

Graduation Report

S. Dior - graduation student HBO-IT Software Engineering

University:

Saxion University of Applied Science

Education:

HBO-IT Software Engineering, Bachelor degree

Company:

Ambient Intelligence (Aml) is a lectorate of Saxion University of Applied Sciences

Student:

Stanislav Dior - 411349

Table of Contents

1. Introduction	3
1.1. Graduation	3
1.2. Graduation Company	4
1.3. Project Approach Tool	4
1.4. Reading Guide	7
2. Assignment	9
2.1. Problem Definition	9
2.2. Problem Analysis	9
2.3. Research Questions	11
3. Approach	12
3.1. Project communication	12
3.2. Agile Development	12
3.3. Project management tool	13
3.4. Cloud for codebase	14
4. Requirements	15
4.1. Time frames	15
Functional	15
Non-Functional	16
4.2. Time planning tool	16
Functional	16
Non-Functional	17
5. Functional Design	18
5.1. Functional design: Time Frames	18
5.2. Functional design: Planning Tool	20
6. Technical Design	28
6.1. Technical design: Time Frames	28
6.2. Technical design: Planning Tool	30
7. Realization	33
7.1. Realization: Refactoring	33
Dependencies	33
Modules	34
Routing	36
7.2. Realization: Time Frames	38
7.3. Realization: Planning Tool	41

8. Evaluation	44
8.1. How to create an activity planning tool that could easily be used by the students?	44
8.2. How to improve the current codebase of the front-end application?	44
9. Recommendations	45
10. Reflection	46
11. References	48

1. Introduction

This chapter brings light to the context of the graduation, such as company information and the description of the project's initial version.

1.1. Graduation

As defined by Saxion University of Applied Science, the HBO-IT bachelor graduation is a final five month long student project, provided by a company. The project realization process is supervised by a company mentor as well as by a mentor assigned by the university.

This graduation is taking place from September 2021 and lasts till February 2022. The goal of this period of time is to prove the student's ability to work under professional conditions and achieve certain results. The student will be assessed by the Saxion Examination Committee including graduation mentors for the following criterias:

- Professional craftsmanship
- Research skills
- Communicative skills
- Learning skills

The process involved several people as the stakeholders. They are highly involved in the outcome of the graduation from the business and education sides. The stakeholders' contacts are below:

Role	Graduation student/employee
Fullname	Stanislav Dior
Tel.	+49 176 64462300
E-mail	411349@student.saxion.nl

Role	Company Supervisor
Fullname	Danny Plass
E-mail	d.plass@saxion.nl

Role	Graduation Supervisor
Fullname	Eelco Jannink
E-mail	e.h.a.jannink@saxion.nl

1.2. Graduation Company

Ambient Intelligence (Aml) is a lectorate of Saxion University of Applied Sciences. Aml focuses on the “smart world”: equipping everyday things with smart, innovative technology. Concepts such as Augmented and Virtual Reality, Machine Learning, Artificial Intelligence and Internet of Things often play a role in this. Aml connects approximately 24 permanent employees of various educational levels and backgrounds. Many of them are also active as teachers at HBO-ICT, CMGT and other Saxion courses. Aml often collaborates with students from these programs to increase students' research skills. In this way they are better prepared for the professional business world and possible (master's) follow-up studies. Aml projects are available to students as Smart Solutions Semester and (graduation) internships. (Saxion, 2021). More information regarding Aml is available on the official website:

<https://www.saxion.edu/business-and-research/research/smart-industry/ambient-intelligence>

One of Aml's projects is Project Approach Tool, which is a platform being in active development and managed directly by Dr. Danny Plass - Oude Bos (Associate professor). The graduation assignment is provided in a scope of this project.

1.3. Project Approach Tool

Project Approach Tool (PAT) is a platform built by Saxion University of Applied Science Ambient Intelligence lectorate, in which students can design a project approach together in a visual environment. The PAT is primarily intended as a brainstorming tool so that students can create a concept for their project approach alone or together.

The platform, unlike the other solutions, brings teachers and students together to build a professional mindset around project planning in real-time.

The current version of the platform can be accessed via the link below:

<http://projectapproachtool.nl>

Note: During the login process check your Spam email folder in case your email client finds it suspicious.

The project focuses on teaching students, helping them to construct a methodical approach to solving their problems. The platform allows students to create project phases, as well as add activity cards to them. Each card follows a specific methodical approach, which differs in color and icons in the application, helping users to easily

navigate planned activities. The following screenshot shows the current *user interface (UI)*:

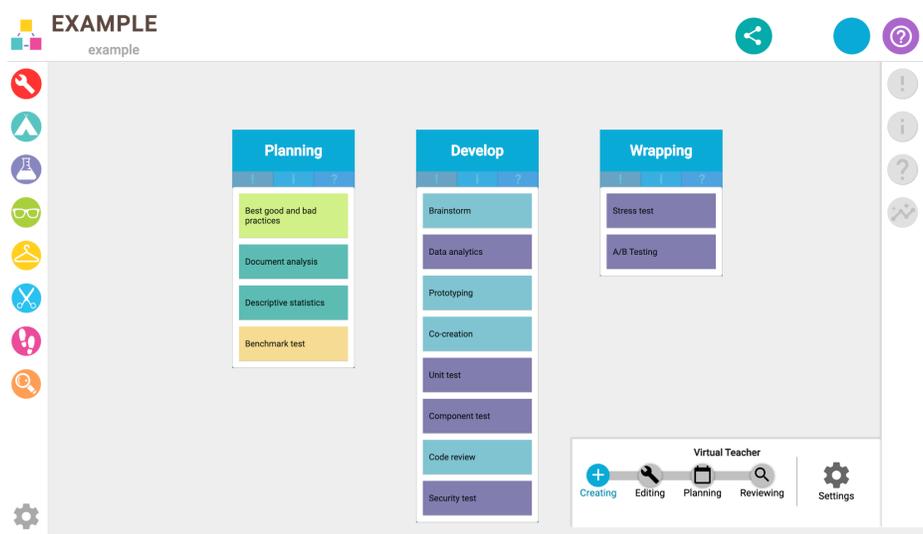


Figure 1.3.a. Project Approach Tool dashboard with phases.

This application allows a user to select and / or create the tasks necessary for the project and receive feedback from the teacher, who has collected in one place information about the tasks for the current project.

To create or receive valid feedback on a project approach, students and teachers use Virtual Teacher (VT). The VT is a set of rules that a project needs to adhere to (for example required Research, a Deliverables activity, etc).

The actions that students and teachers could emit to process a project planning feedback using a Virtual Teacher (VT) shown on the use case diagram below:

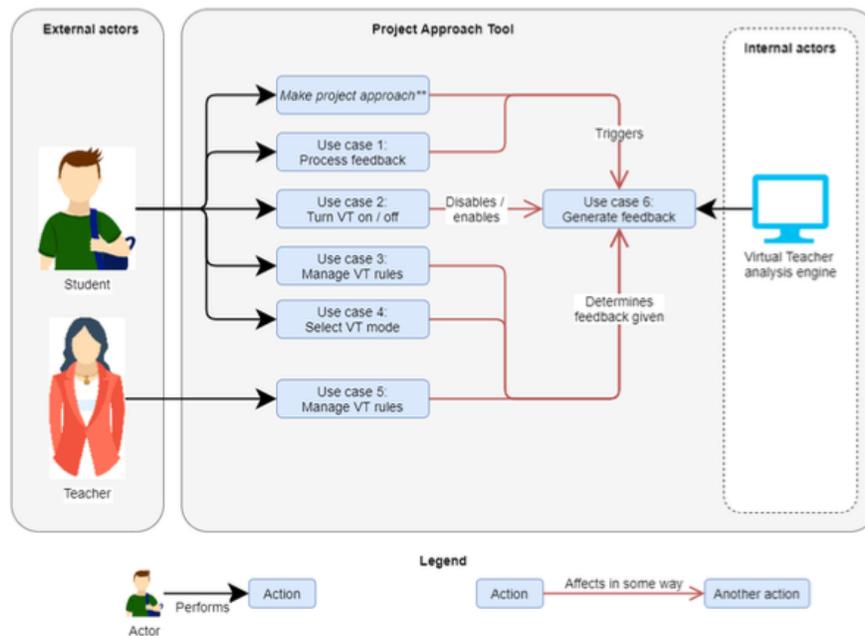


Figure 1.3.b. PAT use case diagram.

The PAT has been developed by three students before this particular graduation started. Every student had a different assignment assigned to him/her in the scope of the platform codebase. It leads to some given constraints regarding certain architecture decisions and the technology stack used for this project. The platform is built using a popular *MEAN* technological stack. The MEAN stack stands for:

- MongoDB
- Express
- Angular
- Node

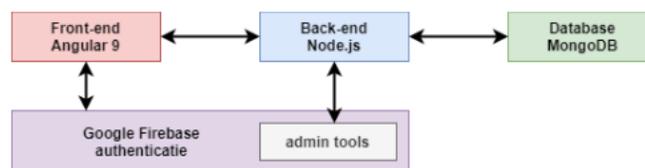


Figure 1.3.c. The MEAN application tech stack.

It would be correct to split the tech stack into two groups, namely the client side (browser) application and the server application, as industry calls them *front-end* and *back-end* respectively:

- Front-end:
 - Node.js
 - Angular

- Back-end
 - Node.js
 - Express
 - MongoDB

As the stack tree indicates both front-end and back-end applications are developed using Node.js, which could be installed on a computer to run JavaScript in an Operating System (OS) environment.

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser.

(Wikipedia, 2021).

The PAT's front-end is built using Angular framework. Based on a Monocube blog post, Angular is one of the most popular frameworks for building complex client applications at any scale. (Monocube, 2021).

The PAT's back-end is built using the Express framework and MongoDB database. Express is the most popular Node.js server framework for developing RESTful APIs. MongoDB is a NoSQL document database with the scalability and flexibility that are essential for the querying and indexing.

NoSQL databases (aka "not only SQL") are non-tabular databases and store data differently than relational tables. NoSQL databases come in a variety of types based on their data model. The main types are document, key-value, wide-column, and graph. They provide flexible schemas and scale easily with large amounts of data and high user loads.

(MongoDB, 2021).

Each tech technology has a large ecosystem of professionals and different solutions to almost any problem a developer could face during a feature realization.

1.4. Reading Guide

The content of this document reports to the reader every stage of the graduation, therefore includes problem description, process details, development choices, etc. Some parts of the report might be not relevant for specific types of readers due to organizational materials or technical details.

To read about business problem, organizational details and the developed solution, it's advised to read the following group of chapters, because they do not encompass any technically related information:

- 2. Assignment
- 3. Approach
- 4. Requirements
- 5. Functional Design
- 8. Evaluation

For the readers interested in technical choices and in the development details, the following reading scenario is useful:

- 2. Assignment
- 4. Requirements
- 6. Technical Design
- 7. Realization
- 8. Evaluation
- 9. Recommendations

In the end of the report the *10. Reflection* chapter informs about writer's personal feedback to the graduation company and the process in general, as well as describes the coherence between HBO-IT level 3 professional competences and the work that was done. This information is useful for the graduation assessors.

For better reading experience the research questions are marked with a special icon:



Research question?

When a reader sees this format he/she can expect a following structure below this icon:

- Research question next to the icon
- Research method
- Results
- Conclusions

2. Assignment

This chapter informs a reader about the context of the main assignment of the graduation. It describes the problem statement, problem analysis and defines a list of research questions.

2.1. Problem Definition

Context:

After the target users of the PAT, which are students, have defined phases and added their activities to each phase, they want to start planning these tasks to have an overview of the required amount of time to complete the activities. The students want to be able to do it on the platform. It allows them to keep the essential information regarding activities, time estimation and the order of tasks to be executed in one place. So the students can dedicate certain time to every task, visualize all planned activities in a single view to avoid adding activities beyond the project's time scope.

Issue:

The current version of PAT doesn't include any functionality regarding time dedication and a planning step for the added project activities in front-end, nor in back-end applications.

Relevance:

There are many ways to let the users plan their activities using time frames. At this point, it's not clear which approach would suit them the most. (Relevance)

Objective:

This document investigates the most suitable way to help and train students to create a solid easy-to-use activities plan.

2.2. Problem Analysis

The purpose of the PAT is to provide students with an environment in which they can design an approach to their activities. In doing so, we want them to think about their project plan and how they will implement and follow-up the activities they want to carry out. However, this goal is hindered in the current version by one major problem, namely: an absence of a time planning step for the activities the students have defined. Current version of the platform doesn't allow users to specify the consecutive order and the time frames of the activities to be executed.

Currently the students use third-party tools to create a visual representation of their project planning. This approach leads to inconsistency between using the same tools for different teams or even different projects made by a single team of students.

The planning tool should help users to plan the research methods and other activities from the very first day of the project up to its deadline in one place. To add this functionality the codebase of the current platform needs to be extended.

In addition to the planning tool, there is a secondary assignment to address a number of issues in the current codebase. This assignment is classified as less complex and therefore serves as a supplement to the main assignment.

Planning Tool

The primary problem of the graduation assignment is a lack of a planning tool, which allows users to plan their activities on a time scale. This functionality should help the students to adjust the activities and have an insight on the amount of time each activity is estimated to occupy as well as set the activities' in a consecutive order.

Refactoring

The secondary problem of the graduation assignment is the current front-end code structure. The project uses a front-end framework, namely Angular. The framework provides a large set of tools out of the box, such as modular distribution, lazy loading and many others, which have not been taken into account in the previous versions. Carrying out this problem will increase the efficiency of work on this project, as well as raise the quality of the code and the general application structure.

Current codebase structure of the front-end application creates certain complexities when adding new features to the application. Therefore, it is recommended to start with a refactoring in the first place and after the front-end application adheres to the good practices, development of a planning tool should take place.

2.3. Research Questions

Main question:

1. How to create an activity planning tool that could easily be used by the students?

Sub questions:

- 1.1. What are the approaches for activities' time planning out there?
- 1.2. What is the most convenient way to input a date range in project planning?
- 1.3. What time units need to be used for student project planning?
- 1.4. What do students use to create Gantt charts?
- 1.5. Is there any planning tool library available for an Angular application?
- 1.6. What are the common user experience standards for a planning tool?
- 1.7. How to add time frames functionality to the activity cards?

Secondary main question:

2. How to improve the current codebase of the front-end application?

Secondary sub questions:

- 2.1. What are the best practices for an Angular application code structure?
- 2.2. Which parts of the current application could and should be refactored?

3. Approach

The description of the approach to the implementation of this project can be divided into several parts:

- Project communication
- Agile development
- Project management tool
- Cloud for codebase

3.1. Project communication

The project communication was established via Microsoft Teams application due to the limited physical communication caused by COVID-19. The meetings with the project manager (company supervisor) were planned weekly. During these meetings the priority of the planned tasks and key decisions were discussed and re-estimated, which helped to keep focus on the crucial functionality that had to be implemented.

3.2. Agile Development

The project time scope is five months. The projects this scale require a management tool to keep track of the progress and the undone, ongoing and completed tasks.

Kanban is a popular framework used to implement agile and DevOps software development. It requires real-time communication of capacity and full transparency of work. Work items are represented visually on a kanban board, allowing team members to see the state of every piece of work at any time. A kanban board is an agile project management tool designed to help visualize work, limit work-in-progress, and maximize efficiency (or flow). It can help both agile and DevOps teams establish order in their daily work. Kanban boards use cards, columns, and continuous improvement to help developers commit to the right amount of work. (Rehkopf, 2018).

Kanban and Scrum are the two most popular agile frameworks, which help to organize the workflow and the process is taken to carry out the assignment and adapt to the changes that usually appear in the software development process. (O'Malley, 2021).

The Kanban management approach provides much more flexibility and no role dependency as Scrum does. Take a look at table below which compares “Scrum” with Kanban:

	Scrum	Kanban
Cadence	Regular fixed length sprints (ie, 2 weeks)	Continuous flow
Release methodology	At the end of each sprint if approved by the product owner	Continuous delivery or at the team's discretion
Roles	Product owner, scrum master, development team	No existing roles. Some teams enlist the help of an agile coach.
Key metrics	Velocity	Cycle time
Change philosophy	Teams should strive to not make changes to the sprint forecast during the sprint. Doing so compromises learning around estimation.	Change can happen at any time

Table 3.2.a. Comparing “Scrum” and “Kanban”

For this project “Kanban” was chosen over “Scrum” as a development management method. The reasons for this decision are:

- Kanban doesn’t include any extra roles to the team structure.
- Kanban flow is much easier to manage if the development team consists of one developer.

3.3. Project management tool

As a project management tool “Notion” was used. Notion is a desktop and web application that helps to organize documentation and allows to create interactive tables with sortable (by priority) task cards. On the screenshot down below you can see how the project’s “Kanban” looks like on Notion:

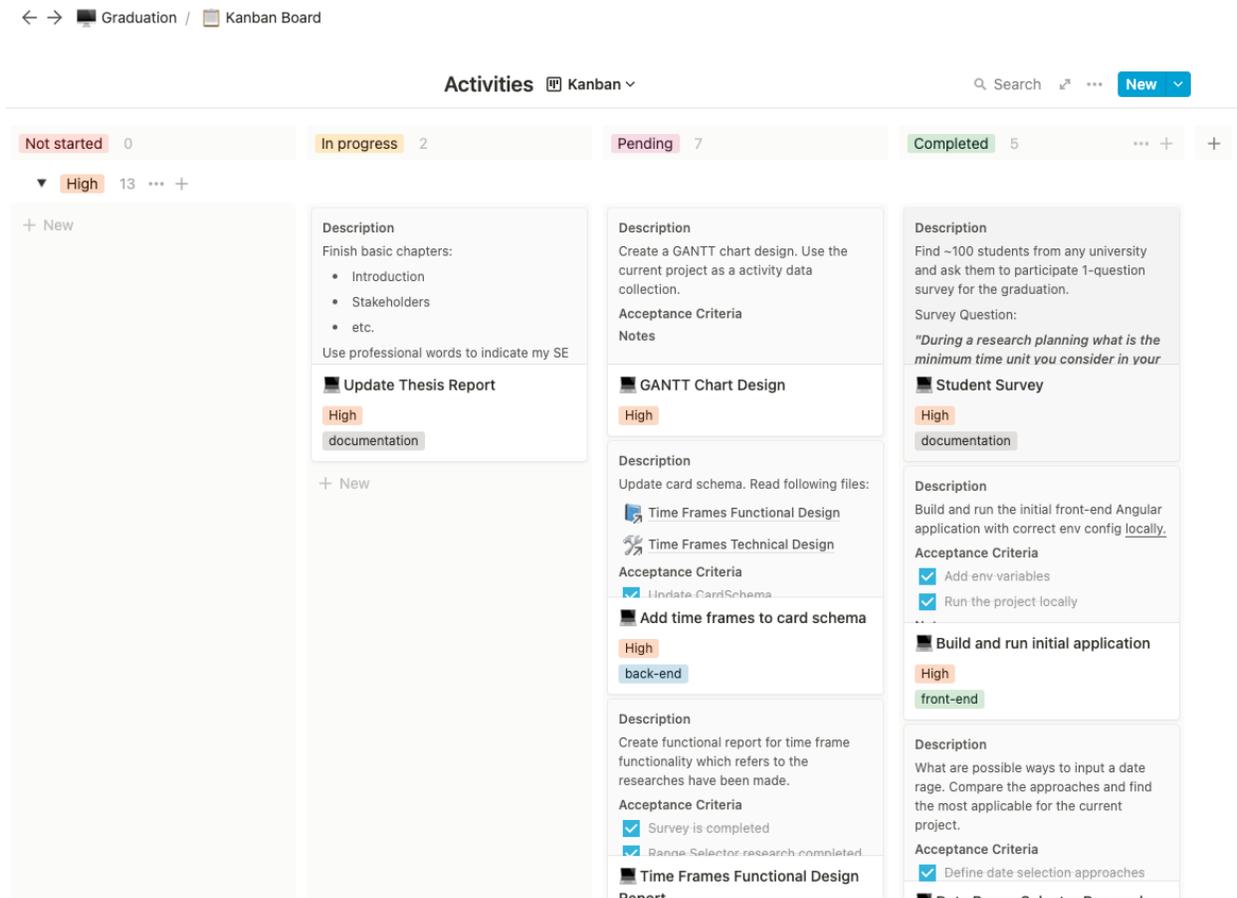


Figure 3.4.a. The project's "Kanban" board.

To find more information about Notion and its pricing plans, follow the link below:

<https://www.notion.so/product>

3.4. Cloud for codebase

The PAT commitment history and current codebase version is stored in "GitHub". GitHub is a code hosting platform for version control and collaboration. This tool allows us to keep track of the changes that have been made to the project's code tree using GIT, which is a version control system. A link below navigates to the project repository:

<https://github.com/SaxionAMI/Project-Approach-Tool>

Note: the repository is only accessible for SaxionAMI employees

4. Requirements

Every requirement on the tables below is approved by the project manager (company supervisor). The requirements were defined by a research, which included several interview sessions with following educational institutions and has been done by previous students who worked on the project in the past semesters:

- Saxion HBO-ICT
- HAN & Hogeschool Utrecht
- Hogeschool van Amsterdam

Based on the communication with the project manager, it was defined that the graduation main assignment splits into two major parts, which have to be carried out by an order defined in this chapter. These two parts are:

1. Time frames
2. Time planning tool

Initial version of the Project Approach Tool doesn't include any activity time boxes, therefore it has to be carried out separately and implemented beforehand. The requirements for each subject are splitted into functional and non-functional. The prioritization is defined by the MoSCoW system (Must, Should, Could, Would not).

The priority of each requirement is assigned with a Minimum Viable Product (MBP) in mind. A MVP requires to include only the essential / core features, so that the product could be used by users to solve their basic problems.

The functional requirements are defined in a user story format:
As a <role>, I want to <goal>, so I can <reason>

4.1. Time frames

Functional

Requirement	Priority (MoSCoW)
As a user, I want to add time frames to an activity card in its detail dialog, so I can define a time box for this activity.	M
As a user, I want to select the start day and the end day for the activity card, so I can count the duration in days.	M
As a user, I don't want to leave empty date inputs, so the	S

card has default start- / end dates values.	
As a user, I want to receive a warning if the end date is earlier than the start date, so I can be sure the time frame is always valid.	S
As a user, I want to see the card list ordered by activity starting dates, so I can get information regarding the activity execution order without opening up the card details.	C

Table 4.1.a. Time frames functional requirements

Non-Functional

Requirement	Priority (MoSCoW)
The system should provide a human readable error message if dates are not correct.	M
The time frames defined by the user should be stored in the database.	M
The time frames are editable.	S

Table 4.1.b. Time frames non-functional requirements

4.2. Time planning tool

Functional

Requirement	Priority (MoSCoW)
As a student, I want to see my method cards on the planning step, so I can start planning immediately.	M
As a student, I want to select a date range for the project, so I can place my planning cards within that range.	M
As a student, I want to drag a card without changing its duration, so I can move it without any required time adjustment.	S
As a student, I want to drag the card's sides, so I can adjust start date and end date accordingly.	S

As a student, I want the cards to be colored the same as the methods (tasks), so I can differentiate them easily	S
As a student, I want to share my planning in view only mode with other students and teachers, so I can get some feedback.	S
As a user, I want to see holidays standing out, so I get informed by the system about this non-working days	C
As a student, I want to see the duration in days in each card, so I don't need to count it myself.	C
As a user, I want to be able to write comments to specific components in the planning view, so it can be improved/changed.	W

Table 4.2.a. Time planning functional requirements

Non-Functional

Requirement	Priority (MoSCoW)
All planning items are within the selected date range and cannot be moved beyond.	M
The planning items dragging adheres to the common user experience standards.	M
The planning items are colorful according to the corresponding activity color.	S

Table 4.2.b. Time planning non-functional requirements

5. Functional Design

In this chapter two functional designs are described, namely: Time frames and Planning Tool. It

5.1. Functional design: Time Frames

The activity cards added by students don't have time boundaries. The students are not able to plan, adjust and visually see the methods' time frames in the current platform. Absence of an ability to add and edit this time information forces the students to keep the time estimation in a separate document or proceeding a project without this crucial to every planning deadline information. This problem leads to the following statement:

The solution to this problem is extending method card entry to include time boundaries information, which could be added during creation of a new method card and be editable. This functionality will allow students to plan and organize their work by time ranges and allow them to give time estimation to each (research) activity.

To understand, what are possible ways for the student to input the date range for an activity the following research was done:



1.2. What is the most convenient way to input a date range for project planning?

Method:

Workshop strategy (DOT Framework). Multicriteria decision making method was used to gain the required information.

Results:

One Date + Duration

Pros:

1. If the starting date shifts the "end" boundary shifts automatically with it.
2. People are used to thinking in duration and not in dates. Therefore, users don't need to calculate To date, so they stay focused on the actual time spending estimation.
3. Easy programmatic access to the method duration.

Cons:

1. Not clear which units are suitable for the duration field (seconds, minutes, hours, days).
2. Changing From could cause undesired time overlap with other methods in the same planning.
3. Difficult to search for overlaps with other planning items programmatically.

Two Dates:

Pros:

1. Recognisable way for a user to provide time frames.
2. The From and To dates are independent.
3. Easy to add time units' precision like hours and minutes.
4. Variety of existing solutions for date and time range selection.
5. Easy to search for overlaps with other planning items programmatically.

Cons:

1. Shifting the starting date of a method requires the user to change both From and To dates.
2. Need to validate for a negative dates' range, when From date comes after To.

Conclusion:

In the current project the option with two dates seems more suitable, since the time frames should be easily adjustable and editable in both manual and programmatic ways. This approach makes it easier to find planning overlaps and most users are familiar with two date ranges.

More detailed research report can be found in the attachment "[research-date-range-approach.pdf](#)".



1.3. What time units need to be used for student project planning?

Method:

Field strategy (DOT Framework), namely a quantitative survey to understand what time units (days, weeks, months) the students currently use when creating a school project planning. The student network was used to distribute the survey.

Results:

The figure below shows that 44 students prefer to use *Days* as a time measurement unit when estimating a research. The rest of students choose *Hours* as base units for time-boxing their research effort:

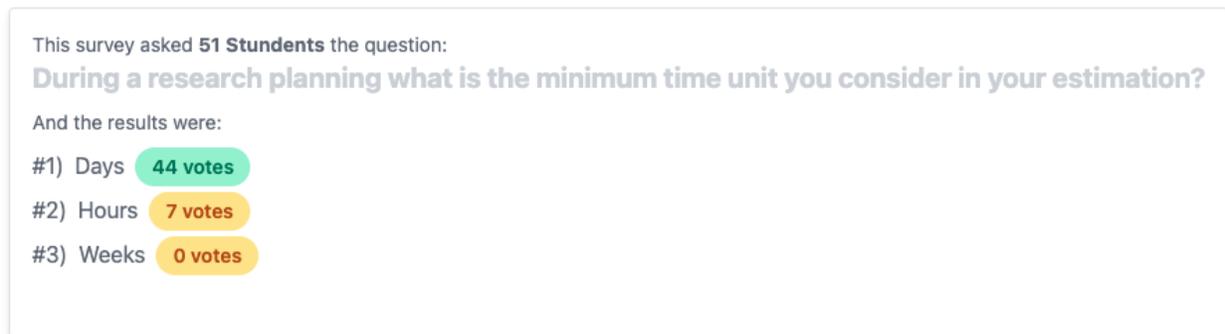


Figure 5.1.a. Time unit research survey results.

Conclusion:

Based on the survey results *Days* are the most widely used units for time-framing research chunks for the scoped majority of students.

More detailed research report can be found in the attachment "*research-time-unit.pdf*".

5.2. Functional design: Planning Tool

The students don't have a visual tool to build up a planning process. At this moment a time frame could be assigned to each method card but there is no way to visually organize and interact with it.



1.1. What are the approaches for activities' time planning out there?

Method:

Library strategy (DOT Framework)

Results:

PERT and Gantt charts are visualization tools that are often used in project management. Both of these charts are used to schedule tasks, monitor and manage the tasks required to complete a project. The difference between the two is that a PERT chart is a kind of network chart, while a Gantt chart is a bar chart.

Gantt chart	PERT chart
Gantt chart is defined as the bar chart.	PERT chart is similar to a network diagram.
Gantt chart was developed by Henry L. Gantt.	PERT chart was developed by the United States navy.
Gantt chart is often used for Small Projects.	PERT chart can be used for large and complex Projects
Gantt chart focuses on the time required to complete a task.	PERT chart focuses on the dependency of relationships.
Gantt chart is simpler and more straightforward.	PERT charts could be sometimes confusing and complex but can be used for visualizing a critical path.

Figure 5.2.a. Table to compare Gantt and PERT charts.

Conclusion:

A Gantt chart is the chart that will be used in this project as a planning tool. The students' projects, which PAT was built for, aren't classified as large and complex projects. A PERT charts main focus to full-fill the dependency graph for big projects where every delayed activity could cause global process changes. A Gantt chart full-fills all requirements set to the planning tool and it is much easier to develop.

More detailed research report can be found in the attachment "*research-planning-tool.pdf*".

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and on the top is a suitable time scale. Each activity is represented by a horizontal bar; the position and length (width) of the bar reflects the start date, duration and end date of the activity (Gantt.com, 2021). This allows you to see at a glance:

- What the various activities are
- When each activity begins and ends
- How long each activity is scheduled to be take in a matter of time
- Where activities overlap with other activities, and how much

- Parallel activities
- The start and end date of the whole project

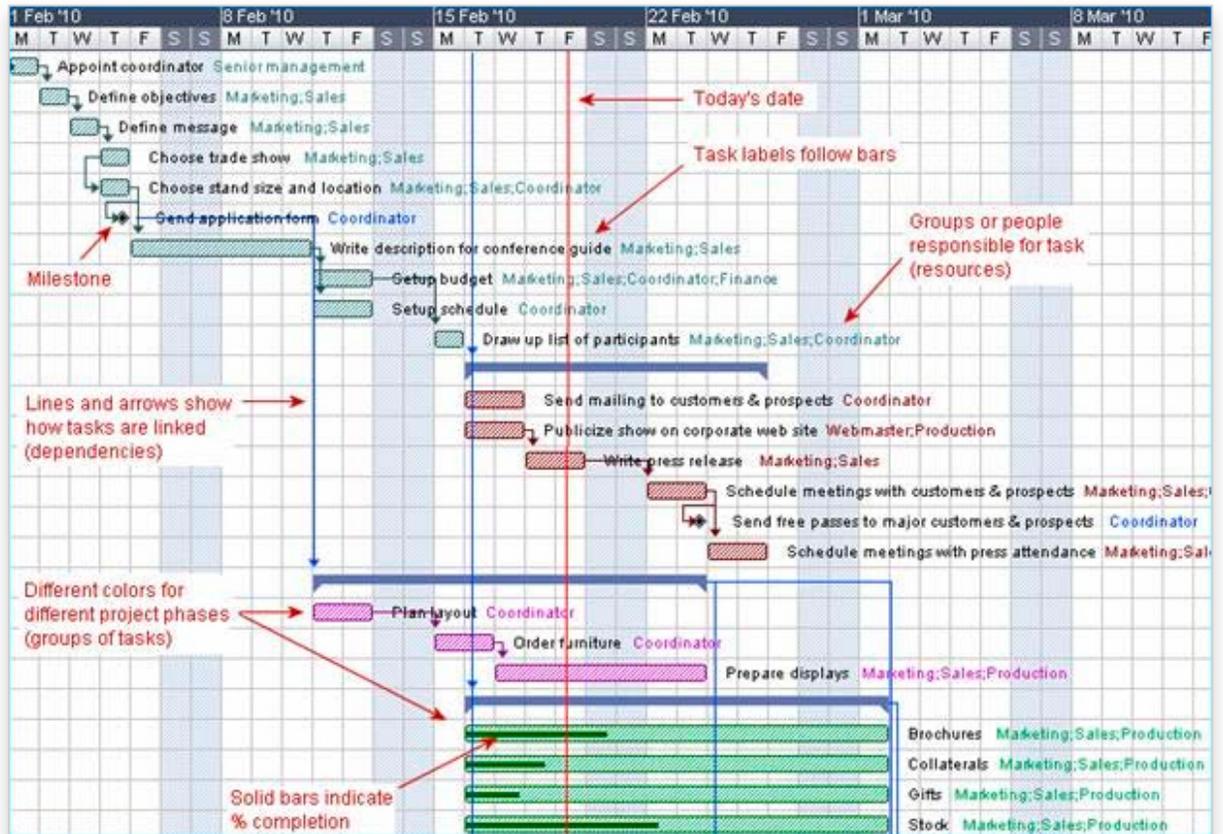


Figure 5.2.b. An example GANTT chart.

A new page with a GANTT chart helps users to see the methods in a consecutive order and provides enough interaction functionality to adjust the time frames for each method. The interaction is created by enabling a drag-and-drop functionality, which allows users to stretch each side of this method to change its start- and -end date, as well as shifting the method card itself by dragging its center horizontally.

The GANTT chart is on a separate page, where the students can concentrate to set up a correct plan. This approach adheres to the functional and non-functional requirements that were defined in the Requirements chapter.

To define the key features of Gantt chart to be included in the PAT platform, a new research question was made:



1.4. What do students use to create Gantt charts?

Method:

Field strategy (DOT Framework). The qualitative survey was done using a network of several students, who answered several open questions.

Results:

To answer this question eight Saxion University of Applied Science students were interviewed. All of the students are Saxion HBO-IT students from first to fourth year of their study.

As a result the following tools are currently used by the students to create GANTT charts:

- Google Sheets (4 students)
- Microsoft Office Excel (3 students)
- TeamGantt (1 student)

The students described the benefits and the drawbacks of the tool they use.

Conclusion:

To summarize the research results, the list below indicates the key features users want a GANTT chart planning tool to include:

1. Be able to share it with a teacher easily by sharing a link, giving an access or being able to export a GANTT chart in a common format (pdf, jpg, etc.).
2. Be able to interact with the GANTT chart using a mouse only. For example drag sides of the tasks to change start/end date and/or just dragging the task itself to change its starting date.
3. A calendar grid view with the dates (just like in any calendar application, f.g. Google Calendar week view).
4. Online collaboration with the team members. At least see each others' updates and changes.
5. Minimum configuration. Some tools require more time to configure than to actually get a use of them.

More detailed research report can be found in the attachment “research-how-students-create-gantt-charts.pdf”.

These five key features the potential users define as the most important. Most of the above described list items are present in the product requirements in the Requirements chapter.

To create a UI design for the GANTT chart view several iterative sessions were made with the project manager (company supervisor). The iterations cycles included there repetitive steps:

1. Prototype
2. Feedback
3. Analyze
4. Go back to 1

The first prototype you can see on the figure below:

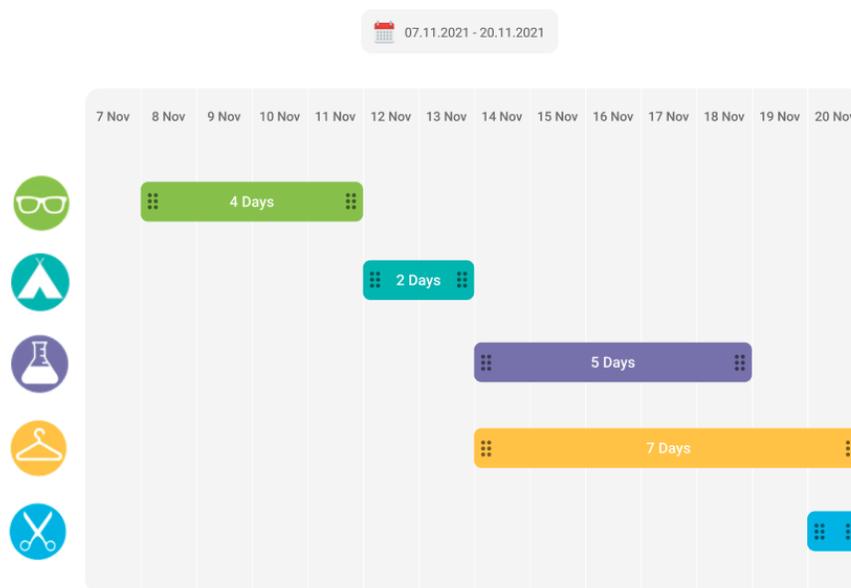


Figure 5.2.c. First GANTT chart design prototype

The feedback made a large impact on how the GANTT chart design was finally defined. Initial design didn't take into account some crucial points:

- The GANTT chart should have activity names on the left side, since multiple activities could be part of the same research method.
- Weekends should be displayed but disabled.

- Zoom in/out functionality to have a better insight view.

Although, some features defined in the first prototype remained approved:

- Dragging icons on the sides of each activity.
- Colorized activities based on the research method they belong to.
- Two interaction possibilities described below.

Dragging a side of a card. This dragging functionality allows a user to change start- (left side) or end date (right side as in the example Figure below). During the dragging process the opposite side remains on its initial position and does not change. When dragging a side of the card the duration is changing reactively without a dropping event with a neat vertical counter animation:

- If duration increases the number fades in and slides from top.
- If duration decreases the number fades out and slides to the bottom.

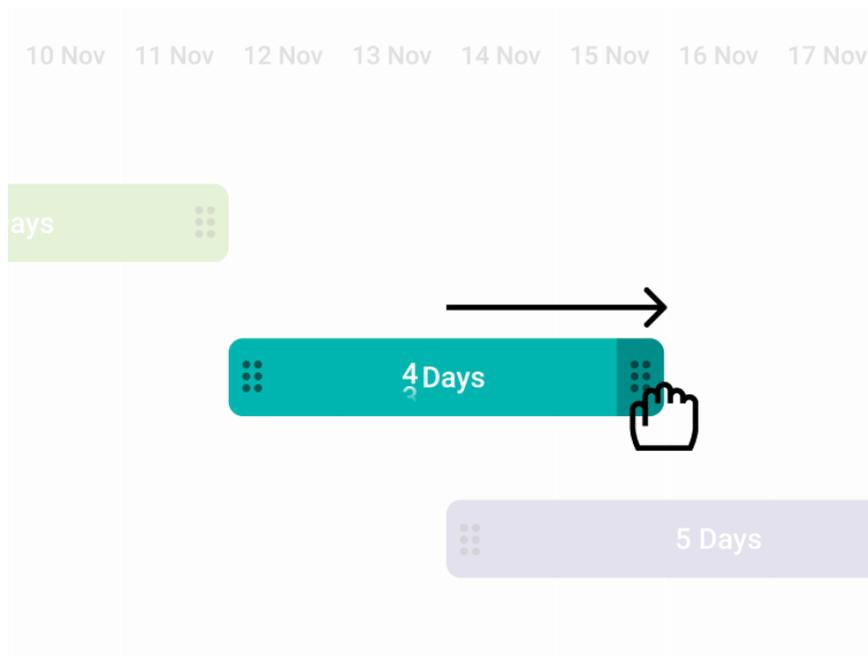


Figure 5.2.d. Side dragging interaction

Dragging body of a card. This dragging functionality allows the user to move the start- and end dates without changing duration of the method. A user can achieve this by clicking between the sides of a card and moving its cursor horizontally as shown in Figure below.

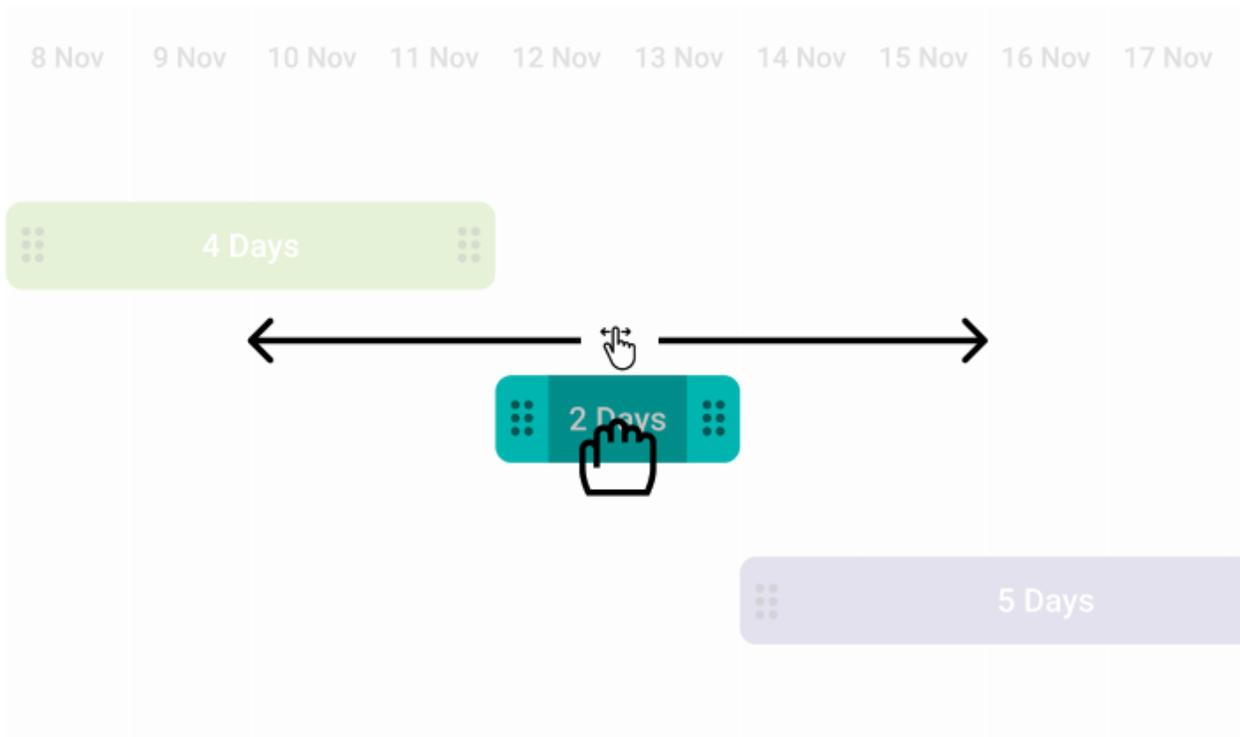


Figure 5.2.e. Card body dragging interaction

The final design was made after a series of repetitive feedback and feedback analysis. The final design covers all required features mentioned by the project manager. It has a standard but simplified version of a usual GANTT chart, since it has less configuration options due to the project time limitations. It includes activity names on the left panel, more extensive date view at the top of the chart and excludes weekends for better work week planning.

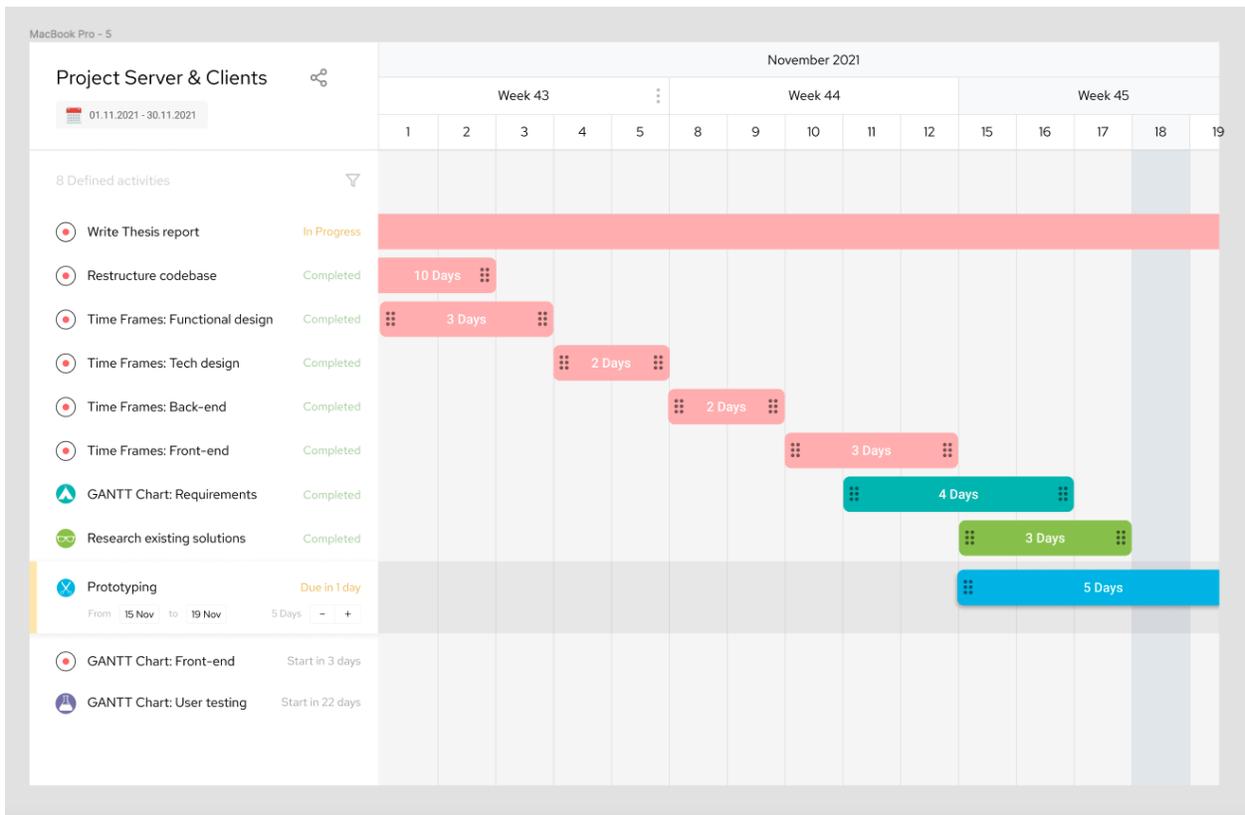


Figure 5.2.f. Final GANTT chart prototype

6. Technical Design

Based on the initial project codebase, the technological stack is given. The project uses MEAN stack, which refers to MongoDB, Express, Angular, Node.js. The technologies are explained in the 1.3. Project Approach Tool chapter.

6.1. Technical design: Time Frames

The server-side code, namely back-end code, keeps the business logic of the entire application and is responsible for data processing and database read & write communication.

The back-end code defines the models of the application. A model is a relational data structure which holds a description for its data properties such as keys, types, default values and requirement constraints as well as database manipulation set of functions (Schema).

The scope of back-end code for this feature encompasses:

1. Update a research method card model, namely *CardSchema*.
2. Add validations.

The card model *CardSchema* needs to be extended with two new data properties *startDate* and *endDate*. Both of the properties should be required with no default value and with type *Date*. Adding these properties and restarting the database will make sure that the next card creation will include these fields visible in the client application (front-end application).

Upon creating or updating a research method card the *startDate* and *endDate* data properties should be validated. The validation rules are following:

- *startDate* is not empty. Error: '*Start date is required to be set*'.
- *endDate* is not empty. Error: '*End date is required to be set*'.
- *startDate* value and *endDate* value are consecutive or the same. Error: '*Start and End dates must be consecutive or the same*'.

The front-end code is responsible for an interaction with a user by visualizing data and allowing the user to trigger certain data processing on the server using RESTful API in our case.

The scope of front-end code for this feature encompasses:

1. Update a research method card model.
2. Add date range picker input to the card creation form component.
3. Validate the user dates input. Display the errors below the date picker input.
4. Send filled out form data to the back-end.
5. Handle back-end validation errors by displaying them to the user. Display the errors below the date picker input.
6. Display the selected date ranges on the research method card components.

The current front-end application uses Angular Material as a common component library. The library includes a date picker with an ability to perform a date range selection. The screenshots of a range date picker component (Material Design, 2017) provided below.

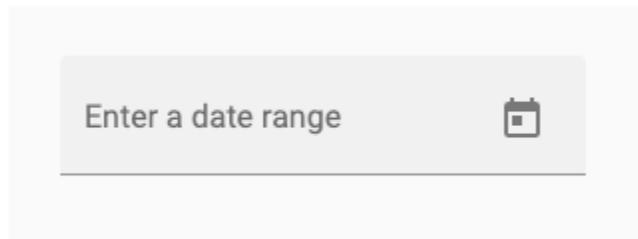


Figure 6.1.a. Date picker is not focused.

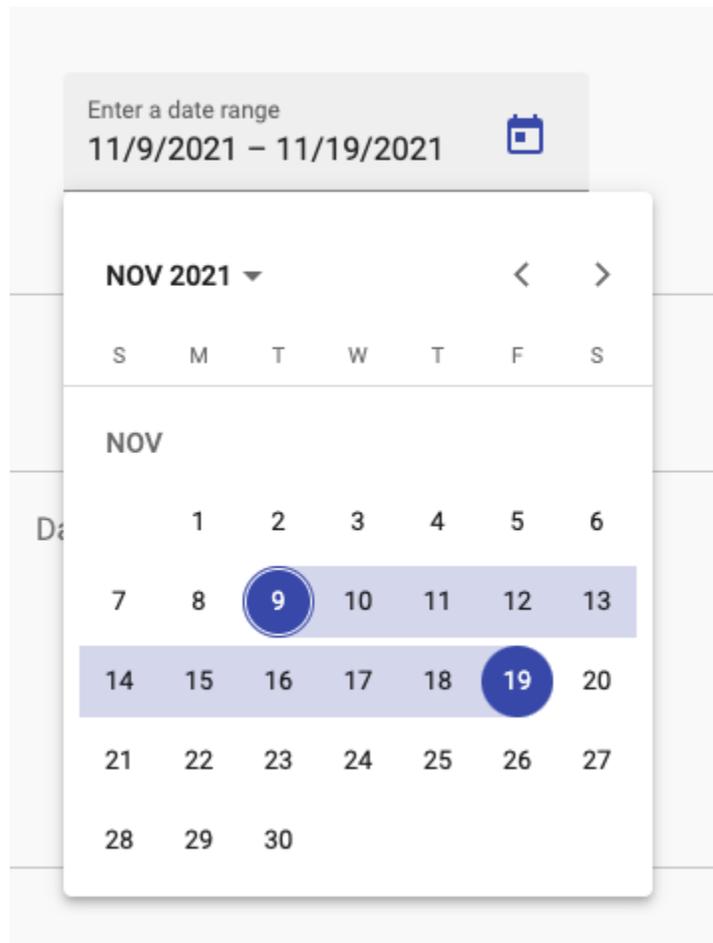


Figure 6.1.b. Date picker is in focus.

6.2. Technical design: Planning Tool

Creation of a GANTT chart view, which is selected for Project Approach Tool as a planning tool, requires a major feature implementation on the front-end side (Angular application).

This feature consists of:

- Creating a new page and adding this to the workspace router configuration.
- Creating a GANTT chart component.
- Gaining data from the server regarding the activities.
- Visualizing content in the GANTT chart component.

Current implementation of the API doesn't allow updating activities with just a request to cards (activities) controller, because the cards that a user creates do not have a schema model and are stored just as an array of entities inside a group. A

group is an array of cards created by a user and it is stored as a workspace property as shown on the class diagram below:

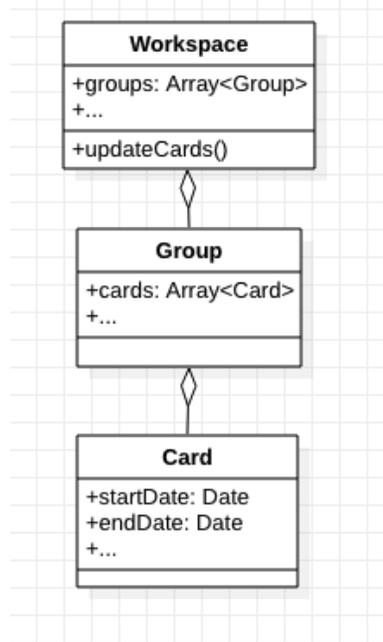


Figure 6.2.a. Workspace-card connection.

Due to this API structure to update a card the entire workspace has to be updated. Therefore an API controller and a route handler has to be to process this request. An activity (card) update process is shown on a sequence diagram below:

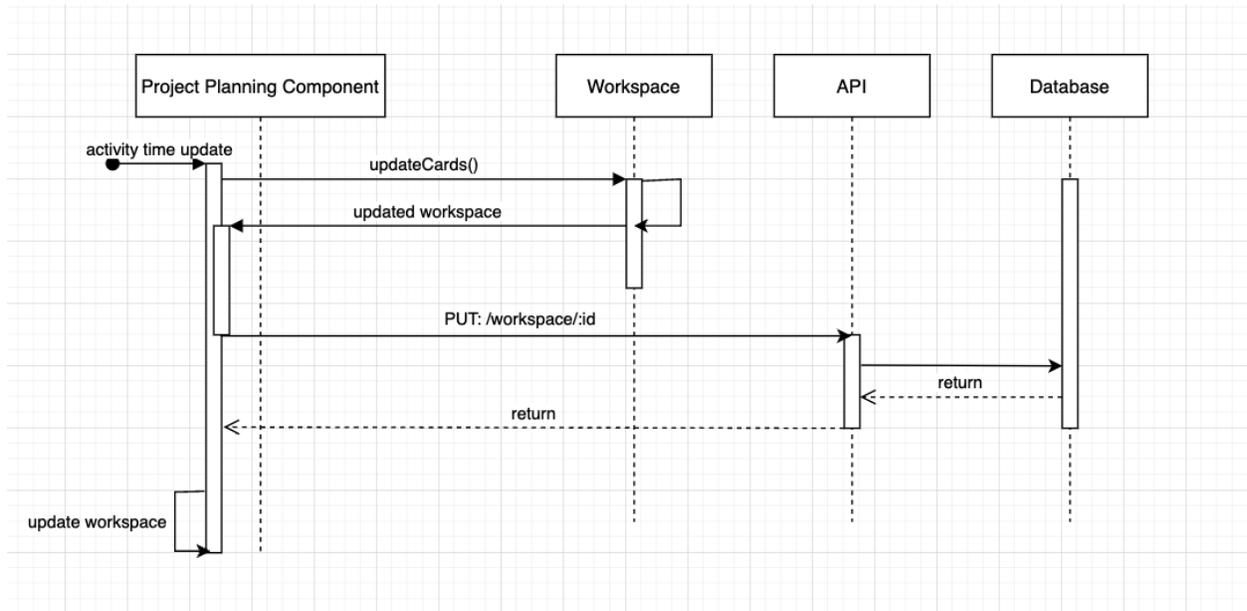


Figure 6.2.b. Update card(activity) sequence diagram.

To create a Gantt chart the 'devexpress' UI component library solution was used. The library has a low-level Gantt chart component that is easily customizable and adheres to the product requirements.

https://js.devexpress.com/Documentation/Guide/UI_Components/Gantt/Getting_Started_with_Gantt/

The library has a pricing free license for non-commercial projects. Since the current project is non-commercial, this library is a fit solution.

7. Realization

In this chapter the realization process is described. During this graduation project, there were few independent features:

1. Refactoring
2. Time Frames
3. Planning Tool (GANTT chart view)

Therefore, each part is described separately in the dedicated sub-chapter below.

7.1. Realization: Refactoring

Application Refactoring is the rewriting of one or more components of an application. Application refactoring is useful for extending the life of an application, improving its performance, adding needed new functionality and strengthening security. (VMWare, 2020).

Dependencies

Based on Patkos Csaba's article (Csaba, 2014), the refactoring process should start with dependencies cleaning, since dependencies are a backbone of the application, which the software relies on.

Therefore, I used a lab strategy research to perform a system test of the current project front-end codebase.

The potential risks discovered during the research:

- Some of the dependencies are outdated / deprecated.
- The project dependencies have 69 vulnerabilities (1 low, 33 moderate and 35 high).

Using `npm audit fix` command from the front-end project root directory, we achieved 13 vulnerabilities less with almost no effort, stating 56 vulnerabilities (1 low, 29 moderate and 26 high). Trying to resolve other vulnerabilities manually causes unstable functioning of the application and throwing incompatibility errors in the user's console.

Read `analyze-front-end-dependencies.pdf` in attached files for detailed information.

Modules

Based on Halodoc (professional development agency which uses Angular) blog article (Halodoc, 2021), the Angular best practices encompass many rules. Despite the adherence to the most Angular best practices, the current front-end codebase violates the essential modularity principle.

The initial PAT application was implemented by a single module responsibility approach. The approach means that every component of the project belongs to the root application module.

This approach works for small Angular applications or prototypes, because it does not require time investment for code splitting and defining domain driven design (DDD) architecture.

The drawbacks of this approach are:

- Difficult project scaling (adding new features).
- Performance issues.
- Failed Single-Responsibility Principle (SRP).
- Failed Don't Repeat Yourself principle (DRY).
- Failed Keep It Stupid and Simple principle (KISS).

Let's enlight these drawback one by one:

Difficult project scaling

When the requirements change and/or new features must be added, the developer is required to know and understand the entire system before being able to update the current codebase. This has a bigger impact as the team grows or the team is dynamic, because every developer becomes responsible for the entire application structured under a single module.

Single module responsibility could also lead to continued Git merge conflicts, which need to be carried out by developers.

Performance issues

Angular compiler builds projects on the server side and ships the bundle JavaScript file to the browser, which allows a developer to split the codebase into the lazy-loaded modules.

Lazy loading is the process of loading components, modules, or other assets of a website as they're required.

Since the initial codebase does not implement any preloading strategy, the Angular compiler builds it all in once and merges into a single JavaScript file. This approach will lead to a huge waiting time on the first page load in the browser at scale. The developers have no control to avoid it using a single module responsibility approach.

Failed Single-Responsibility Principle (SRP)

The single-responsibility principle is a computer-programming principle that states that every module, class or function in a computer program should have responsibility over a single part of that program's functionality, and it should encapsulate that part.

The current codebase does not adhere to this principle, because the AppModule is responsible for the entire application and does not delegate its responsibilities to the sub modules.

Failed Don't Repeat Yourself principle (DRY)

Don't repeat yourself is a principle of software development aimed at reducing repetition of software patterns, replacing it with abstractions or using data normalization to avoid redundancy.

This principle could be achieved by creating common logic components, services, modules and utilities. The current codebase is forcing developers to re-write some shared logic due to lacking side shared modules, utility services and angular directives.

Failed Keep It Stupid and Simple principle (KISS)

The KISS principle states that most systems work best if they are kept simple rather than made complicated; therefore, simplicity should be a key goal in design, and unnecessary complexity should be avoided.

The current codebase breaks this principle by having a single complex module, which cares out the entire application on its own, as well as its container components. Some of the components (f.g WorkspaceComponent) handle too many states and do way more as a developer might have guessed by reading its name or JavaScript documentation. The code that breaks this principle is used to have side effects, which are considered as hard to follow and difficult to debug or maintain.

The initial root code structure and the refactored root code structure illustrated in the Figure below:

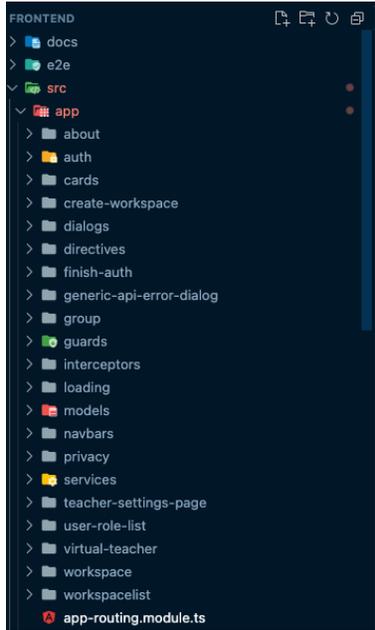
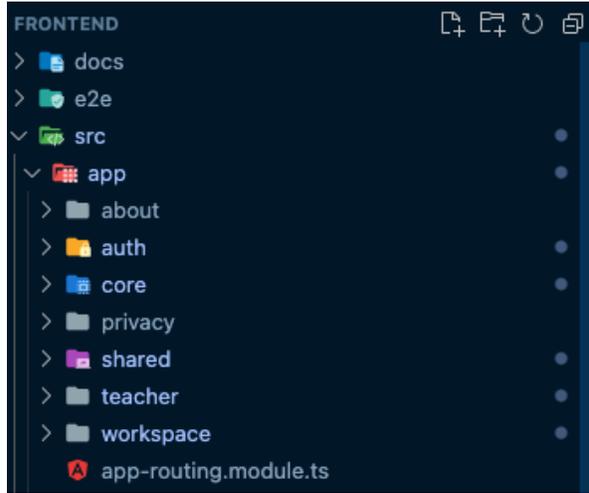
Before refactoring	After refactoring
	

Table 7.1.a. Before & After refactoring session

Routing

The initial PAT application uses Angular Routing mechanism to navigate a user through the Single Page Application. Angular Routing is a powerful tool which comes "out of the box" when generating a new Angular application. The tool allows developers to set up and configure the routing on demand and provides a variety of customization techniques.

An Angular Routing is a module which must be imported into the parent Module as a dependency. A developer is allowed to create many routing modules based on the code split strategy used

The initial project implements a single application routing module which is imported in the main module called application module. The structure of the routes and co-bound components in the application is visualized on the diagram below:

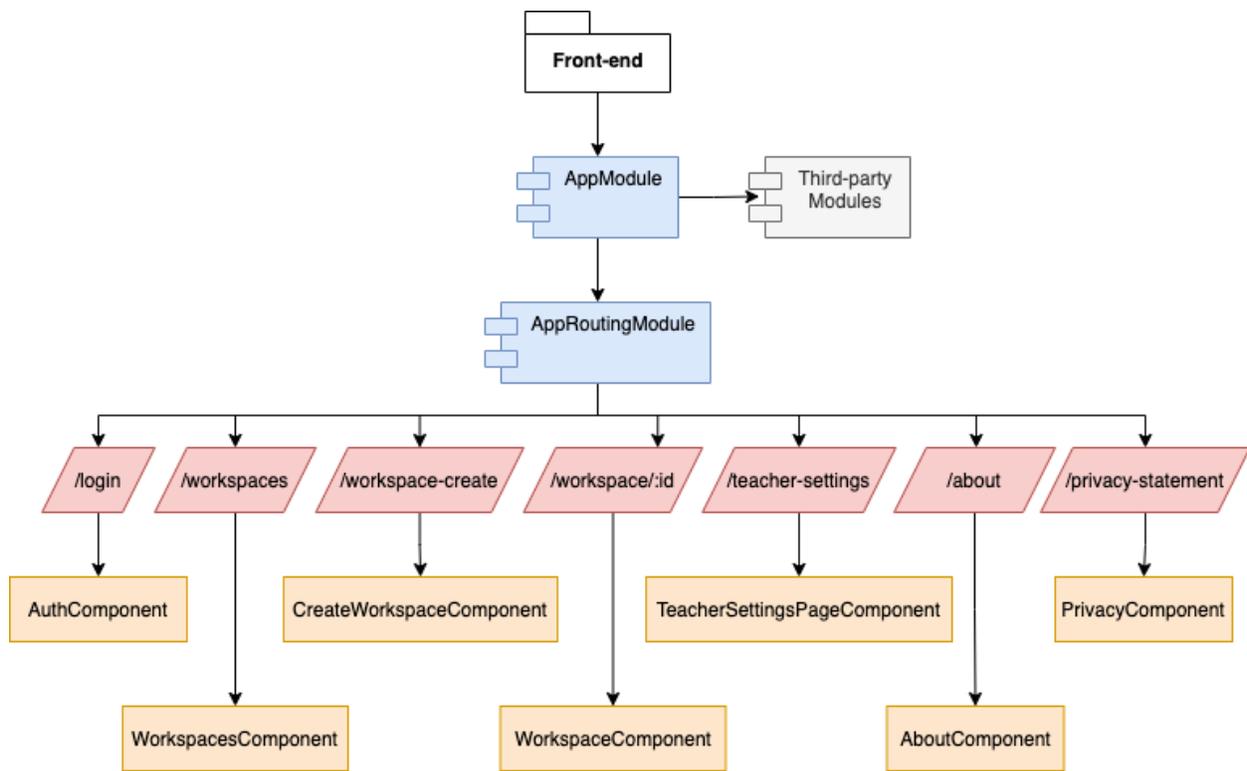


Figure 7.1.b. Initial routing structure.

As the diagram represents, all routes of the application are encompassed by a single routing module called "AppRoutingModule " which is imported into the root module called "AppModule".

This approach forces all components to be defined inside of the root module and be added to the main compiled JavaScript bundle. In this scenario, the application does not have any lazy loading approach, which could lead to optimization issues once the project begins to scale.

The routing refactoring process was to create a routing module for every new module illustrated in the *Module Refactoring* chapter and add page components to corresponding URL path.

7.2. Realization: Time Frames

To add time frames to the activity cards the card schema was updated as shown on screenshot below:

```
const CardSchema = new Schema({
  > id: {...
  },
  > title: {...
  },
  > shortDescription: {...
  },
  > longDescription: {...
  },
  > type: {...
  },
  > note: {...
  },
  > picture: {...
  },
  > color: {...
  },
  > reflectiveQuestions: {...
  },
  > deck: {...
  },
  startDate: {
    type: Date,
    required: [true, "Start date is required"],
  },
  endDate: {
    type: Date,
    required: [true, "End date is required"],
  }
});
```

Figure 7.2.a. Update CardSchema

To apply schema code change the back-end needs to re-run to re-initialize database models with two new fields: startDate and endDate. As the screenshot indicates, these fields are required and the corresponding error message will be returned if no dates are provided.

The front-end code needs to be updated as well to be able to provide these fields to the back-end (server) application. Therefore, the card details has now two additional inputs for startDate and endDate to specify a date range:

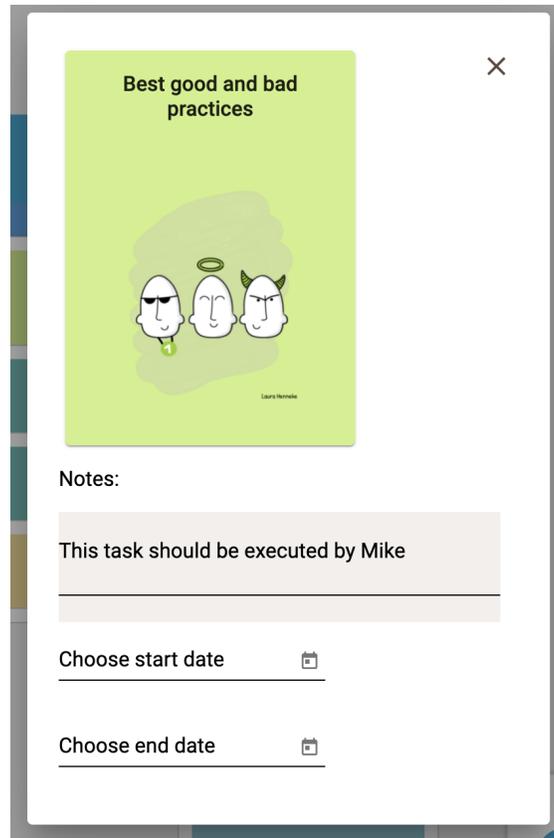


Figure 7.2.b. Updated card details with date inputs.

A card dashboard view component has been updated too. Indicating start and end dates on the card body below its title. It allows users to see the time box of a card without clicking on it and opening its detail view.

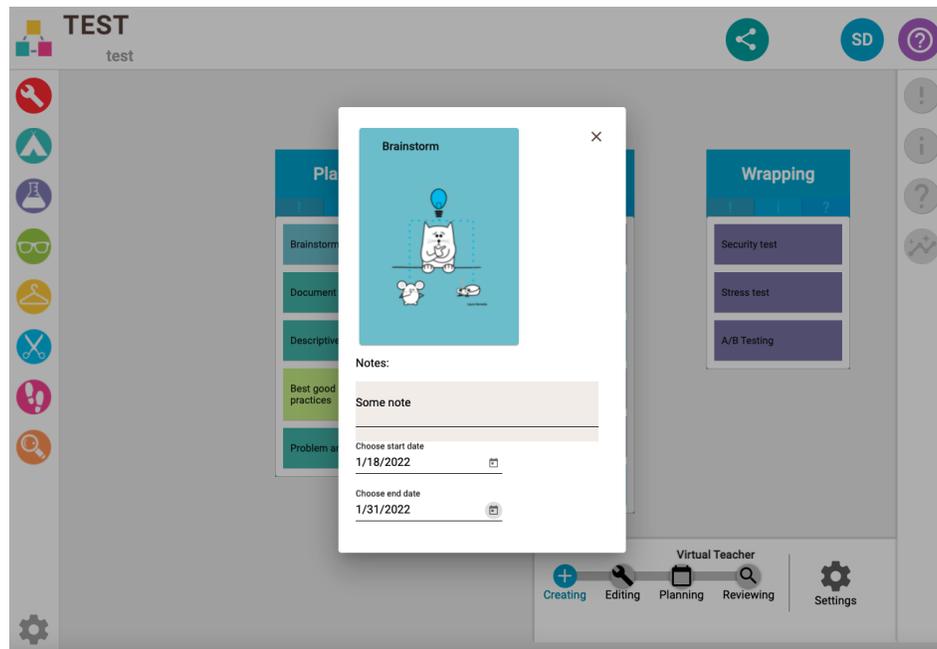


Figure 7.2.c. Choose start/end dates.

The card's start and end dates have been validated with every test scenario and have successfully pass every possible case:

- [✓] Create new card without startDate should display error
- [✓] Create new card without endDate should display error
- [✓] Create new card with startDate comes later than endDate should display error
- [✓] Create new card with consecutive startDate and endDate should create a card
- [✓] Every card in a deck should have startDate and endDate on it
- [✓] Edit a card without startDate should display error
- [✓] Edit a card without endDate should display error
- [✓] Edit a card with startDate comes later than endDate should display error
- [✓] Edit a card with consecutive startDate and endDate should update the card

7.3. Realization: Planning Tool

The final result for the planning page looks as on the screenshot below:

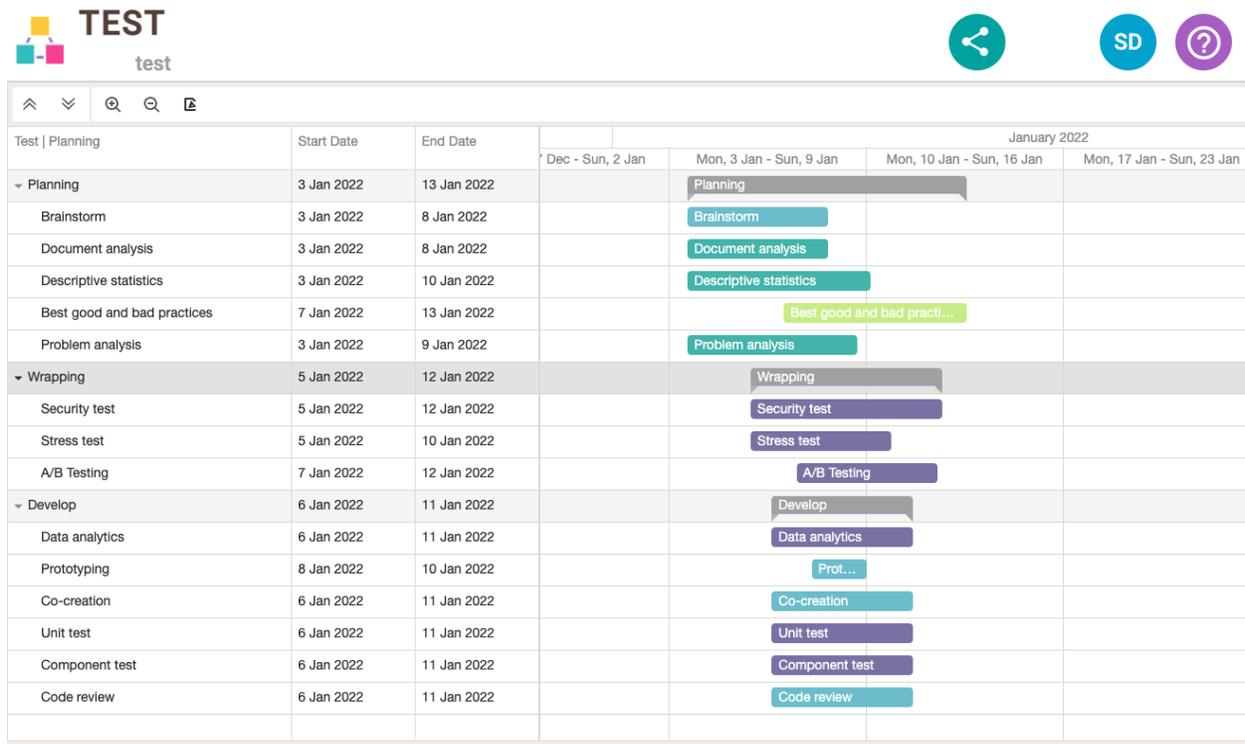


Figure 7.3.a. Final planning page.

This page works on a new route, namely “/workspace/planning/:workspaceId”, which is defined inside the application router.

The biggest challenge of this feature was the data-change events from the Gantt chart component which fired only one change at the time (startDate OR endTime) and just for a single row even when the entire group has been moved on the timescale (f.g Develop, see Figure 7.3.a).

To resolve this issue the following algorithmic approach was implemented using RxJs event manipulation library.

```
// Update cards after 1 second from last change
this.bufferTimer$.pipe(debounceTime(1000)).subscribe(() => {
  this.updateCards(this.updatedTimeBoxesBuffer).subscribe();
});
}
```

Figure 7.3.b. Event buffer.

This buffer timer stores all the events and emits the updated values with a debounce time of 1 second. That means, it only emits an update function if new changes haven't arrived since 1 second, otherwise they are stored and the timer waits another second.

Since the buffer is defined as a RxJs subject, it is easy to tell him that new value has been saved and reset it:

```
if (startDate) {
  updatedFields = {
    ...updatedFields,
    startDate,
  };

  this.updateTimeBoxBuffer({id: data.key, startDate});
  this.bufferTimer$.next();
}

if (endDate) {
  updatedFields = {
    ...updatedFields,
    endDate,
  };
  this.updateTimeBoxBuffer({id: data.key, endDate});
  this.bufferTimer$.next();
}
```

Figure 7.3.c. Resetting the buffer timer.

Updating cards was implemented according to the sequence diagram defined in the technical design (see Figure 6.2.b). The realization of this approach looks as following:

```
private updateCards(changes: any[]) {
  return this.workspace$.pipe(
    map(workspace => new Workspace(workspace)),
    map(workspace => workspace.updateCards(changes)),
    filter(workspace => !!workspace),
    switchMap(workspace => this.workspaceService.updateWorkspaceGroups(workspace)),
  );
}
```

Figure 7.3.d. Update cards implementation.

The Gantt chart was implemented with a customized PDF export. Clicking on this button in the toolbar exports a PDF file with the diagram in full landscape mode to avoid cutted-off edges from the saved file.

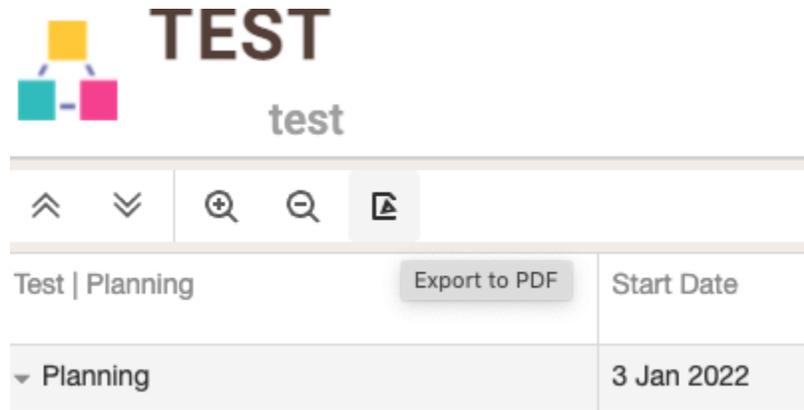


Figure 7.3.e. Export PDF button.

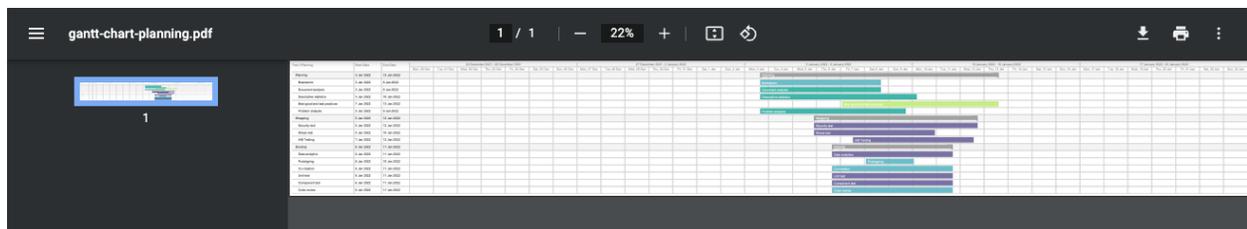


Figure 7.3.f. Generated PDF file with Gantt chart planning.

Created Gantt chart visualization supports all required interactions that were defined in the product requirements:

- Drag and drop activity
- Scratch left side of an activity to update the 'start date'.
- Scratch right side of an activity to update the 'end date'.
- Zoom In
- Zoom out

8. Evaluation

In this chapter the final results are summed up to reflect two main research questions.

8.1. How to create an activity planning tool that could easily be used by the students?

Result

To create a planning step with a Gantt chart, multiple research sub questions had to be answered. A Gantt chart has been added to the project as a project planning step. The students can use the Gantt chart to time box their activities within the project.

Conclusion

In the updated version of the platform students can plan their activities as well as export the planning. This functionality enables users to keep their activities, teachers' feedback and the project time planning in a single platform without a need to use any third party services. This approach should improve students' concentration and make their study more productive due to saved time.

8.2. How to improve the current codebase of the front-end application?

Result

To improve the project's codebase an analysis of the entire codebase was required. During this analysis several issues were discovered, namely:

- Vulnerable dependencies
- Angular module structure

All issues have been refactored and optimized.

Conclusion

Due to refactoring and optimization, the current project development process as well as any further development became easier. A developer will see no warnings in the terminal console during the project building and the codebase is well-structured and much more readable to understand the codebase.

9. Recommendations

The platform functions as it is supposed to and fulfills all required functional and non-functional requirements. However, the platform's codebase and user interface could be improved, therefore this chapter includes my personal recommendations for further development of this platform.

Split workspace component into several sub components.

Currently, the workspace component includes a large piece of logic that covers most of the front-end application's logic and consists of 1200+ lines of code. I recommend to split the component at least into these sub components:

1. Method card/activity left sidebar component.
2. Workspace navigation component.
3. Project activities canvas component.

This approach would simplify the code and improve readability and increase productivity for other developers to update the components.

Add web socket routes for time planning and add it to the project planning component.

The current version of the project planning tool doesn't include real-time updates if one of the user's teammates makes a change to the project planning. It could potentially lead to losing some of the overlapping changes and requires users to refresh the page to get the latest updates from other users on the same project.

Move business logic to the dedicated Angular services.

Keeping business logic in the front-end components is a bad practice. A much better option would be to move all essential business logic and calculations to the dedicated Angular services and inject them in any component that requires this logic.

10. Reflection

Working on the graduation project remotely during the pandemic wasn't that difficult. I had some remote work experience in the past and I believe it helped me to carry out this challenge this time much easier than it might appear to somebody.

Weekly meetings with the project manager (company supervisor) helped me to keep the priority list up to date and stay focused on the important tasks in my Kanban board. My work was performed in the iterative cycles that repeated every week:

- Analyze requirements
- Research solutions
- Develop solution
- Validate the result with the project manager

The project had many challenges along its execution. However, I believe that I pulled a lot of experience during accomplishing it.

Before the project has been started I defined three level 3 competences I am going to demonstrate during the work on this assignment. The competences and what I have done to achieve them described below:

Software Engineering Advice Level 3

To achieve this competence I have written several recommendations for further development aspects, researched and advised on possible solutions (front-end libraries, architecture changes, possible user interface approaches and their technical limitations) to the client, who formulated the requirements clearer.

Software Engineering Design Level 3

To achieve this competence I have created a product technical design report which clearly describes the system to be built and integrated into the current platform codebase.

Software Engineering Realize Level 3

To achieve this competence I have developed and integrated a new functionality block, that allows platform users to plan their project activities right inside the platform and keep it in synchronization with the server/database.

As a result, the pull request encounters [323 files](#) that have been changed (ignoring generated files).

Adding Gantt chart planning process #13

[Edit](#) [Code](#)[Open](#) SImpler wants to merge 6 commits into `master` from `feature/refactoring`

Conversation 0 Commits 6 Checks 0 Files changed 323 +18,600 -20,541

Changes from all commits File filter Conversations Jump to 0 / 323 files viewed Review changes

> 3	.gitignore	Viewed
> 17	backend/authenticationModule.js	Viewed
> 24	backend/controllers/cardController.js	Viewed

Figure 10.a. GitHub Pull Request.

As could be seen on the Figure 10.a the number of new code lines is 18600 and 20541 lines were removed. This negative difference is explained by the refactoring process I have accomplished. As a result, an additional functionality has been developed and the total codebase weight has been decreased without any cut-off in other functionality.

11. References

Saxion University of Applied Science (2021). Ambient Intelligence

<https://www.saxion.edu/business-and-research/research/smart-industry/ambient-intelligence>

Wikipedia (2021). Node.js

<https://en.wikipedia.org/wiki/Node.js>

Monocube (2021). List of 10 Best Front end Frameworks to Use For Web Development

<https://www.monocubed.com/best-front-end-frameworks/>

MongoDB (2021). What is MongoDB?

<https://www.mongodb.com/what-is-mongodb>

MongoDB (2021). What is NoSQL?

<https://www.mongodb.com/nosql-explained>

Rehkopf, M. (2018). What is Kanban?

<https://www.atlassian.com/agile/kanban>

O'Malley, P. (2021). Discover Three Different Types of Agile Project Management Frameworks.

<https://openclassrooms.com/en/courses/4544621-learn-about-agile-project-management-and-scrum/5080431-discover-three-different-types-of-agile-project-management-frameworks>

Material Design (2017). Date Pickers.

<https://material.io/components/date-pickers>

Gantt.com (2021). Gantt chart.

<https://www.gantt.com/>

Mozilla (2021). Single Page Applications.

<https://developer.mozilla.org/en-US/docs/Glossary/SPA>

VMWare (2020). What is application refactoring?

<https://www.vmware.com/topics/glossary/content/application-refactoring>

Csaba, P. (2014). Refactoring code.

<https://code.tutsplus.com/tutorials/refactoring-legacy-code-part-8-inverting-dependencies-for-a-clean-architecture--cms-21659>

Halodoc (2021). Angular best practices.

<https://blogs.halodoc.io/angular-best-practices/>