JOHANNES KEPLER
UNIVERSITY LINZ

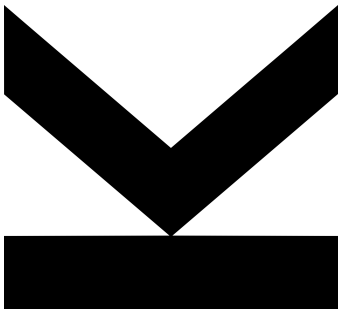
Author
Alexander Burghuber
k12105840

Submitted at the
Institute for System Software

Supervisor
Prof. Dr. Dr. h.c. Hanspeter Mössenböck

September 2024

A Tool for Creating Personalized Semester Schedules



Bachelor Thesis

to obtain the academic degree of

Bachelor of Science

in the Bachelor's Program

Informatik

JOHANNES KEPLER UNIVERSITY LINZ
Altenbergerstraße 69
4040 Linz, Austria
www.jku.at
DVR 0093696

Kurzfassung

Diese Bachelorarbeit beschreibt die Entwicklung eines webbasierten Semesterplaners, der speziell für Studierende der Johannes Kepler Universität (JKU) Linz entwickelt wurde, um die Kursauswahl und die Verwaltung des Stundenplans für eine effiziente Semesterplanung zu ermöglichen. Der Planer zielt darauf ab, die Herausforderungen traditioneller Planungsmethoden zu bewältigen, die oft manuelle und fehleranfällige Prozesse beinhalten. Durch die nahtlose Integration mit den bestehenden Online-Diensten der JKU bietet der Planer aktuelle Kursinformationen, eine konfliktfreie Stundenplanung und eine interaktive Kalenderansicht. Ziel ist es, die Effizienz und Genauigkeit des Kursauswahlprozesses zu verbessern. Derzeit befindet sich die Anwendung im Prototypenstadium, mit geplanter Weiterentwicklung im Rahmen eines Projekts für den Masterstudiengang Computer Science.

Abstract

This bachelor's thesis presents the development of a web-based semester planner specifically designed for students at Johannes Kepler University (JKU) Linz to allow course selection and schedule management for efficient university semester planning. The planner aims to address the challenges associated with traditional planning methods, which often involve manual and error-prone processes. By integrating seamlessly with existing JKU online services, the planner provides up-to-date course information, conflict-free scheduling, and an interactive calendar view. It is designed to enhance the efficiency and accuracy of the course selection process. Currently, the application is a prototype, with further development planned as part of a project for a Master's degree in Computer Science.

Contents

List of Figures

Listings

1 Introduction

Shortly before a semester starts, students at the Johannes Kepler University (JKU), need to choose which courses they are going to attend in that semester. This happens on a specific time and day some weeks before the semester start, where students register for their selected courses using JKU services. An effective academic time plan is crucial for a student's success. The complexity of course selection, curriculum requirements and time management require tools that allow these tasks to be easily solved. This thesis presents the development of a web-based semester planer designed specifically for JKU students to easily provide course selection and schedule management.

The online semester planer addresses several challenges often faced by students shortly before the official registration. Traditional methods of planning usually rely on cumbersome browsing through JKU online services and manual schedule management, all while aligning them to the curriculum and official course attendance order recommendations. This can be very time-consuming and prone to error. Already finished plans also miss out on any changes to the course schedule that may happen before the registration.

This web-based semester planner aims to mitigate these issues by providing a dynamic, user-friendly tool that integrates with JKU's existing online services. Key features include an up-to-date course selection, conflict-free scheduling and an interactive calendar view. By using modern web technologies and a user-friendly design, the app aims to enhance the effectiveness of the course selection process.

Currently, the application is a work-in-progress prototype. The following pages talk about the current state of the semester planner as it is presented as my bachelor thesis. Any future work and completion will happen as part of a project for my Computer Science Master's Degree. The application is currently limited to only test data - no student can currently log in in with their student's account and see their available courses. Nonetheless, the test data is made up of real courses from the JKU's systems, provided by cloned and

anonymised data from actual students. For all limitations and future work, see Section 5.

This thesis starts with the background of a student's university semester planning, then it discusses the implementation of the app and its usage. Lastly, any future work and features will be explored.

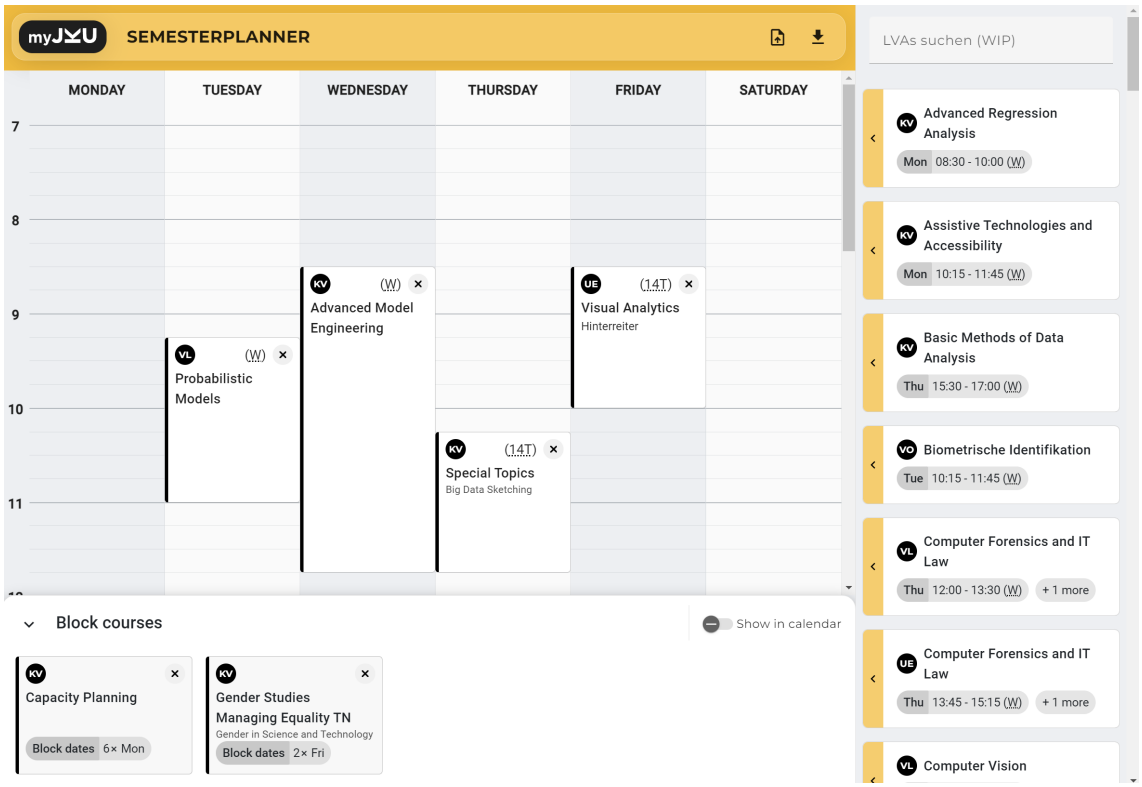


Figure 1.1: First look at the semester planner application

2 Background

In this chapter, we will explore the fundamental concepts and methods involved in planning a university semester, with a focus on computer science students at the JKU. Proper semester planning is crucial for students to successfully get their degrees, manage their time efficiently, and meet all the necessary requirements for graduation. Beside explaining the traditional planning method used by students, tools offered by the JKU are introduced, such as KUSSS and myJKU. Additionally, we will introduce the idea of the semester planner app.

2.1 Planning a University Semester

This section mainly presents semester planning as it currenly happens for computer science students. For other fields of study, it may be conducted differently.

Planning a semester can be troublesome and time-consuming for a lot of students. For many, the planning is not just about selecting the courses they want to attend next semester, but also various other factors may come into play.

For every field of study, there exists a curriculum, which describes many aspects on how the field of study is conducted, including which mandatory courses a student has to attend and how many ECTS this person has to collect to be eligible for the corresponding degree. Some curricula also include a recommended study program, which describes in which order some courses should be attended and in which semester. Students have to keep a close eye on which courses still need to be attended if they want to ensure that they don't have to do additional semesters because of missing courses. A lot of courses are not offered every semester, so an unlucky student may need to be registered for a redundant extra semester.

As mentioned before, students also have to earn a certain amount of ECTS before they can receive their degree. Through the courses that are mandatory a student earns a certain amount of ECTS, but this does not accumulate to the needed sum. The majority of these missing credits can be acquired through non-mandatory courses. In turn, a student also has to keep track of how many non-mandatory courses they still need to attend for the next semester or future ones, which is an important part of the semester planning process.

Additionally, it is increasingly common for university students to hold a job while pursuing their studies. Reasons differ, yet mostly it is to receive job experience even before getting their degrees and to support themselves. If their working hours are not on the weekend or outside common university hours, then their semester planning is also heavily affected by their job. Whole weekdays might need to be blocked off, which can cause problems since some courses are only offered on specific weekdays. If a student's job does not offer flexible working days, then they might need to attend additional semesters to get their degree.

2.2 Traditional planning methods

Students at JKU usually look up the available courses on the university system called KUSSS [1]. Course information is published a few weeks before the start of the registration. In these weeks, students can plan their semester. They look up all courses they might want to attend by browsing through KUSSS. Usually, each course's information and its dates are copied to some kind of digital planning program or on paper. A Microsoft Excel Sheet or one of its counterparts is not uncommon for that. A timetable is created where e.g. each column represents a weekday and each row a quarter-hour. All courses at JKU have their time slots from a fixed quarter-hour to another. One source of error with this method is the copying of the course time slots. A student could misread the time of a course or select the wrong quarter-hour accidentally. Another problem is, that in some rare cases, a course's information is updated again shortly before the registration. This could easily be missed in one's own schedule, since the data is manually entered. Our semester planner app loads real-time data from JKU system, so both of these issues are prevented.

A course at the JKU can have multiple groups, where each group has its own time-slot and possible its own lecturer. The contents of all groups are the same. In the Computer

Science field of study, these courses are often of type "Lab" (Übung). A student has to choose one of these groups or gets assigned to one when selecting multiple. If the student is not yet sure which of the groups to attend, all groups may be entered into his planning program in the meantime. Later the group will be chosen, that best fits into the rest of this person's schedule, and the redundant groups are removed. In our semester planner app, this can be done easily and automatically, without tedious manual rescheduling like in e.g. an Excel sheet.

2.3 myJKU

myJKU is a JKU website that offers many features from KUSSS in a modernized form. From a user-experience standpoint, KUSSS has become somewhat really outdated over the years, so myJKU was developed to offer an alternative to KUSSS's services in a user-friendly way, and it might even fully replace KUSSS in the future [2]. Our final semester planner app will be integrated into myJKU either through direct integration on the website or through an external link from myJKU.

2.4 Planning with the Semester planner app

As seen in Figure 1.1, the semester planner app offers a list of all available courses on the right-hand side of the webpage and a calendar on the left-hand side. The general idea is that user search for and choose courses from the list and add them to calendar through a simple click on a button. Courses are then placed on the calendar at the respective time slot. If a course is already at the same time slot, then all conflicting courses are shown side-by-side in the calendar. Students can select all the courses they might want to register for and then have a clear overview of all their courses on the calendar, where they are also able to easily remove courses from the calendar. This allows for the simple creation of different time schedules, where users can experiment with a variety of course combinations. The planner is personalized, users can log in with their standard JKU account. Courses that the student has already passed are no longer shown in the course list, except those courses where the content can differ each semester. After a student finished planning her semester, the built time schedule can be exported through a variety

of means, although this feature is part of future work. An in-depth overview of all features are describes in Chapter 3.

3 Usage

As already mentioned in Section 2.4, the semester planner app provides an innovative and simple approach for students to create their semester schedule. However, there are still are a variety of features that offer a more complex and personalized schedule if desired. This chapter explains the usage of the semester planner app feature by feature.

Features:

- Show course list
- Search course list
- Add course to calendar
- Remove course from calendar
- Show calendar with selected courses
- Show detailed course information
- Show block courses in collapsible view
- Show block courses directly in calendar
- Automatically save selected courses

More detailed descriptions of each feature are provided in the next few sections. The technical implementation of some features is described later in Chapter 4.

3.1 Initial start-up

As mentioned before, the application is still work-in-progress. After completion, it will be available either through a direct integration in myJKU or as a separate domain, that can be accessed through a link on myJKU. Nevertheless, students will need to log in with their JKU student account to access the semester planner app. This login process will happen with the JKU Single-Sign-On system and is part of future work (see Chapter 5). After login, the user can access the website to its fullest extent.

3.2 Show course list

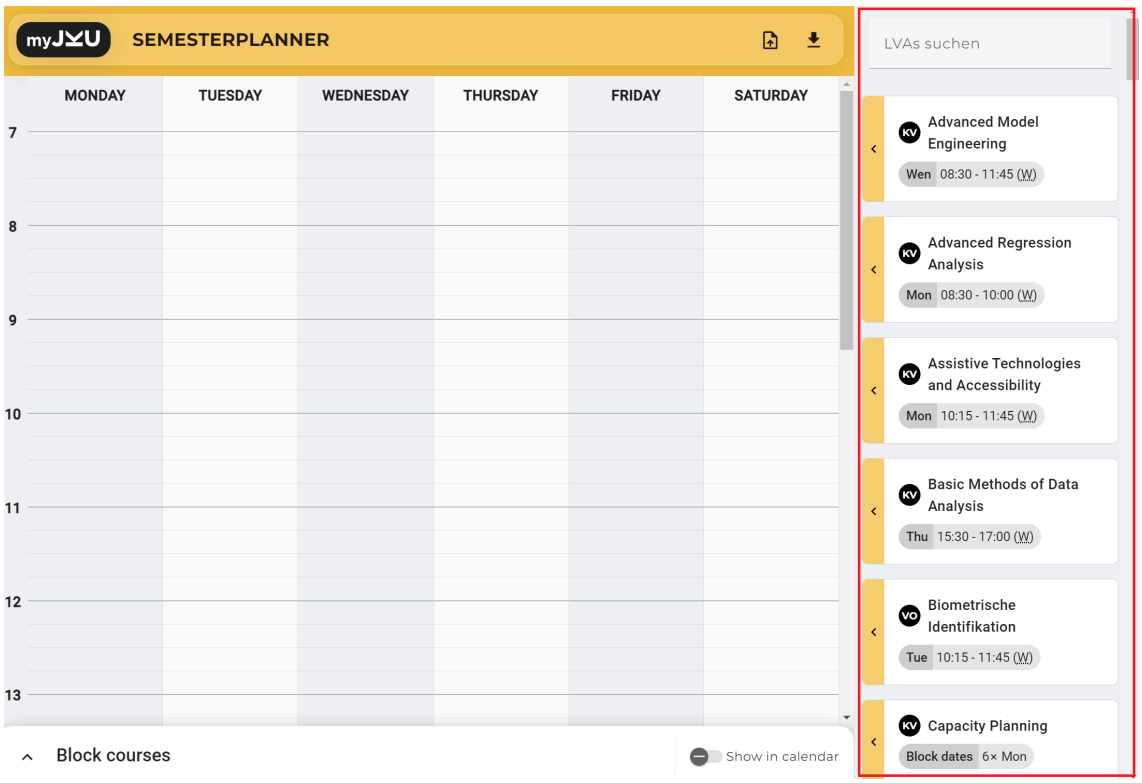


Figure 3.1: The application as it looks on its first start-up, with the course list highlighted in red

Figure 3.1 shows the course list on the right-hand side of the screen. At the top there is a search bar, which is explained in Section 3.3. Below is the list itself, where all available courses for the logged-in student are shown.

These are the courses that the student can register for. Courses that the student has already completed are filtered out beforehand. A completed course is a course that a student has registered for and finished with a positive grade. Students can see their list of completed courses on KUSSS or myJKU. Courses of a specific type are still shown, even if the student has completed them. Those are the courses that can have different content each semester, so a student might want to re-attend them again.

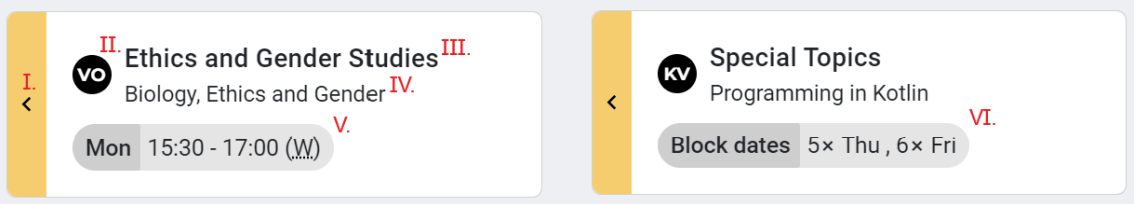


Figure 3.2: Two forms of courses in the list, with elements marked using roman numerals

A variety of different types of courses can appear in the course list. Figure 3.2 shows the two most common types. Specific elements of the courses are highlighted in red roman numerals and are explained below.

I. represents a button that is part of every course item. This button allows the course to be added to the calendar. More on this in Section 3.4.

II. shows the abbreviation of the type of the course itself. *VO* means “Vorlesung” in German, or simply "lecture" in English. Another abbreviation for “Vorlesung” is *VL*. Other common abbreviations include *KV*, for “Kombinierte Veranstaltung" (engl. Combined lecture) or *UE* (engl. Lab).

III. is the title of the course.

IV. is the subtitle of the course, which is optional and not all courses have one.

V. shows a shortened version of the date-time occurrences of this course. In this case, the course is held on Monday at 15:30 till 17:00. The *W* inside the parentheses at the end

indicates the course reoccurs weekly at this weekday and time. Another possible value is *14T*, which means that the course reoccurs fortnightly (every second week).

VI. is similar to V. The difference is that this course has no fixed reoccurring sessions but instead so-called block dates. A course like this can have sessions at any time at any weekday, or in some weeks not at all. The course in the figure has five sessions on a Thursday and six sessions on a Friday, but the time slots could be different each week.

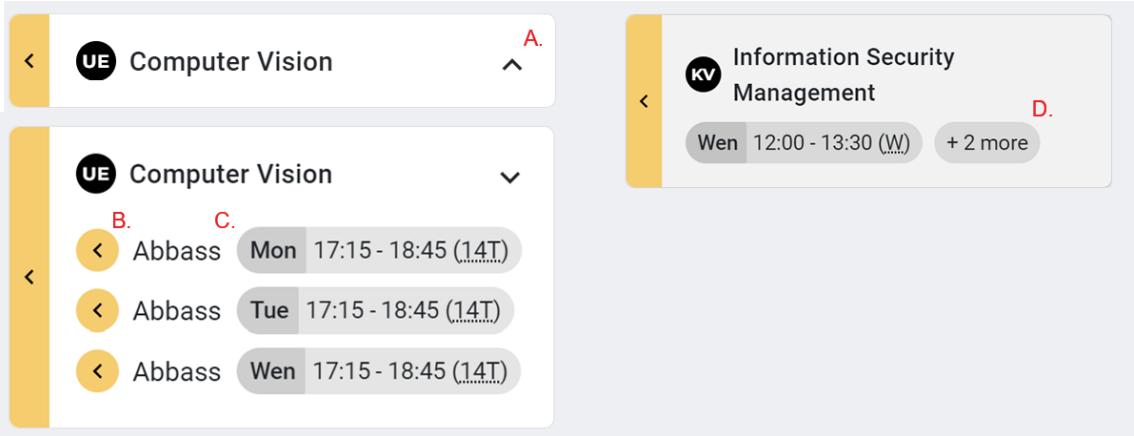


Figure 3.3: Two other forms of courses in the list, with elements marked using letters

There are two additional forms of courses that can appear in the list, as seen in Figure 3.3. The two items on the left-hand side of the figure represent the same course, where the top one shows it in its collapsed form and the bottom one in its expanded form. What makes this course special is that it offers different groups. Groups were explained previously in Section 2.2. The expanded form shows all available groups, whereas the collapsed form shows just the course itself to save screen space on the webpage.

A. marks the button by which the course can be expanded or collapsed.

B. shows three buttons, each belonging to one group. With these buttons, a group can be added to the calendar separately. The button, that was explained in **I**., adds all groups to the calendar.

C. is the last name of this group's lecturer. Some courses have different lecturers for different groups, so the name is a way to distinguish them without the date-time.

D. shows an element similar to **V**. and **VI**. It displays the amount of sessions of a course, that are outside its normal schedule. For example, the course in this figure has its sessions on Wednesday from 12:00 to 13:30 on a weekly basis. The "+ 2 more" indicates, that two more sessions exist that are not happening on this date-time combination.

3.3 Search course list

As shown in Figure 3.1, the top of the course list features a text field that allows users to filter the courses that should be displayed. This filtering functionality enables users to search by title, subtitle, course type (e.g. VL, UE, ...), and lecturer. Courses that do not match the specified search criteria are excluded from the list.

3.4 Add course to calendar

As described in Section 3.2, courses can be added through buttons in each course item in the list. If a course offers multiple groups, then one, some or all groups can be added to the calendar. This allows students to see all groups in the calendar and, after adding the remaining courses, select the one group that fits into their schedule the most. If a course is added to the calendar, it disappears from the course list. It reappears in case the course is removed from the calendar. For a course with multiple groups, the selected groups disappear separately. If all are selected, then the whole course.

3.5 Remove course from calendar

Removing a course from the calendar is as simple as adding one. The calendar view provides a close button at the top of each course item to quickly remove one. This allows fast iteration of different schedules for the student to see which one at the end matches their time management the most. Courses can also be removed through their detailed information view, which is described in Section 3.7.

3.6 Show calendar with selected courses

Figure 3.4 shows the calendar with some added courses. On the left-hand side of the view are the hours of the day, starting from 07:00 until 23:00. All courses at the JKU are offered inside this time frame. The slots in between the hours are quarter-hour steps, which are shown in rows. The weekdays are presented as columns. This calendar grid, which combines weekday-columns with quarter-hour-rows, closely resembles the semester schedule described in Section 2.2.

Course items in this calendar grid have similar properties, as seen in the course list Figures 3.2 and 3.3. The top-left corner has the course type. The top-right corner includes an indicator how often this course is offered ("W" for weekly) and the remove-button, as mentioned in Section 3.5. Below is the course title, followed by the subtitle if present. If the item is a group of a course, that contains multiple groups, the lecturer's last name is also displayed.

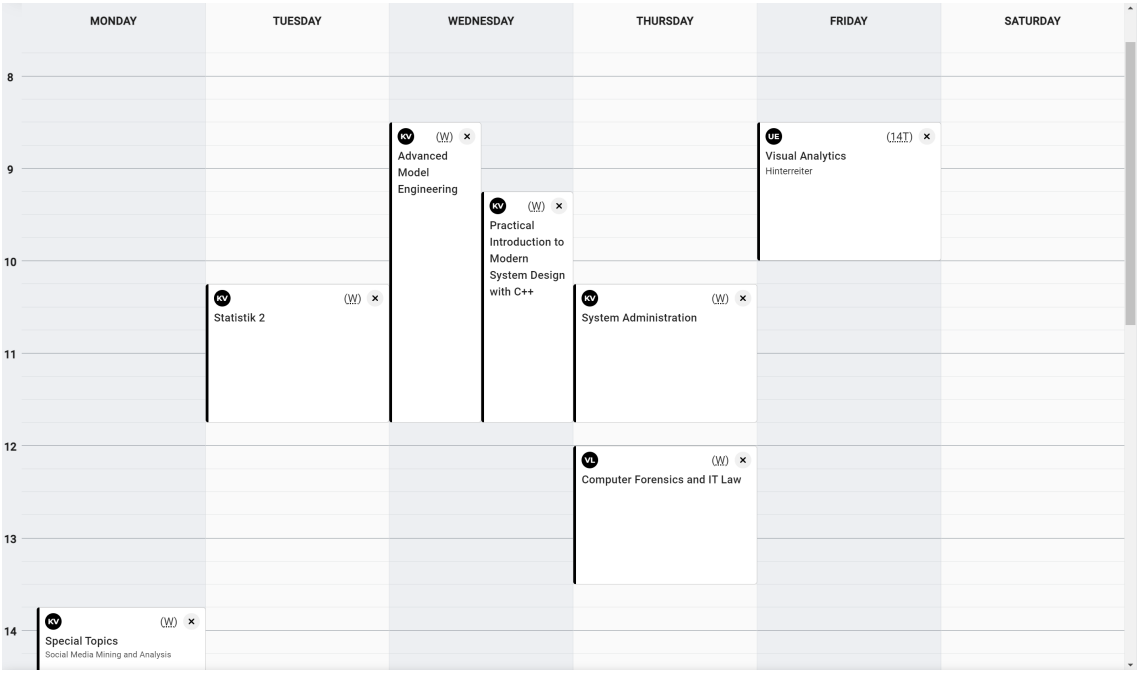


Figure 3.4: The calendar with several selected courses

Courses that are happening on the same weekday at the same time are shown side-by-side inside the calendar grid. The calendar doesn't restrict how many courses are possible at the same time, so students can select as many parallel courses as they want, even if the official university registration doesn't allow that.

Selected block courses are not shown inside the calendar by default. They are first added to a separate view, more on this is explained in Sections 3.8 and 3.9.

3.7 Show detailed course information

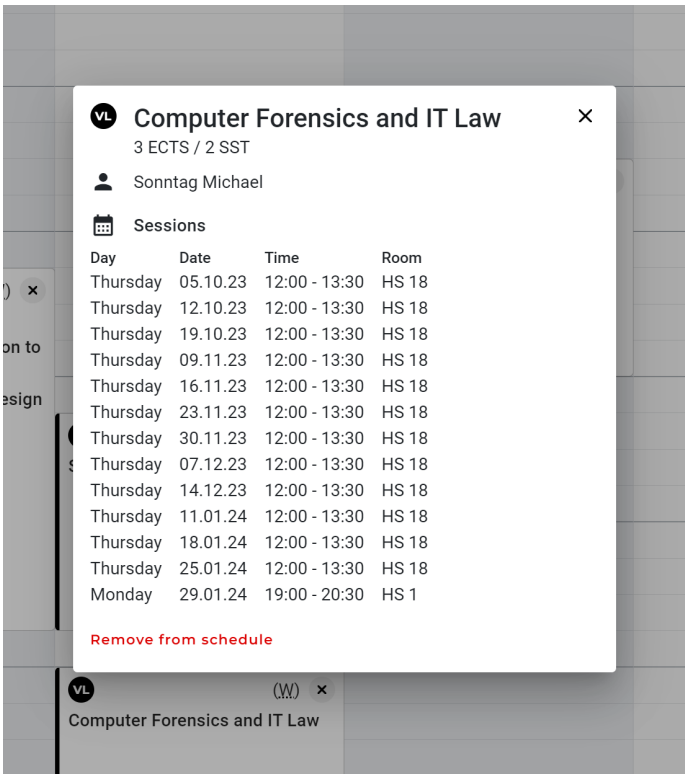


Figure 3.5: The details dialog of a course

Clicking on a course inside the course list or the calendar opens a dialog, containing additional details about the course. Important information like the amount of ECTS that a student earns by completing the course, or the weekly hours per semester (as seen as "2 SST" in Figure 3.5). Additionally, the full name of the lecturer is shown and all sessions

of the course are presented in a table. The table includes the weekday, the date, the time and the room at the university, where the session will be held. Lastly, if the course was already added to the calendar, a button is shown to remove the course from the student's calendar.

If the course has multiple groups, the dialog is presented in a different way. It includes all groups in form of collapsible views. If the dialog was opened by clicking on a group in the calendar, the respective group gets automatically expanded.

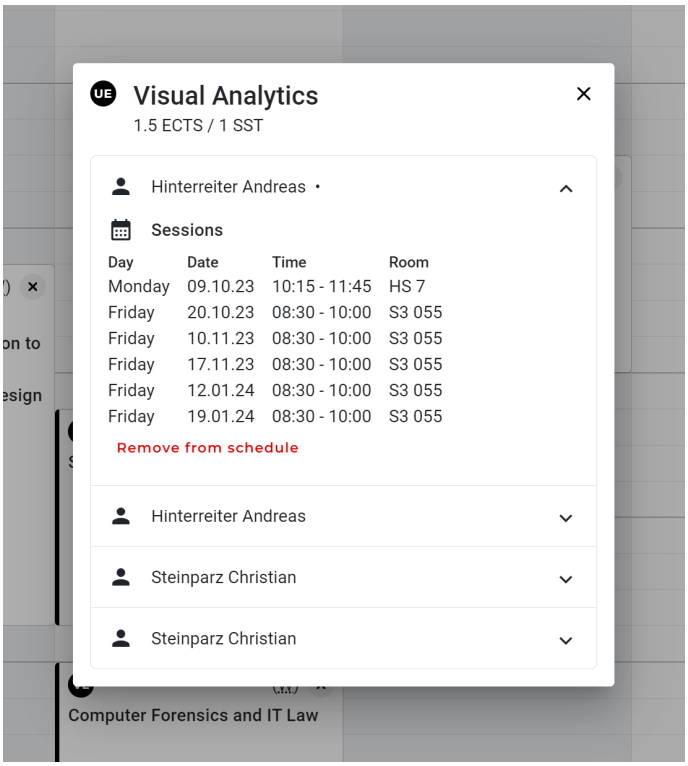


Figure 3.6: The details dialog of a course, that has multiple groups

3.8 Show block courses in collapsible view

When selecting courses, where their sessions are held in block-dates, then they are not directly added to calendar but instead to a seperate view. Since a block course can have sessions at any weekday at any time, the calendar view could get cluttered, so block courses are shown in a seperate collapsible view. This view gets automatically expanded when adding the first block course.

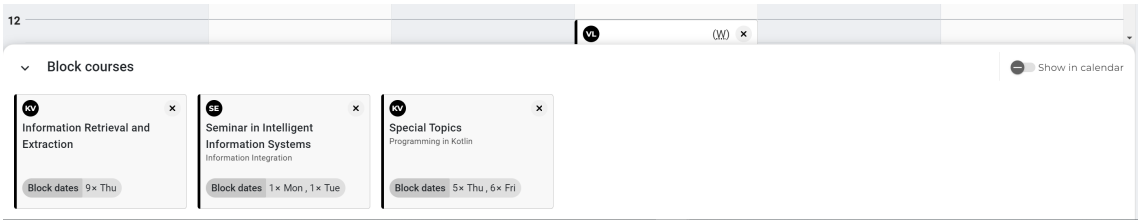


Figure 3.7: The expanded block courses view

3.9 Show block courses directly in calendar

The block courses view includes a toggle switch, that makes the block courses visible directly in the calendar, if desired. This may allow a student to see all weekday and times when any of a block course's sessions could happen, even if a session occurs only on a single day in the whole semester. The block courses shown in the calendar a special left-hand side border to easily distinguish them from other courses. They also show a small, highlighted text indicating how many times a session happens on this exact weekday and time, as seen in Figure 3.8.

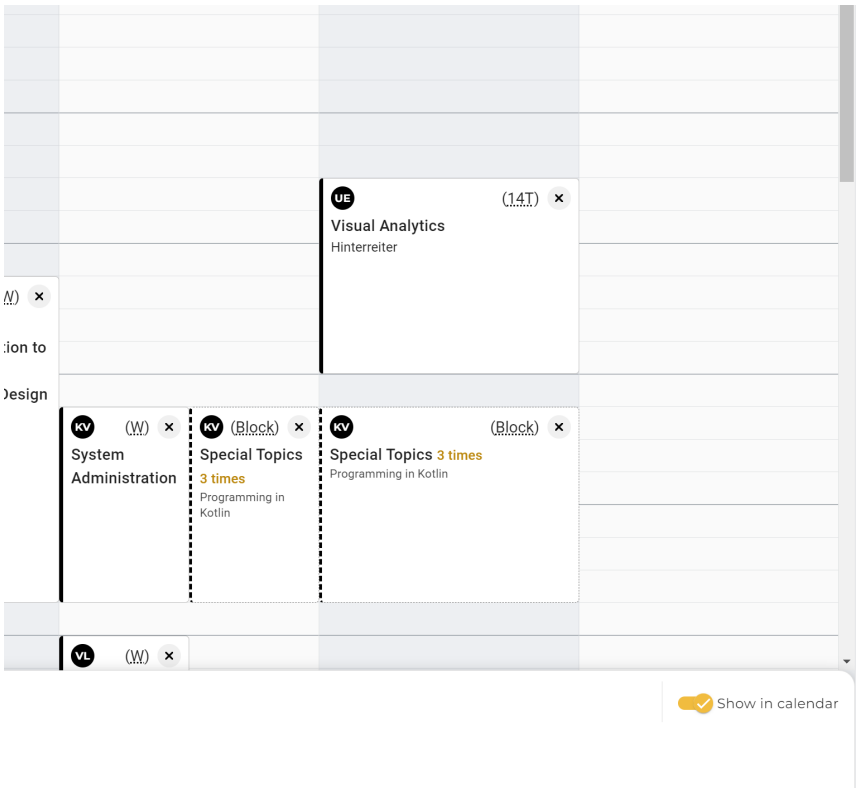


Figure 3.8: Blocks courses shown directly in the calendar

3.10 Automatically save selected courses

Each time a course is added to or removed from the calendar, the semester planner app has a different state. Normally, in web applications, where state is only handled locally on the client, that state would be lost if the browser tab or the web browser itself is closed. To prevent students from needing to reschedule their whole semester after they close the web page, the application automatically saves everything in a local storage belonging to the browser. At the start of every semester, this storage is reset. This happens on every add and removal of a course, so any state is perfectly safe. In the future, users might also export the current state themselves, which is described in Chapter 5.

4 Implementation

As mentioned before, the semester planner application is a web-based tool, i.e. it is hosted on a server and can be accessed through a web browser. Websites are primarily created with the technologies HTML, CSS and JavaScript [3]. In many modern web development projects, those languages are not used directly, but instead handled through a web framework. Angular, React, and Vue.js are just a few examples of the many popular frameworks available [4]. The semester planner app uses Angular. This application also requires dynamic data access to work properly. It utilizes REST to retrieve data from a corresponding server built with Java and the Spring Framework [5].

4.1 Technologies

Angular and Spring are the two main technologies with which the semester planner app was implemented. The next sections explain those frameworks more in-depth.

Angular

Angular is a TypeScript-based web framework for building dynamic websites [6]. TypeScript is a framework of the JavaScript language itself, that adds static typing for improved type safety and maintainability. Angular's core principles are made of a component-based architecture putting focus on modularity and reusability. Dependency injection is also a core part of Angular, which promotes testability and maintainability. One key feature is bidirectional data binding, which synchronizes data between a component's model (TypeScript) and view (HTML). Angular also offers a robust command-line interface (CLI) for efficient project setup and management. With its focus on TypeScript, Angular provides strong typing for improved code quality and developer productivity. Angular has a rich

ecosystem, coupled with performance optimizations and a large community. This makes it a great choice for developing scalable and high-quality web applications [7].

Spring

Spring is a open-source framework for the Java programming language, mainly employed in the development of servers. It provides a cohesive programming and configuration model, which provides the construction of robust and scalable software systems. Central to its core is Inversion of Control (IoC), a design principle where the framework takes responsibility for object instantiation and dependency management, thereby promoting loose coupling and enhancing testability [8]. Spring offers a modular architecture, encompassing modules for core container functionalities, data access, web development and testing. By abstracting away infrastructure concerns, Spring empowers developers to concentrate on core application logic. Moreover, it integrates seamlessly with various technologies and databases, making it a versatile choice for a variety of project requirements [5].

4.2 Integration with JKU systems

To make the semester planning process work for all students of JKU, the application needs to access always up-to-date course information from official JKU systems. The Department of Information Management (IM) at JKU is the central, university-wide service provider of IT infrastructure and digital services [9]. In cooperation with this department, the semester planning application uses a custom-made data interface to obtain the necessary course data. Later on, the JKU log-in system will also be integrated. More on this in Section 5.

```
public record CourseSessionDb(
        @NonNull String courseId,
        @NonNull String groupId,
        @NonNull String semesterCode,
        @NonNull String courseType,
        @NonNull String title,
        @Nullable String subtitle,
        int hoursPerWeek,
        double ects,
        @NonNull String typeOfDates,
        @NonNull LocalDate date,
        @NonNull LocalTime startTime,
        @NonNull LocalTime endTime,
        @NonNull String room,
        @NonNull String lecturer,
        boolean contentIdentical
) {
}
```

Listing 4.1: The CourseSessionDb DTO on the server, directly deriving from the IM API

One key part of the API made by the IM is the simple way to query all sessions of courses held in a specific field of study and a specific semester. This can be done using a SQL query on their database, for which the application was given access to. The semester planner app's server then parses this information into a list of the Data Access Object (DTO) *CourseSessionDb*. As seen in Listing 4.1, this DTO contains a wide variety of data useful for the app.

Another part of the API provides the server with a list of passed courses of a given student. This allows the server to filter out all courses that the student has already passed, so that they are not be shown in the app's course selection list.

The list of *CourseSessionDb* is then transformed through a variety of means to end up as a list of type *Course*. This class is part of the data model (as seen in Figure 4.1), which is offered by a server REST endpoint to clients. After a user logs into the application,

the client queries this endpoint once and caches the received data locally. This reduces network operations to a minimum and, in turn, remove unnecessary loading times.

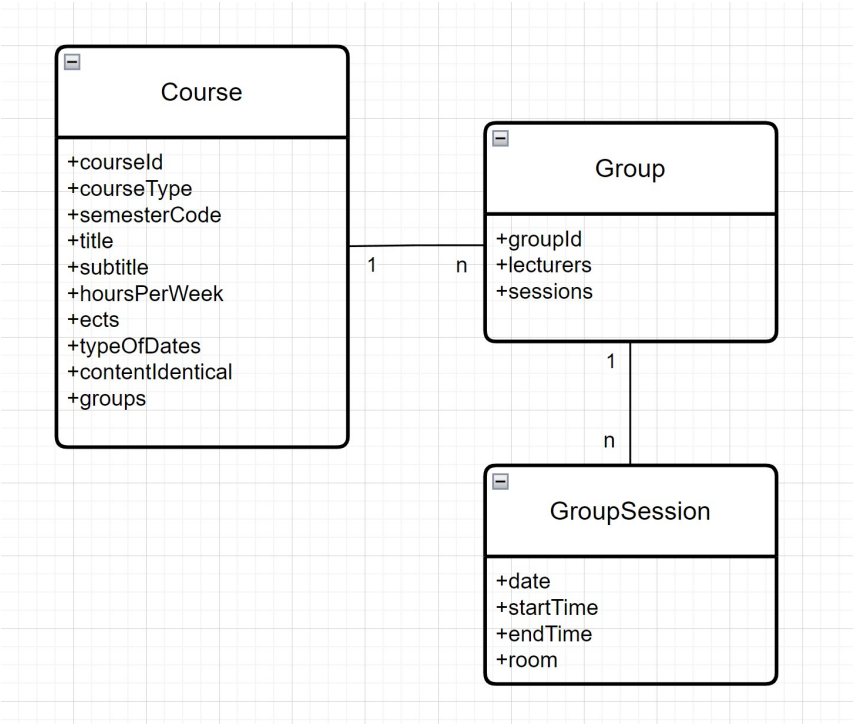


Figure 4.1: The core data model

4.3 Feature implementation details

Chapter 3 described several features provided by the semester planner app. This section delves deeper into some of their implementation details, that allow them to be used effectively by students to plan their semesters.

4.3.1 Show calendar with selected courses

The calendar view is implemented with various CSS layouting systems, which includes absolute-relative, flex and grid positioning. Absolute-relative positioning is used for

overlaying an HTML element with other elements. This is used for placing the quarter-hour markers behind selected courses. Flex positioning is used for showing all weekday columns inside a row. This places the weekdays from left to right, starting with Monday. Monday to Saturday are shown initially. Monday to Friday are normal work days, where most courses take place. Rather uncommon are course sessions on Saturday, but from time to time they occur. Sunday is also possible, but using the KUSSS course search, only one of those occurred in summer semester 2024. The semester planner still handles this use case, and Sunday is automatically added in-case a Sunday course is selected. Grid positioning is used for placing selected courses inside a weekday. The semester planner allows the selection of courses that are happening on the same day at the same time, so they need to be shown side-by-side in the calendar. If this not been the case, then a flex positioning could have been used instead.

4.3.2 Search course list

The course search, as explained in Chapter 3.3, allows a student to search for the title, subtitle, course type or a lecturer of a course. The search happens locally on the device of the user on every input-change of the corresponding text field. Basically, after each keystroke, the course list is updated with the new results.

The search utilizes a special library called Fuse.js [10], which is a lightweight fuzzy-search created in JavaScript. This library runs locally on the device and requires no network operations. Fuzzy-search is a technique that finds matches for search queries that are likely close to the intended result, even if there are typographical errors, missing words, or minor differences in spelling. It ranks thr results based on their similarity to the search input. This is especially useful for handling common human errors and variations in data, which allows students to search for courses, even if they are not completely aware of their course titles, say.

4.3.3 Automatically save selected courses

As described in the Section 3.10, selected courses are automatically saved locally. This way, the student's planned schedule is not lost if he or she logs-out, closes the browser or similar. Selected courses are saved with methods provided by our ScheduleService.

Services are classes that have a distinct purpose and provide functions to the rest of application, which is explained in more detail in Section 4.4. The observer pattern is utilized in our application with the popular JavaScript library called RxJs [11]. This library provides a class called *BehaviourSubject*, where instances of it can hold a single, arbitrary value. A BehaviourSubject provides a function to update this value and a function to observe changes to the value through a given callback. Selected courses are saved in a array, which is the value of a BehaviourSubject. The ScheduleService itself starts observing this subject during it's instantiation. On every observed value change, the selected courses array is taken and saved using the StorageService. This service provides access to the LocalStorage API of a web browser [12], where data is persisted until it is explicitly deleted by the website from which it was saved or by removal by the browser through special means or by the user.

4.4 Client Architecture

The architecture of the web client follows a common Angular architecture, where the main elements are so-called components and services. Components usually represent an user interface element of the webpage, which can be a container holding multiple other components or, for example, just a single button.

A service is a broad concept that includes any value, function, or feature an application requires. Typically, a service is implemented as a class with a specific and well-defined purpose. Angular differentiates between components and services to promote modularity and reusability. A component should expose properties and methods for data binding to act as an intermediary between the view and the application logic. For operations unrelated to the view or the application logic, a component should rely on services. Services are well-suited for tasks like fetching data from a server, validating user input, or sharing data between components using the observer pattern. By encapsulating these tasks in an injectable service class, they become accessible to any component [13].

Figure 4.2 shows the basic architecture of our client. The Data Layer contains services and is split between services that either implement domain logic or access to external data sources. Domain logic services implemented essential application requirements which are provided through methods to components. These include actions such as adding or

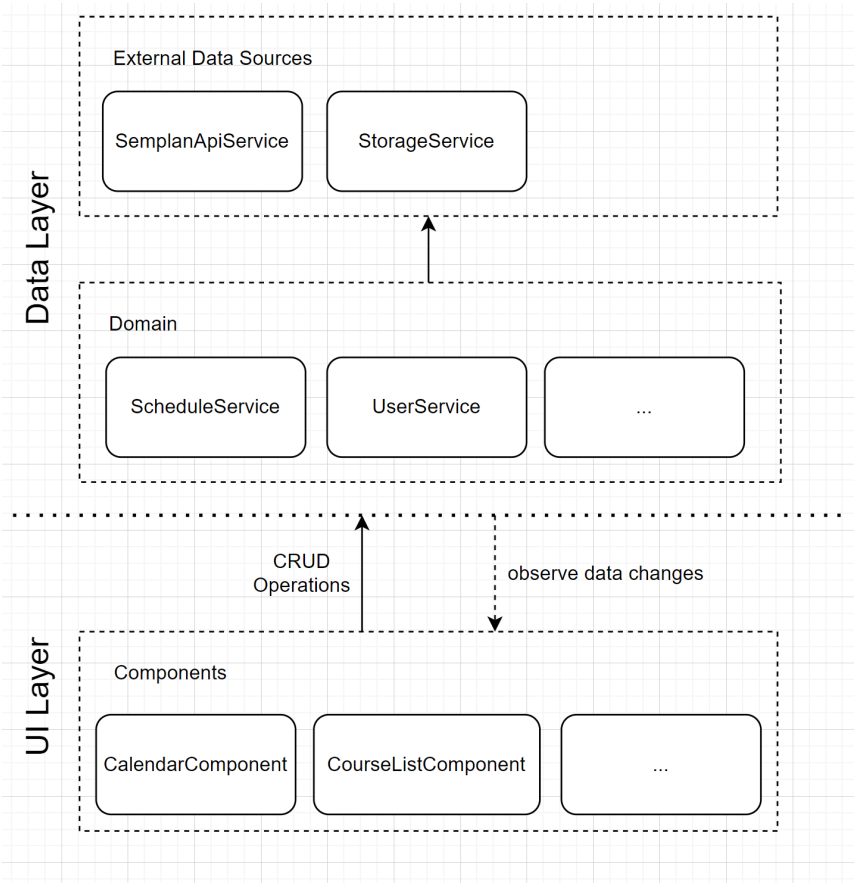


Figure 4.2: Basic client architecture

removing a course from the selection, getting user information, and more. To perform these actions, a service might need to access data sources that are either persisted or can be accessed through web endpoints. This logic is extracted to external data sources services, which can be accessed through other services but not from the components. This ensures a clean single responsibility architecture and layering.

Each component is directly coupled to a single view, which in turn makes these classes part of the UI Layer. Components can access domain services functions that provide CRUD operations or observe data from it.

5 Future Work

As mentioned, the project is not in a finished state. This bachelor thesis only presents a progress report about the development. In this chapter, current limitations of the application are described and there is still some future work that needs to be done to get the semester planner ready for students at the JKU.

5.1 Current Limitations

In the application's current state, a number of limitations apply. Firstly, it is not yet accessible by students but only by developers, which is for a variety of reasons.

During the development of the app, members of the IM provided real data for testing purposes. This consists of actual data from the production database, but anonymized to ensure the GDPR is not validated and no privacy issues occur. This includes real student information without names and fake student numbers. These test student numbers can then be used to query course information, including information related to already completed courses, to filter them out from the course list. In the future, real student numbers are used by logged-in students to the app.

Secondly, the website is not yet hosted publicly on the internet. In the future, it might be integrated directly in myJKU or hosted through a subdomain by the IM.

Lastly, currently no log-in system exists. In the future, JKU's Shibboleth SSO system will be used, which is also used for all other university systems and students use regularly. Shibboleth is a single sign-on (SSO) log-in system for websites. It enables users to sign-in with a single identity to access various system provided by organizations, which most often includes universities or public service organizations [14].

5.2 Feature proposals

During development of our semester planner, many additional features were proposed to make the application even more useful and efficient. Some of these features are already partially implemented and include but are not limited to:

- Adding own custom courses
- Blocking off a time frame or a whole day (e.g. because of work)
- Searching for courses in a specific time frame
- Searching for all courses at the JKU
- Exporting the calendar as PDF
- Exporting the selected courses in a format, from which they can be re-imported later

6 Conclusion

Our semester planner app allows an increased efficiency and accuracy of course selection and schedule management for students at the Johannes Kepler University (JKU). The semester planner tries to solve several challenges related with traditional methods of planning, which often involve cumbersome, manual processes that can be error-prone and inefficient. By integrating with JKU's existing services, the application offers real-time course information, conflict-free scheduling, and an interactive calendar view, thereby streamlining the entire planning process.

Throughout this thesis, we have discussed the key features and functionalities of the semester planner application, including its ability to display a list of available courses, provide detailed course information, and allow for easy addition and removal of courses from the calendar. Additionally, the application supports advanced functionalities, such as handling courses with multiple sessions or groups and integrating block courses into the planning view. The technical implementation of these features uses modern web technologies such as Angular and Spring.

Despite all this, the current prototype remains a work-in-progress, with several limitations. The application is not yet publicly accessible, and real student data cannot be utilized until full integration with JKU's login system is achieved. Future work will focus on overcoming these limitations.

In conclusion, our semester planner prototype demonstrates the potential for significant improvements in semester planning at JKU. By providing a user-friendly, efficient, and accurate tool for semester planning, the application has the potential to greatly enhance the student experience. The planned future enhancements and broader deployment of the application will further solidify its value as a great tool for students navigating their academic journeys.

Bibliography

[1] *LVA-Anmeldung: Erstelle deinen Stundenplan*. de. URL: https://www.jku.at/studium/studierende/dein-studienstart/anmeldung-zu-lehrveranstaltungen/ (visited on 08/11/2024) (cit. on p. 9).

[2] *myJKU - die neue Studierendenplattform der JKU*. de. URL: https://www.jku.at/studium/studierende/myjku/ (visited on 08/11/2024) (cit. on p. 10).

[3] *HTML MDN Web docs*. en. URL: https://developer.mozilla.org/en-US/docs/Web/HTML (visited on 09/25/2024) (cit. on p. 22).

[4] *Most used web frameworks amoung developers 2024*. en. URL: https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/ (visited on 09/25/2024) (cit. on p. 22).

[5] *Spring*. en. URL: https://spring.io/ (visited on 08/15/2024) (cit. on pp. 22, 23).

[6] *TypeScript*. en. URL: https://www.typescriptlang.org/ (visited on 09/25/2024) (cit. on p. 22).

[7] *Angular*. en. URL: https://angular.dev/ (visited on 08/15/2024) (cit. on p. 23).

[8] *Inversion of Control*. en. URL: https://www.baeldung.com/inversion-control-and-dependency-injection-in-spring (visited on 09/26/2024) (cit. on p. 23).

[9] JKU. *JKU - Department of Information Management*. en-US. URL: https://www.jku.at/en/information-management/ (visited on 08/15/2024) (cit. on p. 23).

[10] Kiro Risk. *krisk/Fuse*. original-date: 2012-05-28T04:31:06Z. Aug. 2024. URL: https://github.com/krisk/Fuse (visited on 08/25/2024) (cit. on p. 26).

[11] ReactiveX. *ReactiveX/rxjs*. original-date: 2015-03-15T06:17:10Z. Aug. 2024. URL: https://github.com/ReactiveX/rxjs (visited on 08/25/2024) (cit. on p. 27).

[12] MDN. *Window: localStorage property - Web APIs | MDN*. en-US. July 2024. URL: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage (visited on 08/25/2024) (cit. on p. 27).

[13] Angular. *Angular - Introduction to services and dependency injection*. de. URL: https://v17.angular.io/guide/architecture-services (visited on 08/11/2024) (cit. on p. 27).

[14] Shibboleth Consortium. *Shibboleth Consortium - Shaping the future of Shibboleth Software*. en-GB. URL: https://www.shibboleth.net/ (visited on 08/27/2024) (cit. on p. 29).