Design Report: SPINcms

Making digital signage management usable for non-technical marketing staff

Team 6
Teodor Constantinescu (s2977702),
Tom Heijkamp (s3190536),
Bram Rutgers (s3180794),
Jakob Metin (s3206149),
Serkan Sirin (s3210677)

Supervisor University of Twente: dr. T. Van Dijk MSc
Supervisor Extendas: Arjan Vissers

November 6, 2026

Contents

Glossary **7**

1 Introduction **8**

2 Context and Analysis **9**
2.1 The Client: Extendas . . . 9
2.2 Domain . . . 9
2.3 The Current SPINmedia System . . . 10
2.3.1 Content Model . . . 10
2.3.2 Scheduling . . . 11
2.3.3 User Interface . . . 12
2.4 Stakeholders and Users . . . 12
2.4.1 Personas . . . 13
2.5 Baseline User Research . . . 14
2.6 Core Problems . . . 14

3 Requirements Specification **15**
3.1 Requirement Gathering . . . 15
3.2 Functional Requirements . . . 16
3.3 Non-Functional Requirements . . . 17
3.4 User Stories . . . 17

4 Project Approach **19**
4.1 Method . . . 19
4.2 Team Organisation and Communication . . . 19
4.3 Planning versus Reality . . . 20
4.4 Risks . . . 21

5 Global Design **22**
5.1 Design Principles . . . 22
5.1.1 Simplicity and clarity . . . 22
5.1.2 Consistency . . . 22
5.1.3 Visibility of system status . . . 23
5.1.4 Visual hierarchy and accessibility . . . 23
5.1.5 Error prevention and user control . . . 23
5.2 Conceptual Model . . . 23
5.2.1 Merging of tracks and playlists . . . 24
5.2.2 Separating content management and scheduling . . . 24
5.2.3 Resulting conceptual model . . . 25
5.3 Information Architecture and Navigation . . . 25
5.4 System Overview . . . 26

6 Detailed Design and Implementation **27**
6.1 Scheduling . . . 27
6.2 Tracks . . . 28
6.3 Media Library . . . 29
6.4 Locations and Screens . . . 30
6.5 Dashboard . . . 30
6.6 Visual Design System . . . 31
6.7 Implementation . . . 31

7 Framework and Architecture Recommendation **33**
7.1 Current Situation and Constraints . . . 33
7.2 Evaluation Criteria . . . 33
7.3 Options . . . 34
7.3.1 Move to Blazor's Auto/WebAssembly render modes . . . 34
7.3.2 ASP.NET Core API with a JavaScript single-page application (React, Vue or Angular) . . . 34
7.3.3 Adopt an existing digital signage CMS (e.g. Xibo, Yodeck, ScreenCloud) . . . 34
7.4 Recommendation . . . 35

8 Testing **36**
8.1 Test Plan . . . 36
8.2 Results . . . 37

9 Evaluation, Consequences and Follow-up **39**
9.1 Evaluation of the Product . . . 39
9.2 Evaluation of the Design Process and Collaboration . . . 39
9.3 Consequences for Extendas . . . 39
9.4 Future Work . . . 40

10 Conclusion **41**

11 Individual Contributions **42**
11.1 Tom . . . 42
11.2 Teodor . . . 42
11.3 Serkan . . . 42
11.4 Jakob . . . 42
11.5 Bram . . . 42

12 Use of AI **43**

References **44**

A Appendix **46**

B Source Code **46**

C Raw Research and Test Results **46**

D Design Evaluation **46**

E Project Proposal **57**
E.1 Project, Organization and Requirements . . . 57
E.2 Planning . . . 57
E.2.1 Sprint 1 (14-09 till 25-09) - Design . . . 58
E.2.2 Sprint 2 (28-09 till 09-10) - Development . . . 58
E.2.3 Sprint 3 (12-10 till 23-10) - Development . . . 58
E.2.4 Sprint 4 (26-10 till 06-11) - Polish and Deliver . . . 58
E.3 Deliverables . . . 58
E.3.1 Design Report . . . 59
E.3.2 Presentation Slides . . . 59
E.3.3 Poster . . . 59
E.4 Risk Analysis . . . 59
E.5 AI-use policy . . . 59

F Figures **60**

G Interviews from round 1 **60**
G.1 Interview Plan Round 1 . . . 60
G.1.1 Sample group . . . 60
G.1.2 Questions before tasks . . . 60
G.1.3 Questions after tasks . . . 60
G.1.4 Tasks . . . 61
G.2 Interview 1 . . . 62
G.2.1 Demographic questions . . . 62
G.3 Tasks . . . 62
G.4 After tasks questions . . . 62
G.5 Interview 2 . . . 62
G.5.1 Demographic questions . . . 62
G.6 Tasks . . . 62
G.7 After tasks questions . . . 63
G.8 Interview 3 . . . 63
G.8.1 Demographic questions . . . 63
G.9 Tasks . . . 63
G.10 After tasks questions . . . 63
G.11 Interview 4 . . . 63
G.11.1 Demographic questions . . . 63
G.12 Tasks . . . 64
G.13 After tasks questions . . . 64
G.14 Interview 5 . . . 64
G.14.1 Demographic questions . . . 64
G.15 Tasks . . . 64
G.16 After tasks questions . . . 64

G.17 Interview 6 . . . 65
G.17.1 Demographic questions . . . 65
G.18 Tasks . . . 65
G.19 After tasks questions . . . 65
G.20 Interview 7 . . . 65
G.20.1 Demographic questions . . . 65
G.21 Tasks . . . 65
G.22 After tasks questions . . . 66
G.23 Interview 8 . . . 66
G.23.1 Demographic questions . . . 66
G.24 Tasks . . . 66
G.25 After tasks questions . . . 66

H Interview Summary from round 1 **66**
H.1 Pages . . . 67
H.1.1 Content . . . 67
H.1.2 Playlists . . . 67
H.1.3 Locations . . . 68
H.2 Schedule . . . 68
H.2.1 Home . . . 68
H.3 General Feedback . . . 69

I Interviews from round 2 **69**
I.0.1 Questions before tasks . . . 69
I.0.2 Questions after tasks . . . 69
I.0.3 Tasks . . . 70
I.1 Interview 1 . . . 70
I.1.1 Demographic questions . . . 70
I.2 Tasks . . . 70
I.3 After tasks questions . . . 71
I.4 Interview 2 . . . 71
I.4.1 Demographic questions . . . 71
I.5 Tasks . . . 71
I.6 After tasks questions . . . 71
I.7 Interview 3 . . . 71
I.7.1 Demographic questions . . . 71
I.8 Tasks . . . 72
I.9 After tasks questions . . . 72
I.10 Interview 4 . . . 72
I.10.1 Demographic questions . . . 72
I.11 Tasks . . . 72
I.12 After tasks questions . . . 72
I.13 Interview 5 . . . 73
I.13.1 Demographic questions . . . 73
I.14 Tasks . . . 73
I.15 After tasks questions . . . 73

J Interview summary from round 2 **73**

Glossary

Term	Meaning
SPINmedia	Extendas' digital signage product for fuel pumps
SPINcms	The central, cloud-based web Content Management System (CMS) in which customers manage SPINmedia content.
SPINmmcs	The on-site controller at each location that syncs content from SPINcms and plays it locally.
Media	An uploaded image or video.
Track	An ordered sequence of media clips with a fixed aspect ratio.
Playlist	A set of tracks or media items, played as a time loop, with events that can trigger certain media to take priority
Event	Playback triggered by the dispenser state (Idle, Calling, Authorized, Fuelling, Closed).
Schedule rule	A rule deciding which content plays at which screen and when.
Overruling	A schedule rule property that gives it precedence over other rules.
Draft / Published	Editing model in which changes only go live after publishing.
Superadmin	Extendas staff member who can switch between customer organisations.
SUS	System Usability Scale, a ten-item usability questionnaire.
MoSCoW	Prioritisation method: Must, Should, Could, Won't have.

1 Introduction

Some fuel stations use their fuel pumps not only for dispensing fuel, but also for advertising. They use the build-in screens ons the fuel pumps to show images and videos to customers. Extendas, a Dutch software company in the petrol and car wash industry, offers this through its product SPINmedia.

To be able to do this, the people who run marketing and operations must be able to manage what is shown on the screens themselves. They need to upload new content and turn that into something that plays on a pump screen. They also need to decide what plays where and when. These users are generally not technical.

SPINcms is the cloud-based content management system (CMS) in which customers manage this content. A first version of SPINcms already existed at the start of this project. To find out how well it worked for its users, we interviewed eight non-technical people to see how they worked with it. Although they found the system usable overall, they ran into many problems. The difference between tracks and playlists was unclear to them, and on the scheduling page most participants were unable to complete their tasks at all.

The goal of this project is to improve SPINcms so that non-technical persons can manage the content on their screens without help from the company. We redesigned both the front end and the back end of the system. There were a lot of changes, but the biggest one is combining playlists and tracks. Also, a clearer way of scheduling and a complete new interface. Extendas also asked for a recommendation on which frameworks and architectures best fit the company and its product.

In this report we describe how we want about this. We start with chapter 2 where we introduce the client and the current system. We also show the user research. Then in chapter 3 we list the requirements. Chapter 4 shows how the project was approached. Chapter 5 and chapter 6 shows the global and detailed design of the new system. Chapter 7 contains the framework recommendation. In chapter 8 we show you how the system was tested. In chapter 9 and 10 we look at the results and give a conclusion.

2 Context and Analysis

This chapter gives the background needed to understand the design choices in this report. It first introduces Extendas, the client of this project, and the role of digital signage at fuel stations. It then describes the current SPINmedia system and the stakeholders involved. Finally, it shows you the user research we carried out on the existing CMS system and the problems that followed from it, which is the basis for the requirements in Chapter 3.

2.1 The Client: Extendas

Extendas is a Dutch software company that has been active in the petrol sector since 2001 and in recent years has also focused on the car wash industry. The software is focussed on the daily tasks of a fuel station and a car wash, with a POS system, back-office software, payment solutions, and other supporting software. Many of these products are part of SPIN, which is Extendas' online platform.
Extendas' strategy is to bring all services of a location together into one system. The idea is to make fuel stations a place where customers want to stop, instead of having to stop, and that a smooth customer experience increases the average spend per visit. The SPINmedia fits within this vision. It lets fuel retailers use the screens built into Gilbarco fuel dispensers, to show advertisements and promotions to their customers.
For SPINmedia to be used by real customers, Extendas needs a good, cloud-based Content Management System that non-technical customers can use themselves to upload content and schedule it for multiple locations. This is the assignment of this project.

2.2 Domain

Digital signage refers to digital screens that show centrally managed content, such as advertisements and promotions, mostly at the point of sale. Compared to printed posters, the content can be changed remotely and instantly, and can be changed to the time of day or the situation at the location. For example, when a person is fueling at a fuelpump.
Research shows that digital signage can influence customer behaviour, but that it really depends on the context. Roggeveen et al. (2016) found that digital displays increased sales in large hypermarkets, had little effect in supermarkets, but reduced sales in small convenience stores. Garaus and Wagner (2019) showed that a screen at a grocery checkout reduced perceived waiting time and increased overall store satisfaction.
At fuel stations this all comes together. The screens built into fuel pumps are there when customers are at a moment of forced waiting, namely while they are fuelling. Whether signage pays off depends on showing the right content at the right moment. The people who make these decisions are not IT-persons, but marketing staff at a gasstation company like OK. This makes the usability

of the CMS they use to manage content a deciding factor in whether digital signage delivers value at all.

2.3 The Current SPINmedia System

SPINmedia consists of three parts, shown in Figure 1: a central CMS in the cloud, a controller at each location, and the screens on the forecourt.

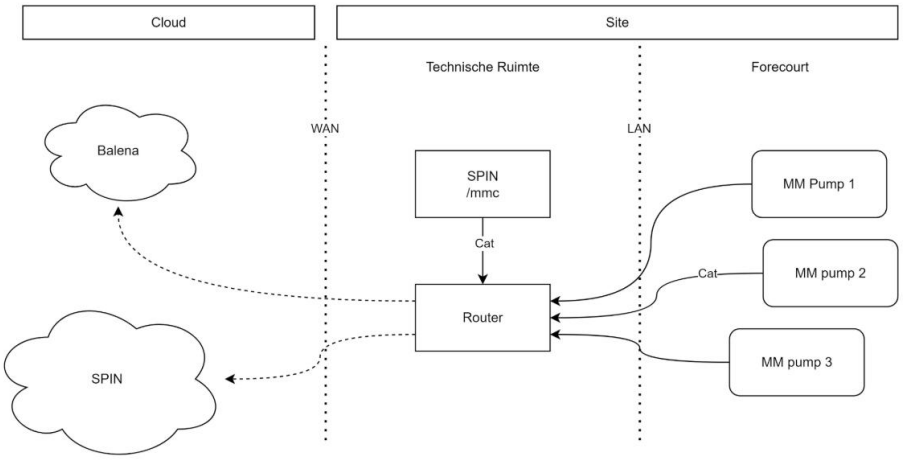


Figure 1: Overview of the SPINmedia system.

SPINcms is a central, web-based Content Management System that runs in the cloud as part of the SPIN platform. Customers use it to manage the content for all their locations. At each location, a small Raspberry Pi called the SPINmmcs is installed on site. It is connected to the local router, which connects it to the internet. The MultiMedia players in the fuel pumps are connected to the same router over the local network. The controller synchronises content from SPINcms and plays it locally on these screens. In addition, the controller is connected to Balena, a cloud platform for managing devices remotely. A controller is linked to its location in the CMS with a one-time linking code.

2.3.1 Content Model

Content in SPINcms is organised in a hierarchy. At the bottom are media files. These are uploaded images and videos. When a file is uploaded, the system takes a thumbnail, its duration, its resolution and its aspect ratio. Images are shown for 10 seconds by default, and both landscape (16:9) and portrait (9:16) formats are supported. Currently, in the field only landscape is being used, but portrait is still possible.

Media can then be combined into a track. You can compare a track to a Spotify playlist, with a media file being a song. There is an order for the clips.

Tracks are combined into a playlist, where you can also add more media files. Each entry in a playlist is played in one of two ways. This can be in a timeloop, both an entire day or specific times (for example, breakfast between 06:00 and 11:00). Next to this, an entry can be linked to an event. This is the state of the fuel pump, such as Idle, Calling, Authorized, Fuelling or Closed. We get these from the pumps themselves.

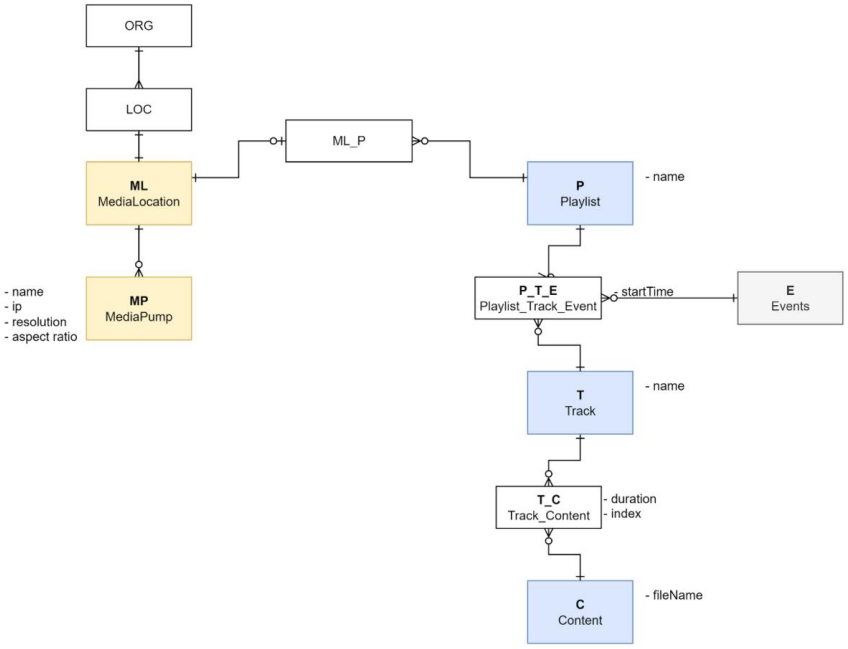


Figure 2: Overview of the SPINmedia Content Model.

2.3.2 Scheduling

What plays where and when is decided by schedule rules. Each rule links a playlist or track to a target. This can be the entire organisation, a group of locations, a single location, a group of devices or a single screen. When several rules apply to the same screen, the more specific target takes priority, so a rule for a single screen overrides a rule for the whole organisation. A rule can be limited to a specific time period, and can also be recurring. A rule can also be marked as overruling, which gives it priority over other rules regardless of the target level.

2.3.3 User Interface

Each organisation has its own environment in the CMS. The interface is currently mainly in Dutch and consists of the pages Overzicht (dashboard), Planning (schedule), Locaties (locations), Afspeellijsten (playlists), Tracks and Media. Technically, SPINcms is a Blazor Server web application, styled with Tailwind CSS and the daisyUI component library.

2.4 Stakeholders and Users

A stakeholder is any person or organisation with an interest in the project. Not all stakeholders need the same attention, so we used Mendelow's power/interest grid(Mendelow, 1981). The grid places each stakeholder on two dimensions: their power to influence the project, and their interest in its outcome. This results in four groups. Stakeholders with high power and high interest, high power and low interest, low power but high interest and low power and low interest. Table 1 lists the stakeholders of this project.

Stakeholder	**Role**	**Power**	**Interest**
Extendas product owner	Decides on priorities and accepts the result	High	High
Extendas development team	Will maintain and extend SPINcms after the project	High	High
Customer marketing staff	Primary users: upload content and schedule campaigns	Low	High
Station owners and managers	Occasional users: change content and check what is playing	Low	High
Extendas support staff	Superadmins who help customers with the CMS	Medium	High
Gilbarco	Manufacturer of the fuel dispensers and their screens	High	Low
Drivers at the station	See the content on the screens, never use the CMS	Low	Low
University supervisor	Assesses the project and this report	Medium	Medium

Table 1: Stakeholders of the project.

The product owner and the development team at Extendas have both high power and high interest. They decide what is built and will have to work with the result after the project ends. We therefore met with them once a week to check our designs and priorities.

The most important group for this project, the people who will actually use the CMS, has the least power over it. Marketing staff and station owners have a high interest in a CMS that is easy to use, but they do not take part in development and cannot influence the design directly. Without going to them immediately, people tend to forget their needs, which is exactly what our user

research showed: the system makes sense to its developers, but not to its users. We addressed this by doing interviews.

Gilbarco has high power but low interest. It has no direct stake in the CMS, but its pumps determine what the screens can show and which pump events are available, so its hardware is a fixed variable for our design. Drivers at the station have neither power over nor interest in the CMS, but they are the reason it exists: the content they see is the final product of everything users do in the system.

2.4.1 Personas

To keep the users in mind during design, we described the main user groups as personas. A persona is a fictional description of a typical user, with a name, background, goals and frustrations. Pruitt and Grudin (2003) found that personas engage a design team more effectively than abstract user descriptions, and that they work best when they are based on real data about users. Our personas are based on the information Extendas (and Bram) gave us about its customers and on our user interviews. Since we could only interview a limited number of people, and none of them were actual customers, the personas should be seen as a first version that can be refined when more data about real users becomes available.

Sanne, marketing coordinator at a fuel retail chain. Sanne (34) works at the head office of a chain with around sixty stations. She is responsible for the campaigns that run on the pump screens across all locations, and plans them weeks in advance, often differing per region. She is comfortable with office software and social media, but has no technical background. She wants to quickly set up a campaign for a group of stations from a certain date, and to check what is playing where. She is frustrated by technical terms she does not understand, and worries about accidentally showing the wrong content on all screens.

Henk, owner of two stations. Henk (58) runs two stations himself and changes the content on his screens about once a month, for example to promote the car wash or a seasonal offer. Because he uses the CMS so rarely, he has to find his way again every time. He prefers a clear, light interface with readable text, and he wants to get his task done without having to call Extendas.

Lars, support employee at Extendas. Lars (29) helps customers when something does not work as expected. As a superadmin, he switches between customer organisations to see their content, schedules and locations. He needs to find out quickly why certain content is or is not playing at a location, for example whether a location is offline or a schedule rule is overruled by another one.

2.5 Baseline User Research

For researching what users want out of this product, we decided to perform user interviews. There are 3 types of interviews that we consideredWilson (2014): Structured interviews, Semi-structured interviews and Unstructured interviews.

For this decision, we considered Pros and Cons for each type of interview (see Table 4).

Ultimately, we decided the best approach for our project would be a semi-structured interview. It both fits our necessity for somewhat consistent feedback, while giving the users the freedom to explore the website, given that most have likely not interacted with such a system before.

For this, we came up with a plan of what an interview, in broad strokes, should be structured like. The exact methodology can be found in the Appendix, but it outlines some demographic questions, a brief description of how the system works and then a list of tasks the interviewee should perform, so we can track speed, understanding and "out of the box" ease of use.

Concerning what type of person we wanted to interview, there were a couple aspects to consider. First, it would have been great to interview exactly people who have worked/will work with the product, such as local marketing consultants. However, due to time constraints we could not get in touch with people of their expertise, so we tried to simulate it. We ended up interviewing 8 people of different educational backgrounds, from and outside the University of Twente, that do not follow a technical study (such as TCS or BIT). Eight people is less then the ideal number of people, but due to time constraints, we could not perform more interviews. However, even though the number of interviews is small, we still can draw valuable feedback from it. It is shown that with only 5 participants, you can usually find around 85% of usability problems.(Nielsen and Landauer, 1993).

The results of the interviews will be laid out in the Appendix and the main feedback points are in the next section.

2.6 Core Problems

#	**Theme**
T1	Unclear model, especially tracks versus playlists
T2	Scheduling does not match how users think (campaigns, calendar)
T3	Unsafe interactions: data loss, hidden or off-screen buttons
T4	Little visibility of the result: what will play where, and previews
T5	Unexplained terms that are mostly only understandable to people in the industry
T6	Long lists without pagination, search or filter
T7	Visual accessibility: font size, contrast, dark default, no responsiveness

3 Requirements Specification

3.1 Requirement Gathering

When we first started working on the project the client gave us a quick explanation of the existing system and the following requirements were in the proposal from the client:

- There needs to be an intuitive interface.
- Users must be able to upload and manage content.
- There needs to be a way for devices to fetch content from the CMS with the least amount of work needed to be done on the existing devices.

Since these requirements are quite broad we spent some time in the first few weeks to define more specific requirements. We did this by using the system ourself, this helped us figure out how the new or changed features would fit in. The final system is quite a lot bigger than what these requirements describe it to be. We decided to not add requirements for all of the parts of the system that we were either not supposed to change or were already functioning fine via the description of the client.

During this process we also gathered requirements from the interviews. That mostly resulted in non-functional requirements, things like text needing to be big enough to read or modals needing to load fast enough. We suspect that we mostly got this kind of feedback because we decided to interview a group without a background in technology, thus resulting in the users focussing on usability and readability.

However, during a few interviews we were also surprised by some technical remarks. Those led to functional requirements like a preview during searching for content and the need for pagination. After the interviews we combined everything we gathered with our own findings and finalised a list of all the requirements that would define the new SPINcms. We gave the requirements a MoSCoW priority rating(Clegg and Barker, 1994), to make sure we wouldn't suffer from feature creep or focus on the wrong parts of the system. The requirements from the client were all assigned a 'Must' or a 'Should', the difference depending on how much value we saw in the requirements after the interviews and using the system ourselves. We then did the same for all the other requirements, giving new features a lower priority so that we finish the basic system first.

Only one requirement got the 'Won't have" priority. One of our ideas was to add authorization so we could set different permission levels for different users. For example to add content to a location instead of the entire organisation. Unfortunately the client told us from the start that this was out of scope.

Finally, after we gathered all the requirements we decided to make user stories for them(Cohn, 2004). This helps us make sure that the system is designed for the user. They also helped us decide what to prioritise on, within the MoSCoW groups of priorities.

3.2 Functional Requirements

ID	Requirement	Priority	Source
F1	Users can upload, preview, rename and delete media.	Must	Client, interviews
F2	Users can combine media into playable content with order and duration.	Must	Client
F3	Users can create tracks with subtracks or content	Must	Client, T1
F4	Users can enable menu options without shifting buttons off-screen.	Must	T3
F5	Users can schedule content for dates, times and recurring periods.	Must	Client
F6	Users can target a schedule at organisation, location group, location, screen group or screen.	Must	Client
F7	Users can see what plays, or will play, on a location or screen at a given time.	Must	T4
F8	Changes only go live on screens after an explicit publish.	Must	Client
F9	Devices fetch content from the CMS with minimal change to existing devices.	Must	Client
F10	Users see a preview whenever they choose content.	Should	T4
F11	Domain concepts are explained where they are used.	Should	T5
F12	Lists support pagination, search and filter.	Should	T6
F13	One concept for “a thing that plays” instead of tracks and playlists.	Should	T1
F14	When deleting content in use, users see where it is used.	Should	Interviews
F15	When schedule rules conflict, users see which rule wins and why.	Should	T2, T4
F16	The dashboard shows what is playing now and what needs attention.	Should	Interviews
F17	Calendar or timeline view of schedules.	Could	T2
F18	Bulk selection of locations and screens.	Could	Interviews
F19	Preview that simulates dispenser events.	Could	T5
F20	Tracks should have pagination.	Could	T6
F21	Users can view the files as a file explorer instead of a list.	Could	T6

3.3 Non-Functional Requirements

ID	Requirement	Priority	Source
Q1	Any action should show feedback within 5 seconds.	Must	T7
Q2	All text should be in a readable font/size, including small hints.	Must	T7
Q3	Tracks should have a publish/draft/save system for reliability.	Must	Client
Q4	The product should be usable by users with no prior experience.	Must	Client
Q5	Uploaded media should be type checked for security.	Must	Client
Q6	The system should keep track of editing actions.	Must	Client
Q7	The system should support accountability through user activity tracking.	Could	Peer Feedback
Q8	Saving or publishing should not take more than 5 seconds.	Could	Client

3.4 User Stories

ID	User Story
US1	As a user, I want to upload media, so that I can use it in my content.
US2	As a user, I want to preview media, so that I can verify its contents before using it.
US3	As a user, I want to rename media, so that I can identify it more easily.
US4	As a user, I want to delete media, so that I can remove files that are no longer needed.
US5	As a user, I want to combine media into playable content, so that I can create content for screens.
US6	As a user, I want to create tracks containing subtracks or content, so that I can structure the content that is played.
US7	As a user, I want to schedule content for specific dates and times, so that I can control when it is played.
US8	As a user, I want to create recurring schedules, so that I do not have to configure the same schedule repeatedly.
US9	As a user, I want to target schedules at organisations, location groups, locations, screen groups or individual screens, so that I can control where content is played.
US10	As a user, I want to see what is playing or will play on a location or screen at a given time, so that I can verify the playback schedule.
US11	As a user, I want to explicitly publish changes, so that I can control when they become active on screens.

ID	User Story
US12	As a user, I want to see a preview whenever I select content, so that I can identify the correct content before using it.
US13	As a user, I want domain concepts to be explained where they are used, so that I can understand the system without external documentation.
US14	As a user, I want to search and filter lists, so that I can find relevant items more easily.
US15	As a user, I want to see where content is used before deleting it, so that I can avoid unintentionally disrupting playback.
US16	As a user, I want to see which scheduling rule takes precedence when rules conflict, so that I can understand why specific content is played.
US17	As a user, I want to see what is currently playing on the dashboard, so that I can monitor the system.
US18	As a user, I want to see what needs attention on the dashboard, so that I can identify issues quickly.
US19	As a user, I want to view schedules in a calendar or timeline, so that I can understand when content is scheduled.
US20	As a user, I want to select multiple locations and screens at once, so that I can configure schedules more efficiently.
US21	As a user, I want to simulate dispenser events in a preview, so that I can verify how content responds to different dispenser states.
US22	As a user, I want tracks to support pagination, so that I can navigate large collections of tracks.
US23	As a user, I want to browse files in a file explorer view, so that I can navigate media in a familiar way.

4 Project Approach

This chapter describes how we approached the project. It describes the method that we followed. It also describes how the team was organised and what the initial planning was. Finally, it describes the risks we identified.

4.1 Method

The project followed an user-centred approach(International Organization for Standardization, 2019). We designed while keeping the user in mind throughout the process, instead of only at the end. This works good with our project, as the main requirement of Extendas was an intuitive interface for non-technical users. The first version of SPINcms showed that the system made no sense to the users, while the developers thought it was understandable.

In practice, we had sprints of two weeks, based on Scrum(Schwaber and Sutherland, 2020). Once a week we had a meeting with Extendas, where we got feedback which we used to set the priorities for the upcoming week. In the first sprint we focus on understanding the codebase and the users. We also had the first round of interviews. The second and third sprints were the development sprints, where we focussed on redesigning and implementation. In the final sprint we did testing, we finished the product and report, gave a presentation and did the last round of interviews.

4.2 Team Organisation and Communication

The team decided to choose a team structure without fixed roles. Because Bram has been working at Extendas for over 3 years already, he took on most of the initial communication and explaining of the current product. Because of this, Bram mostly took on the role of Product Owner. Next to this, we divided tasks based on preference and expertise, and we expected that everyone would roughly contribute the same to the product. You can read about the individual contributions later in this report.

Within the team, we met a few times a week to divide tasks and discuss progress, and kept track of tasks using Whatsapp and the GitHub repository. The code was managed in the existing GitHub repository of the product and the report was written together in Overleaf. We met with the client at Extendas once a week to show our progress and check our designs and priorities. We met with our univeristy supervisor Tom van Dijk biweekly to discuss the process and the report. There were also peer review sessions biweekly.

4.3 Planning versus Reality

The table below compares the original planning from our project proposal with what actually happened.

Period	**Planned**	**What happened**
Week 1 (31-08 – 04-09)	Form group, understand the project	Formed group, first meeting with Extendas.
Week 2 (07-09 – 11-09)	Company visit, proposal approval, codebase access	Company visit, proposal approved, codebase access received
Sprint 1 (14-09 – 25-09)	Design: research company, users and codebase; mockups	First interview round with 8 participants, analysis of the existing system, requirement
Sprint 2 (28-09 – 09-10)	Development: first iteration, client feedback	Started merging tracks and playlists, new interface design.
Sprint 3 (12-10 – 23-10)	Development: second iteration	Separating scheduling from content,
Sprint 4 (26-10 – 06-11)	Polish, testing and delivery	Automated tests (unit, component, integration), final presentation, final round of interviews, report

We did notice that sometimes the sprints overlapped a bit, where the designing took a bit longer on some parts, and other parts were done a bit faster then anticipated. In general, we followed the structure as given above.

4.4 Risks

Risk	Prob.	Imp.	Mitigation	Outcome
No access to the codebase	Low	High	Mockups and designs without integration	Did not occur
Company won't use a new stack or architecture	Medium	Medium	Check technical preferences early	Addressed by keeping the existing Blazor stack and giving a separate recommendation
A student drops out	Low	High	Lower scope with supervisor, or work harder	Did not occur
Slow feedback from the company	Low	High	Clear deadlines for feedback	Did not occur

5 Global Design

This chapter describes the overall design approach that is used for the redesign of SPINcms. Based on the problems that were identified during the user research and requirements stated in Chapter 3, the interface was redesigned with the goal of making the system more accessible for non-technical users. The redesign therefore focuses on simplifying the use of the system without losing any functionality by improving the visibility of important information and creating a more logical and consistent user-experience throughout the entire system. This chapter will address the design principles, the conceptual model, the architecture, and how it all fits together in the system.

5.1 Design Principles

The main goal of the system redesign was to make an user-interface that is more understandable, that can be used by non-technical people. The existing interface was designed more around the technical structure of the system, as described in Chapter 2. This structure is maybe logical and easy to use by developers or technical people, but it is not really intuitive and easy to use for the intended users of the system, such as marketing employees who want to manage the advertisements on the screens of a fuel station.
The user research highlighted various usability problems, for example, unclear terminology, complicated scheduling mechanism, limited feedback on actions and difficulties navigating large amounts of data. These problems formed the basis for the following used design principles.

5.1.1 Simplicity and clarity

One of the most important design principles is making the system more clear and simple. That means that the users of the system should be able to understand the purpose of a page/module and identify the important actions that can be taken without requiring any detailed knowledge of the underlying system. To achieve this we reduce unnecessary complexity, by making clear page titles, remarkable action buttons, and a more structured layout.
Instead of showing pages with a lot of information in technical terms and without a clear structure, the user-interface prioritises the information that is the most relevant to the user's task and more detailed information is shown in dedicated sections where it can be accessed when necessary using clear terminology and a logical structure.

5.1.2 Consistency

Another design principle that is applied is consistency. It is achieved by using the same navigation structure, button styles, typography and familiar layout of pages across the entire system. This reduces the amount of learning for the users required to navigate between different parts of the system.

For example, the main navigation sidebar always stays in the same position with clear active state of the current page, while primary actions are consistent in being highlighted in orange. Also similar information, such as the status of a screen or track is always presented in the same visual style. This consistency helps the user to navigate through the system more easily.

5.1.3 Visibility of system status

Another principle that has been applied is making the current state of the system visible to the users. In a system where digital signage is managed, users need to know if screens are connected, which content is currently displaying and if changes have been successfully synchronized.
The redesigned system therefore uses status indicators, summary cards and notifications. For example, on the locations screen you can select a location and see which devices are on and offline and the synchronisation information is displayed besides the devices.

5.1.4 Visual hierarchy and accessibility

The original interface had limited visual separation between elements with inconsistent and relatively small text. Besides that important action buttons were remarkable and therefore the page and functions of it were not clear. During the redesign we decide to use a default light themed interface using dark text, orange accents and enough spacing between sections(Piepenbrock et al., 2013; World Wide Web Consortium, 2023).
The intention behind this was to make the pages easier to scan and to make a clearer distinctions between primary actions, secondary information and system statuses.

5.1.5 Error prevention and user control

The redesigned system aims to make important actions understandable and recognizable from less important actions. Because managing content can involve actions that have affect on multiple screens of locations, like accidentally removing or publishing content and publishing incorrect schedules. So, changes only go live on screens after an explicit publish and users are informed when deleting content that is currently in use described as requirements in Chapter 3.1 as F8 and F14.

5.2 Conceptual Model

During the initial research, one of the main problems identified was the complexity of the SPINmedia content model. As explained in Chapter 2.3.1, the original system used the following concepts to create the advertisements: media files, tracks, playlists and scheduling rules. Content management and scheduling were also not fully separated because scheduling related settings were also configured within the playlist and tracks.

This content model was intended to give the users a lot of flexibility but it also introduced a complex flow that users had to understand and follow before they can manage their advertisements. Especially the difference between tracks and playlists was not always clear because both were used to organize content, but it was not clear why two separate concepts were needed.
Therefore the conceptual model was redesigned around the following to main principles: merging tracks and playlists into a single concept and separating the content management from scheduling.

5.2.1 Merging of tracks and playlists

In the redesigned system, the two concepts tracks and playlists have been merged into a single concept called a Track. This new track concept combines the functionality of the previous separated concepts without removing any capabilities that were in the original system.
So instead for the users to understand both tracks and playlists and the relationship between them, they can now organise media with one central concept called tracks. It contains the media items, their order and playback settings. This provides the users with a more straightforward flow for managing content.
To realize merging of tracks and playlists not only the front-end of the system was modified, the backend-end and database structure also had to be adapted in order to realize the new unified tracks concept without losing functionality of the original playlists and tracks.

5.2.2 Separating content management and scheduling

Another change of the system was the separation of content management and scheduling related settings.
In the original system, the playlist and tracks contained scheduling-related configurations. That meant that users had to deal with scheduling logic while working on the content.
In the redesigned system, we focused on a clear separation. Tracks are responsible for defining what content should be displayed and the planning section handles when and where a track should be shown. In order to schedule content the users now navigate the the scheduling page and create a new advertisement, they first select a track and then configure the scheduling rules like playback times, target locations or screens and other conditions.
This separation allows users of the system to create and manage their tracks independently of scheduling. Therefore it also makes the relation between content and scheduling easier to understand because each part of the system has a more clearly defined goal and responsibility.
Like merging the playlist and tracks as we described before this change also was not only on the user-interface of the scheduling page but required modifications to the underlying back-end and database. The scheduling content had to be separated from the playlist and tracks and the scheduling logic had to be adapted to work with the new track model.

5.2.3 Resulting conceptual model

The resulting conceptual model now has a clear flow of three main stages. Users have a media page where they can for instance upload images and videos. Then the user combines these media files into a track. Finally the user can publish an advertisement on the scheduling page where they select their made tracks and configure rules like playback and on which location or screen it has to show.
By separating these main stages, the redesigned system is more user-friendly for non-technical users, because they can focus on creating tracks without considering about any scheduling and can manage when and where that content is displayed as a separate task.
This redesign of the conceptual model addresses the problems that were identified in research themes T1 and T2 in Chapter 2.6 and functional requirements F2, F3 and F11 in Chapter 3.1.

5.3 Information Architecture and Navigation

The information architecture describes how the system functionalities are organised, and how users navigate through the different sections.
In the original system, the users had to navigate between multiple pages that contained technical information and different content management and scheduling concepts with limited visual hierarchy, this made it difficult for less experienced and non-technical users to understand where particular functions were meant for and where they were located.
In the new redesigned interface, the system is divided into six main modules: Overview, Planning, Locations, Tracks, Media and System. The modules represent the main tasks that are involved in digital signage management.
The overview page focuses on displaying a summary of the current situation, that includes current displayed content, connected locations and screens, tracks and media. It‘s goal is to be a informative page and a starting point from which users can navigate to specific information and actions. The planning page displays and manages the when and where the tracks are shown. The locations page shows information about physical locations, screens and controllers. The tracks page allows users to create advertisements using media files that are uploaded through the media page. Finally, we have the system page that contains administrative functionality, for example, user management, integrations and system settings.
Furthermore, a clear and user-friendly sidebar is introduced on the left side, to make the above modules easily accessible. The sidebar has a consistent visual styling where the active page is highlighted, this helps the user to always know its current position in the system.
Besides the sidebar each page follows a similar layout. It begins with a clear title and short description about the page. When relevant, it then shows summary cards displaying important information, followed by the section with the main purpose of the page.
Searching and filtering functionality has also been implemented, this is relevant

for organisations that manage many locations, screens, tracks and media files, where searching through long list becomes very inefficient.
The objective of the information architecture is to have a predictable navigation experience where users easily find the functionality of the system they need.

5.4 System Overview

The redesigned interface is the central point from which the users manage their digital signage content, and while the visual structure changed it is implemented to work with the original SPINmedia infrastructure as described in Chapter 2. The complete system consists of the content management system, the on-site controller and the multimedia screens on location. The SPINcms is for managing the content and scheduling the content to the multimedia devices. On location, the SPINmmcs synchronises information from the central system and manages the playback on the multimedia screens.
The typical workflow begins on the media page, where users can upload and manage their media files. Then on the tracks page the user can use those media files to create a track. Once a track has been created, the user can go to the schedule page to publish the track and determine when and where it should be displayed. On the location screen the user can inspect the status of locations, screens, identify connection problems and see what content is displayed on specific locations or screens. Finally, the overview page brings all these different pages together. Instead of requiring the user to go to separate pages and parts of the system, it provides a summary of the relevant information of all the parts of the system.
The purpose of the redesign is to make SPINcms more understandable and manageable, further technical considerations and architectural recommendations are discussed in Chapter 7.

6 Detailed Design and Implementation

This chapter describes how we applied the design principles to the individual modules of SPINcms. The redesign was made to address the usability problems and the design principles from Chapters 2-5. We define three stages of the redesign: the original interface, the first redesign that we made in Figma after the initial interviews, and the final interface that was actually implemented for Extendas. This is important to note because the Figma design we made was a starting point and not aimed to be the actual final product. There were several interactions and layout changes after the Figma design, but still with the same goal as enabling non-technical users to manage content without needing to understand the underlying system architecture.
After the Figma design, we discussed the interface more with Extendas and therefore had a final round of changes. Some elements therefore did change a bit. In the sections below, we identify the improvements we made based on the initial interviews separately from the final changes we made.
The main workflow of the redesign is to upload media, combine some media in a track, add or check locations, and schedule that track to some locations and/or screens. The following sections explain the decisions for the interface behind this workflow and the technical changes required where it was needed.

6.1 Scheduling

Scheduling was one of the most important and serious usability problems in the original system. Users had to create a registry rule that combined the choice between the playlists, tracks and the individual content with dates, recurrence, time windows, exclusivity and multiple target levels. The first interview round we did showed that the users did not understand most of these terms and they could not always see where a rule that they created appeared. Some of the times, the pop-up layout of creating the registry rule pushed controls outside the visible screen. As a result, most participants struggled to complete the scheduling tasks.
We changed the system from form-first approach to a visual planning task in the Figma redesign. This new redesigned page shows the screens as rows and the time as horizontal blocks. The blocks also show with their colours which one is active (dark orange) and which is not active(light orange). We also implemented here a calendar style navigation and a very clear button for adding a schedule. This directly addressed the feedback from the interviews to switch from a list of technical records to something that the users understand more clearly.
The final implementation has still kept the visual approach. However, we improved the schedule creation process. Scheduling a track has now four steps: What plays?, Where does it play?, When does it play?, and Extras. Users need to select a track first, then they choose an individual screen, location, screen group, location group or all the screens. After they have done this, they select the days, time slots, repetition and applicable date range. The optional settings

(rule name and exclusive playback) are put at the end, which makes it more clear that they are optional. We also have implemented a new visual bar that makes the the playback interval a lot easier to understand and visualize. The final schedule page has both day and week views, location filtering and the horizontal schedule rule blocks with their playback period. This also allows them to see the schedule rule, as well as inspect and change an exisiting one. This also now allows to see visible confirmation after saving or adding a schedule rule, which also helps with the feedback. The three questions of what, where and when helped us make scheduling easier to understand without removing any features.

6.2 Tracks

The old SPINcms had two models, Track and Playlist. It turned out to be an incredibly confusing part of the system for most interview partakers. We quickly decided to merge the different functionalities into one model: the Track. This did pose some problems as the Playlist was mainly used during the process of getting the content ready to be used in the SPINmmcs.
Besides changing the model, we also redesigned how users interact with the tracks. In the original dark interface the tracks and playlists were on separate pages, which caused confusion during the interviews. Our Figma design introduced the lighter interface with a clearer overview of playlists and their content. The new track editor allows users to add media or other tracks, change their order, configure events and preview the results on the same page. This has made creating and managing tracks easier to understand.
One of the main requirements of EXTENDAS was that tracks should be able to hold a sub-collection of content. This allows users to create templates, which means less work to create seasonal content for example. The old system handled this by allowing tracks, which consists of content, to be added to playlists. In the new system a Track consists of Trackparts, and a trackpart can either reference a piece of content directly, or reference another track.

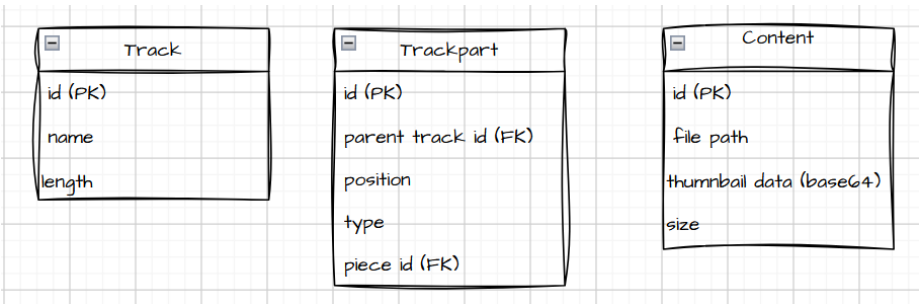


Figure 3: Diagram of the new Track model.

While the new system is simpler for users to create with, there now is a problem of circle dependencies between tracks. A user is free to add a parent

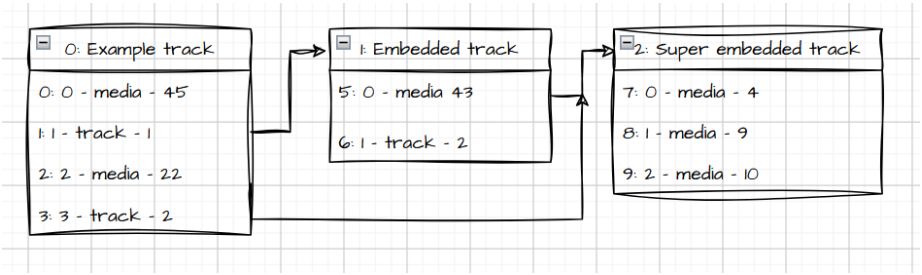


Figure 4: Example of a Track with 2 Subtracks.

track as a sub track, unfortunately the intended use case of the system does not allow for this. When a Track is added as a subtrack all of it's content has to be fetched from the database, while doing this the system checks for circle dependencies.

After the main features were implemented for the new Track system, we had to connect it to the existing architecture. The connection to the SPINmmcs mostly depended on features from the Playlist model which were either deleted or moved. We quickly found these conflicts by removing the original files from the Playlist model and then fixing all the compilation errors. It turned out that the data for the scheduling part of the system was stored in the PlaylistConfiguration model, which was made to reflect the changes in the new system and renamed to 'ScheduleRule'. The PlaylistTrack model which was used both for events and in scheduling was also modified and renamed to 'TrackEvent'. The old system used to put both tracks and events in a PlaylistTrack model, butt he regular logic from tracks was moved to the Track model and 'TrackEvent' now only handles the events for the system.

Unfortunately after these changes the part of the system responsible for getting the data ready to be used by the SPINmmcs was broken again. It was incompatible with the recursive search and fetch function for subtracks in parent tracks, as well as not being ready for events referencing either a track or a piece of content directly. After a quick discussion we saw no use case for events being able to reference a track, so we decided to make it reference a piece of content only. The recursive fetching function was quickly fixed and the entire system was tested and working again.

6.3 Media Library

In the original media page was an upload button and just a list of media files. Although the basic operations were present, like filenames, thumbnails and metadata, they were making the page unclear. Also because of the long list of media files, it didn't support the user with organizing and searching larger collections of media files. From the interviews, the participants requested clearer previews and a easier way to navigate the media files. This was important because selecting a wrong media file would affect every track were that media file is used.

The first Figma redesign of the media page presented the media files grid with folders, a search field and filters. This was a significant improvement because it treated the media files collection as something where the users can browse and organize easily. However, the redesigned media page was more refined into a more complete file explorer layout. This means that the final redesign also uses a folder tree, navigation controls, a list of media files with useful columns, search and filtering, and a toolbar for common media file operations. The users also can switch the way they view the media library and the page displays media file counts and storage information at the top of the media page.
The redesigned file-selection pop-up on the track page reuses the same concept of the media page when adding media files to a track. Besides being more clear and easy to use it also indicates when certain media files are unsuitable for the format of the current track directly in media file list. These changes connect the media page to the track creation workflow and helps users make selections without needing to inspect file dimensions manually.

6.4 Locations and Screens

The original locations page listed all the locations and when clicked open a detail page with address, controller-linking information, technical information and connected multimedia screens. So it gave a lot of technical and unclear information to the user but not an immediate overview of their organization. The Figma redesign gives the user a more clear and organised view of their organization. It combines summary cards, a map, a searchable location list and a detailed view of the connected multimedia screens connected to locations.
The final implementation keeps this structure while displaying information obtained from the connected system. The Locations page still contains an interactive map, with a searchable table and summary cards for locations, screens, pending synchronisation and alerts. We did improve the visual layout of this in the final implementation. Selecting a location now also opens a more focused detail page. The information that is needed is shown clearly such as controller connection, last contact, sync state, storage usage and screen status. More technical identifiers remain available, but are visually separate from the primary information. The location page also includes a schedule timeline of that location, allowing the users to see today's schedule for that location. This design helps both marketing staff and Extendas support employees. Marketing users can check if the intended track is assigned to a screen, while the support team can investigate a disconnected controller or delayed synchronisation wihtout needin ot seacrh through settings that are unrelated.

6.5 Dashboard

The original overview page only showed some summary information, device titles and some very technical network debugging details. It did not provide the user with a clear answer to the everyday question: what is the current status of my locations and screens? Therefore the redesigned overview page acts as the

main starting point op SPINcms.
This was realised by combining today's schedule, location and screen status, track activity and media information into a understandable and user-friendly layout. This redesigned overview page also gave users the option to easily navigate to the different parts of the system to resolve an issue or make a change. The first section displays the daily and optionally the weekly schedule for a selected location. Below that is a summary of the locations, screens, synchronisation, tracks, scheduled tracks and media files. Detailed network debugging information is not relevant for everyday tasks of a user and is therefore moved to another section.
The redesigned overview page displays both the current activity and problems without users having to navigate through different pages of the system.

6.6 Visual Design System

The changes that were made to the original system were not only functional. The first interview round identified several visual problems, for example, the standard dark theme, low contrast secondary text, small labels and usability problems on smaller devices. Users also had to use a horizontal navigation bar and work on different parts of the system where the layout was inconsistent across the them. Therefore we used a standard light themed interface with a fixed sidebar on the left side of the screen, clear page titles, more consistent spacing and remarkable primary actions.
The first Figma prototype uses a white main colour and an orange sidebar. The primary and important actions and button were marked orange, because a white background separates parts of a page without making them look like completely independent tools. The final redesign still has the visual hierarchy of the prototype but it refined the overall appearance even more. The sidebar uses an orange gradient instead of flat orange, the background is changed to a more light-grey colour with content cards in white, the active page is highlighted in the sidebar with a white background and the selected organisation is displayed at the bottom of the sidebar.
The visual language is also improved, the system supports system feedback as well as navigation. Green status labels indicate successful processing or screens that are connected and orange status labels marks primary actions and scheduled items. This practice is implemented in the entire system to make the interface more predictable for the user.

6.7 Implementation

The redesigned system is implemented within the existing SPINcms application, it didn't replace or change the underlying choice of technology. As described in Chapter 2, the application uses a Blazor Server with Tailwind CSS and daisyUI. The SPINmmcs controllers on location still receives the playable content from the central system. Because the original architecture was not changed, it reduced the need for changes that were already deployed on devices and allowed

for more focus on the usability and content model.
The implementation was divided in two levels. In the front-end, a shared navigation structure, repeated page layouts, improved media-management, unified track concept and editor, and a visually improved schedule timeline.
In the backend-end, the replacement of tracks and playlists into just tracks required changes to domain models, database interactions and generation of data for SPINmmcs. In particular, the TrackPart supports nested content, TrackEvent represents event-related playback and ScheduleRule separates scheduling logic from track creation. These changes required compatibility work described in Chapter 6.2 because the visually simpler workflow that was created still had to produce the same type of playable output for the SPINmmcs.
The redesigned system was reviewed with Extendas in the development stage and was then refined into the final version. The second interview round provided an confirmation that this approach was successful because al participants completed their assigned tasks in contrast with the original system where participants failed several tasks. Even though the participants in round 2 succeeded in completing the tasks, the group size was small and was not a sample of the actual users of the system. The remaining feedback about, for example, font sizes, localisation and controls shows that the final redesigned system should still be fine-tuned. The test strategy and technical verification of the implementation are discussed in Chapter 8.

7 Framework and Architecture Recommendation

As part of the project proposal Extendas wrote for us, they requested an analysis of their current product framework and to offer alternatives. This section will be about breaking down their current approach, comparing it to other alternatives, and giving them a final recommendation.

7.1 Current Situation and Constraints

SPINcms is built using .NET and C# for digital signage, with Blazor pages for the interface with connectivity through SignalR. Styling is kept consistent with Tailwind CSS and daisyUI. For data storage, it uses PostgreSQL.

For deployment, there are 6 different Docker containers:

- **spimcms**: the webapp itself
- **api**: the API that the controllers (MMCS) use to connect the screens
- **distribution**: handles the distribution of content to the controllers
- **analyser**: media file inspection
- **replica**: handles data relating to organizations and their location
- **supporting services**: different proxies

There are constraints that the company would need to keep in mind if switching to a different architecture. For example, the time and resources it would take the development team to get familiar enough with the framework to ensure good quality implementation. Any alternative would also need to work with the screens. Of course, it would also need to integrate nicely within the entirety of the SPIN ecosystem.

7.2 Evaluation Criteria

When looking at alternatives, there are several factors to keep in mind, such as similarity to the current system, integration with the existing SPIN ecosystem, reliability, speed, security, etc. Because of this, we will rank each each option we find in a weighted (High-Medium-Low importance) table with the following criteria:

- **Maintainability and team fit - High**
- **Migration cost and risk - High**
- **Reliability and offline behaviour - High**
- **Security - High**

- **Integration - Medium**
- **Scalability - Medium**
- **Cost - Low**

7.3 Options

Taking those restrictions into account and looking at other industry standards, we managed to find 3 alternatives to the current SPINcms implementation.

7.3.1 Move to Blazor's Auto/WebAssembly render modes

Since .NET 8, Blazor can render components on the server first and then hand them over to WebAssembly running in the browser (Microsoft, 2024). Interactive pages such as the schedule board or the track editor would then run on the client and stop using server resources. Larger benefits are that it's still using Blazor, which makes it easy to integrate and not everything needs to be changed. The downside is that the UI can no longer access services and the database directly: it needs a proper API layer, and the first page load gets heavier.

7.3.2 ASP.NET Core API with a JavaScript single-page application (React, Vue or Angular)

This is an industry standard, which makes the future hiring pool large for Extendas and it is well documented. However, it means rewriting more than sixty existing Razor components in a different language. It would also consume a lot of resources, as in the time it would take for employees to get familiar with this framework.

7.3.3 Adopt an existing digital signage CMS (e.g. Xibo, Yodeck, ScreenCloud)

Off-the-shelf platforms offer scheduling, media management and player software immediately. However, they do not integrate with SPIN for organisations, locations and access rights. They do not support the built-in pump screens. Commercial products usually charge per screen, which scales badly across many stations. This would also mean Extendas no longer offers their own services, which comes with the benefit of customer support and a "closeness" with the clients.

Table 3: Comparison of framework options against the evaluation criteria

Criterion	Weight	Blazor Auto	API + SPA	Off-the-shelf
Maintainability and team fit	High	++	−	+
Migration cost and risk	High	+	−−	−−
Reliability and offline behaviour	High	+	+	+
Security	High	+	+	∘
Integration	Medium	++	+	−−
Scalability	Medium	+	++	+
Cost	Low	++	+	−
Weighted score		**23**	**4**	**−3**
Rank		**1**	**2**	**3**

++ very good (2), + good (1), ∘ neutral (0), − poor (−1), −− very poor (−2). Weights: High = 3, Medium = 2, Low = 1. Weighted score = $\sum$ (points × weight).

7.4 Recommendation

Taking these options into consideration, if Extendas is looking to shift their approach, we would recommend them to slightly alter the current implementation and use Blazor's Auto render mode. Compared to the other options, it clearly wins in team fit, cost and integration, because of how closely it resembles the current CMS. There are upsides and downsides to this, as mentioned in section 7.3.1. The reason we ourselves did not implement this during development is because we cannot make these decisions in the name of the company. They know best what was previously tried, where their bottlenecks are and what they consider to be non-issues.

8 Testing

This chapter will focus on the testing strategy for this project, specifically what the tests do, how they work and why such tests were chosen.

8.1 Test Plan

Most of the testing we have done in previous University projects have been Unit tests: check different functions on some sample values and check if the outcome is the one developers would expect. Alongside these tests, both due to good web-testing practice and the advice of our supervisor, we decided to also perform automatic tests. That is, testing not only if functions behave correctly in the back-end but also if there are any inconsistencies between front-end actions and back-end processes.

When looking at which tools to use for testing, we came across a few "obvious" choices: nUnit is a .NET testing framework which already matches our .NET project environment; bUnit is specifically made for Blazor component rendering and testing;

For the automatic webpage interaction tests, we had to pick two services: one that handles the API calls and one that runs the browser. The tool we are going for to be a SPIN API replica WireMock, because it uses C#, which is already convenient for us after working with Blazor pages, which also use C#, also because it does not require an external application, which makes it quick to adapt for the company as well. For the browser handler, we found a couple viable options options: Selenium WebDriver and Playwright. Selenium is a popular choice for testing, however for our objective it has one important factor that made us not choose it: manual waiting times. After performing a browser click using Selenium, we would have to hardcode the amount of time to wait until the next browser click. This waiting time accounts for front-end - back-end communication and element loading. Playwright however automatically checks every few milliseconds whether every element is done loading, which not only makes the tests easier to write, but also less prone to errors. An additional, smaller point, is that Playwright also uses C#, which again facilitates integration within the rest of the code. So in the end, WireMock and Playwright are the chosen tools for the job. For a visual representation of the testing framework, see Figure 29.

We have decided to split the tests into 4 categories:

1. **Unit tests** - Usual unit tests, checking function outputs or correct back-end functionality (e.g. file names are parsed correctly, it only allows certain file types etc.)

2. **Component tests** - Razor pages get rendered in memory, where proper connectivity between components is tested (e.g. a click opens the correct window, interactions between different components are, or aren't, allowed)

3. **Integration tests** - A dummy database is created to check if queries execute correctly and if restrictions are properly applied (e.g. if a piece of content is currently in use, it cannot be deleted; if Organization A uploaded a piece of content, someone using Organization B's account cannot access that content)

4. **End-to-end (E2E) tests** - WireMock pretends to be the SPIN Replica API. Playwright opens a Chromium tab to host the tests. This simulates interaction, performing different tasks, checking outputs and evaluating errors.

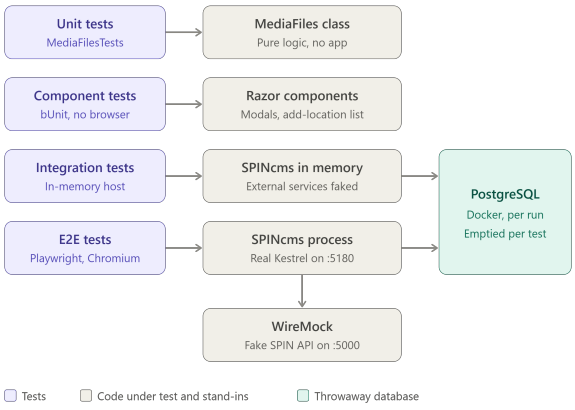


Figure 5: Diagram showcasing testing framework

8.2 Results

In the Testing folder we added a README that explains the testing architecture, how to run the tests and what each test does. This is so Extendas employers have an easy time getting up to speed on what we have deployed if the company decides to use our service. Firstly, testing prerequisites:

- Docker must be running for the Integration and E2E tests
- Chromium must be installed on the machine

For some tests, a testing method is ran with multiple examples, so below we will go into detail on the number of tests, what they cover and how they performed.

1. **Unit tests - 9 methods, 23 tests - 23 passes, 0 failures**

   Tests tackle mainly media files. This includes name cleaning, when the name of the file is too long, it has spaces, it contains special characters etc. These names must be trimmed correctly. It also checks file extensions,

such that it allows PNG, JPG and MP4 files, but refuses all other types. For an added layer of security, it also checks for executable or HTML files renamed to PNGs.

2. **Component tests - 20 tests and methods - 20 passes, 0 failures**

   Tests the Blazor pages, whether buttons clicked send the correct signal. For example, closing a window actually closes said window, or that buttons are available/not available in the right scenarios.

3. **Integration tests - 34 tests and methods - 34 passes, 0 failures**

   Mainly PostgreSQL tests, checking whether creating, editing, uploading or deleting files and folders work correctly. Also checks for the right permissions depeding on the account, such that accounts under different organizations cannot access each other's content. Also a few login related tests, concerning session tokens and credentials.

4. **E2E tests - 32 methods, 44 tests - 44 passes, 0 failures** These tests are sort of a combination of the Component and Integration tests, but where these 2 sides are actually connected. Browser clicks are simulated using a temporary database, checking if the Blazor pages update correct and if the right updates happen in the database. Some of these tests include uploading and editing a file and adding a new location on the Locations page.

Regarding the scope of the tests, we tried to make them as inclusive as possible, especially in areas of high security importance or where we've personally encountered difficulties designing. The 100% pass rate indicates that the code is functionally correct. Both in testing using this framework and playing around ourselves, we could not find any unexpected behaviours. Obviously, it could still be that there are minor hidden flaws, but none that we managed to uncover.

9 Evaluation, Consequences and Follow-up

In this chapter we evaluate the result of the project. We describe to what extent the system meets its goals and requirements, how the process and collaboration went, what the result means for the company and future work recommendations.

9.1 Evaluation of the Product

The goal of this project was to improve SPINcms so that non-technical customers manage content without help from Extendas. The first round of interviews showed two main problems: the unclear difference between tracks and playlists (T1), and a scheduling page that most participants could not use at all (T2).

The difference between tracks and playlist was unclear. We merged this into one concept, a track. This track can now also hold other tracks. The scheduling was made way clearer, by making it more visual in a way that people are used to.

There are a few things that are limited. None of the participants in the interviews were actual customers. They were people who are non-technical like the intended users, but they don't have the domain knowledge that the actual users do have. The results therefore say more about how easy the system is to learn, but not how it performs in daily use by marketing staff. The number of participants was also small.

9.2 Evaluation of the Design Process and Collaboration

Starting the project with user interviews on the existing system was good. Without them, we would probably have focused on improving the existing interface, while the interviews showed that the main problems were in the underlying concepts.

The weekly meetings with Extendas meant that our designs were checked regularly and that decisions could be made quickly. We do think that the communication towards Extendas could have been better, and that our supervisors could be kept more up-to-date throughout the process.

9.3 Consequences for Extendas

The redesign has a few consequences that Extendas needs to take into account before the new version can replace the current one.

The existing data must be migrated. Customers who already use SPINcms have everything in the old content model. This has to be converted to the new model, so no playlists and the new scheduling rules.

Some functionality also changed. Events can now only trigger a piece of media, not a track. We did not see any use case for this.

The new interface also has English as its default language, not Dutch. Most customers are Dutch, so they need to change it to Dutch if they want to use it in Dutch manually.

Finally, Extendas now gets an automated test suite. This makes it easier to change the system without breaking anything.

9.4 Future Work

Based on the results of this project, we recommend the following next steps.

The most important next step is to test the new system with actual customers of Extendas. Real marketing staff and station owners can confirm whether the redesign also works for experienced users with domain knowledge.

Also, a migration of existing customer data is needed before the new version can be released, as described in Section 9.3. This is very important too.

Finally, the framework and architecture recommendation in Chapter 7 can be used as a basis for long-term decisions about the technical direction of SPINcms.

10 Conclusion

The goal of this project was to improve SPINcms so that non-technical customers of Extendas can manage the content on fuel pumps without help from Extendas.

Interviews with eight non-technical participants showed that the original system did not meet this goal. The participants could not tell tracks and playlists apart, and tasks were difficult. These interviews showed that the main problems were not only in the interface, but in the concepts underneath it.

We redesigned SPINcms at both levels. In the content model, tracks and playlists were merged into a single concept, the track, which can contain both media and other tracks, so that no functionality was lost. On top of this, the interface was redesigned for non-technical users. The resulting system is covered by 121 automated tests, which all pass.

Whether the redesign fully reaches its goal cannot yet be concluded with certainty. The redesign directly addresses the problems found in the interviews, but it has not yet been checked by actual customers of Extendas. Testing the new system with real customers, and migrating their existing data to the new model, are very important.

Overall, the project has turned SPINcms from a system that made sense mainly to its developers into one that is structured around the tasks of its users. With these next steps, it provides Extendas with a solid foundation for a CMS that its customers can use on their own.

11 Individual Contributions

11.1 Tom

- Design and implementation for merging track and playlists.
- Adapting SPINmmcs bridge logic after removing playlists.
- Interviews round 1 + summary
- Interviews round 2 + summary
- Requirements
- Final presentation

11.2 Teodor

- Testing design and implementation
- Coming up with the interview strategy with Tom

11.3 Serkan

- Creating the inital Figma design based on the first round of interviews
- Implementing the Figma design for the overview, scheduling, locations and media pages
- Refining all the pages after final suggestions from the company

11.4 Jakob

- Creating the inital Figma design based on the first round of interviews
- Implementing the Figma design for the overview, scheduling, locations and media pages
- Implemented the locations page on the front-end and back-end

11.5 Bram

- Implementing the new styling
- Implementing language localization
- Implementing Light/Dark mode
- Implementing different "quality of life" features (eg. how buttons and dropdown lists behave)

12 Use of AI

No LLMs or other generative AI tools were used during the design of the system or the writing of this report.

During implementation, Claude 5.5 was used to assist with minor front-end development tasks and to generate translations for the localization feature. It was also used to replace hardcoded text with localization IDs and create a comprehensive README for the testing scheme. All AI-generated code was thoroughly reviewed and tested before being merged into the system.

References

Dai Clegg and Richard Barker. *Case Method Fast-Track: A RAD Approach*. Addison-Wesley, Reading, MA, 1994.

Mike Cohn. *User Stories Applied: For Agile Software Development*. Addison-Wesley, Boston, MA, 2004.

Marion Garaus and Udo Wagner. Let me entertain you – increasing overall store satisfaction through digital signage in retail waiting areas. *Journal of Retailing and Consumer Services*, 47:331–338, 2019. doi: 10.1016/j.jretconser.2018.12.008.

International Organization for Standardization. ISO 9241-210:2019 ergonomics of human-system interaction – part 210: Human-centred design for interactive systems, 2019.

Aubrey L. Mendelow. Environmental scanning – the impact of the stakeholder concept. In *Proceedings of the Second International Conference on Information Systems (ICIS)*, pages 407–417, Cambridge, MA, 1981. URL `https://aisel.aisnet.org/icis1981/20`.

Microsoft. ASP.NET core Blazor render modes. `https://learn.microsoft.com/en-us/aspnet/core/blazor/components/render-modes`, 2024. Official documentation, accessed 6 November 2026.

Jakob Nielsen and Thomas K. Landauer. A mathematical model of the finding of usability problems. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, pages 206–213. ACM, 1993. doi: 10.1145/169059.169166.

Cosima Piepenbrock, Susanne Mayr, Iris Mund, and Axel Buchner. Positive display polarity is advantageous for both younger and older adults. *Ergonomics*, 56(7):1116–1124, 2013. doi: 10.1080/00140139.2013.790485.

John Pruitt and Jonathan Grudin. Personas: Practice and theory. In *Proceedings of the 2003 Conference on Designing for User Experiences (DUX '03)*, pages 1–15. ACM, 2003. doi: 10.1145/997078.997089.

Anne L. Roggeveen, Jens Nordfält, and Dhruv Grewal. Do digital displays enhance sales? Role of retail format and message content. *Journal of Retailing*, 92(1):122–131, 2016. doi: 10.1016/j.jretai.2015.08.001.

Ken Schwaber and Jeff Sutherland. The Scrum guide: The definitive guide to Scrum: The rules of the game. `https://scrumguides.org/scrum-guide.html`, 2020.

Chauncey Wilson. *Interview Techniques for UX Practitioners: A User-Centered Design Method*. Morgan Kaufmann, 2014. ISBN 9780124103931. doi: 10.1016/C2012-0-06209-6.

World Wide Web Consortium. Web content accessibility guidelines (WCAG) 2.2. https://www.w3.org/TR/WCAG22/, 2023. W3C Recommendation, 5 October 2023.

A Appendix

B Source Code

This project adapts Extendas' actual deployed system. The code and its iterations exist in the private GitHub repository of the company. For any questions and potential access to review the code, contact Arjan Vissers, Extendas CTO.

C Raw Research and Test Results

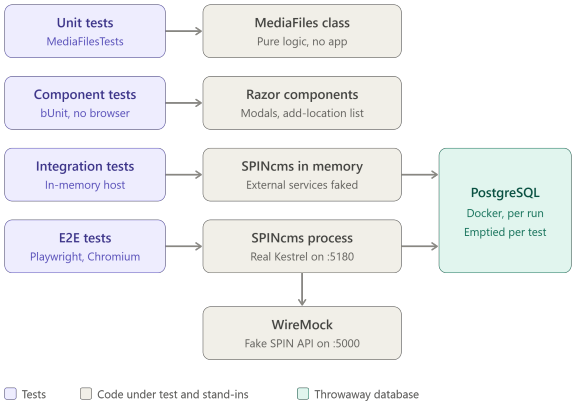


Figure 6: Diagram showcasing testing framework

D Design Evaluation

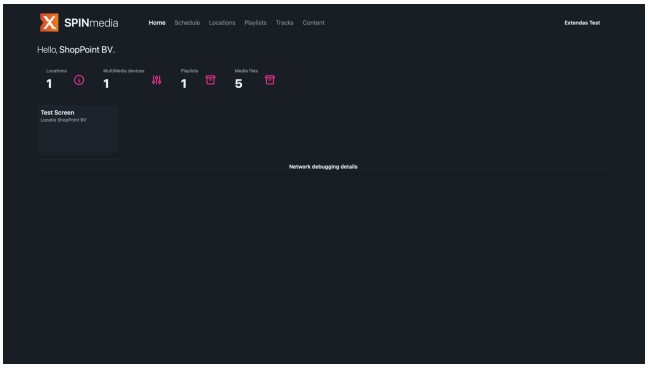


Figure 7: Old overview page

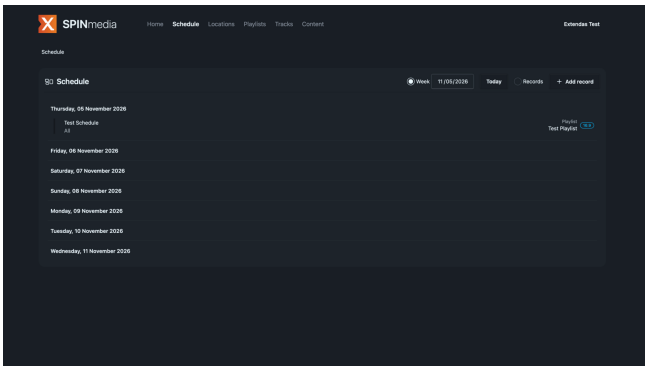


Figure 8: Old schedule page

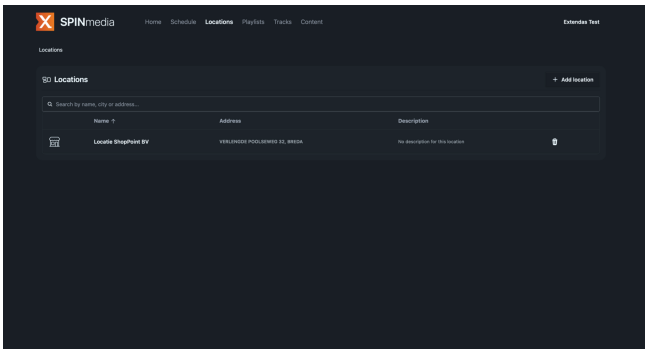


Figure 9: Old locations page

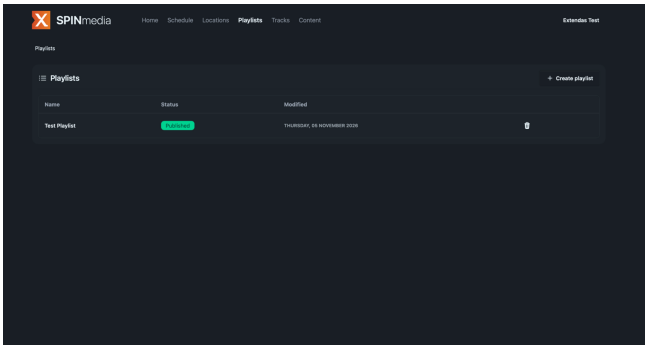


Figure 10: Old playlist page

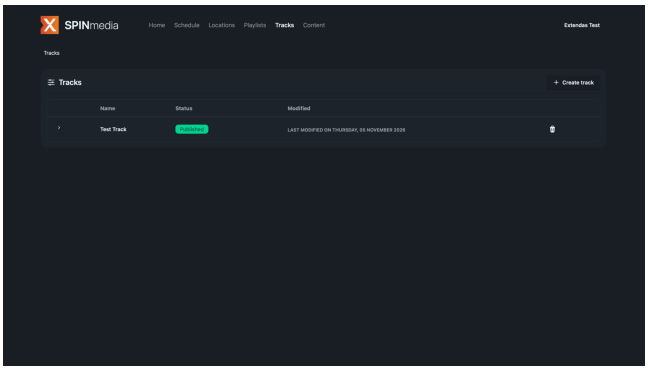


Figure 11: Old track page

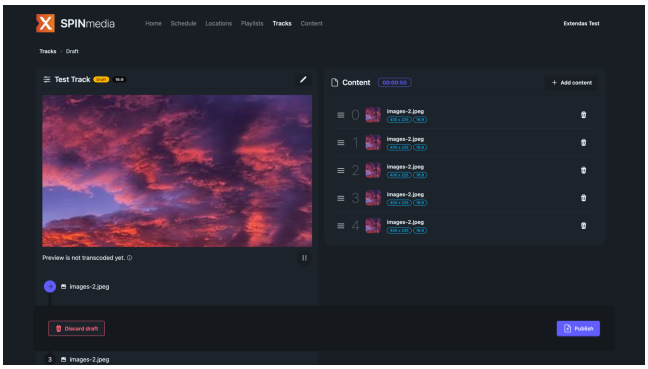


Figure 12: Old track editor page

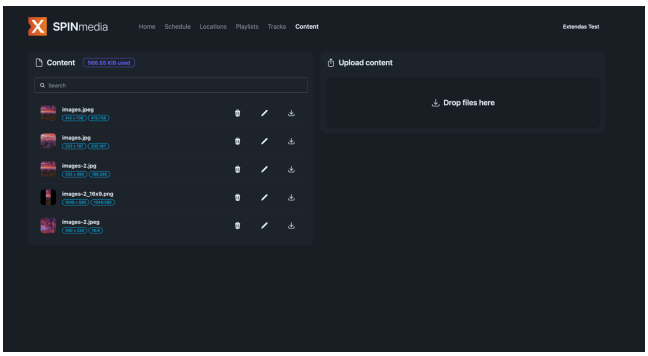


Figure 13: Old content page

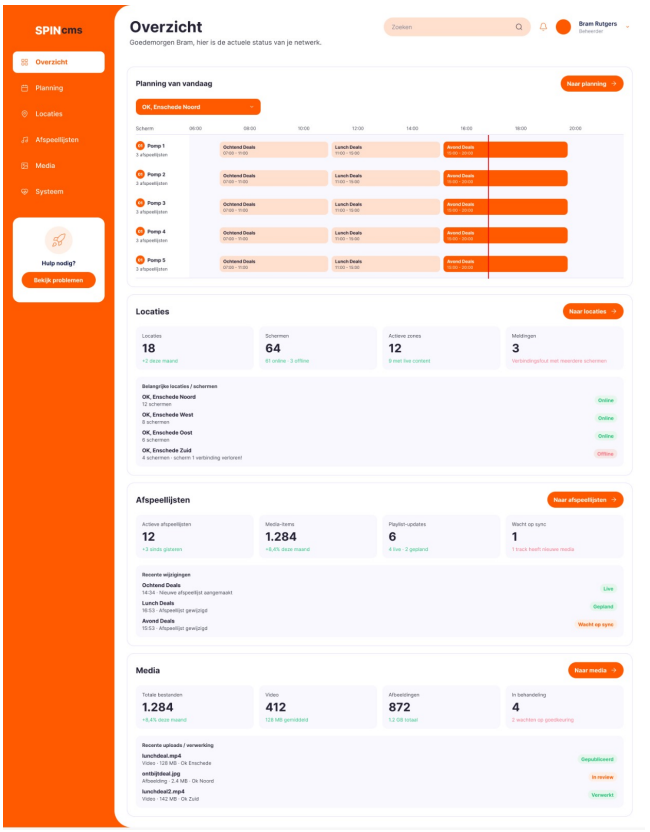


Figure 14: Prototype overview page

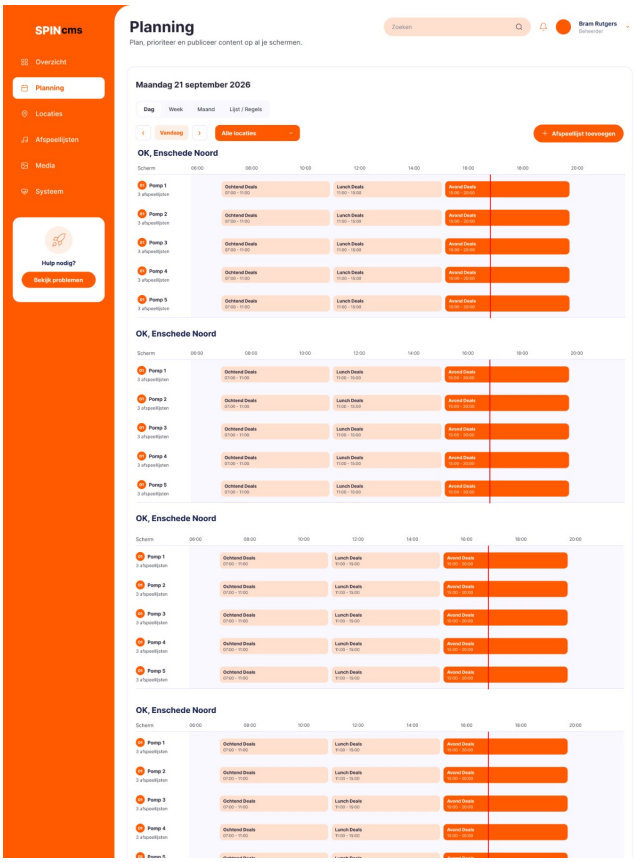


Figure 15: Prototype schedule page

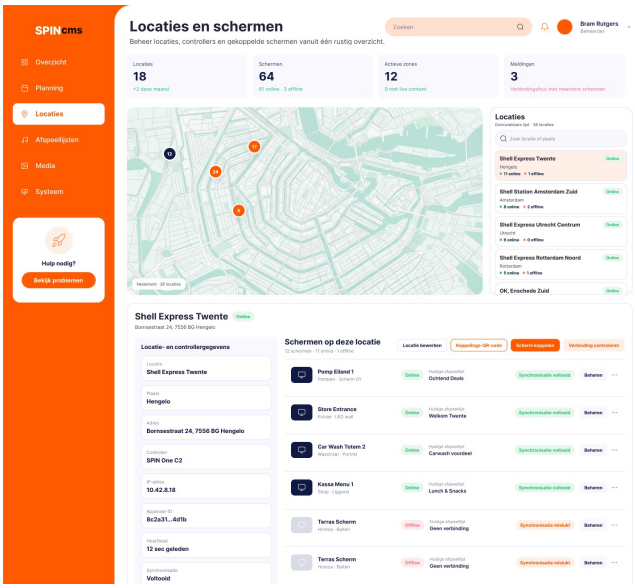


Figure 16: Prototype locations page

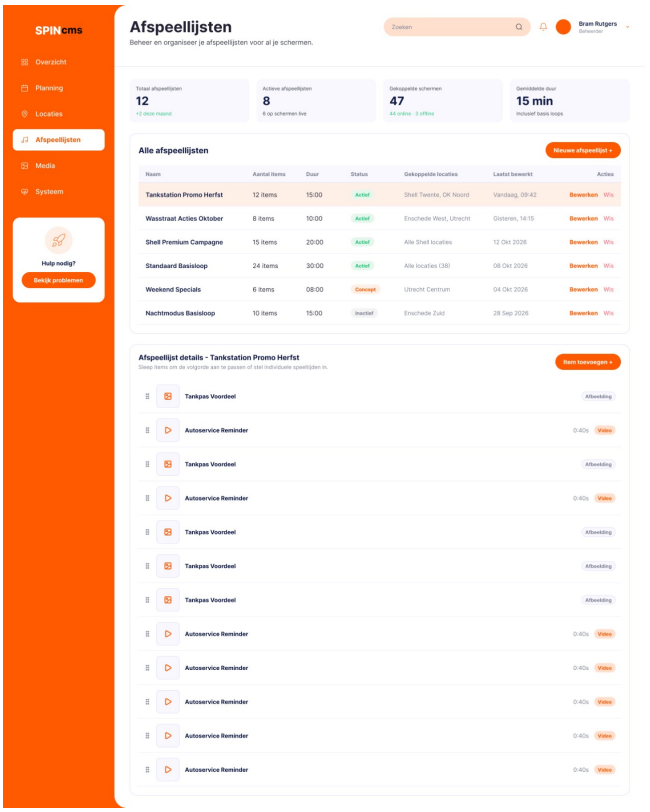


Figure 17: Prototype track page

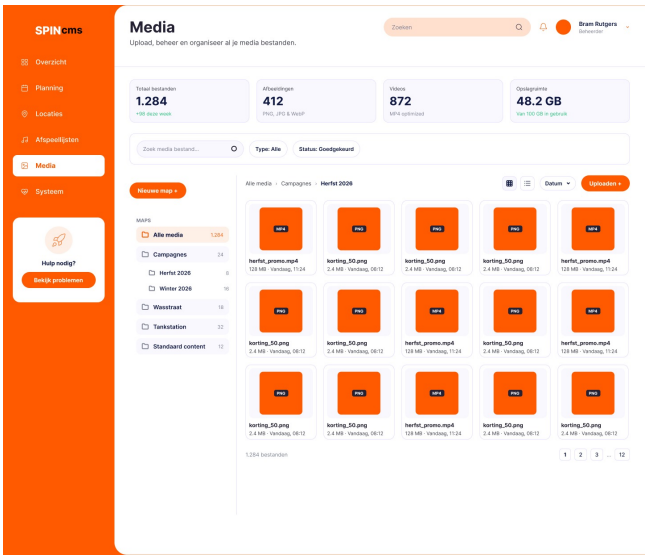


Figure 18: Prototype media page

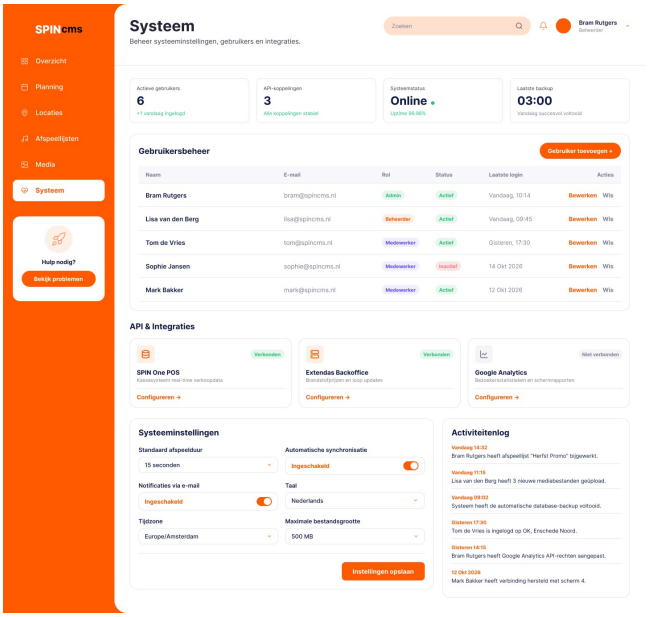


Figure 19: Prototype system page

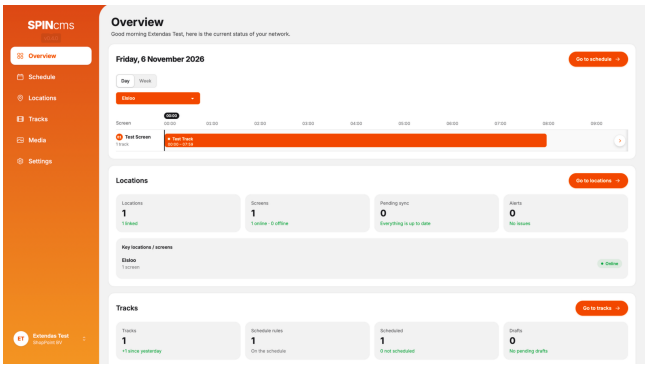


Figure 20: Final overview page

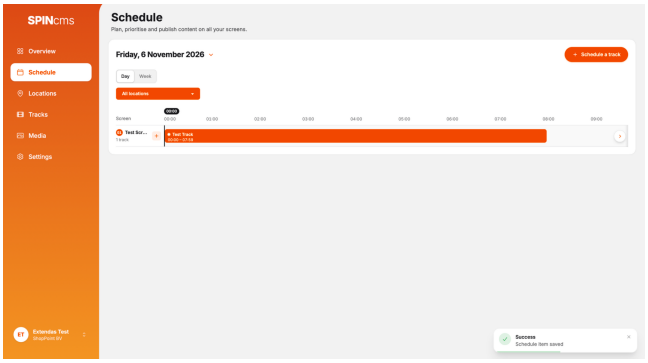


Figure 21: Final schedule page

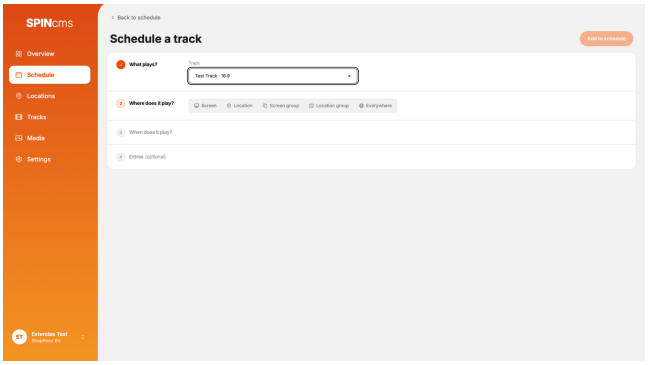


Figure 22: Final scheduling track page

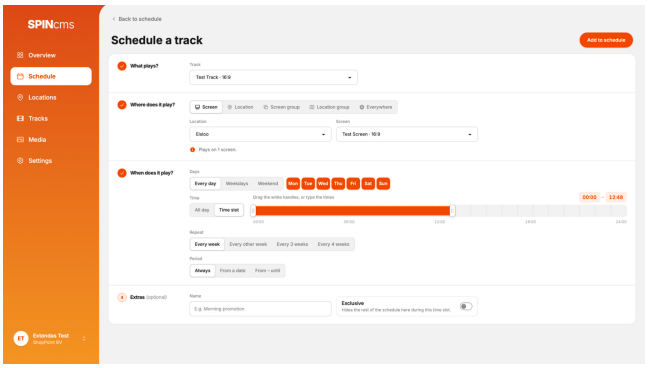


Figure 23: Final scheduling track page

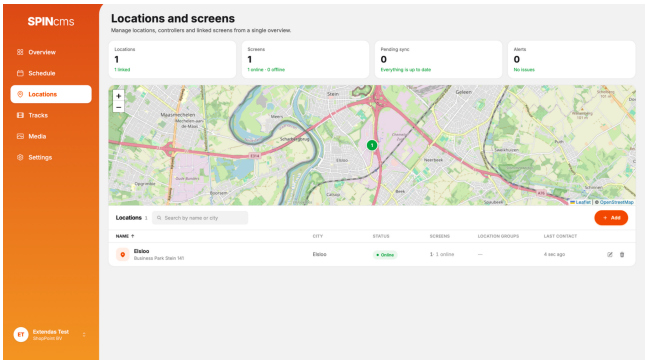


Figure 24: Final locations page

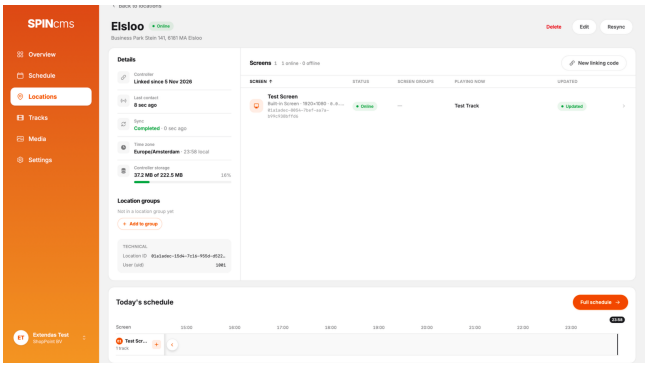


Figure 25: Final locations specific page

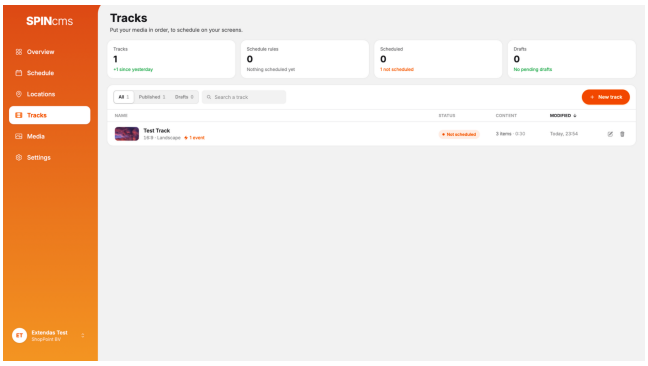


Figure 26: Final tracks page

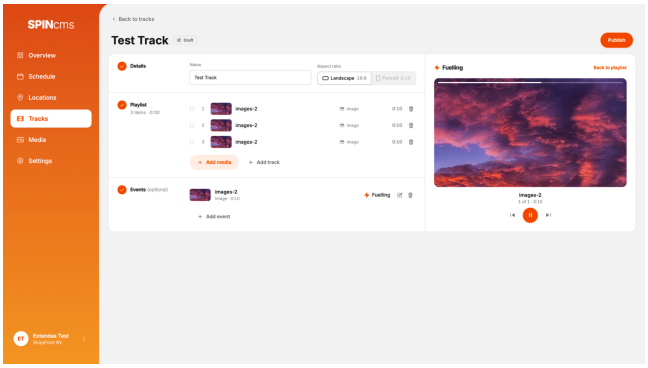


Figure 27: Final tracks editor page

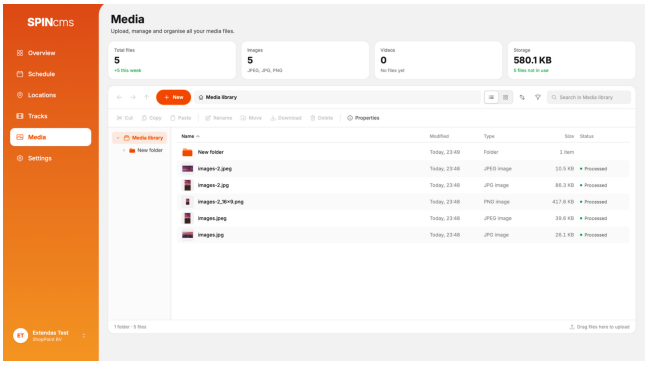


Figure 28: Final media page

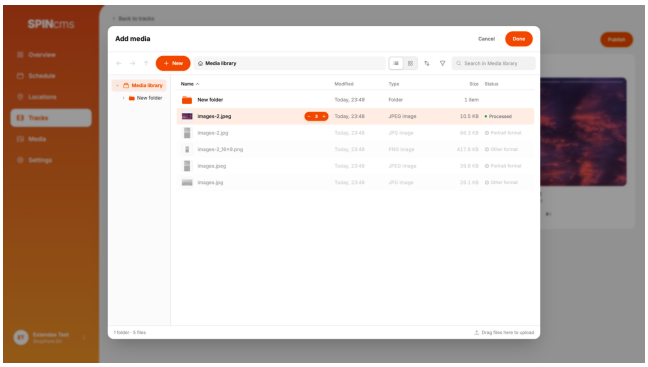


Figure 29: Final add media page

E Project Proposal

E.1 Project, Organization and Requirements

Our project revolves around redesigning, both front-end and back-end, Extendas' client-side Content Management System (CMS). They supply the necessary hardware and software to businesses, such as car washes and gas stations, to run advertisements and to have full control over the displayed content, currently limited to fuel points from Gilbarco.
These are the requirements set by the client:

- The system must have an intuitive interface
- Non-technical users must be able to upload and manage content
- There needs to be a way for devices to fetch content from the CMS with the least amount of work needed to be done on the existing devices

In addition to these requirements, we must also explore different possibilities for architecture and framework and present the best fit for the product and company.
As a team, our internal organizational structure is flexible, without well-defined leadership roles. As we do not have access to the codebase yet, we have not divided any tasks, but these will be done by personal preference and expertise. Everyone is expected to have about the same input, both in the project and the report.

E.2 Planning

We have decided on a four-sprint approach, with each sprint lasting two weeks. Before our actual sprints start in week 3, we used the first two weeks for preparation and the project kick-off:

Week 1 (31-08 till 04-09): We formed our project group and started understanding the project.
Week 2 (07-09 till 11-09): On Monday, September 7th, we visited the company for a tour and received a brief overview of the system. We also found a supervisor and aim to have a meeting on September 10th to get their approval on our finished project proposal. We will spend the rest of the week trying to get access to the codebase.
Throughout the entire project, we will update the final design report during every sprint to ensure it stays accurate and is finished well on time.

E.2.1 Sprint 1 (14-09 till 25-09) - Design

During this sprint, we want to focus on 2 main things: gathering information about the company, the clients, and the codebase and designing the system based on requirements given by both the company, as well as the clients. We want to look at the architecture of the current system and decide whether or not we agree with the structure or propose changes. We are planning to make a few mockups on the design, presenting this and refining it before actually implementing the system .

E.2.2 Sprint 2 (28-09 till 09-10) - Development

In the first development sprint we'll use the information gathered and the designs that we made to start implementing the first iteration of the system. At the end of the sprint, we will show the system to the client and gather useful feedback.

E.2.3 Sprint 3 (12-10 till 23-10) - Development

Using feedback from the client on our first iteration of the project, we'll start working on our second and improved iteration.

E.2.4 Sprint 4 (26-10 till 06-11) - Polish and Deliver

During the last sprint, we will focus on finalizing existing features, implementing the last small features, and dotting the I's. Besides that, we will test the system thoroughly to make sure all the features work correctly.

E.3 Deliverables

There are 3 deliverables for the project: the Design Report, slides for the final presentation and a poster outlining our project. The deadline for all these deliverables is November 6th.

E.3.1 Design Report

This document will be almost equally time consuming as the project itself. Because of this, we are planning on working on both in tandem. Once a topic relevant to the report has been approached practically, we will document it to make sure the report is not left behind the project. Our goal is for the report to not take much longer past the project's completion.

E.3.2 Presentation Slides

These will be done close to the presentation deadline, November 4th, since they do not take more than a few hours to complete, and will provide a good visual aid for our presentation

E.3.3 Poster

This will be done likely within the last week, after the whole (or at least most) of the project is done. It will outline the functionality of the project, the successes and failures, visual guides and other relevant details.

E.4 Risk Analysis

Risk	**Prob.**	**Imp.**	**Mitigation**
No access to the codebase	Low	High	Mockups and designs without integration
Company won't use a new stack or architecture	Medium	Medium	Check technical preferences early
A student drops out	Low	High	Lower scope with supervisor, or work harder
Slow feedback from the company	Low	High	Clear deadlines for feedback

E.5 AI-use policy

The company has a liberal policy on using generative AI while working on products. The company encourages employees to use generative AI by giving them access to a shared Claude Pro environment. Therefore, we think using generative AI to assist in the development of the project should be fine. All code generated by AI will be checked by us, and never blindly copy-and-pasted.

F Figures

Structured Interviews		Semi-structured Interviews		Unstructured Interviews	
Pros	**Cons**	**Pros**	**Cons**	**Pros**	**Cons**
1. Unillateral consistency between interviews, thus it is easy to compare each question directly.	1. The average person does not have enough experience with a CMS to answer theoretical questions about them.	1. Still gives a level of consistency between the interviews, while leaving more room for diversity in each interview approach.	1. The interviewer needs to be careful not to lean too much into structured or unstructured territory.	1. Gives the interviewer full freedom to mold the interview based on each interviewee.	1. Interviews risk differing too much, so we do not have enough consistency to measure outliers vs. recurring opinions.

Table 4: Pros and cons of structured, semi-structured, and unstructured interviews.

G Interviews from round 1

G.1 Interview Plan Round 1

G.1.1 Sample group

Since the main objective of this project is to create a system that can be used by non-technical people we'd like our interviewees to not have too much technical knowledge. While it might be nice to get some data from people that do have that background, we mainly focussed on people without it.

G.1.2 Questions before tasks

These questions were asked to get a general impression of our demographic and to make sure we didn't interview people with very similar backgrounds.

- Age
- Study and/or job
- Technical background

G.1.3 Questions after tasks

After the tasks we asked some questions to get the actual feedback from the tasks. It's important to note that we also took notes during the tasks to learn directly from the interactions with the system.

- During the steps of the tasks, did anything feel slow or hard to figure out?
- Did the general workflow feel correct or did you have to move back between the steps a lot?
- Do you have any notes about the layout or the looks of the system?
- Are there features that you feel like you're missing?

G.1.4 Tasks

The following list are the tasks that we gave to the participants. The list includes multiple tasks, not all participants did all tasks, most did 1 or 2.

- 1
  - Upload specific files
  - Create a playlist with the new content
  - Schedule the playlist
  - Add any event to it
  - Change the scheduling to not all screens on a location
- 2
  - Upload specific files
  - Create 2 tracks with any content
  - Create a playlist
  - Have the playlist include a track
  - Add a track as an event to the playlist
  - Assign playlist in the schedule
- 3
  - Remove a track from a playlist
  - Remove the playlist from the schedule
  - Add an event to the playlist
  - Re-add the playlist to the schedule
- 4
  - Find 2 playlists in the schedule, name them

G.2 Interview 1

G.2.1 Demographic questions

- Age: 18
- Job/Study: Mechatronics MBO / Store employee
- Technical background: No coding, proficient computer user.

G.3 Tasks

Tasks performed: 1 and 4
Task 1 was given up after trying, but wanted to continue so we did 4 as well.

G.4 After tasks questions

- Failed after creating the tracks/playlist/content so schedule was the problem.
- Preview after adding new content to a track resets, which seems a wrong design choice
- Fine
- No

G.5 Interview 2

G.5.1 Demographic questions

- Age: 57
- Job/Study: Secretary/Project Manager
- Technical background: Using software like the office suit

G.6 Tasks

Tasks performed: 2 and 4

During the interview the participant was unable to follow most of the steps. Having to back track through the system quite a bit. Next to that the font choice and background colour appeared to be hard to read as the participant had to zoom in from time to time.

G.7 After tasks questions

- The add record menu is slow to open. The scheduling system was not understandable. Playlists versus tracks are unclear.
- Workflow is fine and the choice of words for buttons between pages are good.
- Hard to read at times, dark mode seems unprofessional
- Locationless records for the schedule.

G.8 Interview 3

G.8.1 Demographic questions

- Age: 21
- Job/Study: Chemical Science Engineering
- Technical background: coding for mathematical things in R or mathlab

G.9 Tasks

Tasks performed: 2 and 3

During the interviews he asked a lot of questions but in the end most tasks were completed.

G.10 After tasks questions

- The scheduling system was a challenge.
- Workflow seemed to be in reverse, going from right to left instead of left to right.
- Schedule layout seemed wrong
- Final preview before publishing a track or playlist

G.11 Interview 4

G.11.1 Demographic questions

- Age: 60
- Job/Study: Business executive
- Technical background: highschool coding classes, office suit.

G.12 Tasks

Task performed: 1

During the tasks the participant expressed some frustrations and asked questions. Task was not fully completed.

G.13 After tasks questions

- Scheduling was too hard and too many options at once. Tracks vs Playlist was confusing.
- Workflow was fine.
- Layout was fine. Font/style choices made things hard to read (grey/dark combo)
- Better preview when working with content in the playlist and track pages.

G.14 Interview 5

G.14.1 Demographic questions

- Age: 28
- Job/Study: Humanitarian Engineer / Plastic Researcher
- Technical background: python coding, regular computer user

G.15 Tasks

Task performed: 1, 2, 3

This participant performed a lot of tasks, and got more familiar with the system than others. After a bit of rough start with the scheduling part, it went quite well.

G.16 After tasks questions

- Schedule was described as not functional with long loading times.
- Playlists vs Tracks was unclear in the workflow
- Layout of the add record menu for the schedule shifts offscreen.
- Pagination for content and location pages.

G.17 Interview 6

G.17.1 Demographic questions

- Age: 32
- Job/Study: Retail manager
- Technical background: Software for work

G.18 Tasks

Task performed: 3

Participant was familiar with software used only by employees for work. This seemed to help as they were used to having to figure out this kind of software.

G.19 After tasks questions

- In general all menus and specifically schedule were described as clunky
- Events from playlists vs tracks was unclear
- Add record is not clear in schedule
- No suggestions

G.20 Interview 7

G.20.1 Demographic questions

- Age: 25
- Job/Study: MBO pedagogiek
- Technical background: social media, regular computer use

G.21 Tasks

Task performed: 3, 4

The participant said during the tasks that it was quite a frustrating experience because of having to go back a lot and not understanding the scheduling system, task 3 was not completed.

G.22 After tasks questions

- In general all menus and specifically schedule were described as clunky
- Workflow seemed reversed and had to go back constantly. English wasn't great, maybe add language support?
- No
- Language support

G.23 Interview 8

G.23.1 Demographic questions

- Age: 21
- Job/Study: Technical Computer Science
- Technical background: Both design and implementation of digital systems

G.24 Tasks

Task performed: 2, 4
The tasks part went quite fast, the participant just kept pressing any button until the task was completed. We found out that some records disappear in the schedule if not added correctly.

G.25 After tasks questions

- Too much information at once in schedule, time can be entered both in playlist and schedule menu. Did not understand difference between tracks and playlists. Schedule menu broke layout and hid components
- Workflow seemed to be in the wrong order, should be flipped
- General looks were fine, layout was also fine except for modals. Did not work on mobile at all.
- Pagination, better overview page

H Interview Summary from round 1

After the first round of initial interviews using the original EXTENDAS Content Management System we can finally make some general statements about the current and next design of the system.
First we'll start with an overview of the feedback by going through it per webpage. Afterwards there will be a general feedback part.

H.1 Pages

H.1.1 Content

While the general feedback was not necessarily negative on this page, during the testing part most users experienced some hick-ups on this page and had to ask for additional help. Most of the problems were with the 'Create Track' menu, but also on the initial view of the page, where it lists all the current tracks, some users got stuck. In the interview afterwards most users still described it as an usable page there was still some feedback:

- Pagination for the tracks
- Edit/publish button for the tracks, in addition of clicking on the name
- Add Content menu can cover publish button, hiding it from the user
- Don't restart the preview when adding content, preview the just added content
- Allow the user to select a piece of content to preview
- Combine the indices of the content on the right with the preview controls on the left

H.1.2 Playlists

During both the before and after interview it became quite clear that a big issue with the system was the difference (or lack thereof) between playlists and tracks. For the new design we have already decided to make a system where they are combined into one concept, while allowing for the same functionality. In general the experiences of the users were very comparable to that of the tracks page, according to their feedback:

- Pagination for the playlists
- Add track button also allows to add regular content
- Preview of track/content in the lists
- Publish button can be hidden from the user by one of the sub menus
- More explanations of what an event means/when it is fired
- Add track menu shows no information about what the time data is used for
- Previews should be used when adding content instead of just names

H.1.3 Locations

The locations page was a bit underused in hindsight as it was not included in most of the tests the users had to do. However, when it was used by a user it was obvious that this page was visually clear and easy to use. During the interview after the users were done, there were some suggestions:

- Pagination
- Easier way to select, group or see all the locations

H.2 Schedule

This page performed the worst out of all pages by a large margin. It had many issues ranging from UI layout hiding buttons below the screen, to incorrectly entered schedules not showing up anywhere. This meant that most users were unable to perform the tasks that they were given during the interview, meaning that they also felt unable to give feedback on the page as they didn't quite grasp what it was supposed to do.
After some further prodding and giving some hints to the users the following feedback was given:

- Add record does not make sense to non-technical people
- Information of existing schedules are all over the place
- Labels/dates/titles/tags have no meaning to the user
- Pagination or other way of displaying it instead of long list
- Buttons move off screen without becoming scrollable when enabling options
- Long loading time for the menu
- Confusion between selecting a playlist, track or content
- Previews for selecting a track/playlist
- Unclear what the role is of the Organisation, location, location group, device, device group options
- Time window and exclusive options are unclear what they do
- Clicking next to the window closes it, possibly deleting a lot of information on accident

H.2.1 Home

Since this page only provides some data for the user to see, it was not included in any of the tests, however that also shows that for any of the users stories it was useless, so we should think about whether we want to keep this page or change it in some way to have more use cases and functionality.

H.3 General Feedback

In general most users considered the system usable. While a lot of issues and hick-ups were experienced during the testing phase of the interview, even though they were all first time users of this system. The last questions of the interview asked for some general feedback:

- All users that were quite a bit older than the students, noted that the default dark mode of the system made it feel unprofessional
- The light grey secondary font colour was hard too read at times
- While the pages at the top of the screen in the navbar were nice to use to navigate between the different steps of creating content, they felt like they were in the wrong order at times
- Website felt fast and responsive to the user in most cases
- None of the pages are responsive enough to be used on mobile devices/smaller screens

I Interviews from round 2

A second round of interviews was organised in the last sprint, after development was finished. While we were still open for suggestions or changes we were mostly looking for a confirmation that this system was in fact more usable than the original.

I.0.1 Questions before tasks

These questions were asked to get a general impression of our demographic.

- Age
- Study and/or job
- Technical background

I.0.2 Questions after tasks

After the tasks we asked some questions to get the actual feedback from the tasks. It's important to note that we also took notes during the tasks to learn directly from the interactions with the system.

- During the steps of the tasks, did anything feel slow or hard to figure out?
- Did the general workflow feel correct or did you have to move back between the steps a lot?
- Do you have any notes about the layout or the looks of the system?
- Are there features that you feel like you're missing?

I.0.3 Tasks

The following list are the tasks that we gave to the participants. The list includes multiple tasks, not all participants did all tasks, most did 1 or 2.

- 1
  - Upload specific files
  - Create a track with the new content
  - Schedule the track
  - Add any event to it
  - Change the scheduling to not all screens on a location
- 2
  - Upload specific files
  - Create 2 tracks with any content
  - Have a track include another track
  - Add an event to the main track
  - Assign the track in the schedule
- 3
  - Remove the track from the schedule
  - Add an event to the track
  - Re-add the track to the schedule
- 4
  - Find 2 tracks in the schedule, name them

I.1 Interview 1

I.1.1 Demographic questions

- Age: 25
- Job/Study: MBO pedagogiek
- Technical background: social media, regular computer use
- Participated before as candidate 7

I.2 Tasks

Task performed: 3, 4

Since this participant did the exact same tasks as the first time, it was quite a nice way to see the improvements. There was one bug but it turned out fine.

I.3 After tasks questions

- Schedule and the removal of playlists were seen as an improvement
- Workflow was fine
- Font could be hard to read in darkmode
- No suggestions

I.4 Interview 2

I.4.1 Demographic questions

- Age: 21
- Job/Study: TCS
- Technical background: TCS
- Participated before as candidate 8

I.5 Tasks

Task performed: 1,2,3,4

We were able to add all the tasks as the time for each task was drastically improved in the new system.

I.6 After tasks questions

- Speed of the system was fine. Some errors in localization
- The workflow was experienced as far better
- No
- An authorization/authentication system could be added

I.7 Interview 3

I.7.1 Demographic questions

- Age: 18
- Job/Study: Mechatronics MBO / Store employee
- Technical background: regular computer use
- Participated before as candidate 1

I.8 Tasks

Task performed: 1,2,3

We did most of the tasks as we went through them quickly. The participant continued to say throughout the process that it was much better.

I.9 After tasks questions

- Better, but the primary orange colour is ugly
- Workflow was fine
- No
- No suggestions

I.10 Interview 4

I.10.1 Demographic questions

- Age: 19
- Job/Study: Applied Physics
- Some python coding, but nothin major

I.11 Tasks

Task performed: 1,2,3,4

All tasks were performed and there were no general remarks during the interview.

I.12 After tasks questions

- It was good, uploading content was experienced as slow but this might have been the network.
- No
- No
- No

I.13 Interview 5

I.13.1 Demographic questions

- Age: 21
- Job/Study: International Business Administration
- Technical interest in coding but never started, regular computer use

I.14 Tasks

Task performed: 1,2,3,4

All tasks performed, not much complaining, except editing a track when published seemed weird for them, but it followed the requirements from the client.

I.15 After tasks questions

- Publish buttons can still hide important components
- Font could be hard to read at times
- Good
- No

J Interview summary from round 2

This interview round was organised after the 2 development sprints were completed. We tried to get back to the original interviewers, but since we had a shorter time frame we were unable to reach all of them. In the end we interviewed 3 participants from the old interviews and 2 new ones. This will only consist out of a short summary of all the feedback we received on the new system.

All of the major issues from the last interviews were addressed. The schedule was re-designed and tracks and playlists were merged. The feedback from everyone on this was positive. All participants were able to complete their tasks, where they were not able to do this in round 1. Some participants even said that they noticed the changes that were made that they suggested, which was fun. Some suggestions were still made:

- Publish button can move and/or hide components.
- Localization seemed wrong in some places.
- Wrong orange colour
- A permission system might be nice

- Font is still small/hard to read