkon Stensland, Vamsidhar Gaddam, Marius Tennøe, Espen Helgedagsrud, Mikkel Næss, Henrik Alstad, Asgeir Mortensen, Ragnar Langseth, Sigurd Ljødal, Østein Landsverk, Carsten Griwodz, Pål Halvorsen, Magnus Stenhaus, and Dag Johansen. “Bagadus: An integrated real-time system for soccer analytics.” In: *ACM Transactions on Multimedia Computing, Communications and Applications* 10 (Jan. 2014), 14:1–14:21. DOI: 10.1145/2541011.
- [10] Dag Johansen, Håvard Johansen, Tjalve Aarflot, Joseph Hurley, Åge Kvalnes, Cathal Gurrin, Sorin Zav, Bjørn Olstad, Erik Aaberg, Tore Endestad, Haakon Riiser, Carsten Griwidz, and Pål Halvorsen. “DAVVI: A Prototype for the next Generation Multimedia Entertainment Platform.” In: *Proceedings of the 17th ACM International Conference on Multimedia*. MM ’09. Beijing, China: Association for Computing Machinery, 2009, pp. 989–990. ISBN: 9781605586083. DOI: 10.1145/1631272.1631482. URL: <https://doi.org/10.1145/1631272.1631482>.
- [11] Dag Johansen, Pål Halvorsen, Håvard Johansen, Håkon Riiser, Cathal Gurrin, Bjørn Olstad, Carsten Griwodz, Aage Kvalnes, Joseph Hurley, and Tomas Kupka. “Search-based composition, streaming and playback of video archive content.” In: *Multimedia Tools and Applications - MTA* 61 (Nov. 2012), pp. 1–27. DOI: 10.1007/s11042-011-0847-5.
- [12] Maximilian T. Fischer, Daniel A. Keim, and Manuel Stein. “Video-Based Analysis of Soccer Matches.” In: *Proceedings Proceedings of the 2nd International Workshop on Multimedia Content Analysis in Sports*. MMSports ’19. Nice, France: Association for Computing Machinery, 2019, pp. 1–9. ISBN: 9781450369114. DOI: 10.1145/3347318.3355515. URL: <https://doi.org/10.1145/3347318.3355515>.
- [13] László Gyarmati, Haewoon Kwak, and Pablo Rodriguez. “Searching for a Unique Style in Soccer.” In: *ArXiv abs/1409.0308* (2014).
- [14] Sam Williams. *Behind the Badge: The physicist who leads Liverpool’s data department*. June 2020. URL: <https://www.liverpoolfc.com/news/behind-the-badge/398645-ian-graham-liverpool-fc-behind-the-badge>.
- [15] Bruce Schoenfeld. *How data (and some breathtaking soccer) brought Liverpool to the cusp of glory*. May 2019. URL: <https://www.nytimes.com/2019/05/22/magazine/soccer-data-liverpool.html>.
- [16] *Post match summary reports*. URL: <https://www.fifatrainingcentre.com/en/fwc2022/post-match-summaries/post-match-summary-reports.php>.
- [17] Greg Mathieson. *Opposition analysis at Liverpool FC: Part 1*. Nov. 2020. URL: <https://www.nacsport.com/blog/en-gb/Users/opposition-analysis-at-liverpool-fc-part-1>.
- [18] Duncan Ritchie. *Klipdraw at Liverpool FC*. Mar. 2021. URL: <https://www.klipdraw.com/blog/en/Users/klipdraw-liverpool-fc>.

- [19] Phil Hay. *How prozone sparked a football analytics boom*. Nov. 2020. URL: <https://theathletic.com/2193722/2020/11/16/prozone-analytics-ramm-mylvaganam-analysis-premier-league/>.
- [20] Di Salvo Valter, Collins Adam, McNeill Barry, and Cardinale Marco. "Validation of Prozone ®: A new video-based performance analysis system." In: *International Journal of Performance Analysis in Sport* 6.1 (2006), pp. 108–119. DOI: 10.1080/24748668.2006.11868359. eprint: <https://doi.org/10.1080/24748668.2006.11868359>. URL: <https://doi.org/10.1080/24748668.2006.11868359>.
- [21] Paul Bradley, Peter O'Donoghue, Blake Wooster, and Phil Tordoff. "The reliability of ProZone MatchViewer: A video-based technical performance analysis system." In: *International Journal of Performance Analysis in Sport* 7 (Sept. 2007), pp. 117–129. DOI: 10.1080/24748668.2007.11868415.
- [22] Rabi Musa. "THE ADVANCEMENTS AND USE OF TECHNOLOGICAL STRATEGIES IN PERFORMANCE ANALYSIS OF SOCCER: AN UPDATE." In: *Journal of Physical Education Research* Volume 3 (June 2016), pp. 34–47.
- [23] Philipp Seidenschwarz, Martin Rumo, Lukas Probst, and Heiko Schuldt. "High-Level Tactical Performance Analysis with SportSense." In: *Proceedings of the 3rd International Workshop on Multimedia Content Analysis in Sports*. MMSports '20. Seattle, WA, USA: Association for Computing Machinery, 2020, pp. 45–52. ISBN: 9781450381499. DOI: 10.1145/3422844.3423053. URL: <https://doi.org/10.1145/3422844.3423053>.
- [24] *Hudl pricing*. URL: <https://www.hudl.com/pricing>.
- [25] *Upload sorter to HUDL • Hudl Sportscode support*. [Accessed April 17, 2023]. URL: <https://www.hudl.com/support/hudl-sportscode/hudl-sportscode-integration/upload-video-data-to-hudl/upload-sorter-to-hudl>.
- [26] *Sports video analysis software*. [Accessed April 17, 2023]. URL: <https://www.nacsport.com/index.php?lc=en-gb>.
- [27] *Leeds United and footovision: A blossoming partnership*. [Accessed April 17, 2023]. URL: <https://www.footovision.com/leeds-united-and-footovision-a-blossoming-partnership>.
- [28] Hans W. Barz. *Multimedia networks : protocols, design, and applications*. eng. Hoboken, 2016.
- [29] Abdelhak Bentaleb, Bayan Taani, Ali C. Begen, Christian Timmerer, and Roger Zimmermann. "A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP." In: 21.1 (2019), pp. 562–585. DOI: 10.1109/COMST.2018.2862938.
- [30] Ian Sommerville. *Software engineering*. Pearson, 2011.
- [31] Armin Lawi, Benny L. E. Panggabean, and Takaichi Yoshida. "Evaluating GraphQL and REST API Services Performance in a Massive and Intensive Accessible Information System." In: *Computers* 10.11 (Oct. 2021), p. 138.

- ISSN: 2073-431X. DOI: 10.3390/computers10110138. URL: <http://dx.doi.org/10.3390/computers10110138>.
- [32] William Bugden and Ayman Alahmar. *Rust: The Programming Language for Safety and Performance*. 2022. arXiv: 2206.05503 [cs.PL].
- [33] *Stack overflow developer survey 2022*. URL: <https://survey.stackoverflow.co/2022/#section-most-loved-dreaded-and-wanted-programming-scripting-and-markup-languages>.
- [34] Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Voshall, and Werner Vogels. “Dynamo: Amazon’s Highly Available Key-Value Store.” In: *Proceedings of Twenty-First ACM SIGOPS Symposium on Operating Systems Principles*. SOSP ’07. Stevenson, Washington, USA: Association for Computing Machinery, 2007, pp. 205–220. ISBN: 9781595935915. DOI: 10.1145/1294261.1294281. URL: <https://doi.org/10.1145/1294261.1294281>.
- [35] Avinash Lakshman and Prashant Malik. “Cassandra: A Decentralized Structured Storage System.” In: *SIGOPS Oper. Syst. Rev.* 44.2 (Apr. 2010), pp. 35–40. ISSN: 0163-5980. DOI: 10.1145/1773912.1773922. URL: <https://doi.org/10.1145/1773912.1773922>.
- [36] Josiah L. Carlson. *Redis in action*. Manning, 2013.
- [37] Microsoft. URL: <https://www.microsoft.com/licensing/docs/view/Service-Level-Agreements-SLA-for-Online-Services?lang=1>.
- [38] Andreas Husa, Cise Midoglu, Malek Hammou, Steven A. Hicks, Dag Johansen, Tomas Kupka, Michael A. Riegler, and Pål Halvorsen. “Automatic Thumbnail Selection for Soccer Videos Using Machine Learning.” In: *Proceedings of the 13th ACM Multimedia Systems Conference*. MM-Sys ’22. Athlone, Ireland: Association for Computing Machinery, 2022, pp. 73–85. ISBN: 9781450392839. DOI: 10.1145/3524273.3528182. URL: <https://doi.org/10.1145/3524273.3528182>.
- [39] *Full match | Liverpool 3-1 manchester city | FA community Shield 2022-23*. Oct. 2022. URL: <https://www.youtube.com/watch?v=cTqY53zWypk>.
- [40] *Netherlands vs Norway Handball Women’s World Championship Spain 2021*. Dec. 2021. URL: https://www.youtube.com/watch?v=Us-N0g_M558.
- [41] Rensis Likert. “A Technique for the Measurement of Attitudes.” In: *Archives of Psychology* 140 (1932), pp. 1–55.
- [42] Jim Gray. *To BLOB or Not To BLOB: Large Object Storage in a Database or a Filesystem*. Tech. rep. MSR-TR-2006-45. Apr. 2006, p. 10. URL: <https://www.microsoft.com/en-us/research/publication/to-blob-or-not-to-blob-large-object-storage-in-a-database-or-a-filesystem/>.
- [43] *Video-on-demand digital media - azure solution ideas*. URL: <https://learn.microsoft.com/en-us/azure/architecture/solution-ideas/articles/digital-media-video>.
- [44] *Introduction to Azure Blob Storage*. URL: <https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction>.

- [45] M. Factor, K. Meth, D. Naor, O. Rodeh, and J. Satran. “Object storage: the future building block for storage systems.” In: *2005 IEEE International Symposium on Mass Storage Systems and Technology*. 2005, pp. 119–123. DOI: 10.1109/LGDI.2005.1612479.
- [46] URL: <https://cloud.google.com/learn/what-is-object-storage>.
- [47] Andreas Husa, Cise Midoglu, Malek Hammou, Pål Halvorsen, and Michael A. Riegler. “HOST-ATS: Automatic Thumbnail Selection with Dashboard-Controlled ML Pipeline and Dynamic User Survey.” In: *Proceedings of the 13th ACM Multimedia Systems Conference*. MMSys ’22. Athlone, Ireland: Association for Computing Machinery, 2022, pp. 334–340. ISBN: 9781450392839. DOI: 10.1145/3524273.3532908. URL: <https://doi.org/10.1145/3524273.3532908>.
- [48] Manish Saraswat and R.C. Tripathi. “Cloud Computing: Comparison and Analysis of Cloud Service Providers-AWs, Microsoft and Google.” In: *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*. 2020, pp. 281–285. DOI: 10.1109/SMART50582.2020.9337100.
- [49] Daniel P Siewiorek and Robert S Swarz. *Reliable computer systems*. 3rd ed. Natick, MA: A K Peters, Oct. 1998.
- [50] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. “Argon2: New Generation of Memory-Hard Functions for Password Hashing and Other Applications.” eng. In: *2016 IEEE European Symposium on Security and Privacy (EuroS and P)*. IEEE, 2016, pp. 292–302. ISBN: 1509017518.
- [51] Jos Wetzels. *Open Sesame: The Password Hashing Competition and Argon2*. Cryptology ePrint Archive, Paper 2016/104. <https://eprint.iacr.org/2016/104>. 2016. URL: <https://eprint.iacr.org/2016/104>.
- [52] M. T. Goodrich and Roberto Tamassia. *Introduction to Computer Security: Pearson New International Edition*. Vol. New international edition, first edition. Pearson, 2014. ISBN: 9781292025407. URL: <https://search-ebshost-com.mime.uit.no/login.aspx?direct=true&db=nlebk&AN=1418217&site=ehost-live>.

