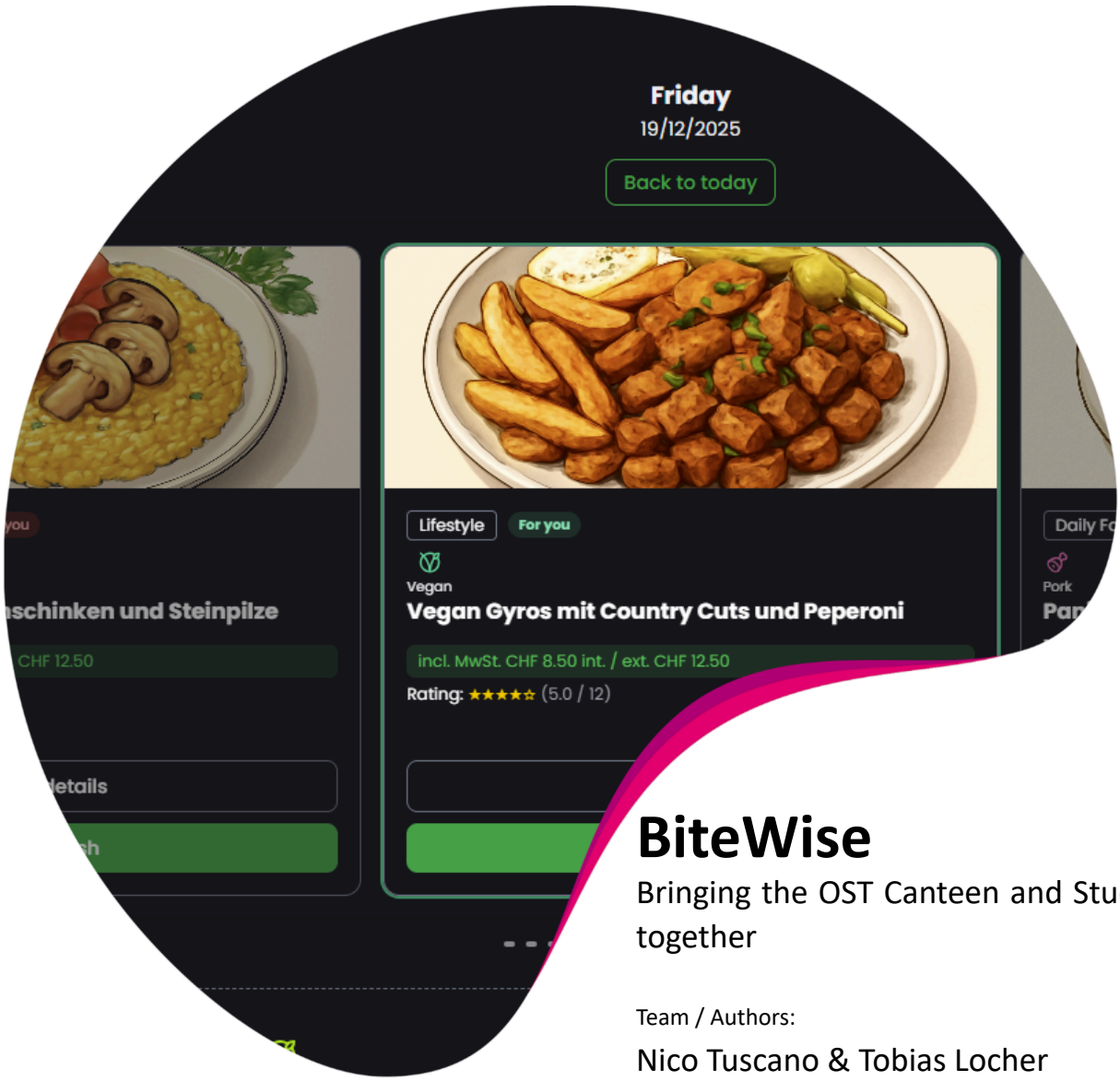

BiteWise

Bringing the OST Canteen and Students together

Team / Authors:

Nico Tuscano & Tobias Locher

Project advisors:

Prof. Dr. Mitra Purandare

Clemens Meier

Date:

19.12.2025 – HS 2025

Studienarbeit:

OST – Eastern Switzerland
University of Applied Sciences
Campus Rapperswil - Jona

Degree:

Bachelor of Science in Computer Science

Abstract

BiteWise is a web-based sub-platform developed within the existing Smart Eating ecosystem to address the specific needs of the canteen at the University of Applied Sciences of Eastern Switzerland (OST). Smart Eating is an ongoing research and development project at OST that focuses on nutrition planning, AI-supported enrichment of food recipes, and serves as the technical and conceptual foundation for BiteWise.

At present, the OST canteen publishes its menu plans mainly as static PDF documents and receives only limited feedback from students, primarily via generic online forms. This results in fragmented data and provides little actionable insight for systematic menu improvement.

The objective of this project is to design and implement a functional MVP that brings students and the OST canteen closer together by establishing a structured feedback loop between students and canteen and transforms the existing communication into an interactive, student-centered system. The platform enables students to access current canteen menus digitally, receive personalized dietary recommendations based on individual preferences, and provide structured feedback on individual meals or on the canteen in general. At the same time, menu ratings and qualitative feedback are made directly accessible to the canteen, supporting evaluation and informing future menu planning.

An OST-wide questionnaire was conducted during the requirements analysis, providing early insights into student expectations, priorities, and overall acceptance of the proposed solution. An iterative UX process from prototype design to validation followed, and a summative usability and acceptance test with the OST students was conducted first on the prototype and later on the fully implemented application to validate the MVP. The results indicate that BiteWise improves usability, transparency, and feedback quality compared to the existing PDF- and form-based approach, confirming its effectiveness as a functional MVP for the OST canteen and providing a solid foundation for future extensions.

Management Summary

BiteWise is a web-based sub-platform developed within the Smart Eating ecosystem to address the specific context of the canteen at the University of Applied Sciences of Eastern Switzerland (OST). Smart Eating is an ongoing research and development project at OST that focuses on digital nutrition planning and AI-supported enrichment of food recipes. BiteWise builds on this foundation and transfers its concepts and technologies to the campus canteen environment.

Currently, the OST canteen publishes its menu plans mainly as static PDF documents. Student feedback is collected only to a limited extent and primarily via generic online forms. This approach results in fragmented and weakly structured data, making it difficult for the canteen to gain meaningful insights into student preferences and to systematically improve menu planning.

The objective of this semester project was to design and implement a functional Minimum Viable Product (MVP) that establishes a structured feedback loop between students and the canteen. BiteWise introduces an interactive, student-centered system. Students can access up-to-date canteen menus digitally, view detailed nutritional information, and receive personalised dietary recommendations based on individual preferences.

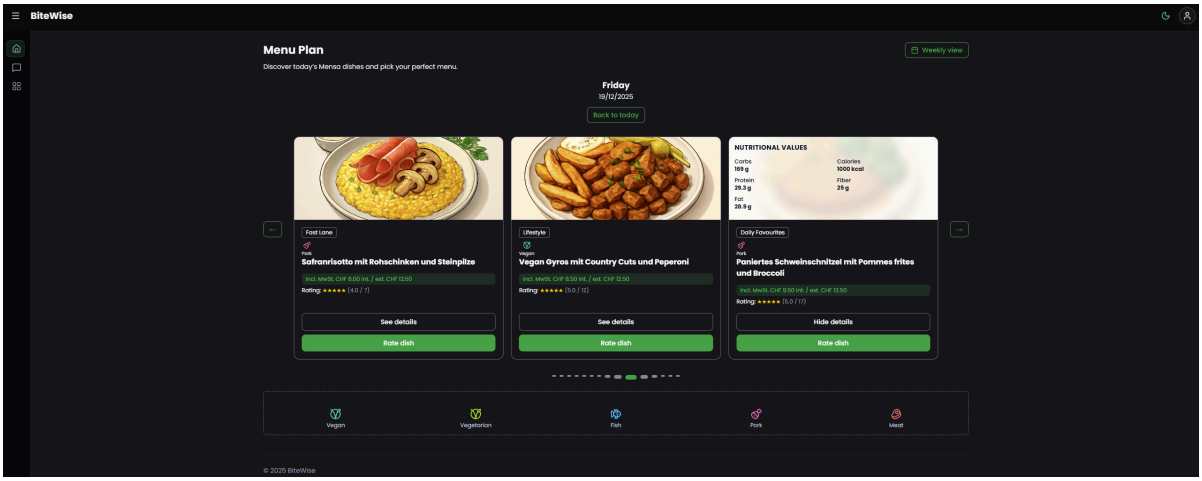

Figure 1: Daily Menu View

Personal dietary matching highlights menus that best fit a user's preferences and visually distinguishes unsuitable options.

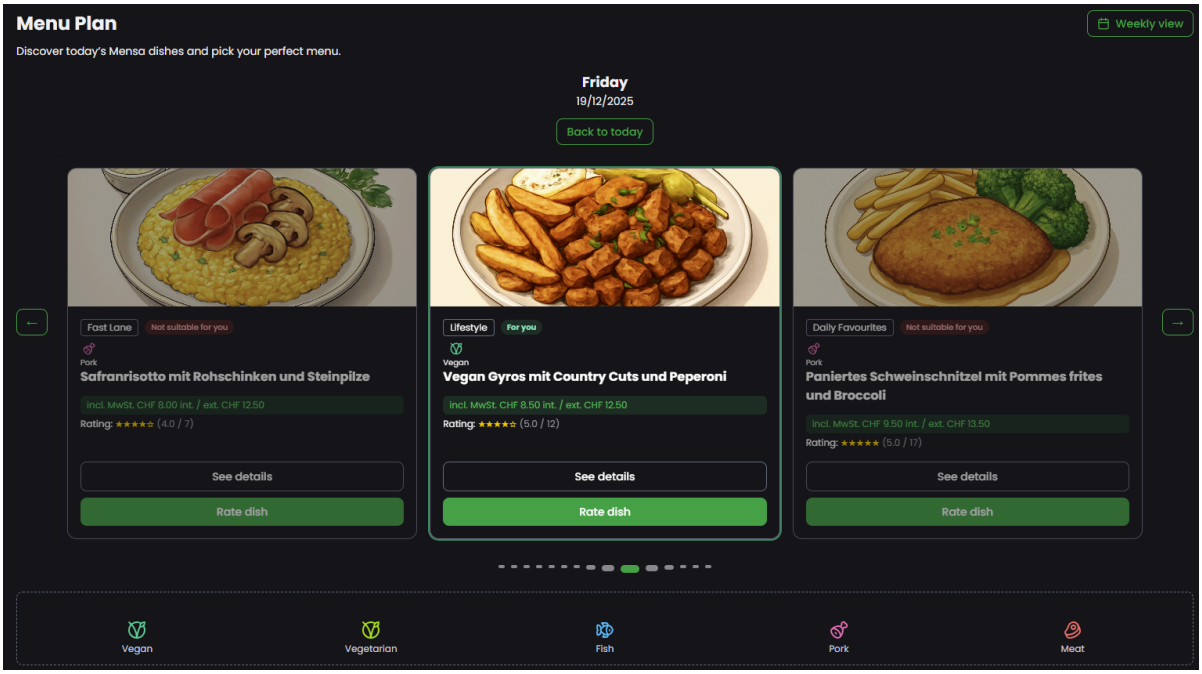

Figure 2: Personal Dietary Matching

In addition, students can provide structured feedback on individual meals as well as submit general feedback or menu ideas to the canteen.

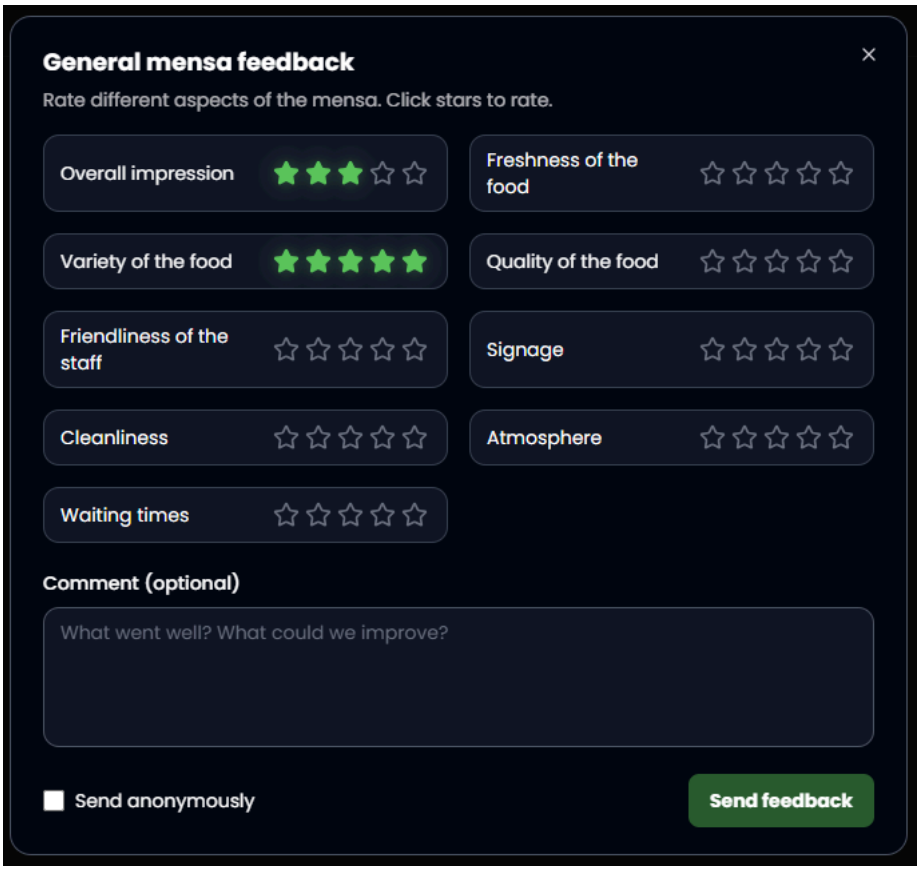

Figure 3: General Feedback View

Through a dedicated canteen dashboard, canteen staff can import and manage menus as well as access menu ratings and qualitative feedback, enabling better visibility into student responses and supporting future menu planning decisions.

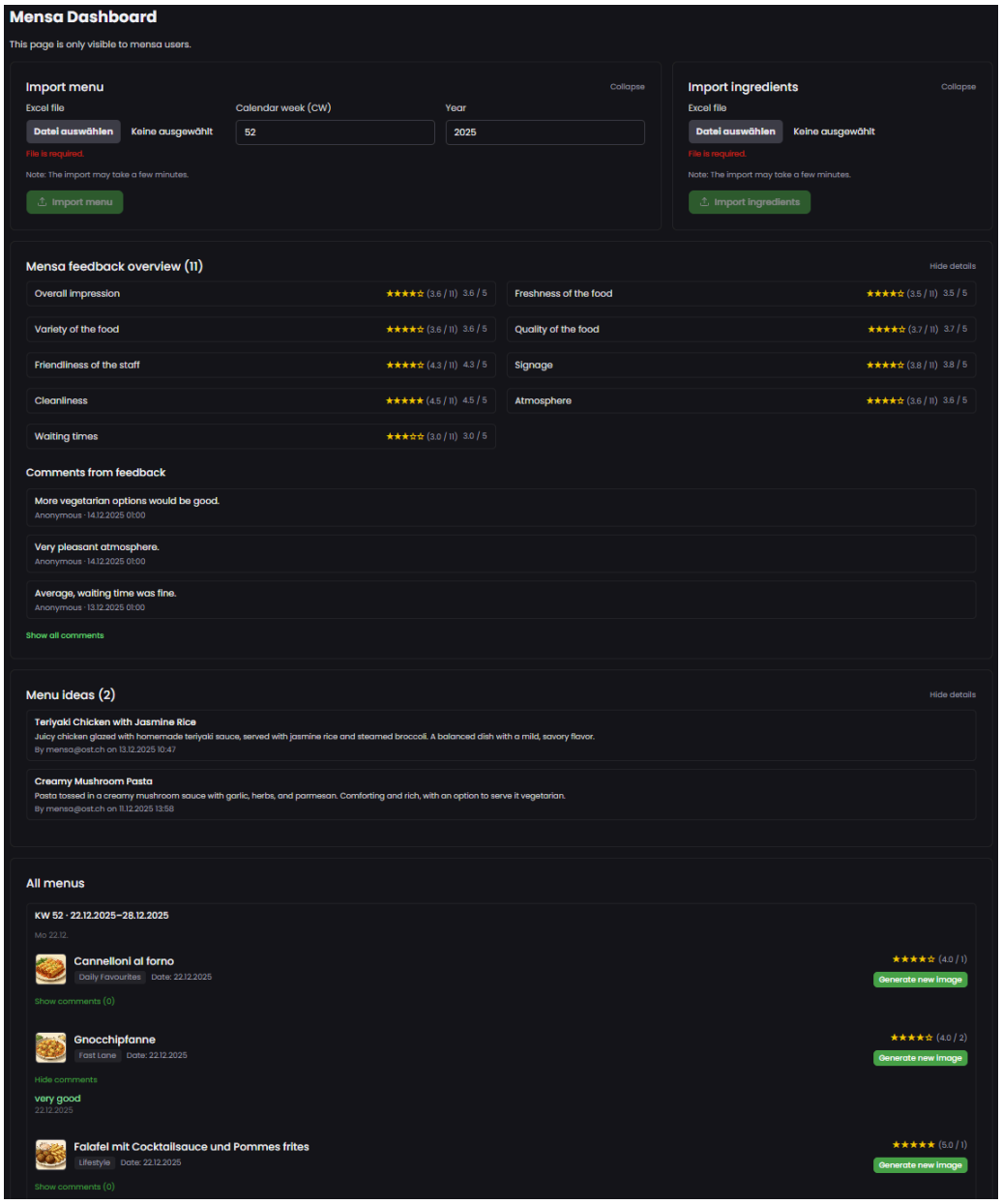

Figure 4: Canteen Dashboard

From a technical perspective, BiteWise was implemented using the same core technology stack as the existing Smart Eating project to ensure consistency and seamless integration. The frontend is implemented with React, providing a responsive and modern user interface. Backend services are realized through an ASP.NET Core API, while a PostgreSQL database is used. In addition, a Python-based service is integrated for AI-supported image generation. Although BiteWise is presented to end users as a standalone and clearly separated platform, its implementation is closely integrated with the existing Smart Eating services and functionality.

The development process followed established software engineering practices. An initial requirements analysis was conducted for both students and canteen stakeholders, followed by architectural design and iterative implementation. A strong focus was placed on user experience: an early UX prototype was created, evaluated through usability tests, and refined before implementation. After completion of the MVP, summative usability and acceptance tests were conducted with OST students on the fully implemented web application.

The evaluation results show that BiteWise improves usability, transparency, and feedback quality compared to the existing PDF- and form-based approach. Test participants were able to complete core tasks without guidance and expressed a generally positive impression of the system. These findings confirm that BiteWise is suitable as a functional MVP for the OST canteen and provide a solid foundation for future extensions, such as broader analytics, extended canteen functionality, and deeper integration into long-term menu planning processes.

Table of Contents

1 Introduction

This semester project is based on the already advanced Smart Eating project of the Institute for Software (IFS) at the University of Applied Sciences of Eastern Switzerland (OST), which deals with critical global challenges such as healthy nutrition, affordable food and environmental protection. (OST, 2025) The aim is to develop a sub-platform that brings students and the OST canteen closer together. This chapter explains the background and aim of this thesis in more detail.

1.1 Background

The Smart Eating platform for nutrition planning is already at an advanced stage of development and has already been part of several SAs and BAs. It has a recipe database that is enriched with nutritional information using AI. It is also already possible to address individual needs thanks to the integration of official reference values for nutrient intake from the Federal Office of Public Health (FOPH).

1.2 Task

The following sub-chapters are copied and translated from the official task.

1.2.1 Aim of this work

The aim of this semester project is to develop a sub-platform that connects students with the canteen on their campus. The aim is to draw students' attention to canteen meals that both suit their health needs and correspond to their personal preferences. At the same time, the data generated in this process will be incorporated into the canteen's menu planning in the medium and long term.

The focus is on the following core functions:

1. Live Menu Sync – real-time synchronisation of the dishes on offer (personalised)
2. Feedback Loop for the cafeteria – direct feedback from students
3. Personal Dietary Matching – personal recommendations based on nutritional profiles
4. Hybrid Meal Planning Mode – combination of cafeteria offerings with individual eating habits
5. Sustainability Insights – evaluations of sustainability and food waste

1.2.2 Task

The following subtasks are to be completed

- Analysis:
  - Analyse the needs of students and canteen management and derive the non-functional requirements (NFR) from this
  - Determine functional requirements (FR) (necessary and optional)
- Architecture & Design:
  - Select a suitable architecture and technology stacks – appropriate for the platform's existing services
  - Design data model and interfaces
  - Create mock-ups for central app functions
- Implementation & testing:
  - Implement core functions (see objective of the thesis)
  - Document whether FRs and NFRs could be fulfilled
  - Develop and implement test concept
  - Describe expansion options

1.2.3 Expected results

- Functioning app prototype with a selection of the defined core functions
- Documentation of architecture, requirements, tests and ideas for expansion
  - The documentation should document progress and the final result
- Initial evaluations of how feedback and matching data can be incorporated into long-term menu planning

1.3 Context Delimitation

The aim of the project was to create a functional sub-platform for the canteen that makes the best possible use of the existing Smart Eating project for our use cases. Due to time constraints, the goal is to create a functional MVP, but it is rather unrealistic to expect to create a fully finished product that can be used immediately in everyday university life. The focus is primarily on the needs of the students, but also on the canteen, which is why a requirements analysis with design process and usability tests was carried out.

The aim of the project is not to improve existing Smart Eating processes, such as nutrition calculation, but to use them as effectively as possible for our application. The focus is also more on the platform than on precise nutrition data, as these are use cases of the existing Smart Eating project.

Any necessary changes to the existing Smart Eating platform required for successful implementation will be documented, together with the steps needed to achieve a deployable product.

1.4 Organisational basic conditions

The sub-platform will be developed as part of an SA. Each team member will receive 8 ECTS credits for this, which corresponds to an approximate workload of 240 hours. With two students, this results in a total workload of 480 hours. Spread over 14 weeks of project time, this amounts to an approximate average of 17 hours per person per week.

2 Requirements Analysis

This chapter deals with the functional and non-functional requirements and how they were defined and collected.

2.1 Requirements Elicitation

Before functional and non-functional requirements can be defined, they must first be collected. This can be done through various methods, such as interviews, surveys and document analysis. The goal is to gather as much information as possible about the needs and expectations of the stakeholders.

In a first step, the official task description is used to define basic requirements. From that, two main stakeholder groups were identified: the canteen and the students. Thus, a more specific requirements analysis for these groups was needed.

2.1.1 Collecting canteen Requirements

The canteen requirements were collected through interviews and direct communication with the canteen manager (Florian Wunderlich). In the first meeting, a lot of useful information was gathered about the needs of the canteen.

Important findings included:

- The lack of feedback from the students
- How the canteen currently defines its menus: based on sales numbers.
- How the canteen currently uploads its menus: They create a PDF and upload it to several locations (e.g. for screens and websites).
- It's challenging to get nutritional data from the canteen, since they only keep track on nutritional information for the ingredients and not the whole meal. These are saved in an excel spreadsheet, that is not easily accessible for non-canteen employees.
- The canteen has not much time to fill the data into our application on their own. So, it has to be as easy as possible for the employees to fill in the data.
- The canteen is very interested in gathering feedback and additional nutritional data from the students.
- Allergens cannot be displayed in the application, due to risk of false information. Students have to ask the canteen for this information.

After that, through email and personal conversations, more specific questions were answered and some vouchers for future usability tests were obtained. The final defined requirements are listed and described in detail in the next chapter.

2.1.2 Collecting Student Requirements

The student requirements were collected through an online questionnaire with Microsoft Forms that was sent to the students. The target group was all students, lecturers and employees of OST at the Rapperswil campus. Other OST locations were excluded, since the project will only be implemented at the Rapperswil campus for our SA.

The goal of the questionnaire was to gather information about how the students currently use and think about the canteen, and if the need for such a project like BiteWise exists. Some questions were also influenced by the feedback given of the canteen.

The questionnaire then was analysed and evaluated. Important results included, that most students were not aware of the possibility to give feedback to the canteen. Also, not a lot of students usually track their diet, but most are still interested to see nutritional information of the meals. The majority of the students are interested in the project and would like to use it now after finding out about it. The final defined requirements are listed and described in detail in the next chapter.

2.2 Functional Requirements

The functional requirements of the project were defined using use cases and diagrams. They are based on the task and were formulated in more detail through requirements analysis.

2.2.1 Actors

The application will use four different main actors and are described in the table below.

Actor	Description
User	Guests of the canteen, e.g. students, OST employees, lecturers, etc. They use the app to view menus, give feedback and receive recommendations.
Admin	Person with extended permissions, e.g. someone from the canteen, who can create, update, or delete menus. Distinguished by authorization from a normal canteen Employee.
Mensa Employee	Employee of the canteen who mainly uses the dashboard to view evaluations, feedback, and statistics provided by the users.
Scheduler	System component that executes automated tasks such as importing weekly menus or generating nutrition data.

Table 1: Actors

2.2.2 Use Case Diagram

The use case diagram is a visual representation of the system's functionality, showing the interaction between actors and use cases. It helps define the system's scope, identify what the system does, and clarify how users achieve their goals within the system without detailing internal operations. The colors represent the priority of the use cases, which are described and explained in more detail in the next chapter.

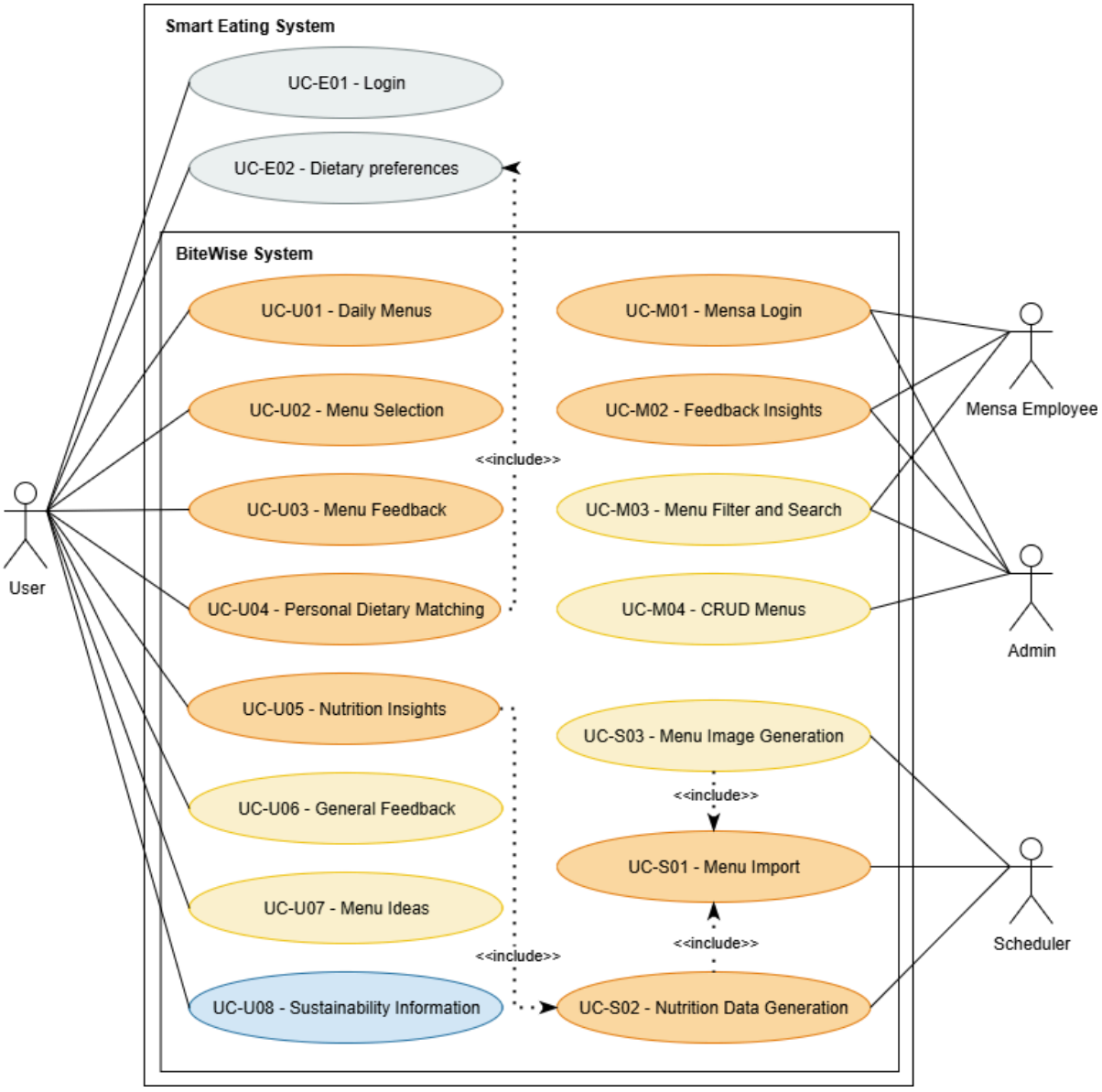

Figure 5: Use Case Diagram

,

2.2.3 Use Cases

We distinguish three categories of use cases: automated processes handled by the scheduler, functionalities from the users' perspective (Users view), and functionalities for the canteen employees (dashboard view).

The use cases are defined according to the model of Craig Larman (Larman, 2004) .

In addition, there are use cases that must be present but are already included in the current Smart Eating Project. These are listed first.

We use three priority levels (High / Mid / Low) for classification. They define what is essential for the MVP in this SA and what is less important. The more detailed classifications are described in more detail here.

Priority	Meaning
High Prio	These are the critical core functions and must be implemented for the MVP within this SA.
Mid Prio	Will be implemented according to priority and depending on available time, optional for the MVP, but it would be good to implement at least some of them.
Low Prio	Nice to have; currently not planned for this SA. Can be implemented in future projects or if there is time left over here.

Table 2: Priority Legend (Use Cases)

These following use cases are described as they were initially defined and do not represent the final state of implementation. For details on how the use cases were implemented in the final product and which aspects were adjusted, see chapter **Implementation** .

2.2.3.1 Existing Use Cases

UC-E01 - Login
Users can register and log in with an account.

Table 3: UC-E01 - Login

UC-E02 - Dietary preferences
Users can enter their dietary preferences in the system.

Table 4: UC-E02 - Dietary preferences

2.2.3.2 Users View

UC-U01 - Daily Menus	***High Prio***
Users can view the daily menus.	

Actors	User
Preconditions	None
Frequency of Use	High
Triggers	User opens the app or navigates to the menus page.
Main Scenario	1. The system automatically loads the current daily menus from the DB. 2. The menu for the current day is displayed, including dish names and basic information like Vegan.
Alternative Scenario	• If the canteen is closed, the app indicates no meals are available today. • The user can navigate to view menus of the following week or the previous week.

Table 5: UC-U01 - Daily Menus

UC-U02 - Menu Selection	***High Prio***
Users can select which menu they actually consumed so that it is added to their personal meal plan.	

Actors	User
Preconditions	User is logged in and daily menus are available.
Frequency of Use	Medium
Triggers	User opens the daily menu overview and chooses a menu.
Main Scenario	1. The system displays a ‘Select this menu’ option for each menu of the day. 2. User selects the menu they have eaten. 3. The system saves the selection for the user’s personal meal plan. 4. The selected menu is visually marked as chosen in the daily overview.
Alternative Scenario	• If the user does not select a menu, the personal plan for that day remains empty. • If the user wants to change their selection, they can update it.

Table 6: UC-U02 - Menu Selection

UC-U03 - Menu Feedback	***High Prio***
Users can rate each menu they have eaten using stars or text.	

Actors	User
Preconditions	User is logged in and has selected a menu they have eaten (see UC-U02 Menu Selection).
Frequency of Use	Medium
Triggers	User has selected a menu they have eaten.
Main Scenario	1. User fills out the necessary feedback form fields. 2. User submits the feedback form. 3. Feedback is saved in the DB and linked to the selected menu.
Alternative Scenario	• If the user decides not to give feedback, they can close or skip the feedback dialog without submitting.

Table 7: UC-U03 - Menu Feedback

UC-U04 - Personal Dietary Matching	***High Prio***
Users get personal recommendations for the menus based on their preferences.	

Actors	User
Preconditions	User is logged in and has entered dietary preferences in the system.
Frequency of Use	Medium
Triggers	User opens the menu overview
Main Scenario	1. The system compares the current canteen menus with the user's stored preferences (e.g. vegetarian, vegan, nutritional values). 2. The best-matching menu is outlined in green to indicate an optimal fit. 3. Menus that do not match at all with the user's preferences (e.g. meat for vegetarians, pork for users avoiding it) are greyed out or visually dimmed. 4. User sees why a menu is recommended (e.g. "fits vegetarian preference").
Alternative Scenario	• If the user has not set preferences or is not logged in, the system simply shows all menus without recommendations and provides a link to the preferences page.

Table 8: UC-U04 - Personal Dietary Matching

UC-U05 - Nutrition Insights	***High Prio***
Users can see the nutrition values of each menu.	

Actors	User
Preconditions	None
Frequency of Use	High
Triggers	User clicks on see details button for the respective menu
Main Scenario	1. System loads the nutritional values for the menu from the DB. 2. Nutritional values like Calories, Protein, Carbohydrates, Fat and Fiber are displayed.
Alternative Scenario	• If a value for a nutrition category of a menu is missing, "N/A" is displayed for that category.

Table 9: UC-U05 - Nutrition Insights

UC-U06 - General Feedback	***Mid Prio***
Users can provide general feedback to the canteen via a text field.	

Actors	User
Preconditions	User is optionally logged in
Frequency of Use	Rare
Triggers	User clicks on feedback page from navigation.
Main Scenario	1. User fills out the necessary feedback form fields. 2. User submits the feedback form. 3. Feedback form is saved in the DB.
Alternative Scenario	• If the user is not logged in, the feedback will be sent anonymously. • If the logged-in user wishes, they can send the feedback anonymously by ticking the “send anonymously” checkbox.

Table 10: UC-U06 - General Feedback

UC-U07 - Menu Ideas	***Mid Prio***
Users can submit their own menu suggestions.	

Actors	User
Preconditions	User is logged in.
Frequency of Use	Rare
Triggers	User navigates to the menu ideas page.
Main Scenario	1. User opens the menu ideas form in the app. 2. User enters the details of a suggested dish (e.g. name, description, optional ingredients). 3. User submits the suggestion via the provided button. 4. The system stores the suggestion and makes it available for the canteen employees to review.
Alternative Scenario	-

Table 11: UC-U07 - Menu Ideas

UC-U08 - Sustainability Information	***Low Prio***
Users can see how sustainable the menus are in comparison using a score (1-5 stars).	

Actors	User
Preconditions	None
Frequency of Use	High
Triggers	User clicks on see details button for the respective menu
Main Scenario	1. System loads the sustainability score for the menu from the DB. 2. Sustainability score gets displayed as a star rating from 1-5.
Alternative Scenario	• If no sustainability score for the menu exists, the score is displayed with a placeholder value like “N/A”.

Table 12: UC-U08 - Sustainability Information

2.2.3.3 Canteen Dashboard

UC-M01 - Mensa Login	***High Prio***
The canteen can login to their account to get access to the dashboard.	

Actors	Mensa Employee, Admin
Preconditions	Actor has valid login credentials.
Frequency of Use	High
Triggers	Actor opens the dashboard page.
Main Scenario	1. Actor enters username and password on the same login page used by students. 2. The system verifies the credentials and checks the account role. 3. If the account has dashboard permissions, the dashboard view is loaded. 4. Dashboard features (e.g. feedback insights, menu management) become accessible.
Alternative Scenario	• If the account does not have dashboard permissions, the user is redirected to the normal user view.

Table 13: UC-M01 - Mensa Login

UC-M02 - Feedback Insights	***High Prio***
The actor can view user feedback for each of their menus and menu ideas on a dashboard page.	

Actors	Mensa Employee, Admin
Preconditions	Actor is logged in with a canteen Employee or Admin role.
Frequency of Use	High
Triggers	Actor logs in with a canteen Employee or Admin account.
Main Scenario	1. System loads feedback for the menu from the DB. 2. Feedback gets processed and displayed with some graphs to get the most out of the user feedback. 3. Students menu ideas gets displayed.
Alternative Scenario	• If no feedback is found for the menu, an empty section with an info text is shown.

Table 14: UC-M02 - Feedback Insights

UC-M03 - Menu Filter and search	***Mid Prio***
The actor can filter and search for existing menus.	

Actors	Mensa Employee, Admin
Preconditions	Menus exist in the system
Frequency of Use	Medium
Triggers	Actor filters or searches for a menu.
Main Scenario	1. The system provides filter options: Vegan, Vegetarian, No Pork, Lactose-free, Gluten-free. 2. The system provides a search option to search by menu name. 3. User selects one or multiple filters (logical AND) or searches by menu name. 4. The menu overview updates dynamically and only shows menus that meet the selected criteria. 5. User can remove filters or search string to return to the full menu list.
Alternative Scenario	• If no menus match the selected filters or search string, the system displays a message such as “No menus available for the chosen criteria”.

Table 15: UC-M03 - Menu Filter and search

UC-M04 - CRUD Menus	***Mid Prio***
An admin can create, view, update, and delete menus.	

Actors	Admin
Preconditions	Admin is logged in.
Frequency of Use	Medium
Triggers	Admin navigates to the menu management section in the dashboard.
Main Scenario	1. The system shows a list of all menus currently stored in the system (imported or created). 2. Admin can create a new menu by entering dish details (name, description, dietary info, etc.). 3. Admin can open a menu to view all details. 4. Admin can update an existing menu (e.g. correct mistakes, add missing information). 5. Admin can delete a menu that is no longer valid. 6. The system saves the changes and updates the data for users.
Alternative Scenario	• If the menus have already been imported correctly, manual changes are optional and usually not needed.

Table 16: UC-M04 - CRUD Menus

2.2.3.4 System

UC-S01 - Menu Import	***High Prio***
Every week, the scheduler automatically imports the canteen PDFs from the current canteen website and uses them to create the menus.	

Actors	Scheduler
Preconditions	The canteen website is accesible.
Frequency of Use	High
Triggers	Scheduler automatically triggers once a week.
Main Scenario	1. Scheduler gets the canteen PDF from the canteen website. 2. The PDF gets analysed and the menus of the week are created. 3. Additional values such as nutritions, sustainability and more are generated. 4. Everything gets saved in the DB.
Alternative Scenario	• If the website is not accessible, the scheduler job fails and a warning is displayed for the Admins.

Table 17: UC-S01 - Menu Import

UC-S02 - Nutrition data generation	***High Prio***
The scheduler automatically generates nutritional information from the imported menus.	

Actors	Scheduler
Preconditions	Menus have been successfully imported into the system.
Frequency of Use	High
Triggers	Scheduled task runs after menu import is completed.
Main Scenario	1. The scheduler processes the imported menus. 2. The system enriches each menu with nutritional information (e.g. calories, proteins, etc.). 3. The calculation is based on the currently used AI for nutrition data from Smart Eating. 4. The generated nutrition data is stored together with the menus. 5. Users can later view these values in the web app (e.g. via Nutrition Insights).
Alternative Scenario	• If required nutritional information is missing or incomplete, the system marks the data as “not fully available”.

Table 18: UC-S02 - Nutrition data generation

UC-S03 - Menu Image Generation	***Mid Prio***
The scheduler automatically generates a suitable image for each menu using AI.	

Actors	Scheduler
Preconditions	Menus have been successfully imported into the system.
Frequency of Use	Medium
Triggers	Scheduled task runs after nutrition data generation is completed.
Main Scenario	1. The scheduler processes all imported menus without existing images. 2. The system uses an AI model to generate a realistic image that visually represents each menu. 3. The generated image is linked to the corresponding menu entry in the database.
Alternative Scenario	• If image generation fails, a default placeholder image is used instead. • If the AI service is temporarily unavailable, the job retries at the next scheduled run.

Table 19: UC-S03 - Menu Image Generation

2.2.4 Use Case Order & Planning

After implementing the first use case, the other use cases were prioritised in order and approximate sprints were estimated in which they should be completed.

Use Case ID	Use Case	Priority	Planned Completion	Completed in
UC-U01	Daily Menus	*High Prio*	Sprint 6	Sprint 8
UC-U02	Menu Selection	*High Prio*	Sprint 7	Sprint 10
UC-U03	Menu Feedback	*High Prio*	Sprint 7	Sprint 10
UC-S01	Menu Import	*High Prio*	Sprint 8	Sprint 10
UC-S02	Nutrition Data Generation	*High Prio*	Sprint 8	Sprint 11
UC-U05	Nutrition Insights	*High Prio*	Sprint 9	Sprint 12
UC-U04	Personal Dietary Matching	*High Prio*	Sprint 9	Sprint 12
UC-M01	Mensa Login	*High Prio*	Sprint 9	Sprint 11
UC-M02	Feedback Insights	*High Prio*	Sprint 10	Sprint 12
UC-U06	General Feedback	*Mid Prio*		Sprint 11
UC-U07	Menu Ideas	*Mid Prio*		Sprint 11
UC-S03	Menu Image Generation	*Mid Prio*		Sprint 11
UC-M04	CRUD Menus	*Mid Prio*		-
UC-M03	Menu Filter and Search	*Mid Prio*		-
UC-U08	Sustainability Information	*Low Prio*		-
UC-E01	Login	*Existing*	Sprint 6	Sprint 6
UC-E02	Dietary Preferences	*Existing*	with UC-U04	Sprint 12

Table 20: Use Case Order & Planning

2.3 Non-Functional Requirements

In addition to the functional requirements, several non-functional requirements (NFRs) were defined. They describe quality aspects of the system that are not directly related to specific features but are essential for usability, reliability, performance, and data protection. To ensure clarity and verifiability, the NFRs were formulated according to the SMART principle (Specific, Measurable, Achievable, Relevant, Time-bound).

The following NFRs were derived from the project's objectives, the analysis of user and canteen needs, and the task definition. Each NFR is documented with its verification approach, priority, and criteria for successful fulfillment.

NFR-1 - Responsive Design	***High Prio***
ID	NFR-1
Title	Responsive Design
Description	The web app must provide a responsive user interface that adapts to smartphones (≥360px width), tablets (≥768px width) and desktops (≥1200px width) to ensure smooth usability across screen sizes.
Verification Tools	Manual tests on 3 different devices (e.g. PC, Tablet, Smartphone)
Defined on	03.10.2025
Verification Date	18.12.2025
Fulfillment Level	Fulfilled
Verification Details	The UI is tested on a representative set of devices and screen resolutions. All central functions (view menus, feedback, dashboard) remain fully usable without layout errors or inaccessible elements.

Table 21: NFR-1 - Responsive Design

NFR-2 - Intuitive Usability	***High Prio***
ID	NFR-2
Title	Intuitive Usability
Description	The web app must be intuitive to use for both students and canteen employees.
Verification Tools	Usability tests with students and canteen employees
Defined on	03.10.2025
Verification Date	12.12.2025
Fulfillment Level	Partially fulfilled
Verification Details	In a usability test with 3 students, 100% of the participants were able to complete core tasks without external help. Usability tests were not conducted on canteen employees due to time constraints.

Table 22: NFR-2 - Intuitive Usability

NFR-3 - Menu Import Reliability	***High Prio***
ID	NFR-3
Title	Menu Import Reliability
Description	The system must successfully import canteen menus in at least 90% of the tested cases.
Verification Tools	Functional tests with 10 example canteen menus
Defined on	03.10.2025
Verification Date	18.12.2025
Fulfillment Level	Fulfilled
Verification Details	Verification is performed by running the manual import on the canteen dashboard with 10 different self-created menus. 10 out of 10 menus are correctly imported without data loss (dish names, dates, dietary tags etc.).

Table 23: NFR-3 - Menu Import Reliability

NFR-4 - Data Privacy & Anonymity	***High Prio***
ID	NFR-4
Title	Data Privacy & Anonymity
Description	User data, including personal information and dietary preferences, is already stored securely in the existing Smart Eating platform and is not part of this project's implementation. However, our sub-platform must handle these data in a way that ensures they remain protected and are never exposed to unauthorized parties. Matching results and dietary information may only be used in aggregated or anonymised form for canteen insights. It must never be possible for the canteen to link data back to an individual user, where not necessary.
Verification Tools	Code reviews, database inspection, functional tests with sample accounts
Defined on	03.10.2025
Verification Date	18.12.2025
Fulfillment Level	Fulfilled
Verification Details	Verification ensures that no sensitive user data is leaked to the dashboard, dietary preferences are only visible to the user, and anonymous feedback cannot be linked back to a user account.

Table 24: NFR-4 - Data Privacy & Anonymity

NFR-5 - Multilingual Support	***High Prio***
ID	NFR-5
Title	Multilingual Support
Description	The web app must support both German (default) and English. Users can switch languages dynamically within the UI. 100% of all informational and interface texts in the application must be available in both languages. Imported menu texts from the canteen PDFs are automatically translated into English during the import process (see UC-S01 Menu Import).
Verification Tools	Manual tests by switching the UI language and verifying that all texts are displayed correctly in both German and English.
Defined on	12.10.2025
Verification Date	18.12.2025
Fulfillment Level	Fulfilled
Verification Details	Verification ensures that all user-visible texts, including menus, labels, and error messages, are correctly localized.

Table 25: NFR-5 - Multilingual Support

NFR-6 - Page Load Performance	***Mid Prio***
ID	NFR-6
Title	Page Load Performance
Description	All main pages of the application must load in less than 1 second in 90% of the cases under normal usage conditions.
Verification Tools	Performance tests with Lighthouse
Defined on	03.10.2025
Verification Date	18.12.2025
Fulfillment Level	Fulfilled
Verification Details	Verification is performed by measuring load times of the menu overview, feedback page and dashboard. Each page loads in <1s in at least 9 out of 10 test runs under OST WLAN conditions.

Table 26: NFR-6 - Page Load Performance

2.4 Domain Model

For the domain model, the domain requirements were analysed first. Together with the supervisors, it was decided that the existing model should be expanded with as few entities as possible and that as much of the current Smart Eating project as possible should continue to be used. To this end, a compromise was made to allow for a slightly more complex representation in some areas, but this has the advantage that, for example, the nutritional value generation of the recipes can continue to be used. This also results in some restrictions, which are described in more detail in the model.

The domain model shown only shows the existing entities and properties of Smart Eating that are relevant in our context. As already mentioned, these are the recipes, the nutritional values for personal dietary matching, the user management including feedback, the images and the tagging.

It only had to be expanded for the daily menu plan of the canteen, the menu ideas and the general canteen feedback.

Although the domain model is closely related to the database schema, it does not represent the actual database model. Instead, it provides a conceptual overview of the core entities and their relationships within the domain.

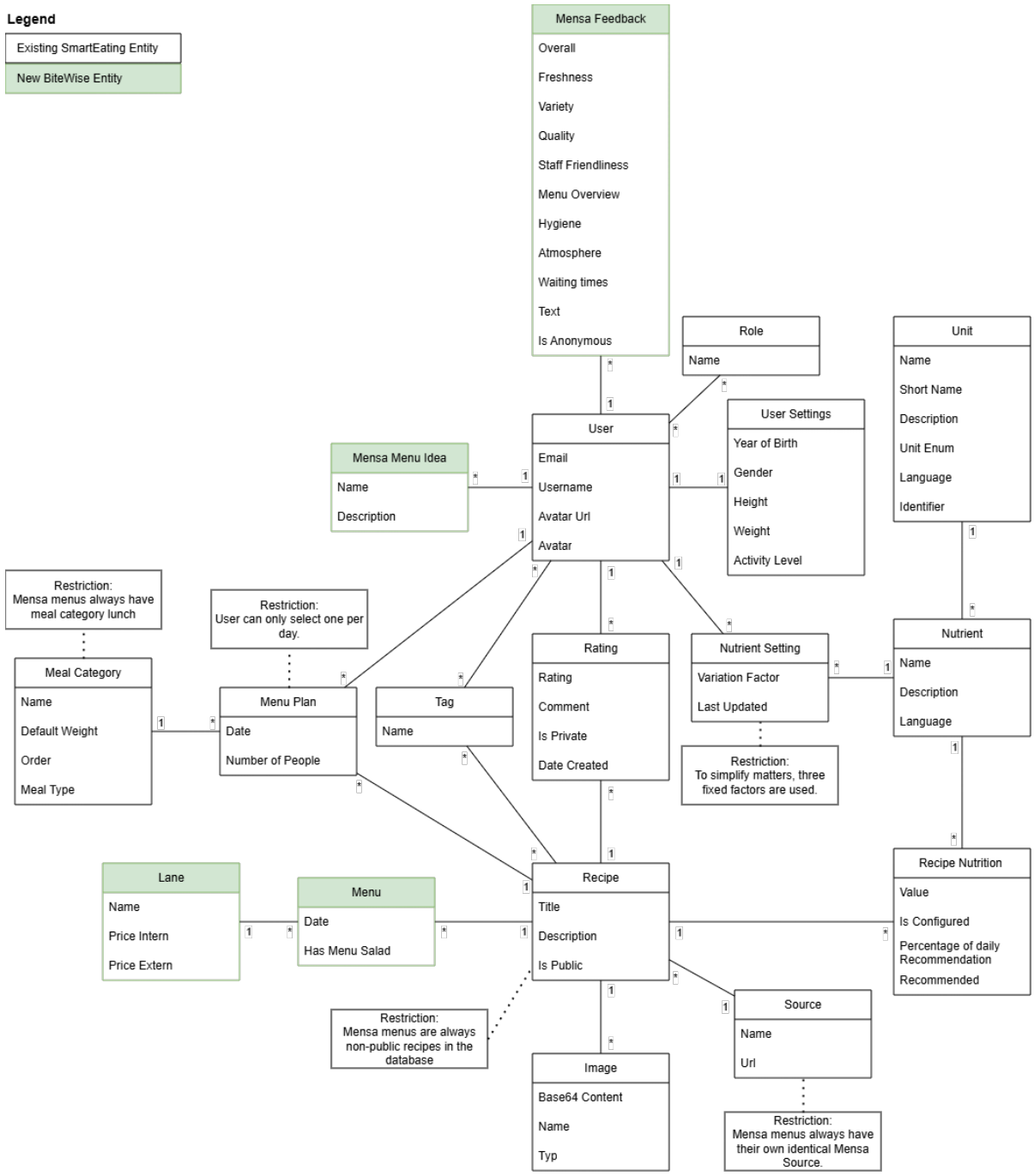

Figure 6: Domain Model

3 Design & Prototyping

After analysing the requirements, the next step was to visualise them in a prototype. We decided to focus on the user view, as this is ultimately what appeals to OST students and is used by as many of them as possible. It is also quite feasible for us to conduct usability tests with the students, which is also part of our project and will be described in more detail in the following chapters.

3.1 Figma Prototype

We chose Figma for prototype development. It offers many features, especially for making prototypes interactive and thus ideal for usability testing. We can also already work with Tailwind, which we plan to use later on the actual website. We also made use of Figma AI during development. With precise application and good prompting, it was very helpful in designing more efficiently and making the prototype interactive.

The following image shows the start page of the initial prototype. The entire prototype can be seen in the comparisons in the **Results & Pain Points chapter** and later in the **Final Prototype chapter** .

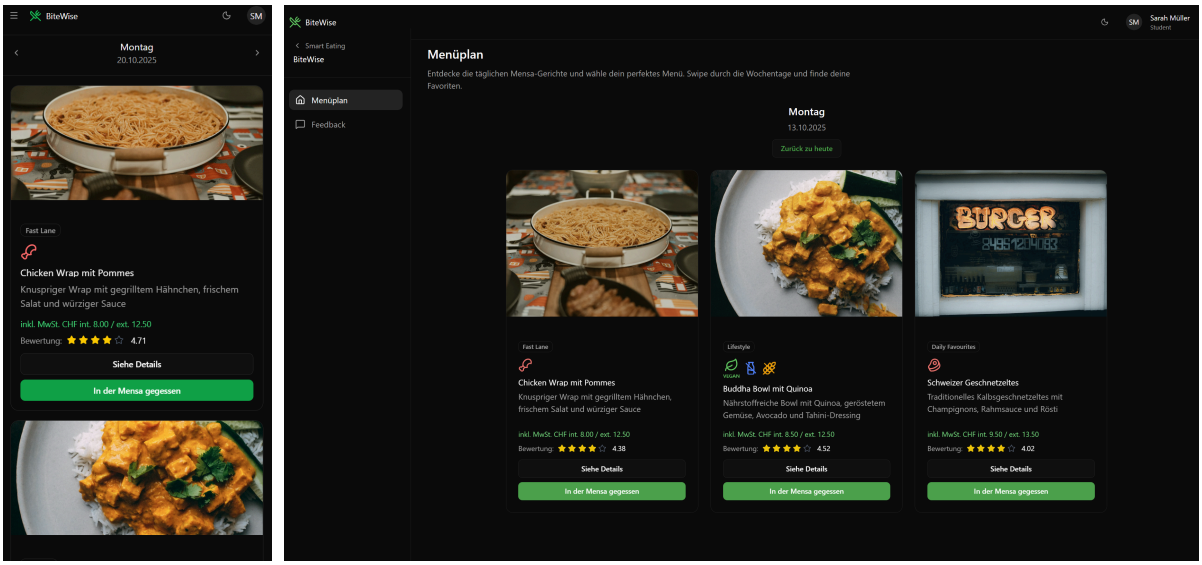

Figure 7: Start Page Initial Prototype

Our focus was on creating a user-friendly and intuitive design that also looks good. Our goal was to achieve a more attractive and less generic design than in the existing Smart Eating Project. That's another reason why we used Tailwind.

In addition to the desktop version, the app and therefore also the prototype should look good and work well on mobile devices. Since such web apps are increasingly being used on mobile devices today, we decided to take a mobile-first approach when designing the Figma prototype.

3.2 Usability Tests

After designing the Figma prototype, it was important for us to get initial feedback at this stage. To do this, we decided to conduct usability tests directly on the clickable Figma prototype.

The usability tests take place in the last week of the prototype phase.

3.2.1 Test form & content

The tests take place in an undisturbed room at OST. For simplicity, both of us sit next to the test subject without interfering in the test. There is a test leader who conducts the test and gives initial instructions at the start. The second person takes notes.

The test subject performs the tests on a device provided and prepared by us. This setup simplifies the process for the participant and minimises the risk of technical issues. We plan to conduct tests on both notebooks and smartphones, with at least two sessions per device type.

The screen of the test device is recorded and, with the consent of the test subject, the audio as well.

A test should take approximately 15 minutes.

3.2.2 Pre- and post-interview

Before the test, we conduct a pre-interview to obtain some basic information about eating habits, tracking and visits to the canteen.

After the tests, there is a post-interview. We ask about the overall impression of the web app, what participants liked, and what could be improved. We also ask spontaneous questions that arose from our observations during the test.

The aim of the interviews is to gain additional insights that were not apparent from the observations.

We have already predefined the fixed questions, which can be found in Appendix B .

3.2.3 Screener

Before selecting our test participants, we defined clear criteria to ensure a representative group. To this end, we created a screener that defines the target audience and selection criteria for the usability tests.

The screener ensures that the selected participants represent the actual target audience of the BiteWise platform. This helps to obtain relevant insights into usability, user expectations, and potential improvements.

Primary target group
OST students who regularly use the canteen. (at least 4 people)

Secondary target group
- Students who are interested in healthy eating and, ideally, already track their diet. (at least 2 people)
- Students who do not eat certain foods (e.g. vegan, vegetarian, lactose-free) (at least 1 person)

Additional group (optional)
Canteen staff who will test the dashboard.

Inclusion criteria
- Enrolled student at OST
- Studies at the Rapperswil campus
- Interest in nutrition/canteen offerings
- Openness to digital tools and motivation to test new features
- Willingness to participate in a 15-minute test

Exclusion criteria
- Individuals who are or were involved in the development of the Smart Eating platform

Recruitment strategy
- Distribution through personal conversations, via lecturers or information flyers
- Voluntary participation, with a canteen voucher for lunch as a thank you.

All participants give their consent before the session starts, especially regarding screen and audio recording. Personal data and recordings are handled confidentially and used exclusively for evaluation purposes.

3.2.4 Test Scenarios

Before the tests, we created various test scenarios, which the person reads through during the test and carries out the task without further assistance. The aim of the test scenarios is to tell a kind of story and not to dictate too much to the user what they should do. This allows us to test whether the test person understands the UI or, if necessary, has other ideas based on intuition.

We created nine test scenarios, covering all the functions of the user view. These can be found in Appendix C .

3.2.5 Tests performed

Usability tests were done with 5 students, 4 male and 1 female. They were from three different degree programmes. Two of them already track what they eat, two pay attention to a lactose-free diet, and one is vegetarian.

We conducted the tests on the device preferred by each person. In most cases, this was the smartphone, which was what we expected. We conducted four tests on smartphones and one on a desktop device.

So, we had a good mix and hit the goals of the screener. This helped us get useful insights from the tests, which can be found in the next chapter.

3.3 Results & Pain Points

This chapter discusses the test results obtained from the test observations and pre- and post-interviews. The focus is on the pain points and how we improved the prototype accordingly. To show the differences, image comparisons of the initial and final prototypes are provided for each discussed point. Screenshots are sometimes shown for mobile or desktop version, to illustrate the differences better. The interview transcripts and our test reports can be found in Appendix D and Appendix E .

3.3.1 Overall Impression

The general impression of the test subjects regarding BiteWise was very positive. All of them found the prototypes to be fundamentally good and were able to benefit from them. Fortunately, there was also a lot of constructive feedback and other perspectives, which helped us to draw important conclusions. This will be discussed in the following chapters.

3.3.2 Daily Menu View & Menu Feedback

Many people still had difficulties with the icons and their meanings in the Daily Menu View. For example, one person did not know whether the lactose icon meant lactose-free or the opposite. This was also due to the fact that the hover effects do not work on mobile phones. We have therefore labeled all icons and added a legend at the bottom.

The star rating was well understood, but some still looked for comments. However, this was also due to the vague wording of the test case. Nevertheless, we have decided not to include comments, as we do not believe that they should be made public.

The rating itself worked well, but some people did not understand that you can rate the menu after you have marked it as eaten using the button. Consequently, we need to adjust the button naming.

A test person said that the prices and menu titles did not stand out enough. These have now been enlarged and made more visually prominent.

We also planned for users to be able to swipe from day to day on their cell phones, but almost no one used this feature or even thought about it. For the actual app, it would also be a good idea to make the day and the buttons at the top sticky when scrolling. However, this was not implemented in Figma due to complexity issues.

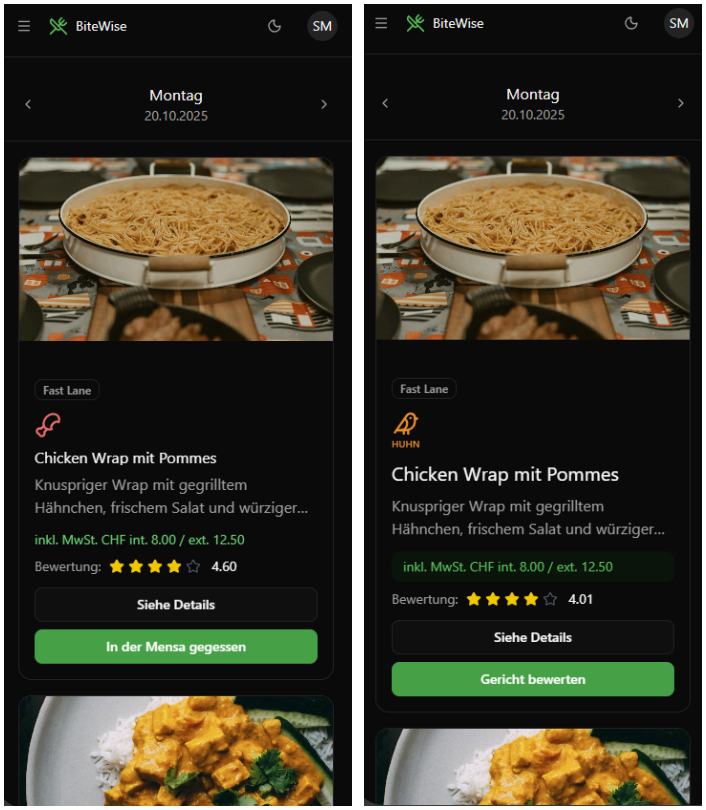

Figure 8: Daily Menu View Comparison

3.3.3 Weekly View

When navigating, there was often a desire for a weekly view, as this allows you to get an overview of the entire week more quickly. This was the biggest point in the redesign, and we have designed a new view accordingly, which is intended to display the information in a compact form. You can easily switch back and forth between the views. The start view remains the current day view.

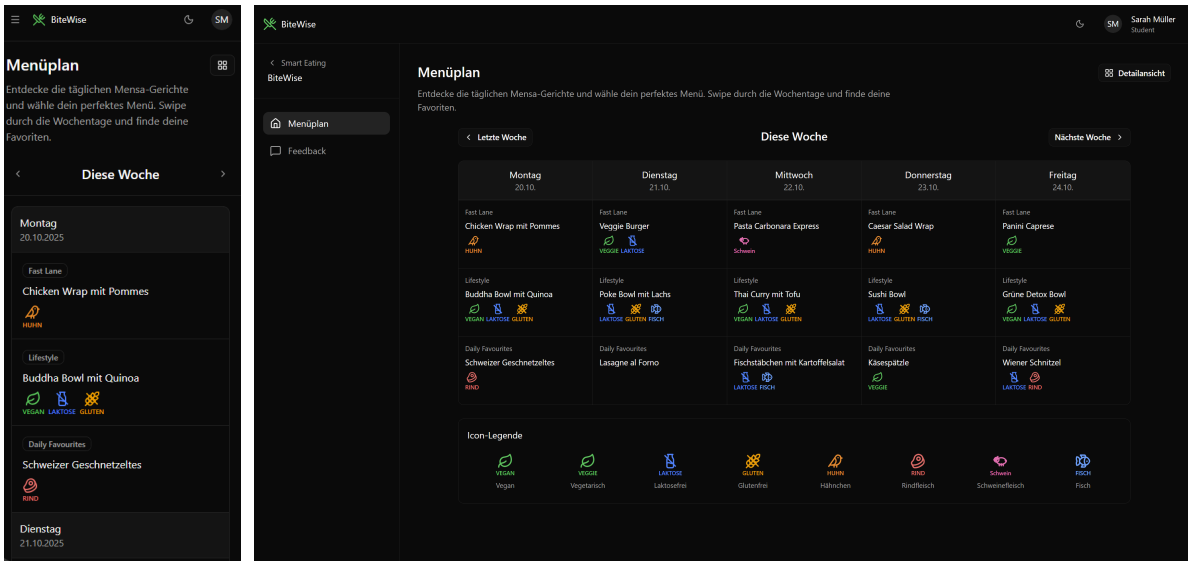

Figure 9: Menuplan Weekly View

3.3.4 Nutrition Data & Sustainability Score

The test subjects found the nutritional values to be good and also considered them useful. There was no desire for more information. The situation was different with the sustainability score. The use of stars, as in the user rating, led to confusion. Also, the test subjects, especially those from the renewable energies and environmental technology program, wanted a little more information about how this score was calculated. That's why we changed the icons and added some information.

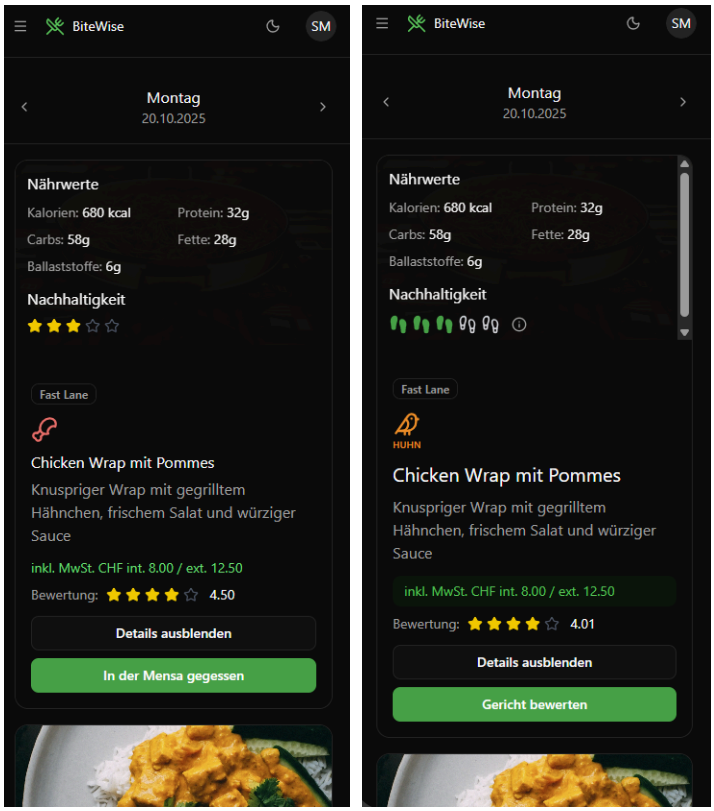

Figure 10: Nutrition Data & Sustainability Score Comparison

3.3.5 Menu images

There was frequent criticism that the images did not match the menus. We were aware of this and want to change it in the real app. It was exciting to find out whether the test subjects would be open to AI images, as it would be difficult to obtain real images on a daily basis. All test subjects liked the idea of AI images. We have already shown them some possible generated images, and the students preferred the images that were not too realistic, as this avoids false expectations. However, this will be examined in more detail at a later date.

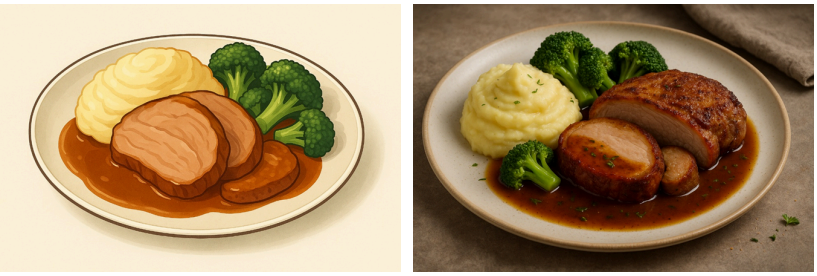

Figure 11: Menu Image Examples from AI

3.3.6 Personal Diet Matching

The configuration of the meal matching data worked well for most people. Some clicked on vegan and then vegetarian as well. This is possible, but not necessary. It would be possible to do this automatically during implementation. The icons were changed to the new versions, which were better understood.

Regarding the suggestions, a test person that the suggested menu was not prominent enough, so we will keep that in mind.

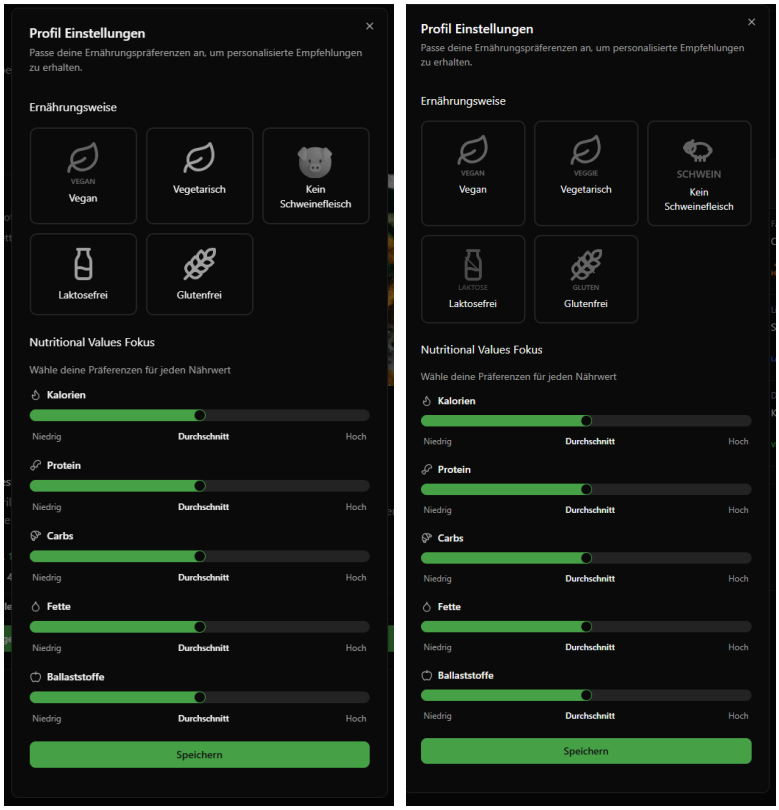

Figure 12: Personal Diet Matching Comparison

3.3.7 Feedback and Menu ideas

Most participants found the forms to be appropriate and easy to use, but some felt that confirmation messages such as “Feedback has been sent” were lacking. This should be implemented.

The screens for the feedback and menu ideas were well received, and no changes were made here.

3.4 Final Prototype

This chapter shows screenshots for each screen of the final prototype. For each screen, the mobile and web version are shown side by side. These images serve as visual references for the MVP implementation.

A video of the final prototype can also be found here: https://youtu.be/Sgmw8EBkhZw .

3.4.1 Menu Plan Screen - Home

After login, the user is presented with the menu plan screen. Here, they can see the three daily menus for the current day and navigate to other days by clicking left and right in the carousel. For each menu, the user sees the price, icons for dietary preferences, ratings from other students and a menu description with a title.

Covered Use Cases: UC-U01

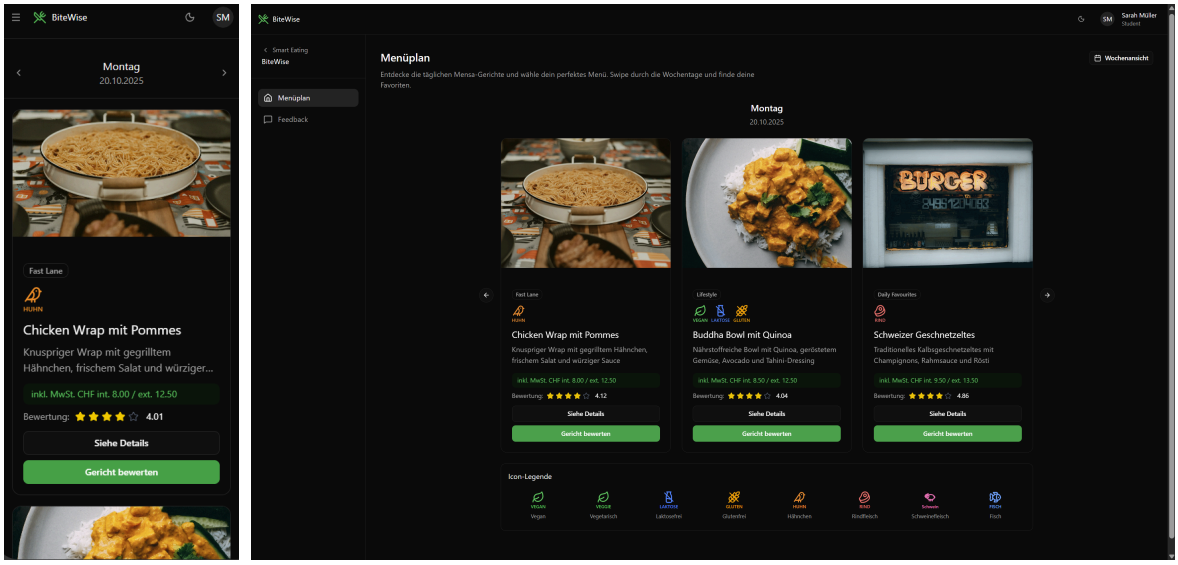

Figure 13: Menu Plan View

3.4.2 Menu Plan Screen - Weekly View

The user can switch the view to a weekly view by clicking the week icon in the top right corner. The weekly view shows the same menu plan as the daily view, but with a more compact layout.

Covered Use Cases: UC-U01

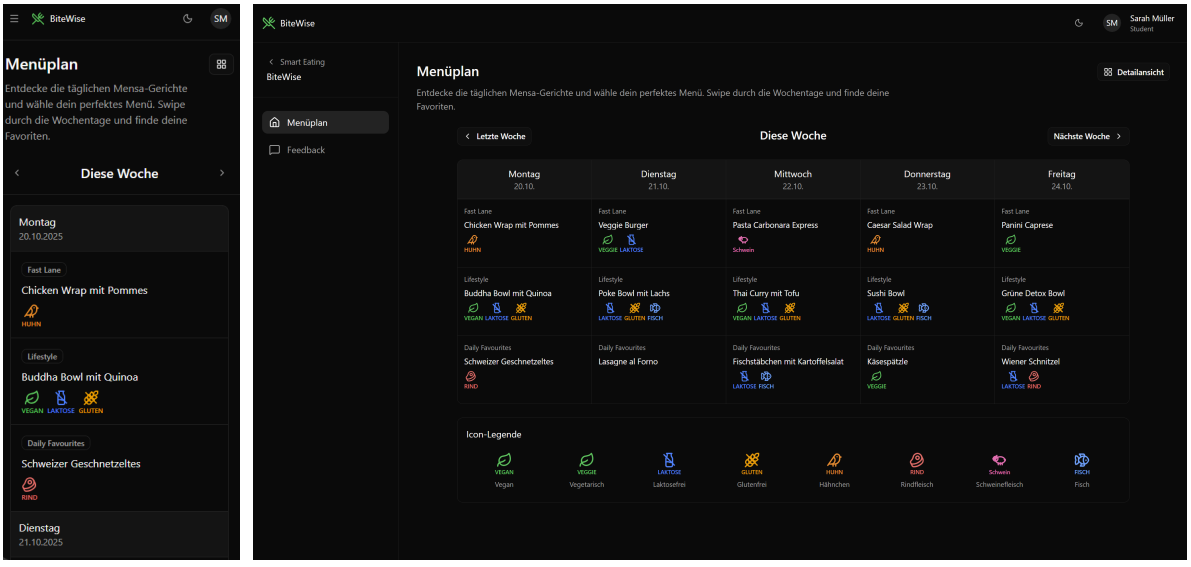

Figure 14: Menu Plan Weekly View

3.4.3 Menu Plan Screen - Detail View

By clicking the “See details” button, the user can see more information about the menu. The user can see the nutritional information for each menu and a sustainability score. By clicking the info icon next to the sustainability score, the user can see more information about how the score is calculated.

Covered Use Cases: UC-U05 , UC-U08

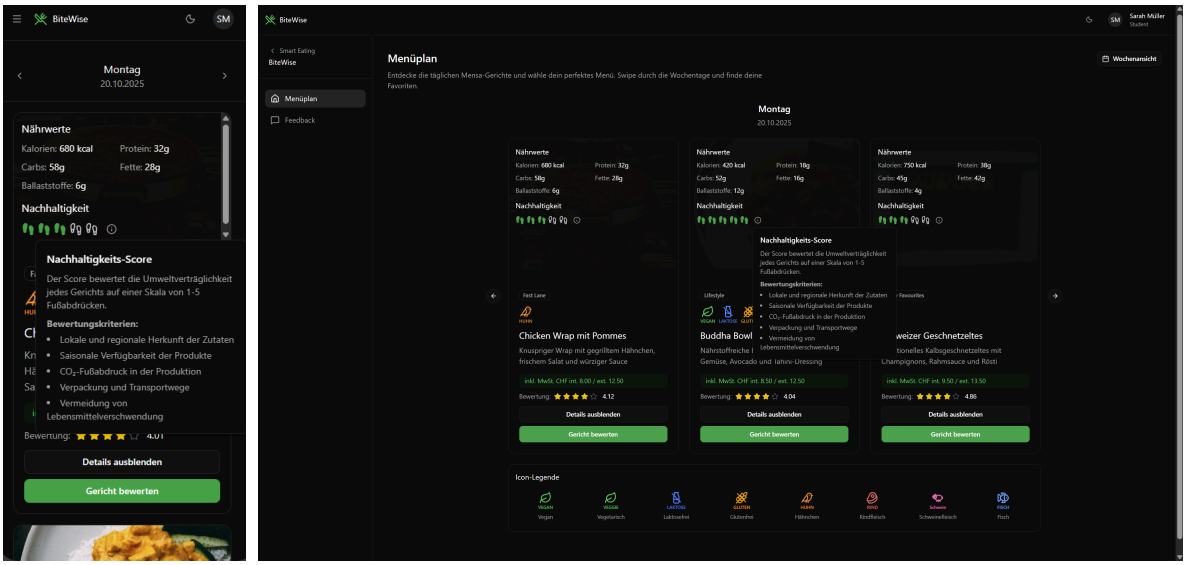

Figure 15: Menu Plan Detail View

3.4.4 Menu Plan Screen - Dietary Matching View

After the user sets their dietary preferences in their profile, they get personalised recommendations for the menu plan. The user can see the menu plan with the recommended items highlighted in green and unsuitable menus are greyed out.

Covered Use Cases: UC-U04

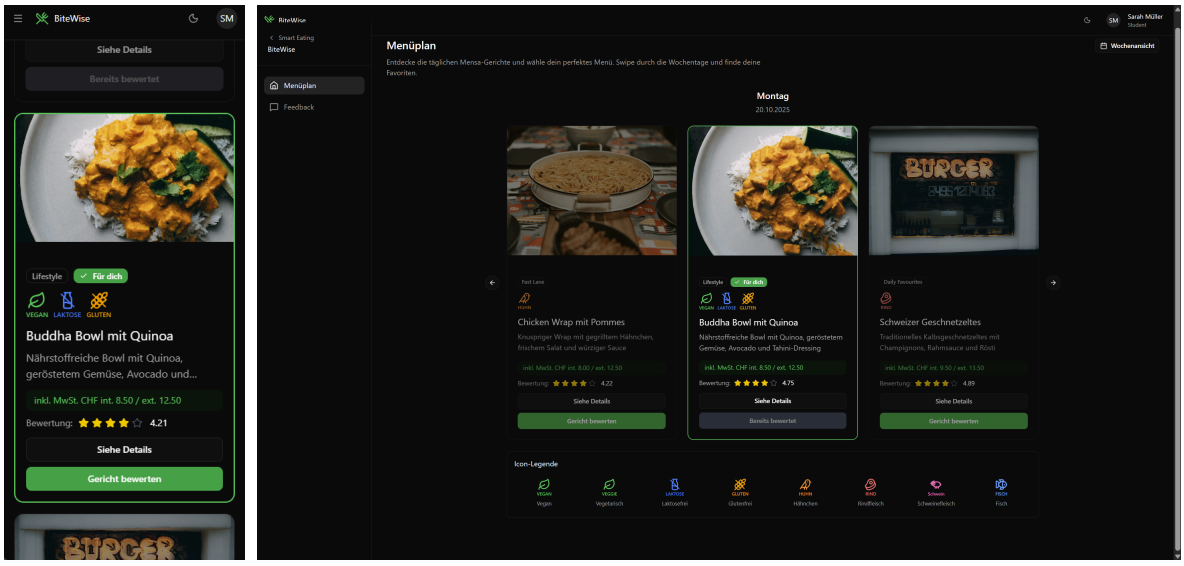

Figure 16: Menu Plan Dietary Matching View

3.4.5 Menu Feedback Screens

The user can give feedback on the menu plan by clicking the “Give feedback” button. After clicking the button, the user is reminded that the menu can only be rated, when the menu has been eaten in the canteen. The user then can rate the menu from 1 to 5 stars and give an optional comment. After the user submits feedback, their rating is factored into the other students overall score and displayed.

Covered Use Cases: UC-U03

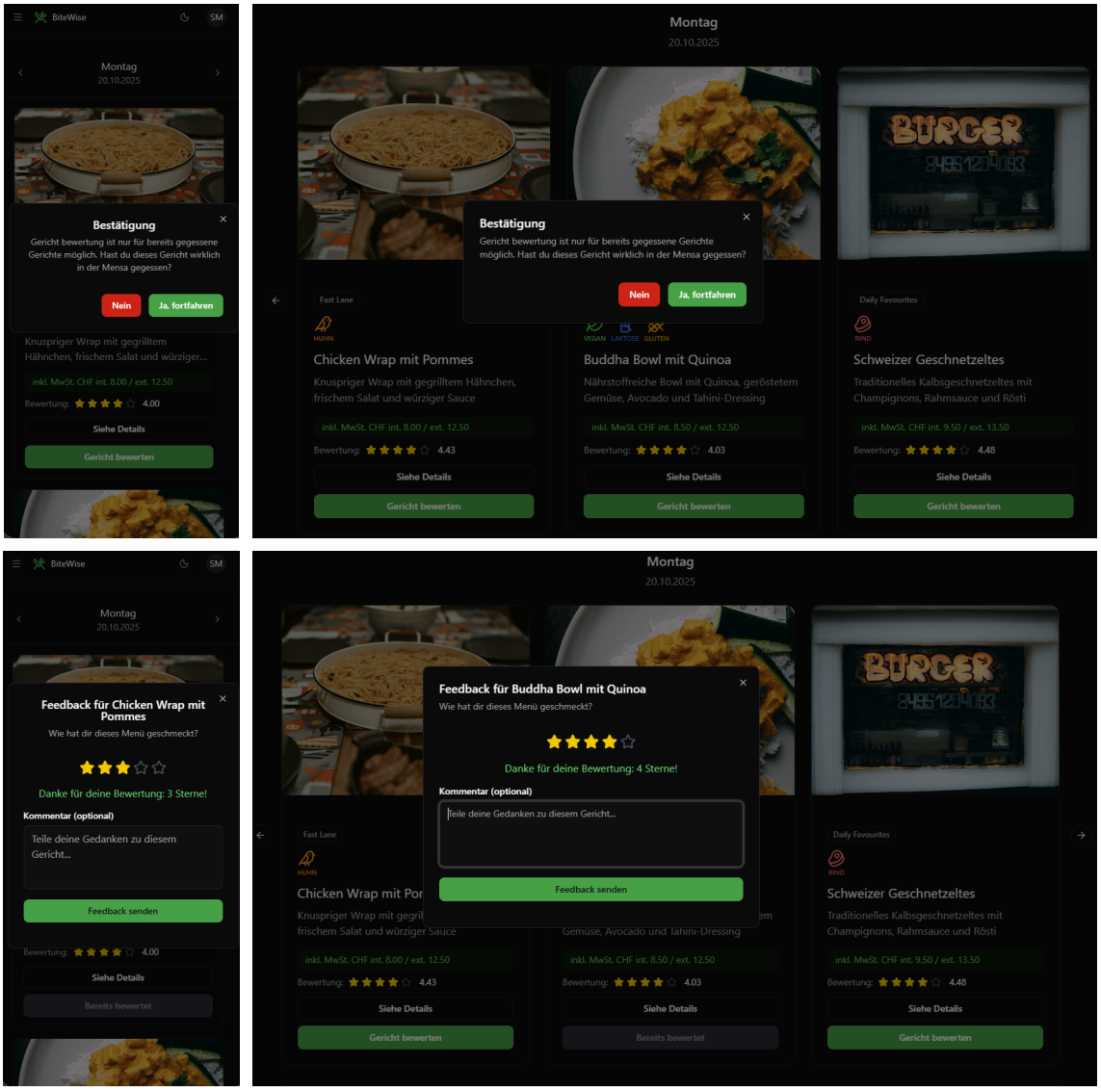

Figure 17: Menu Feedback Views

3.4.6 Feedback Screen - Home

The user can also give feedback in other ways. By clicking on the “Feedback” option in the menu, the user is presented with two options: general feedback and menu suggestion.

Covered Use Cases: -

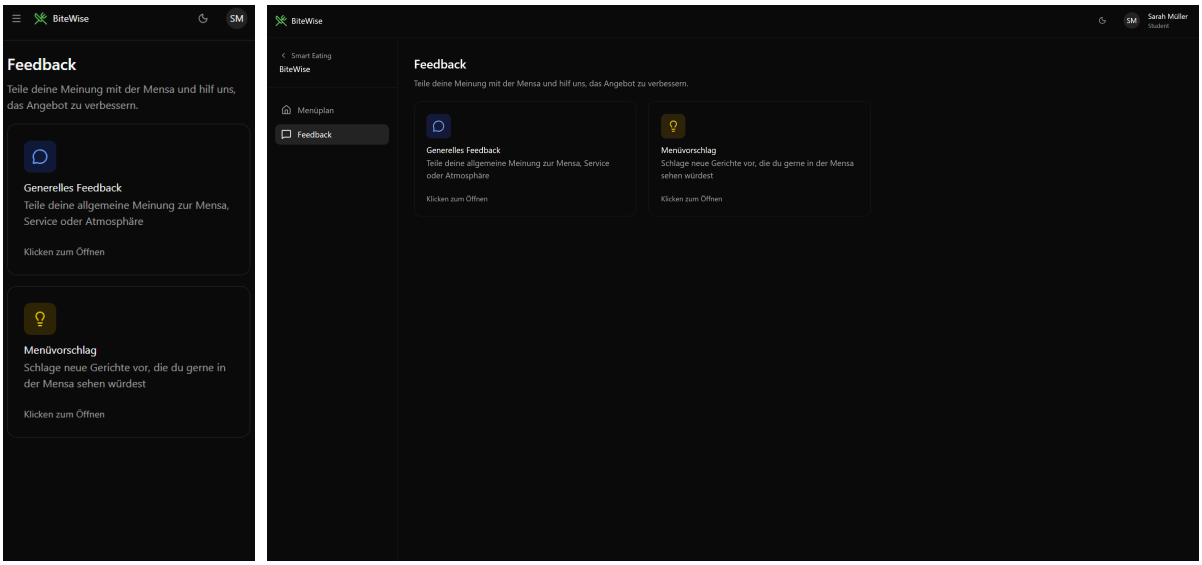

Figure 18: Feedback Home View

3.4.7 Feedback Screen - General Feedback

By clicking on the “General feedback” option, the user can give general feedback about the canteen. The user can rate the canteen from 1 to 5 stars in various categories and give an optional comment. They can also choose whether they want to submit the feedback anonymously or not.

Covered Use Cases: UC-U06

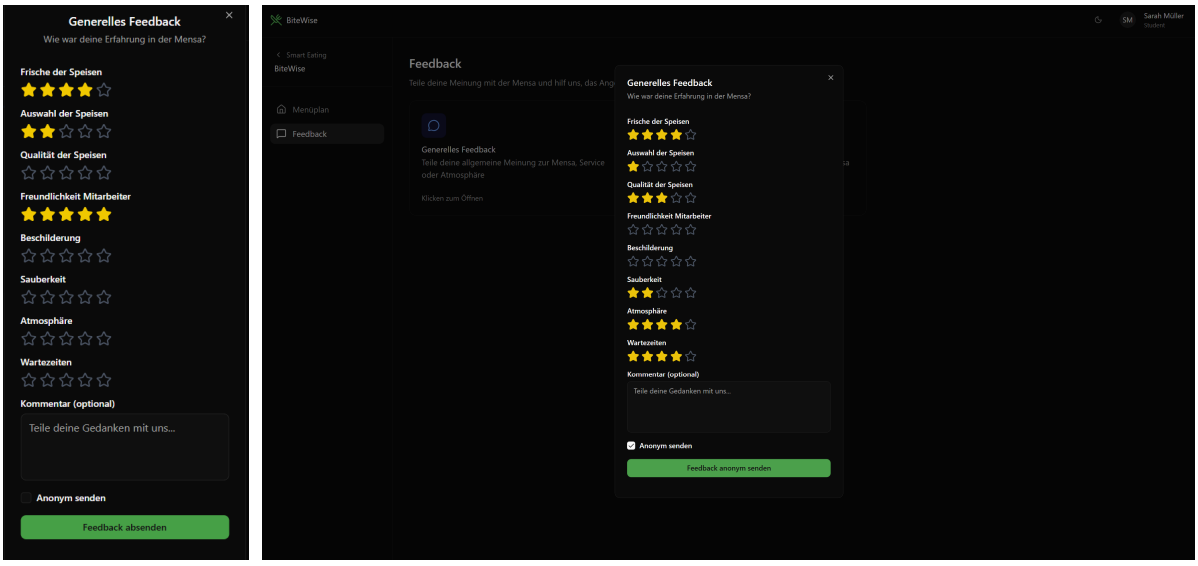

Figure 19: Feedback General View

3.4.8 Feedback Screen - Menu Suggestion

By clicking on the “Menu suggestion” option, the user can give a suggestion for a menu. The user can name their menu and give an additional description for their special menu.

Covered Use Cases: UC-U07

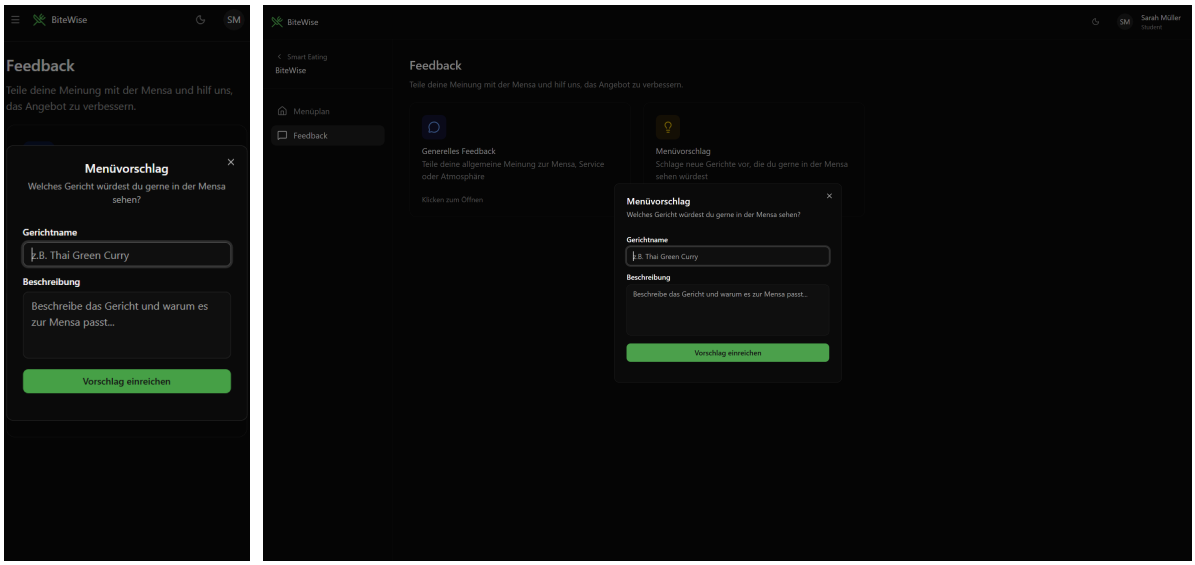

Figure 20: Feedback General View

3.4.9 Profile Screen - Menu

By clicking on the Profile icon in the top right corner, the user can access settings related to their profile. They have two options: Set their dietary preferences and settings.

Covered Use Cases: -

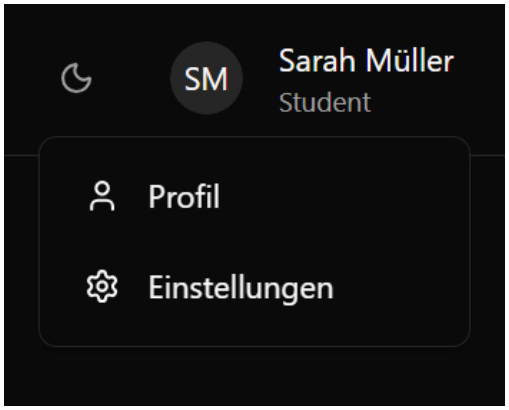

Figure 21: Profile Menu View

3.4.10 Profile Screen - Dietary Preferences

By clicking on the “Profile” option in the profile menu, the user can set their dietary preferences. They can choose from a list of dietary preferences and can prioritize certain nutrients and deprioritize others. Suitable menus are then personalised and highlighted in the menu plan.

Covered Use Cases: UC-E02 , UC-U04

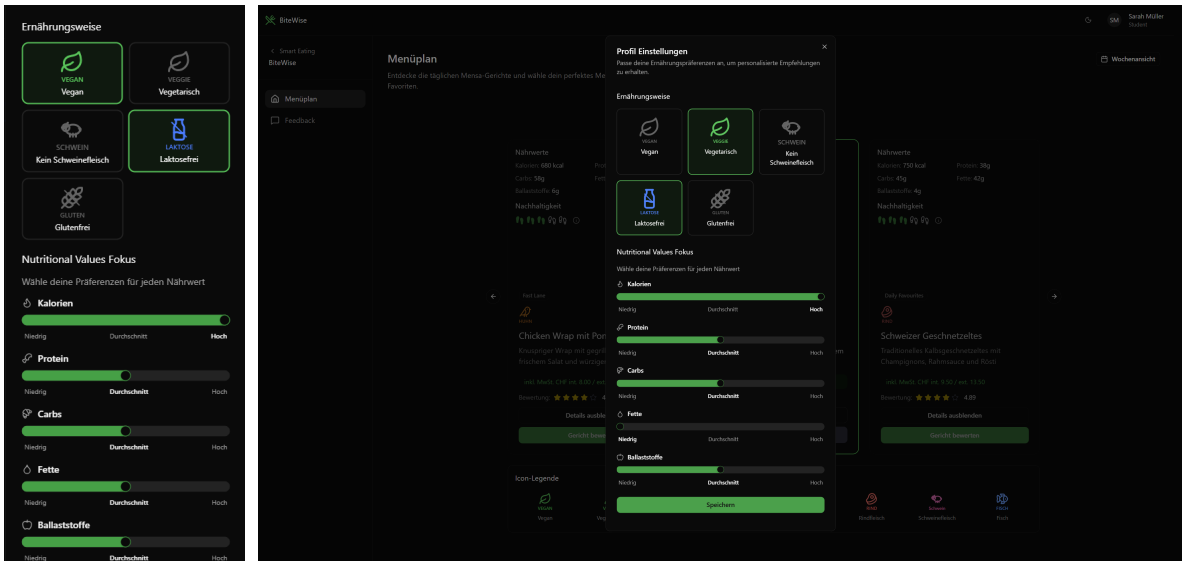

Figure 22: Profile Dietary Preferences View

3.4.11 Profile Screen - Settings

By clicking on the “Settings” option in the profile menu, the user can set their profile settings.

Covered Use Cases: -

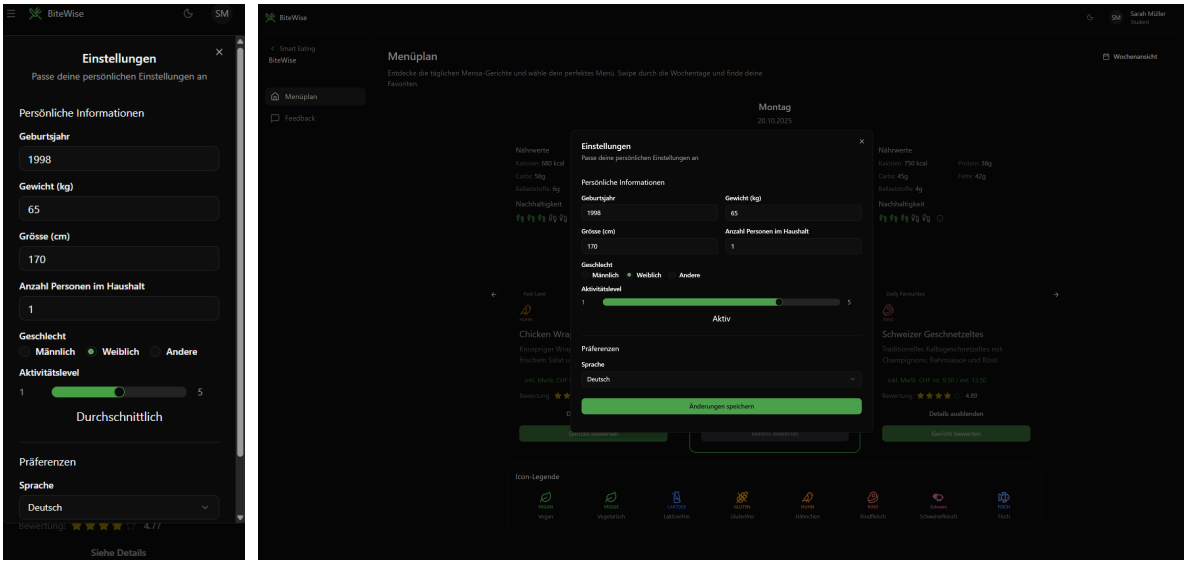

Figure 23: Profile Settings View

3.4.12 Light & Dark Mode Comparison

The user can switch between light and dark mode by clicking the icon in the top right corner.

Covered Use Cases: -

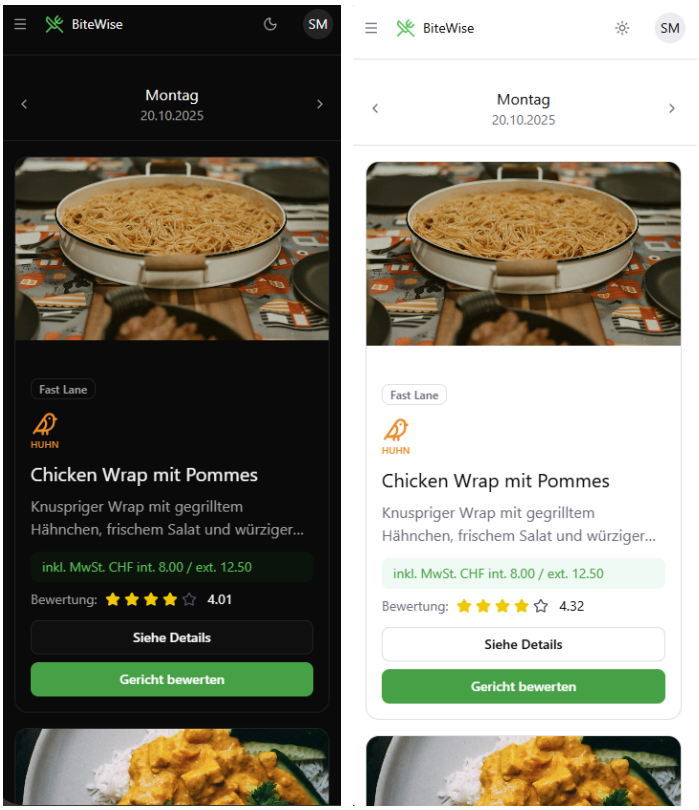

Figure 24: Light & Dark Mode Comparison

4 Architecture

4.1 Technologies

The following section explains the reasons behind our choice of technologies.

As we are developing a sub-platform of Smart Eating, we have decided to use the same technologies wherever possible and appropriate. This ensures consistency, easier integration, and reduced maintenance effort across the ecosystem, especially for future work at IFS or other student projects. In addition, we already have experience with most of these technologies, which further supports efficient development.

4.1.1 .NET & C#

We use .NET with C# in the backend, following the same approach as the current web application. This choice guarantees compatibility and simplifies knowledge transfer within the project.

4.1.2 Entity Framework Core

Entity Framework Core is used as the object-relational mapper (ORM) in the backend. It provides a strongly typed data access layer and enables mapping between the domain model and the PostgreSQL database. EF Core supports migrations, allowing the database schema to evolve in a controlled and versioned manner. This aligns well with the Clean Architecture approach, as persistence concerns are encapsulated within the infrastructure layer.

4.1.3 React & TypeScript

For the frontend, we use React with TypeScript, just as in the main platform. React provides a proven, component-based framework that enables rapid development and reusability of UI components across sub-platforms. TypeScript adds static typing to JavaScript, improving code quality, maintainability, and tooling support.

4.1.4 Redux Toolkit

Redux Toolkit is used for global state management in the frontend. It provides a structured approach to Redux by introducing slices, reducers, actions, and asynchronous thunks. This enables a predictable, unidirectional data flow and keeps business and state logic separate from presentation components. Redux Toolkit significantly reduces boilerplate and follows current best practices.

4.1.5 Tailwind CSS

Tailwind CSS is used for styling the frontend. It is a utility-first CSS framework that enables rapid UI development by composing styles directly in the markup. This approach reduces the need for custom CSS files and promotes consistency across the application. Tailwind also supports responsive design and adds a modern look, which fits well with the requirements of the BiteWise web app.

4.1.6 Swagger & NSwag

We continue to use Swagger for API documentation and testing, as already established in the existing platform. This creates a clear contract between backend and frontend and makes it easier for developers to get started, something that has already proven helpful for us. In addition, we integrate NSwag to automatically generate client code from the Swagger specification.

4.1.7 Axios

Axios is used in the frontend for HTTP communication with the BiteWise API and the Smart Eating API. It provides a promise-based abstraction over HTTP requests and supports features such as request and response interceptors, centralized error handling, and automatic JSON transformation. Using Axios keeps API communication consistent and encapsulated outside of React components.

4.1.8 Testing

For backend testing, we rely on a .NET testing stack. xUnit is used as the main test framework due to its simplicity and good integration into the .NET ecosystem. Moq is used to mock service dependencies in controller tests, allowing isolated verification of controller behaviour without invoking the actual business logic. For data access and service tests, Entity Framework Core with the InMemory provider is used to simulate the database layer without requiring a real database instance.

For frontend testing, we use Jest together with React Testing Library. Jest serves as the test runner and assertion framework, while React Testing Library focuses on testing components from a user perspective.

4.1.9 Database

The BiteWise database was implemented using PostgreSQL. This decision was primarily based on existing experience with PostgreSQL and the fact that the existing Smart Eating database is already based on the same technology.

4.1.10 Python & OpenAI

The image generation service was implemented using Python. Python is commonly used for AI integrations and was already established in the Smart Eating project for other AI-related services. The service requires OpenAI for image generation.

4.1.11 npm

npm is used as the package manager for the frontend. It is responsible for managing third-party dependencies such as React, Redux Toolkit, Tailwind CSS, and testing libraries. npm scripts are also used to standardize common development tasks, including building, testing, and running the application locally. Using npm ensures a reproducible and consistent development environment across all team members.

4.2 Database

The existing Smart Eating project already consists of the user database and the recipe database. An additional BiteWise database has now been added for our SA. The following chapters explain the benefits and architecture of the individual databases in more detail.

4.2.1 Databases

Database	Technology	Description
smarteatingdb	PostgreSQL	The existing SmartEating database contains all recipes and associated nutritional values, sources, ingredients, images, quantities, etc.
userdb	SQLite	The existing user database can also be used. Compared to the other databases, it has been implemented in SQLite.
bitewisedb	PostgreSQL	The BiteWise database was newly created for this project and is used for new entities that cannot be represented in the current SmartEating DB. These include, for example, menu assignments to days, menu ideas, and general canteen feedback. However, most entities are attempted to be represented using the SmartEating database.

Table 27: Database Descriptions

4.2.2 Context Diagram

The database context diagram shows how the BiteWise app interacts with the different databases, as well as the high-level relationships between the databases themselves. The arrow directions between the databases should be interpreted as follows: the bitewisedb knows the smarteatingdb and the userdb, and the userdb knows the smarteatingdb, but not vice versa.

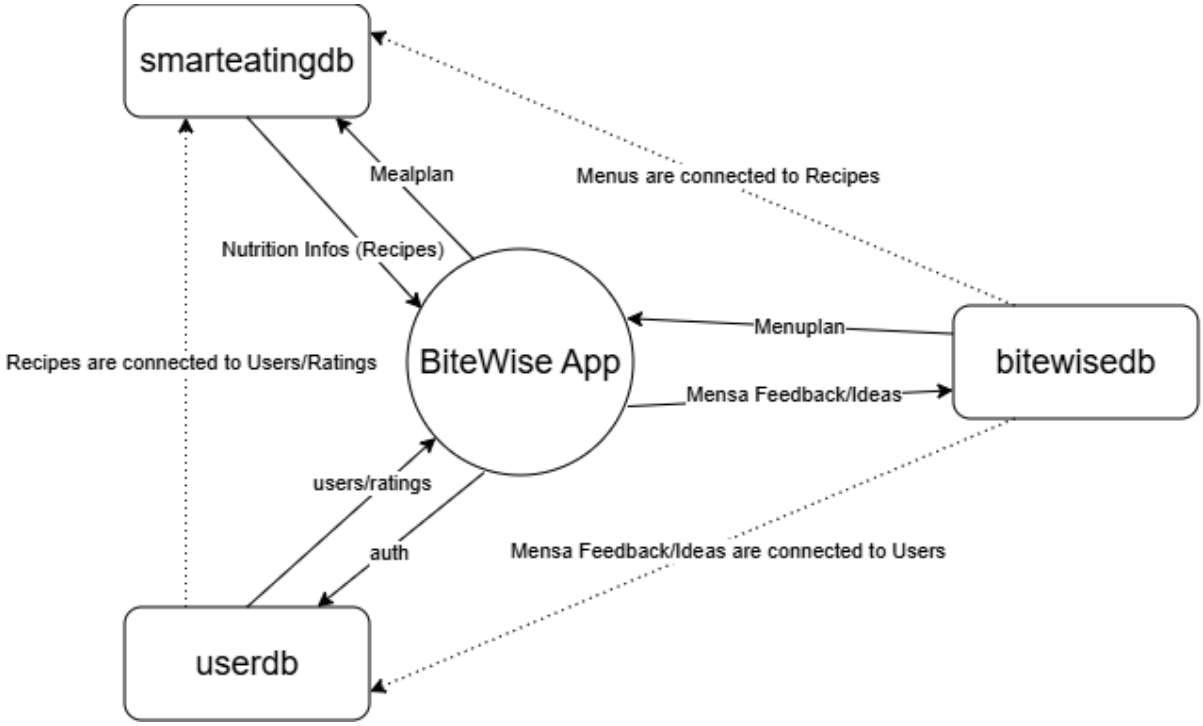

Figure 25: Database Context Diagram

4.2.3 Entity Relationship Diagram

The following entity relationship diagram (ERD) shows the databases, entities and their relationships within and between the databases. As it would otherwise have been almost impossible to display, the SmartEating DB and the User DB have been shortened by removing some entities that are irrelevant to our project or the illustration. For example, the locale tables and shopping tables have been removed. The black lines represent relationships within a database and the pink lines represent relationships between databases.

As you can see, this makes the whole thing rather complex, which is why the following chapters show the shortened ERDs of the individual databases.

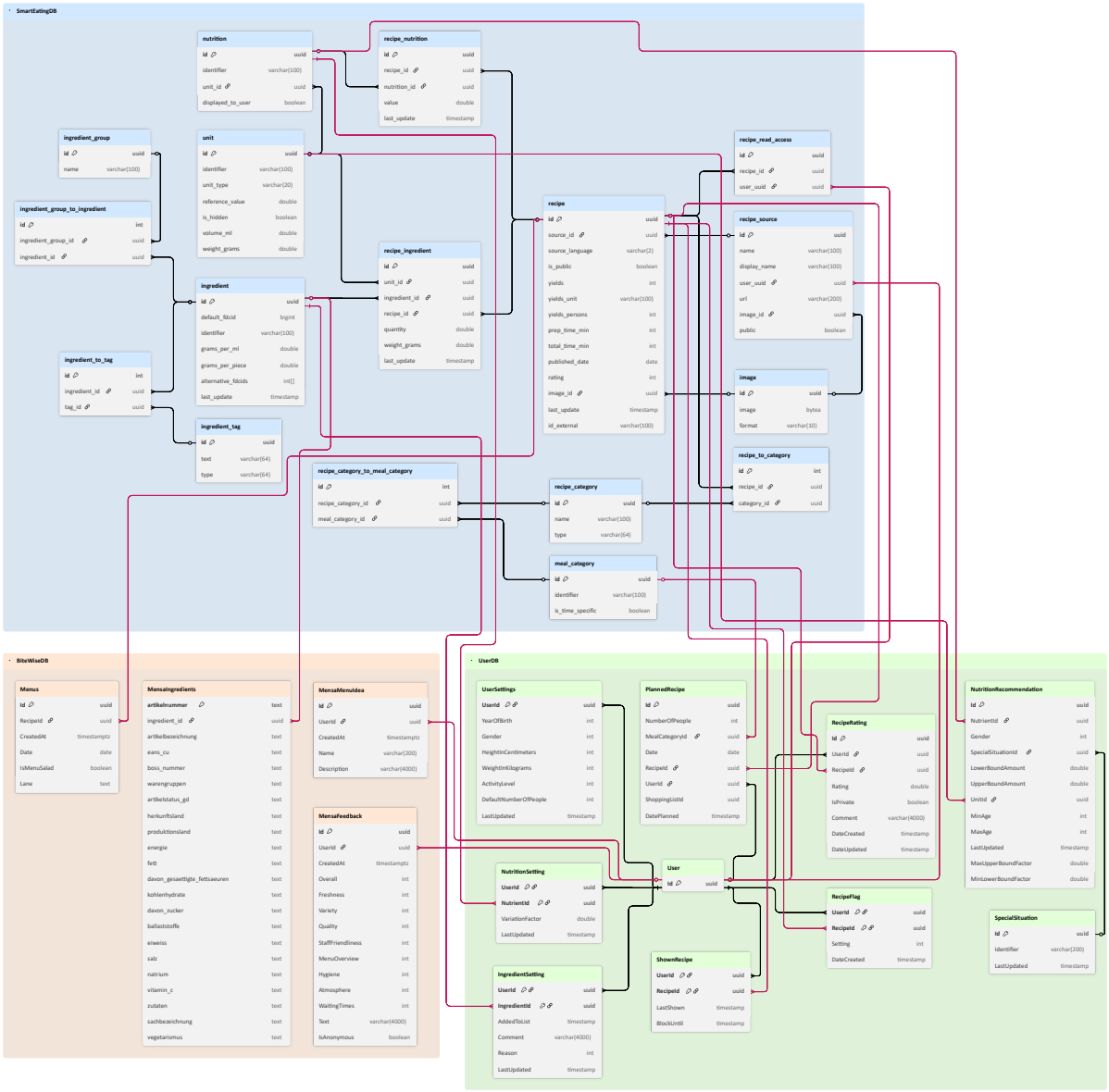

Figure 26: ERD all Databases

4.2.3.1 ERD SmartEatingDB

The SmartEatingDB maps all core data that is central to the existing Smart Eating platform. This includes recipes, ingredients, nutritional values, categories and metadata such as images and sources. It represents the largest and most complex data area and will continue to be used for our new sub-platform.

The ERD shows in particular the strongly interconnected recipe entities (recipe, ingredient, nutrition, image, etc.) as well as the assignment tables used to structure, categorise and link recipes with ingredients. This data forms the basis for all calculations, nutritional analyses and recipe displays within the app.

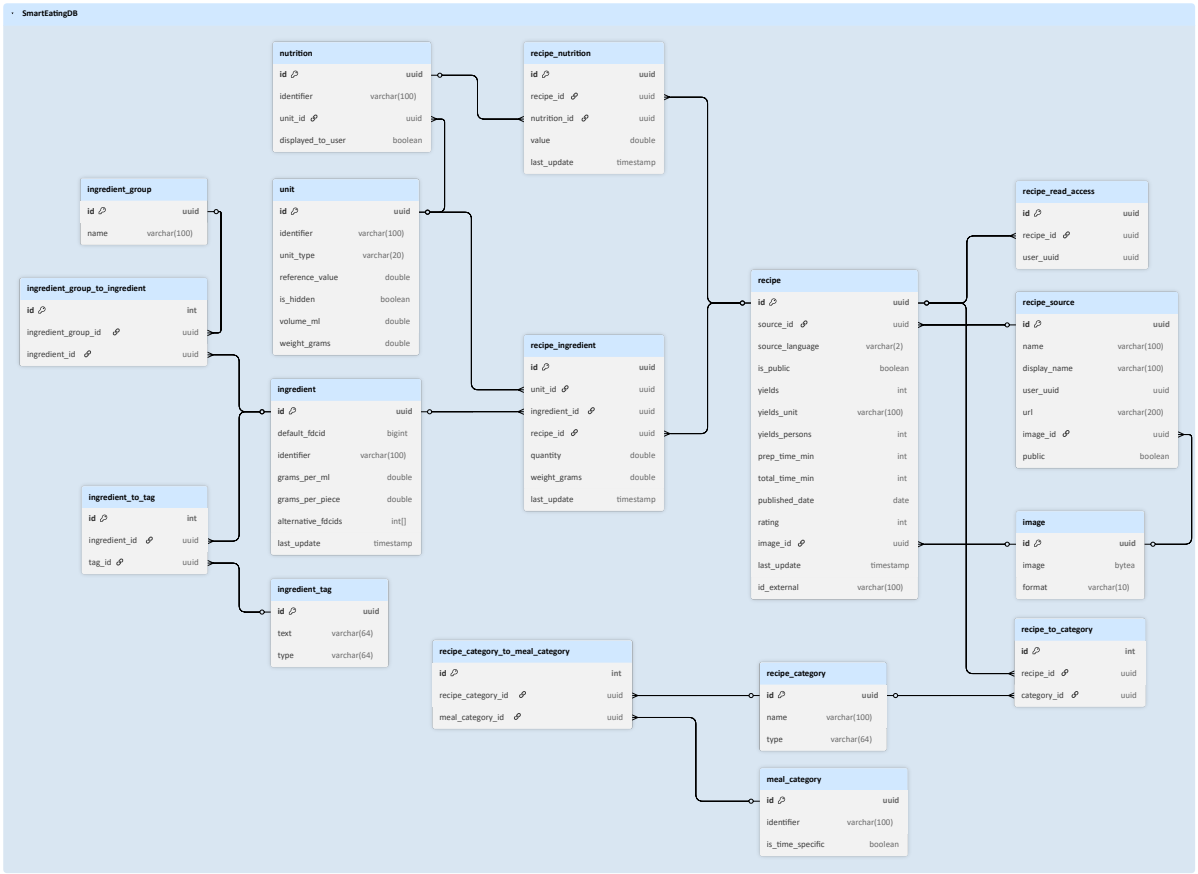

Figure 27: ERD SmartEatingDB

4.2.3.2 ERD UserDB

The UserDB contains all user-related data from the existing Smart Eating platform. This includes settings, preferences, personal details, recipe flags and feedback-related entities such as ratings.

The UserDB is particularly important for the BiteWise project, as many newly introduced features (e.g. Feedback or Menu Ideas) refer to the user. Therefore, there are numerous cross-DB connections from the new BiteWiseDB as well as from the SmartEatingDB to the UserDB. We also use the ratings from this database for our menu ratings.

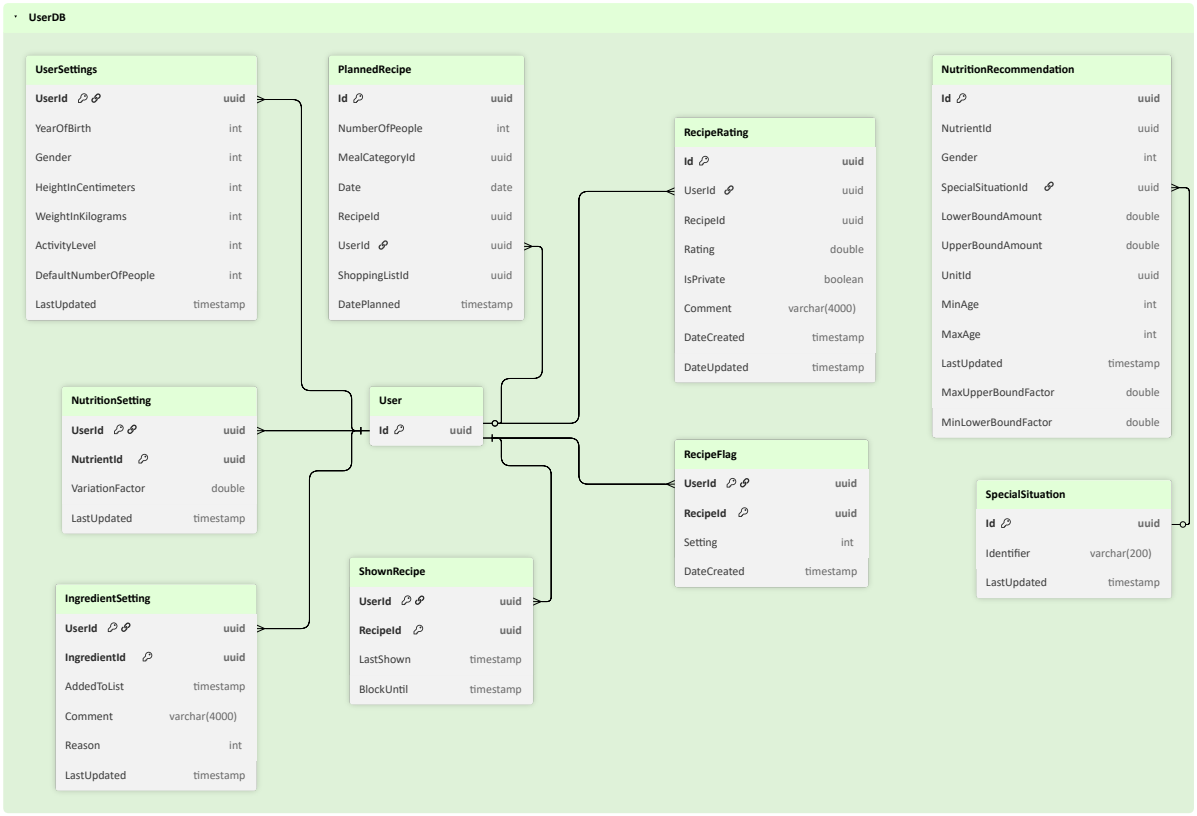

Figure 28: ERD UserDB

4.2.3.3 ERD BiteWiseDB

The “BiteWiseDB” was created specifically for this project and contains all specific entities that could not be mapped in either the “SmartEatingDB” or the “UserDB”. These include daily menus, canteen feedback and menu ideas. This data is closely linked to the UX concept of the BiteWise app, as it captures information that was not provided for in the existing Smart Eating structure.

The “MensaIngredients” table is a special mapping table that is filled after the canteen ingredients import and is used for mapping during the menu import. It is therefore linked to the ingredients from the “SmartEatingDB”.

Compared to the others, this ERD is displayed in its entirety with all entities. The ERD shows the comparatively slim structure of the BiteWiseDB. The clear separation of this DB allows new features to be developed without having to change the complex Smart Eating structure.

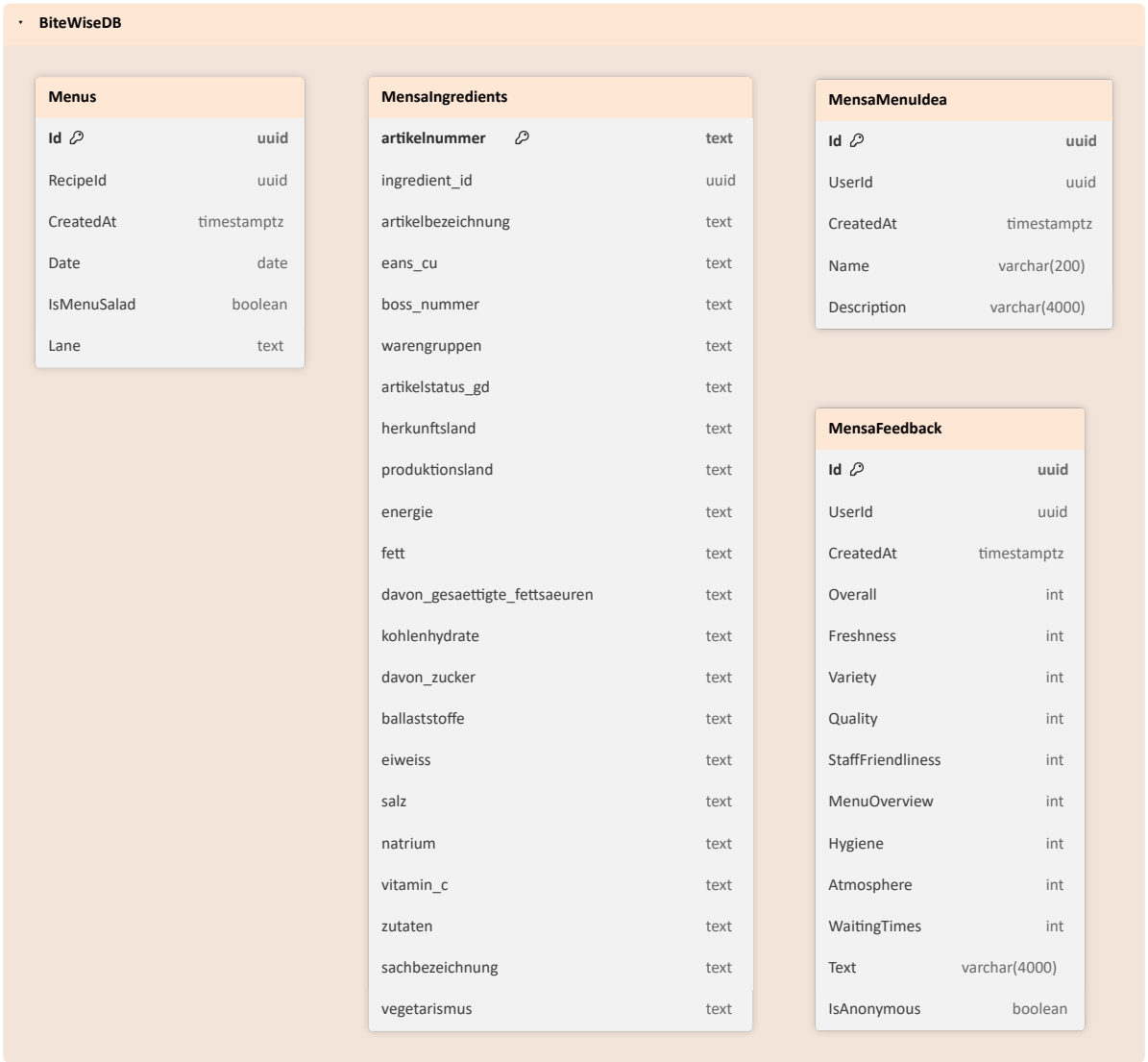

Figure 29: ERD BiteWiseDB

4.3 Software Architecture

4.3.1 C4 model

The following C4 diagrams describe the architecture of the BiteWise subsystem within the broader Smart Eating platform (Brown, 2025) . The model consists of four detail levels: the system context, the containers, and the frontend and backend components. These diagrams illustrate how students and Canteen employees interact with the system, how responsibilities are distributed among the software elements, and how BiteWise integrates with external systems such as Smart Eating and the image generation service. The goal of the model is to provide a clear, structured view of the system's technical composition and the interactions between its parts.

The broader Smart Eating platform includes several additional internal services that are not shown in this C4 model. For example, the Smart Eating API (specifically the WebApp API used by BiteWise) makes use of other subsystems, such as the Nutrition Service, to handle nutritional analysis. Since BiteWise does not interact with these services directly, they were left out to keep the diagrams clear and easy to understand. The C4 views therefore focus only on the external systems that BiteWise actually communicates with: the Smart Eating WebApp API and the Image Generation Service.

4.3.1.1 Context

The system context diagram shows the primary user groups and all external systems interacting with BiteWise. Students use the web application to browse menus and submit feedback, while canteen employees rely on dashboard and import functionalities. BiteWise integrates with the Smart Eating platform to access recipe and nutrition data, and with an external image generation service to automatically create menu images. The diagram clearly defines the subsystem boundary.

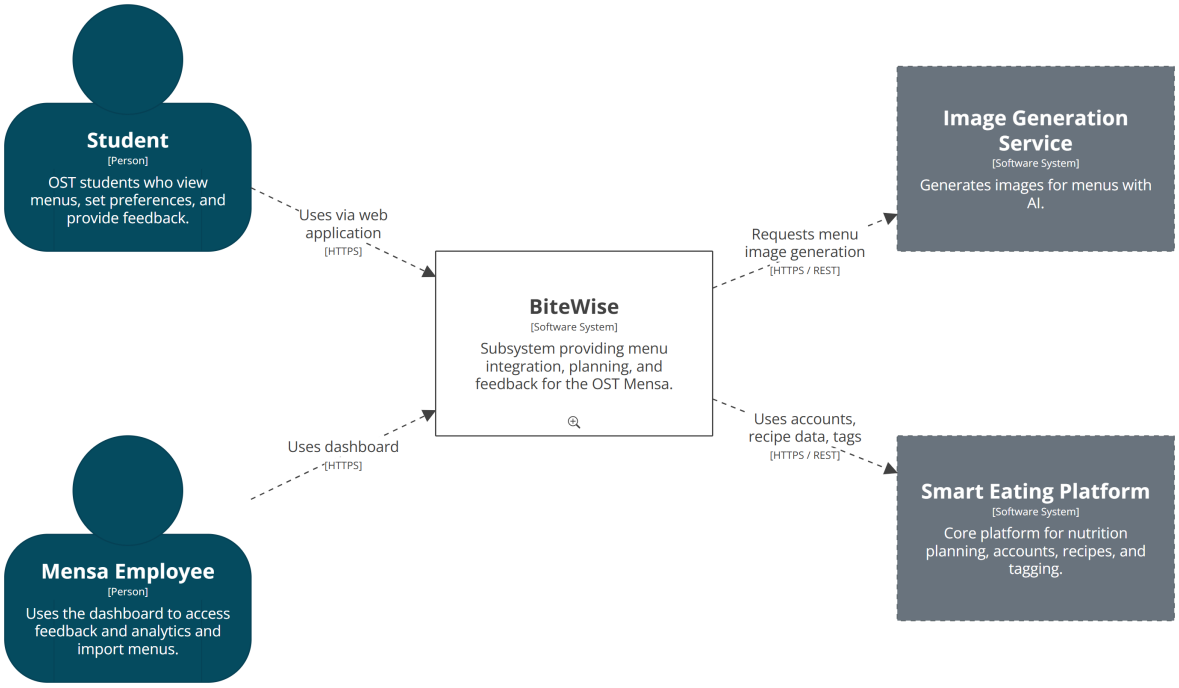

Figure 30: C4 Context Diagram

4.3.1.2 Containers

The container diagram outlines the three core building blocks of BiteWise: the React-based Web App, the ASP.NET Core API backend, and the PostgreSQL database. The Web App communicates with both the BiteWise API and the Smart Eating API, depending on the type of information required (e.g., recipes, nutrition data, tags, images, or user settings). The BiteWise API, in turn, manages menus, feedback, imports, and delegations to external services such as Smart Eating and the image generation service. The image generation service acts as a dedicated AI component for creating menu illustrations. Overall, this view provides a clear understanding of the system's high-level structure and the responsibilities of the involved containers.

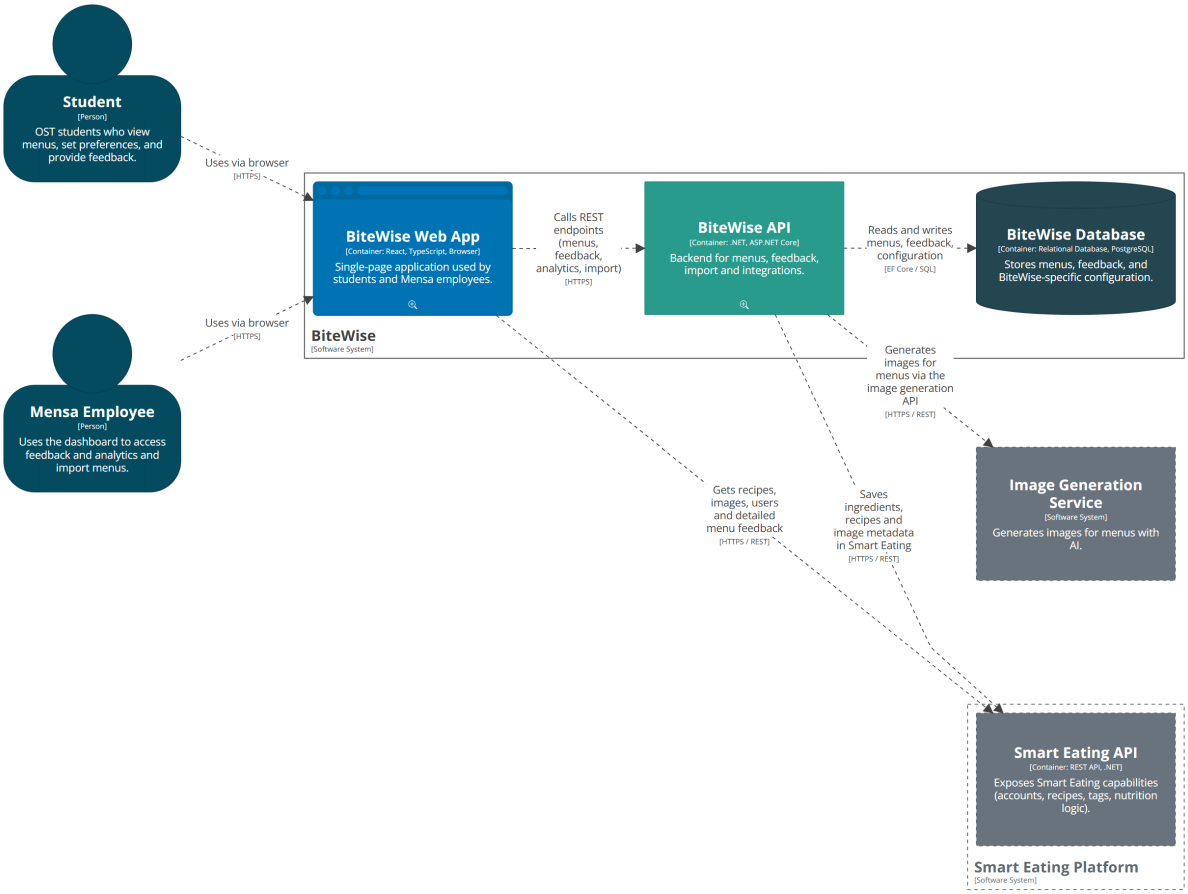

Figure 31: C4 Container Diagram

4.3.1.3 Backend Components

The backend component diagram decomposes the API into controllers, services, and parsers. Controllers expose HTTP endpoints and delegate business logic to the corresponding services, which encapsulate use cases such as menu management, imports, or image generation. Parsers provide specialised functionality for processing Excel-based menu and ingredient data. Integration with Smart Eating and the image generation service is handled through dedicated service components. The architecture separates concerns cleanly and supports maintainability and extension.

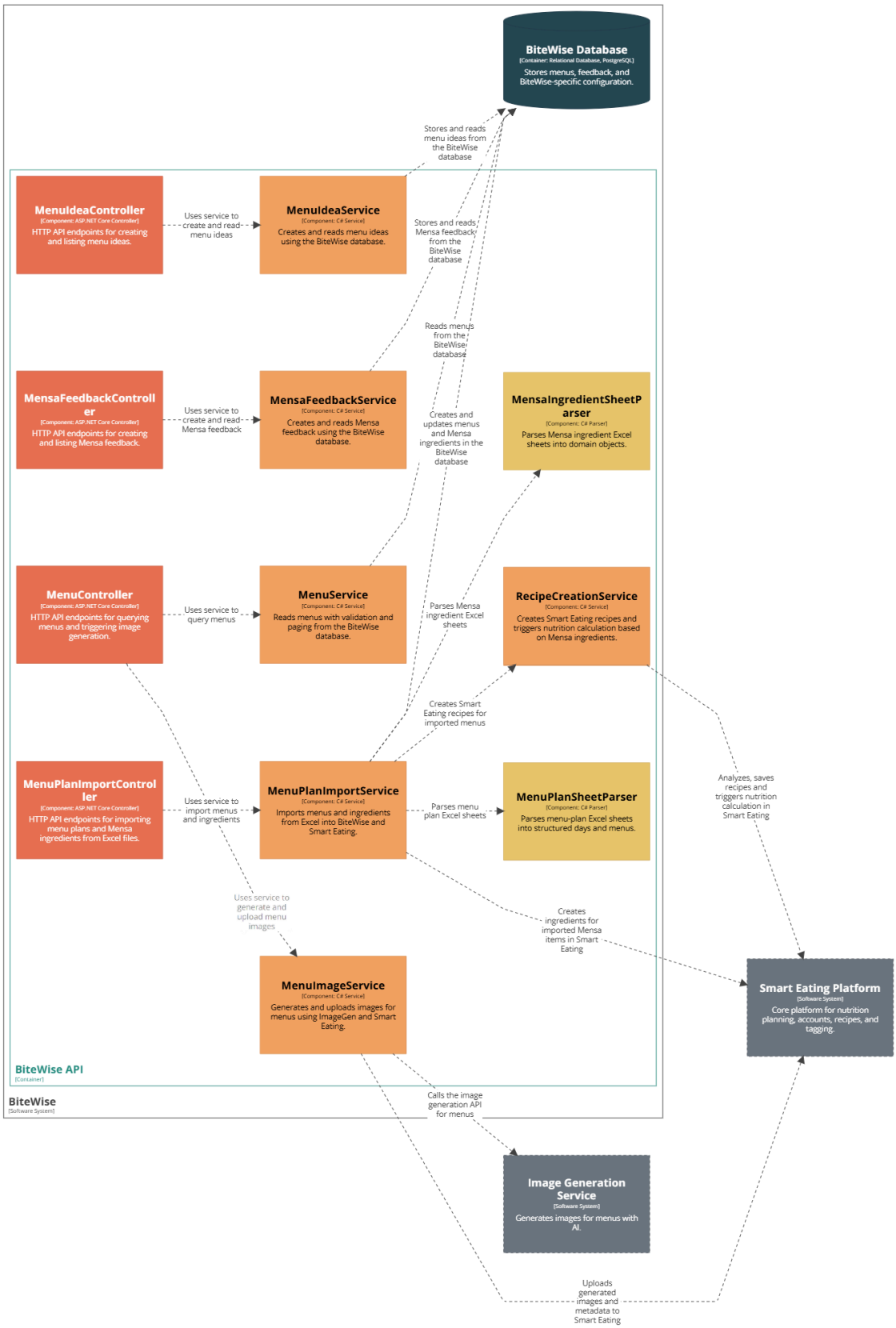

Figure 32: C4 Backend Components Diagram

4.3.1.4 Frontend Components

The frontend component diagram shows the main React components of the BiteWise Web App. These represent the higher-level functional areas of the application. Each of these components communicates with the BiteWise API or the Smart Eating API depending on the data it requires. While the actual implementation contains additional subcomponents and UI abstractions, these are intentionally omitted here to maintain clarity and focus. This abstraction level provides a clear overview of the Web App's structure without diving into technical detail that would exceed the scope of this documentation.

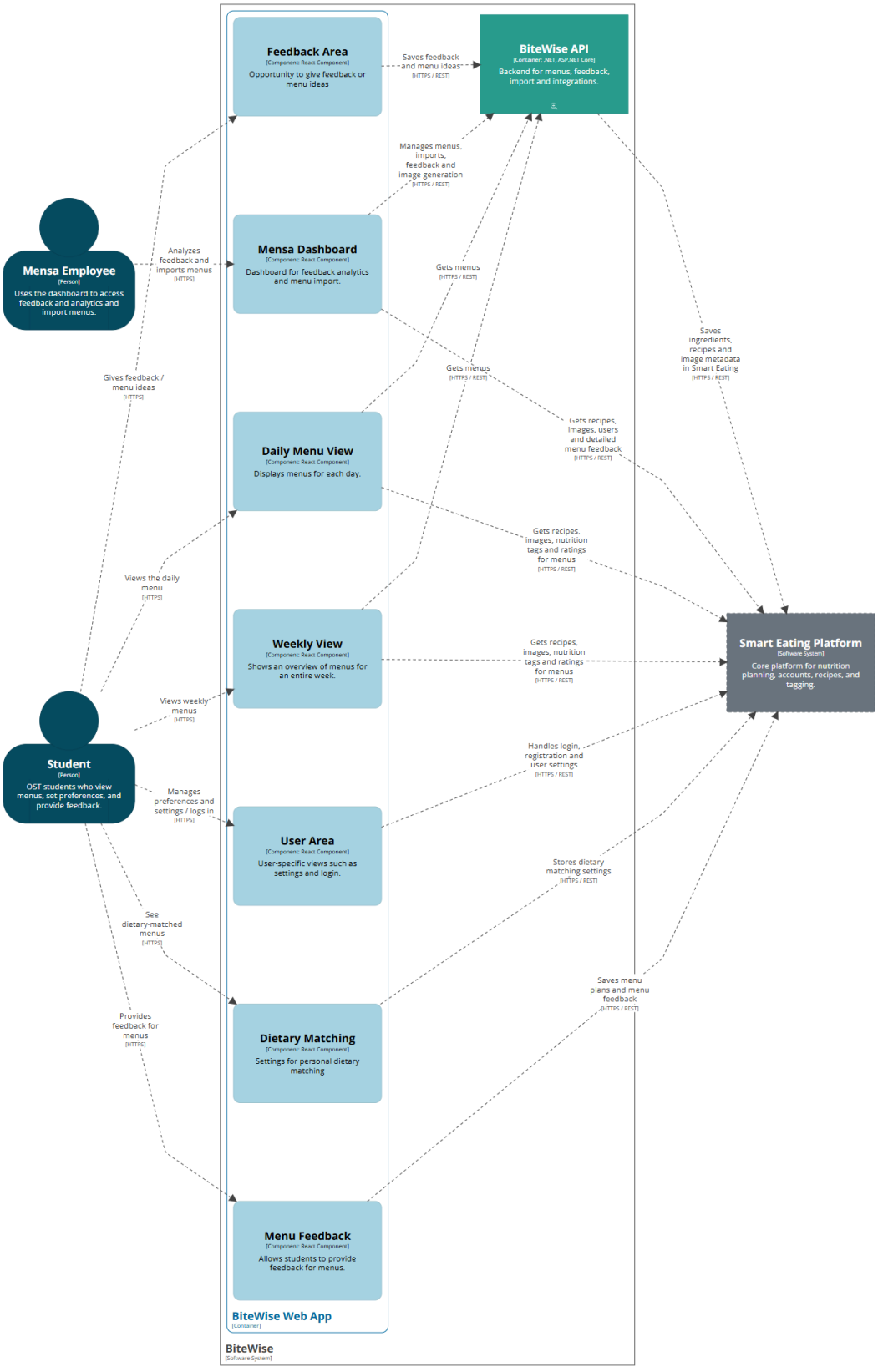

Figure 33: C4 Frontend Components Diagram

4.3.2 Backend Architecture and Project Structure (Clean Architecture)

The BiteWise backend follows the principles of Clean Architecture (Martin, 2017) . The core domain model is isolated from technical concerns, while infrastructure and framework code is kept in separate projects.

At a high level, the projects can be mapped to the typical Clean Architecture layers:

Project	Meaning
OST.Bitewise.Domain	Contains the core domain model (entities, value objects) and domain-level abstractions for persistence or external dependencies (infrastructure interfaces). The domain layer is framework-independent and does not contain application use cases or transport-specific types.
OST.Bitewise.Services	Implements the application layer and use case orchestration (business logic), based on the domain model and domain interfaces. This layer remains framework-independent; it may contain use-case DTOs and service interfaces when required by the chosen data contracts. It depends only on the Domain project.
OST.Bitewise.Infrastructure	Provides framework-dependent implementations of the domain interfaces, including EF Core-based persistence and technical integrations. This project contains the DbContext, migrations and import/IO-related implementation details.
OST.Bitewise.Api	Hosts the ASP.NET Core Web API (controllers, authentication/authorization, request handling) and acts as the composition root. It wires up dependency injection and exposes the HTTP endpoints used by the BiteWise Web App.
OST.SmartEating.WebAppClient	Provides typed clients for the Smart Eating WebApp API, used by the application layer to integrate with Smart Eating functionality (e.g. recipes).

Table 28: Clean Architecture Projects

This separation makes the domain and application logic testable and independent of ASP.NET Core and the database. Infrastructure details (such as EF Core and Excel handling) can be replaced without touching the domain layer, and external integrations are encapsulated behind clear interfaces.

The BiteWise solution includes two dedicated test projects to ensure correctness and reliability across different architectural layers. The separation between service-level and API-level tests follows the Clean Architecture principles and allows each layer to be validated independently with an appropriate test scope.

Project	Meaning
OST.Bitewise.Services.Tests	Contains unit tests for the application and domain logic in the Services layer. These tests verify business rules, use case behavior, and service interactions in isolation from infrastructure and web concerns.
OST.Bitewise.Api.Tests	Contains integration tests for the ASP.NET Core API layer. The tests focus on controller endpoints, request validation, authentication and authorization.

Table 29: Test Projects

The following diagram complements the project overview by visualising the concrete folder structure of each backend .NET project. It illustrates how the solution is organized in practice and how the projects map to the Onion/Clean Architecture layers.

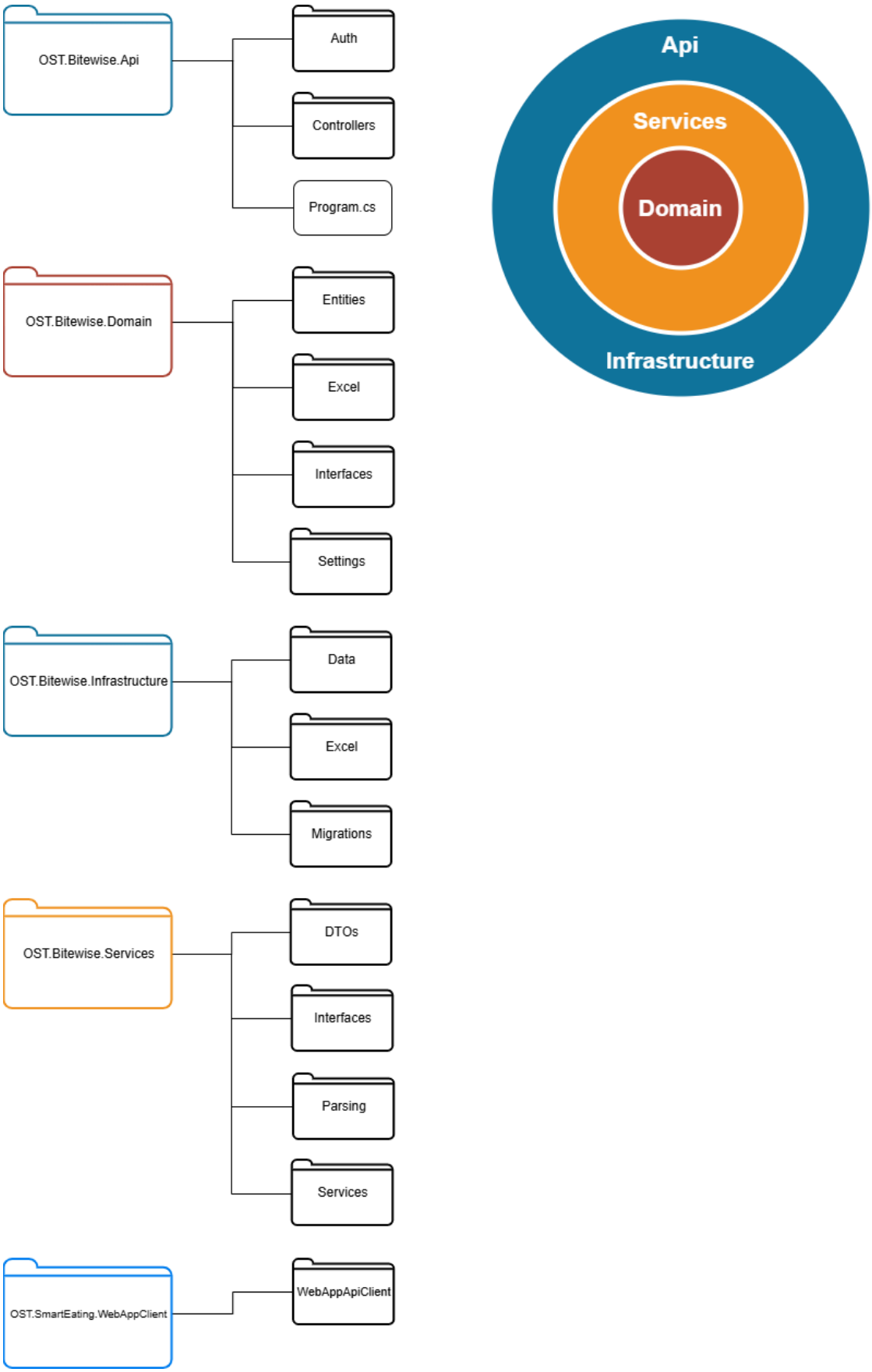

Figure 34: Backend Project Structure & Onion Diagram

4.3.3 Frontend Architecture (React & Redux)

The BiteWise frontend is implemented as a single-page application using React and follows a unidirectional data flow based on the Redux pattern. This architecture provides a clear separation between presentation logic, state management, and side effects, supporting maintainability and predictable behavior.

At the core of the frontend architecture is a centralized Redux store that represents the application state. The state is structured by feature domains (menus, feedback, user, dietary-matching and mensa), each encapsulated in its own Redux slice.

4.3.3.1 State Structure and Slices

Each slice defines:

- an initial state representing the domain data and loading or error states,
- reducers that describe how the state changes in response to actions,
- generated action creators for synchronous state updates.

Slices are implemented using Redux Toolkit, which reduces boilerplate and enforces best practices such as immutable state updates. This approach ensures that domain-specific logic remains localized and easy to reason about.

4.3.3.2 Actions, Reducers and Data Flow

UI components do not manipulate state directly. Instead, user interactions (e.g. submitting feedback or changing preferences) dispatch actions to the Redux store.

Reducers are pure functions that react to these actions and compute the next application state.

4.3.3.3 Asynchronous Logic with Thunks

All asynchronous operations, such as HTTP requests to the BiteWise API or the Smart Eating API, are handled via Redux thunks. Thunks encapsulate side effects and follow a consistent lifecycle:

- Dispatch a pending action to indicate loading state
- Perform the API call
- Dispatch either a fulfilled action with the received data or a rejected action with error information

This pattern keeps API interaction logic out of React components and avoids duplicated request handling across the UI.

4.3.3.4 Selectors and Component Integration

Selectors provide a stable and reusable abstraction for accessing state. They decouple React components from the internal structure of the Redux store and allow derived data (e.g. menus aggregated with recipes and ratings) to be computed efficiently.

React components subscribe to the store using hooks and selectors and focus solely on rendering and user interaction. This results in mostly stateless, declarative components that are easy to test and reuse.

4.3.3.5 Architectural Benefits

The combination of React and Redux establishes a predictable data flow: User interaction → Action → Reducer / Thunk → Store update → UI re-render.

This architecture aligns well with the complexity of BiteWise, where multiple views depend on shared data such as menus, feedback, and user preferences. It enables clear responsibility boundaries, simplifies debugging, and provides a solid foundation for future extensions.

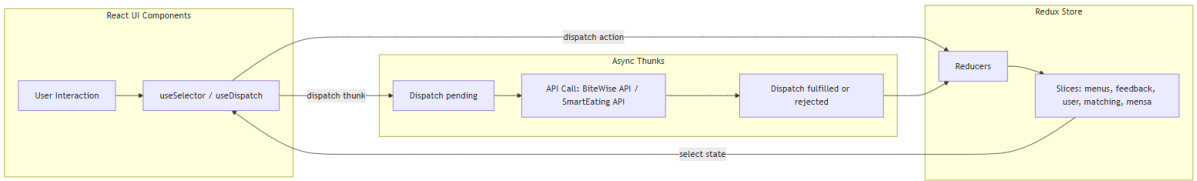

Figure 35: React Redux Flowchart

4.3.4 Deployment and Container Extensions (BiteWise)

The existing Smart Eating Docker setup was extended with several BiteWise-specific containers to support the new functionality introduced in this project. The overall container architecture follows the established Smart Eating approach and integrates BiteWise seamlessly into the existing ecosystem.

Container	Description
bitewise_db	A dedicated PostgreSQL database container for BiteWise. It stores all BiteWise-specific entities that cannot be represented in the existing Smart Eating or User databases, such as daily menus, menu ideas, and general canteen feedback. The clear separation allows independent schema evolution without impacting the core Smart Eating databases.
imagegeneration	A new internal service responsible for AI-based image generation. The container exposes a protected HTTP endpoint that generates images using the OpenAI API. Access is restricted via an internal API key and the service is intended exclusively for backend-to-backend communication within the Smart Eating infrastructure.
bitewise	The main BiteWise application container. It hosts the BiteWise ASP.NET Core backend API and serves as the integration point between the frontend, the BiteWise database, the Smart Eating WebApp API, and the image generation service.

Table 30: New Docker Containers

5 Implementation

This chapter presents the implemented views of the application and describes what users can do within each view as well as the processes that are executed in the background. In addition, the covered use cases are referenced where applicable.

It should be noted that the initially defined use cases were deliberately not reworked in detail. While they largely remain valid, several implementation details evolved during the project compared to the originally envisioned approaches. These deviations are summarised in the relevant sections to make the design and implementation decisions transparent.

5.1 Menu Plan Page

The menu plan page acts as a homepage for the BiteWise App as a student. The student can choose between two views to look at the canteen menus. On both views the student directly sees the most important info for each menu: Lane category of the menu, icons for dietary preferences, the menu title, price and ratings for the menu from other students.

At the bottom of the page is an icon legend to better understand which icon stands for what dietary preference. Currently, the preferences identified are: Vegan, vegetarian, fish, pork and meat. This info is taken from the tags returned by the nutrition info endpoint from the backend.

5.1.1 Daily View

This is the default view of the menu plan page, here the student directly sees the three daily menus from the canteen for the current day. In this more detailed view, the menus are depicted with AI generated look-a-like pictures. By clicking or swiping left and right in the carousel, the student can look at the next or previous three daily menus. The menus for the current, next and previous week can be viewed.

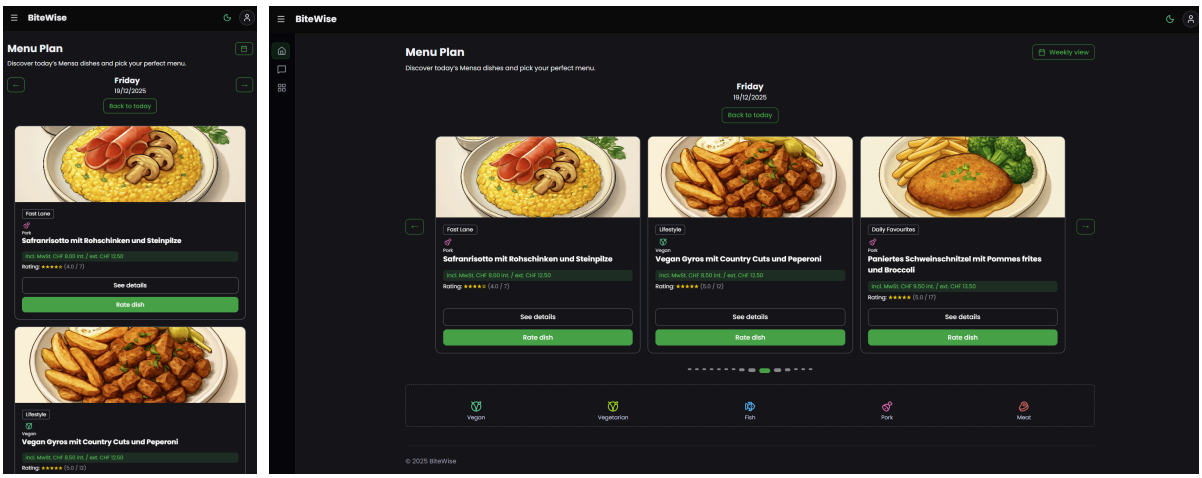

Figure 36: Daily Menu Page

5.1.2 Weekly View

For a more compact view, the user can switch views by clicking on the respective button on the top right corner. In the weekly view, all 15 menus for the whole week are shown to get a faster overview. Here, the menus are not shown with a picture to save space. Similar to the daily view, the week can be switched to view the next or previous menus. When a menu on the weekly view is clicked, the user gets redirected back to the daily view to the exact day when this menu is served.

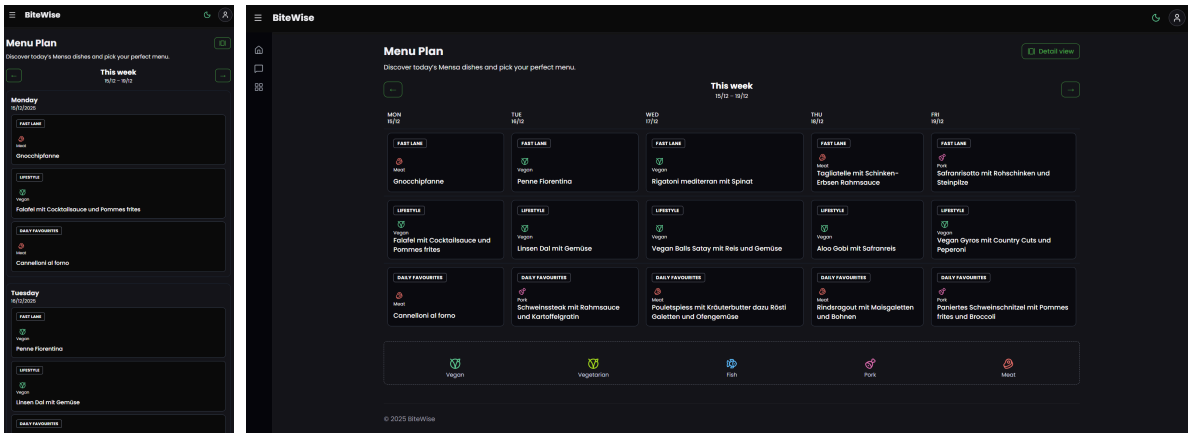

Figure 37: Weekly Menu Page

5.1.3 Sequence Diagram

The following sequence diagram summarises the data retrieval process of the Daily Menu View. It highlights the separation between the BiteWise API (menu meta data) and the SmartEating API (recipes, nutrition, ratings and images).

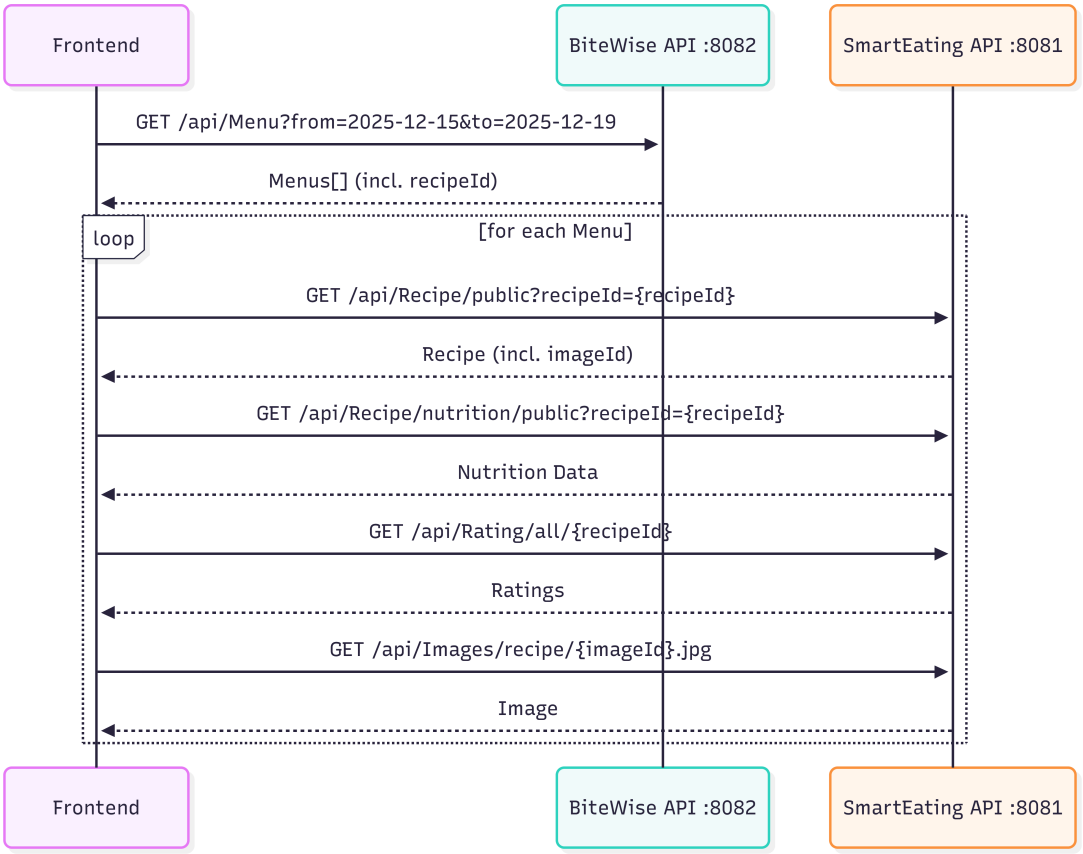

Figure 38: Sequence Diagram Daily Menu View

5.1.4 Covered Use Cases

Use Case ID	Use Case	Priority	Deviations
UC-U01	Daily Menus	*High Prio*	Decided to add an additional view “Weekly View” after feedback from the usability tests. Decided to drop the dietary preference icons for lactosefree and glutenfree, since these are harder to determine by an AI whether they contain traces of it. After further investigating how the tags to determine the dietary preference icons of the menus are set by Smart Eating, we discovered that no differentiation of meat, except pork is made. So we grouped chicken and beef to one “meat” tag.

Table 31: Covered Use Cases Menu Plan Page

5.2 Menu Feedback

The student has the possibility to give feedback to each of the menus the canteen serves. For this, the rate dish button on the bottom of each menu in the daily menu view can be used. The student first gets asked if they have really consumed the menu at the canteen, since they must have eaten it to be able to give honest feedback. Here we rely on the honesty and cooperativeness of each student to ensure correct data. If they confirm, they can give a star rating and a comment for the menu.

5.2.1 Hybrid Meal Planning

A part of this SA is to also integrate some features of the Smart Eating platform with BiteWise. For this, the user has a personal mealplan in their Smart Eating profile to keep track of what they ate. If such a user eats lunch at the canteen, the chosen meal automatically gets saved in their mealplan too. The student can also decide to skip giving the feedback, if they only want to save the menu in their mealplan.

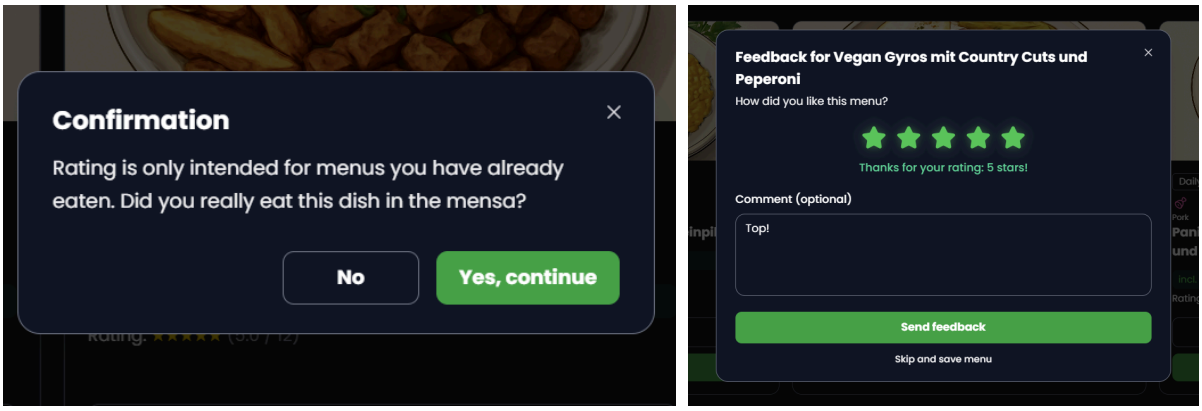

Figure 39: Menu Feedback Forms

It is important to note that the meal plan is currently stored in SmartEating and is therefore not yet displayed in BiteWise. This is a limitation of the current solution and may be expanded in the future.

5.2.2 Sequence Diagram

The sequence diagram illustrates the submission flow for menu feedback. Besides creating a rating entry, the selected menu is persisted in the user's Smart Eating meal plan.

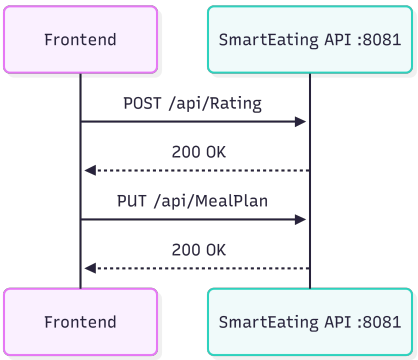

Figure 40: Sequence Diagram Menu Feedback

5.2.3 Covered Use Cases

Use Case ID	Use Case	Priority	Deviations
UC-U02	Menu Selection	*High Prio*	Is saved in the meal plan, but this cannot be viewed in BiteWise itself.
UC-U03	Menu Feedback	*High Prio*	-

Table 32: Covered Use Cases Menu Feedback Page

5.3 Feedback Page

The Feedback Page allows students to actively contribute feedback to the canteen. They can choose between submitting general canteen feedback or proposing new menu ideas. For menu ideas, the user must be logged in. This requirement ensures that the canteen can contact the student in case of questions and also supports incentive models, as the canteen may offer rewards when a submitted menu idea is successfully implemented.

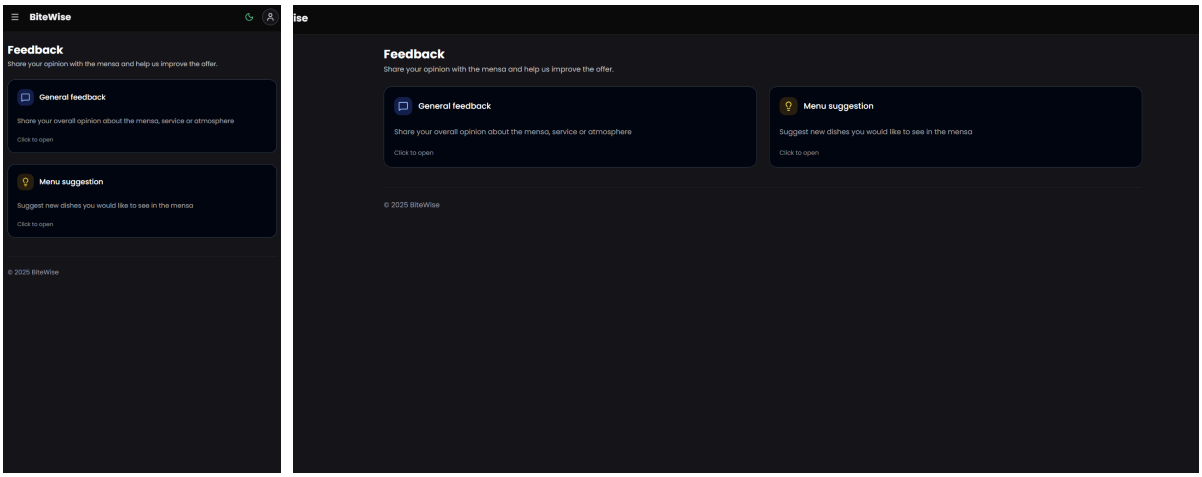

Figure 41: Feedback Page

5.3.1 Canteen Feedback

In the general canteen feedback section, students can rate the canteen across multiple categories using star ratings. The content of this form is based on an existing feedback form already used by the canteen. In addition to the ratings, students can submit a written comment. Depending on their preference, this comment is either displayed to the canteen together with their email address or anonymously.

After submitting the feedback, the user receives a toast notification confirming that the feedback was successfully sent, provided no errors occurred during submission.

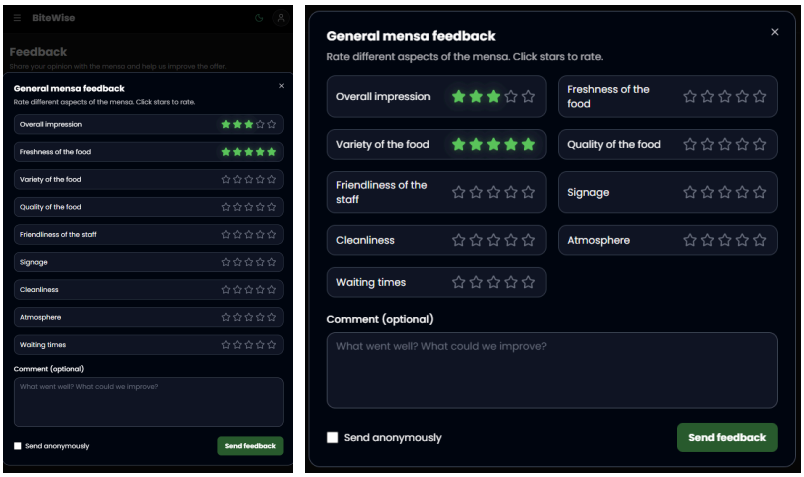

Figure 42: Canteen Feedback View

5.3.2 Menu Idea

In the menu idea section, students can submit suggestions for new dishes by providing a title and a description. The description field can be used for any additional information, including references or external links. As with the general feedback, a toast notification is displayed after successful submission to confirm that the menu idea was sent successfully.

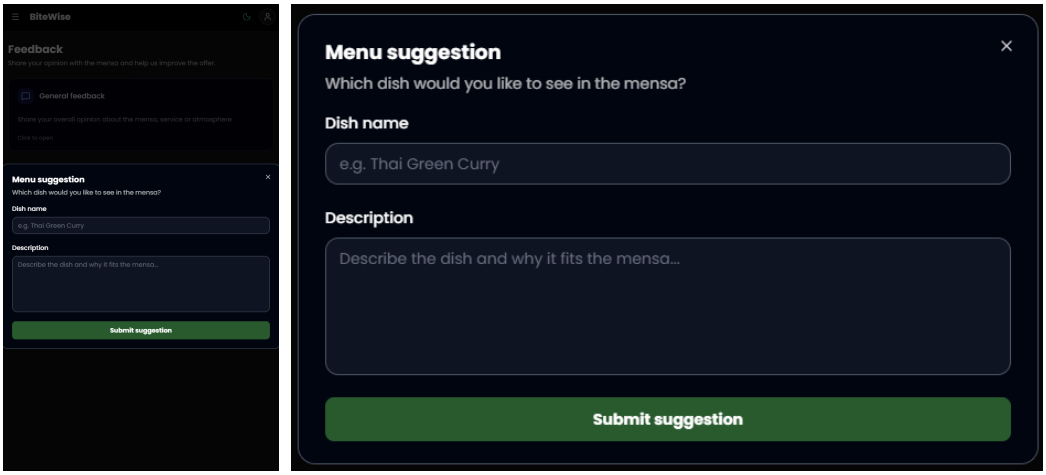

Figure 43: Idea Suggestion View

5.3.3 Backend Storage

All feedback data is persisted via the newly introduced BiteWise API and stored in the BiteWise database. If feedback is not submitted anonymously, the ID of the currently logged-in user is stored alongside the entry. The referenced users originate from the existing Smart Eating user system and are reused within BiteWise.

5.3.4 Sequence Diagrams

The sequence diagrams show how general canteen feedback and menu ideas are submitted. In both cases, the frontend sends the user input to the BiteWise API, where the entries are persisted and made available for evaluation in the canteen Dashboard.

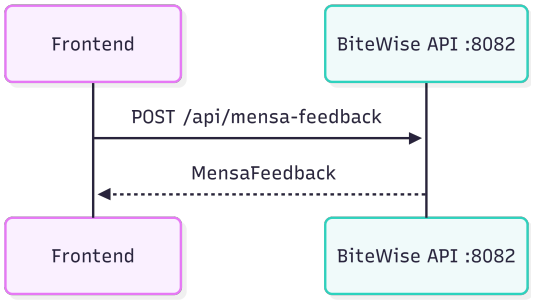

Figure 44: Sequence Diagram canteen Feedback

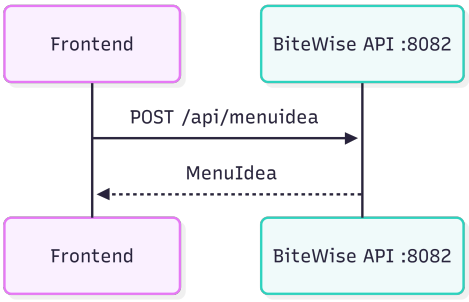

Figure 45: Sequence Diagram Menu Idea

5.3.5 Covered Use Cases

Use Case ID	Use Case	Priority	Deviations
UC-U06	General Feedback	*Mid Prio*	-
UC-U07	Menu Ideas	*Mid Prio*	-

Table 33: Covered Use Cases Feedback Page

5.4 Canteen Dashboard

The canteen Dashboard was introduced to support functionalities that are exclusively relevant and allowed for canteen employees. From the beginning, it was clear that delivering a fully mature and comprehensive dashboard within the scope of a semester project would be unrealistic, as the primary focus remained on the student view. Nevertheless, a functional and extensible dashboard was successfully implemented, providing a solid foundation for future development.

Some features, such as the menu import, were initially planned as fully automated background processes (e.g. cron jobs). However, based on discussions with the canteen and the provided data, these functionalities were integrated directly into the dashboard to allow more control and transparency. The design decisions, implemented features, and limitations of the canteen Dashboard are described in detail in the following chapters.

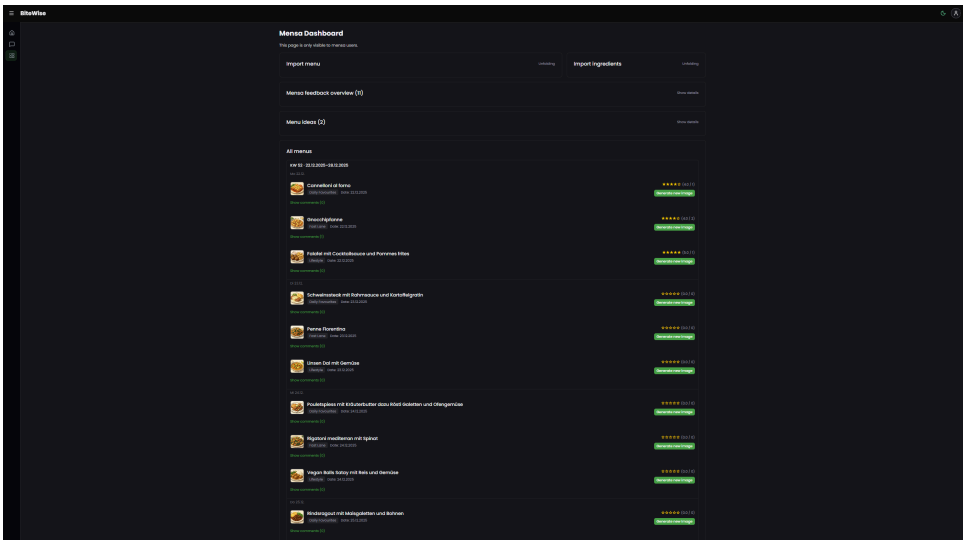

Figure 46: Canteen Dashboard Page

5.4.1 Menu & Ingredient Import

The initial concept for the menu import was to automatically generate menus directly from the weekly PDFs published on the canteen website. During the project, however, the canteen provided an Excel file containing the menu plan together with detailed ingredient information. This data source cannot be accessed automatically, as it is not publicly available via the canteen website.

As a result, the import process was redesigned to be executed manually via the canteen Dashboard.

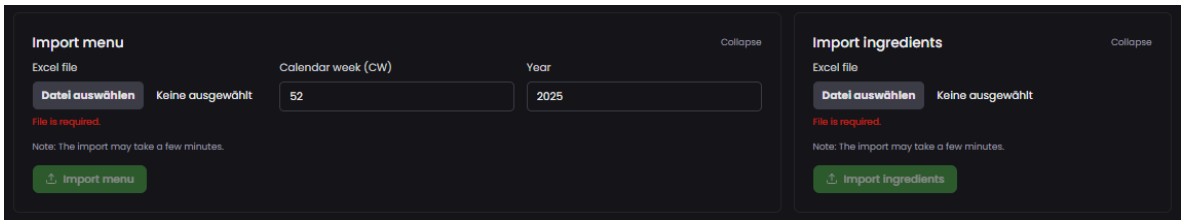

Figure 47: Canteen Dashboard - Import View

This approach allows canteen employees to upload the provided Excel files directly. Since the ingredient data is delivered separately from the menu structure, an additional dedicated ingredient import was required. This separation ensures that menus and ingredients can be processed, validated, and reused independently during the import workflow.

5.4.1.1 Ingredient Import

The ingredients import is implemented as a dedicated dashboard feature that allows canteen employees to upload the ingredients Excel file via a simple form. After the upload is completed, the user receives immediate feedback through toast notifications indicating whether the import was successful or if an error occurred. The import process takes approximately one minute.

The feature also supports uploading updated Excel files. Existing ingredients are updated based on their “Artikelnummer” (article number), while new ingredients are added automatically. This enables the canteen to continuously upload the latest

version of their ingredients file without additional manual cleanup, ensuring that the data always remains up to date. The article number was chosen as the primary identifier, as it is stable across Excel versions and used to identify ingredients in the menu plan excel.

All imported entries are stored in the “mensa_ingredients” table. In addition, each ingredient is mapped to the Smart Eating ingredients domain using the “Sachbezeichnung” as the identifier, and the corresponding “ingredient_id” is stored with the “mensa_ingredient” entry. This mapping was originally introduced to support nutritional value calculation based on ingredient data.

As part of this approach, the preanalysed recipe mechanism of Smart Eating was also evaluated, which allows structured data to be provided alongside the analyze request. However, this option did not prove useful for the canteen use case either, due to the complexity of the Smart Eating project and its strong focus on classic recipe structures. A lot of time was spent trying things out here and understanding the whole structure and mechanisms of smart eating.

Although this approach could not be fully realized within this project and is therefore not used for nutrition calculation (see Nutrition Calculation chapter), it was intentionally retained. For potential future extensions of Smart Eating, this mapping and the gained insights may become relevant again.

To support this integration, a new endpoint (POST “api/ingredient”) was added to the Smart Eating backend to add ingredients from BiteWise.

The diagram shows the external API interactions of the ingredient import.

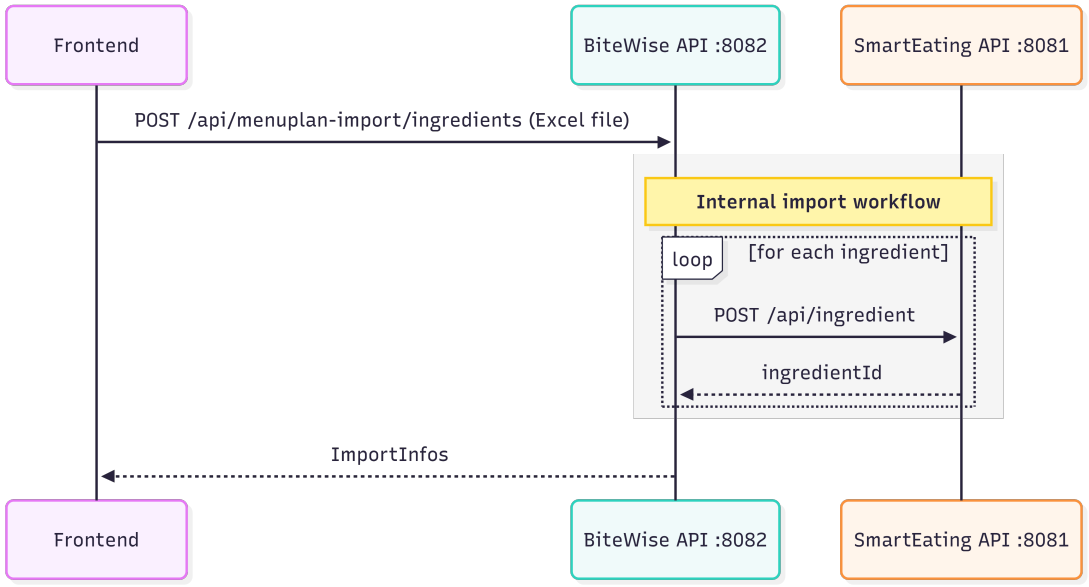

Figure 48: Sequence Diagram Ingredient Import

5.4.1.2 Menu Import

Menus can be imported on a weekly basis via the Menu Import feature in the canteen Dashboard. The uploaded Excel file must strictly follow the structure of the test menu plan provided by the canteen If this structure is respected, the menus are imported correctly and associated with the corresponding calendar week. Existing ingredients must already be present in the system. However, they do not need to be re-imported for every menu import.

During the import process, the user specifies a calendar week and a year. By default, the week after next is preselected. This reflects the canteen workflow, as the upcoming week is usually already visible to students, while the week after next is prepared in advance and imported before it becomes publicly available.

The menu import process takes a relatively long time due to the subsequent nutrition calculations performed via the Smart Eating services, which are described in more detail in the Nutrition Calculation chapter . On average, a single import takes approximately ten minutes. During this time, no intermediate progress is shown to the user. Once the import has completed, a toast notification is displayed to inform the user about the success or failure of the operation and remains visible until dismissed.

After the menu import has finished successfully, the image generation for the imported menus is triggered sequentially. This process is described in more detail in a later chapter. Once the import is completed, the canteen employee can find the imported menus in the list displayed below the import section. If the most recent week was imported, it appears immediately at the top of the list. If an older week was imported, the user may need to scroll to locate the corresponding entries.

This sequence diagram outlines the weekly menu import from an Excel file. For each imported menu, a Smart Eating recipe is generated via the analyze/save workflow. Also the subsequent image generation is triggered from the dashboard UI per imported menu.

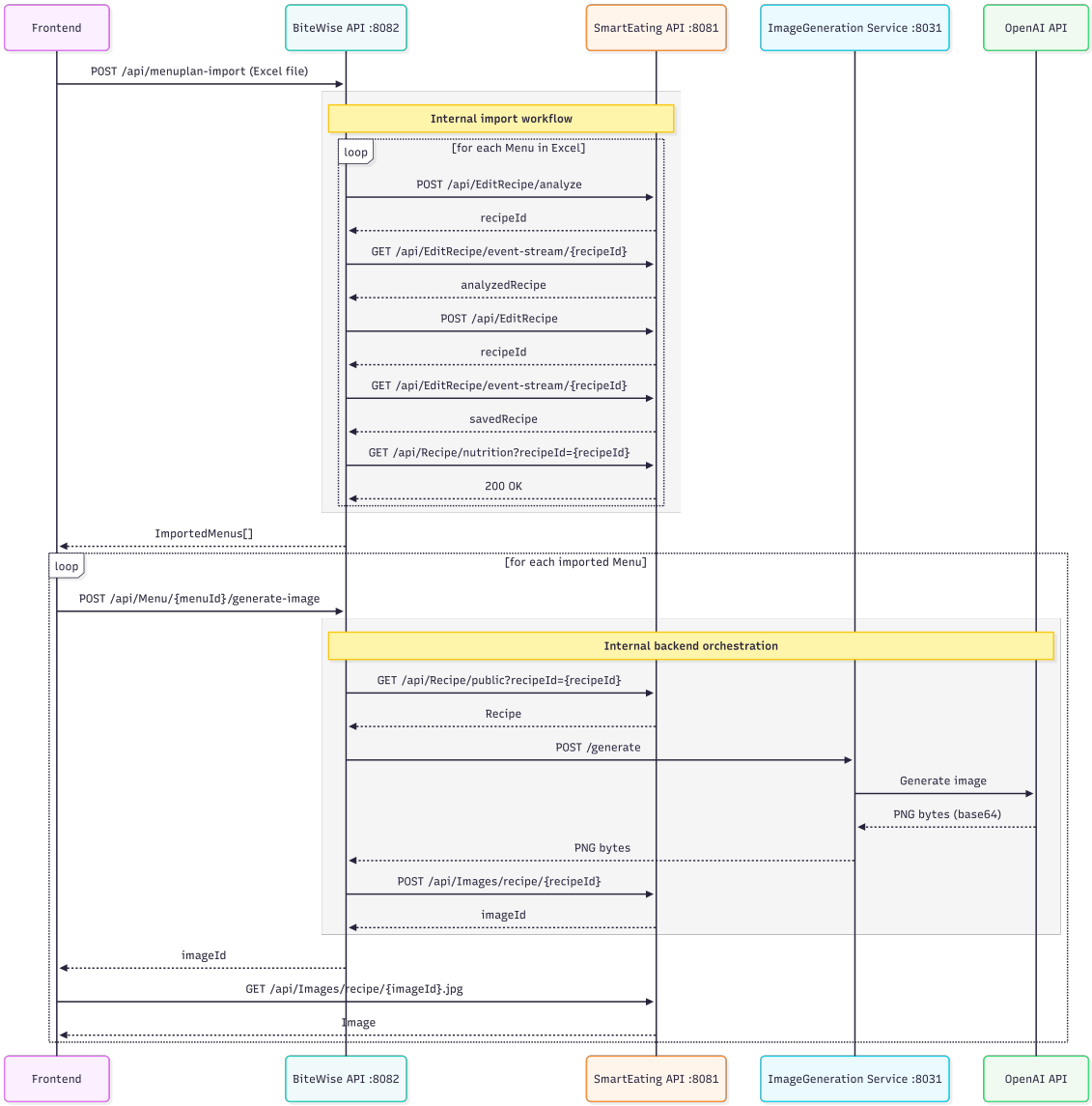

Figure 49: Menu Ingredient Import

5.4.1.3 Smart Eating Recipe Creation & Nutrition Data Generation

During the menu import process, a recipe is created in Smart Eating for each imported menu in order to reuse existing Smart Eating functionalities as much as possible. This interaction is handled by the BiteWise backend, which communicates directly with the Smart Eating backend. The generated recipe IDs are then stored with the corresponding menu entries.

Recipes are created using the Smart Eating Analyze endpoint. This endpoint is text-based and is designed to analyze a textual recipe description, automatically generating tags and nutrition values. This approach is not optimal for the canteen use case, as ingredients cannot be passed explicitly. Instead, ingredients are re-generated by AI based on the provided text input.

Nutrition values are subsequently calculated using the USDA FoodData database. This dependency is also the main reason why ingredients could not be manually mapped and the nutrition service called directly. Canteen ingredients are often product-level items rather than granular ingredients and do not have corresponding USDA identifiers, which makes direct mapping infeasible.

Using a prompt-based text description works reasonably well for generating certain tags, such as dietary labels (e.g. vegan). To improve the results, the prompt also includes the list of products (canteen ingredients) belonging to the menu, in order to provide as much context as possible to the AI. However, inconsistencies still occur. Even when using identical prompt text, a menu such as Vegan Gyros may be classified as vegan in some executions and as meat-based in others, despite the additional ingredient context.

An example of such a prompt is shown below:

```
  Recipe Title: Vegan Gyros mit Country Cuts und Peperoni

  Es folgt eine Liste mit Produkten, deren Unter-Ingredients und allen Spalten mit weiteren Infos:
  Artikelnummer;Artikelbezeichnung; // ...
  171656500000;Vegan Gyros TK 2x2.5kg;  // ...
  172487300000;Vegan Sce Cafe dParis0.8l;  // ...
  171672000000;CountryCutsTK2x2.5kg gew;  // ...
  171785500000;Peperoni grill TK 4x1kg;  // ...

  Erstelle ein passendes Rezept für eine Mensa-Hauptmahlzeit. Wichtig: für eine Portion (also 1
Person)
```

The generated nutrition data is generally of low quality. In many cases, the calculated values are unrealistic and typically far too low. This is likely due to Smart Eating being strongly optimized for classic recipes rather than complex canteen menus, where ingredient composition and portion sizes cannot be inferred reliably enough to map meaningful USDA ingredients.

This limitation would need to be addressed in a future evolution of BiteWise. For this project, however, the primary objective was to integrate and reuse Smart Eating as much as possible. Therefore, exact nutrition values were not the main focus, and the described approach was accepted.

A lot of time was spent trying things out here and understanding the entire structure and mechanisms of Smart Eating. In addition, the Smart Eating nutrition service often encountered an exception ‘Error processing recipe: “volume_ml”’ in the *_calculate_grams* function, which also occurred in the live version of the Smart Eating Website. We fixed this so that the Analyze always runs, as this is essential for our import and calculation.

5.4.2 Image generation

Early during the prototype design phase, it became clear that menu images would significantly improve the overall look and perceived quality of the menu plan UI. At the same time, it was also evident that the canteen cannot realistically create and upload images for every menu on a daily basis. As an initial fallback, it was agreed with the supervisor that placeholder images could be inserted manually into the database for testing purposes. However, the project was able to go one step further and implement automated image generation.

As already mentioned, images are generated automatically after a successful menu import, as long as the user remains on the dashboard page. In addition, images can also be generated manually for individual menu entries using the Generate Image button. Once the generation process finishes, the image is immediately displayed in the corresponding menu list item. If the generated image is not satisfactory, it can be regenerated at any time using the same action.

5.4.2.1 New Image Generation Service

To enable this functionality, a new service named imagegeneration was introduced within the Smart Eating ecosystem. The service is not specific to BiteWise, but implemented as a generic component that generates images via the OpenAI API based on a text prompt. It exposes an HTTP endpoint that accepts a prompt and optional generation parameters (e.g. size, model, quality) and returns the generated image as PNG bytes. This allows other Smart Eating sub-platforms or future projects to reuse the same service without modification.

To protect the service from external access, it is secured using an internal API key. The endpoint can only be called internally by providing a valid internal API key via a request header, which is configured through environment variables. This ensures that the image generation service is not publicly accessible and can only be used by trusted backend components within the Smart Eating infrastructure.

The BiteWise backend makes direct use of this mechanism and calls the imagegeneration service in a backend-to-backend communication setup. By supplying the internal API key, BiteWise can securely trigger image generation without exposing the service to external clients or the frontend.

5.4.2.2 OpenAI API Image Generation Pricing

The image generation service allows configuration of parameters that influence image quality, format, and pricing. Higher quality or larger images result in increased token consumption. The total price is determined by the number of input and output tokens and the price per token of the selected model. According to OpenAI documentation, this includes both image generation tokens and prompt input tokens (OpenAI, 2025a)

For development, a cost-effective combination of the gpt-image-1-mini model with low quality settings was used, generating images in a landscape format of 1536x1024. With these settings, an image consumes approximately 400 output tokens, while the image prompt contains around 330 tokens (OpenAI, 2025a; 2025b)

Based on these values, the cost per generated image is below CHF 0.01. In contrast, using high quality settings with the gpt-image-1 model results in approximately 0.25 CHF per image, leading to weekly costs of around CHF 3.75 for the canteen (OpenAI, 2025c)

5.4.2.3 Sequence Diagram Image Generation

The diagram depicts how menu images are generated and uploaded. The frontend triggers image generation per menu, while the backend orchestrates the calls to the internal imagegeneration service and the SmartEating API. The resulting imageId is returned to the frontend, which then loads the image via the public image endpoint.

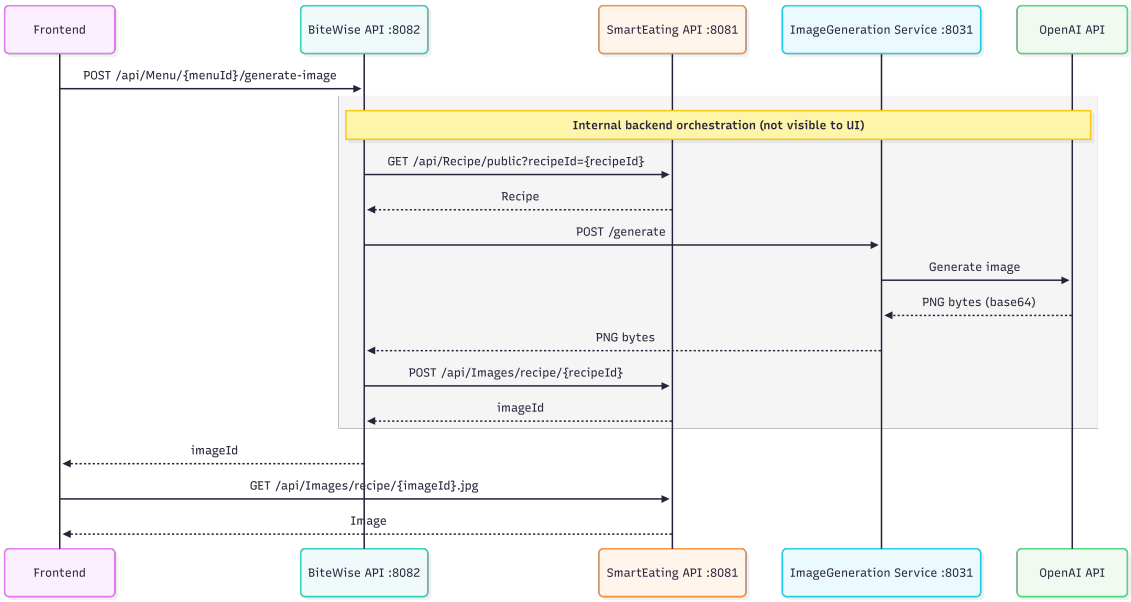

Figure 50: Sequence Diagram Image Generation

5.4.3 Feedback Insights

The Feedback Insights section provides canteen employees with an aggregated overview of all feedback submitted by students. This includes menu-specific ratings, general canteen feedback, and submitted menu ideas. The goal of this section is to make student feedback easily accessible.

5.4.3.1 Menu Feedback

For each menu entry in the menu list, the canteen can see the average star rating submitted by students. By expanding a menu entry, all related comments can be viewed. Menu comments are always displayed anonymously to the canteen, ensuring that individual students cannot be identified.

Figure 51: Canteen Dashboard - Menu Feedback View

Menu feedback is implemented using the existing Smart Eating recipe rating system, where ratings are stored in the user database. These ratings are retrieved via the Smart Eating Web App API and aggregated for display in the canteen Dashboard.

The endpoint for retrieving ratings in Smart Eating was protected for the respective user. It had to be adapted so that all users could retrieve it, as the data is also needed in the Daily Menu View when users are not logged in, and the canteen user also must be able to retrieve the ratings.

5.4.3.2 General Canteen Feedback

In the general feedback section, the canteen can view the three most recent feedback entries by default. By expanding the section, all submitted feedback entries become accessible. In addition to written comments, the dashboard displays the average star ratings per category as well as the total number of submitted feedback entries. If a user chose not to submit the feedback anonymously, their name is displayed alongside the corresponding comment.

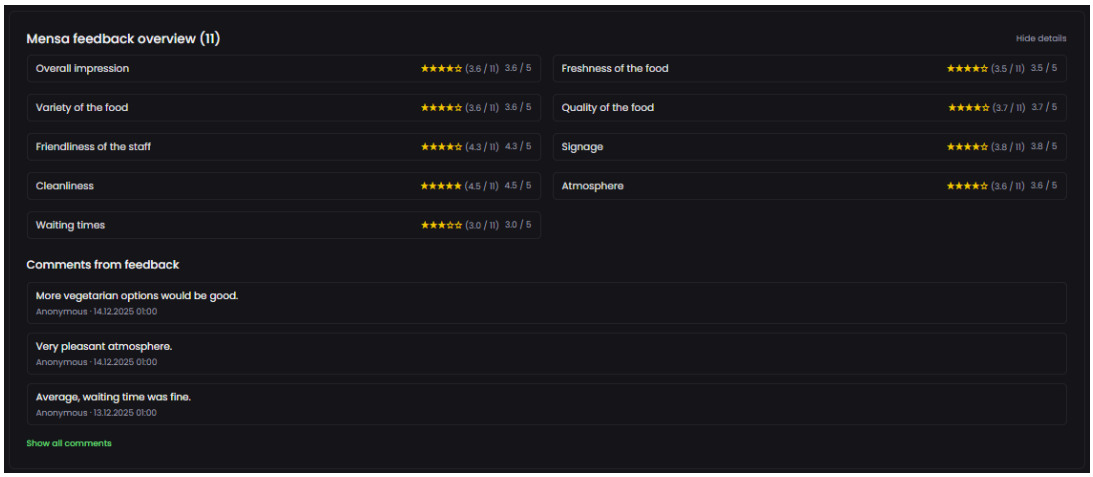

Figure 52: Canteen Dashboard - Feedback View

The rating categories and data model for general canteen feedback were newly designed and implemented specifically for BiteWise and are stored exclusively in the BiteWise database.

5.4.3.3 Menu Ideas

All submitted menu ideas from students are listed in a dedicated section. The three most recent suggestions are shown first, with the option to expand the list to view all entries. For menu ideas, the submitting user is always visible to the canteen, allowing follow-up or clarification if needed.

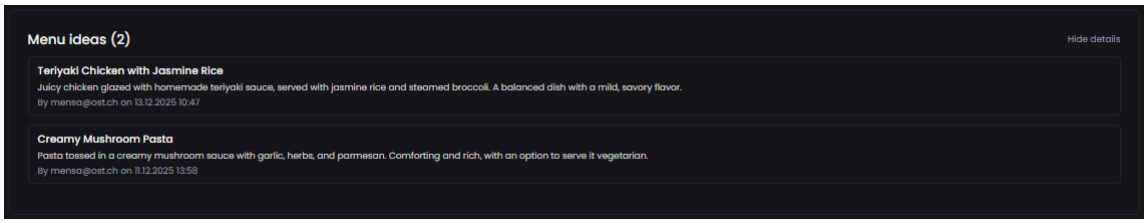

Figure 53: Canteen Dashboard - Menu Ideas View

Menu ideas were implemented as a new BiteWise-specific feature and are persisted in the BiteWise database.

5.4.4 Menus List

To allow the canteen to review both previously served menus and menus that have already been prepared for future weeks, the dashboard provides a menu list view. The menus are grouped and ordered by calendar week and day, giving canteen employees a clear and structured overview of all available menus across past and future weeks.

For each menu entry, the list displays the menu title, the corresponding lane, the generated image, and the current average rating and comments. Images can be regenerated directly from this view, as described earlier, which allows quick visual adjustments without navigating to a separate screen. The ratings displayed on the right side provide immediate insight into student reception and support informed menu planning for upcoming weeks.

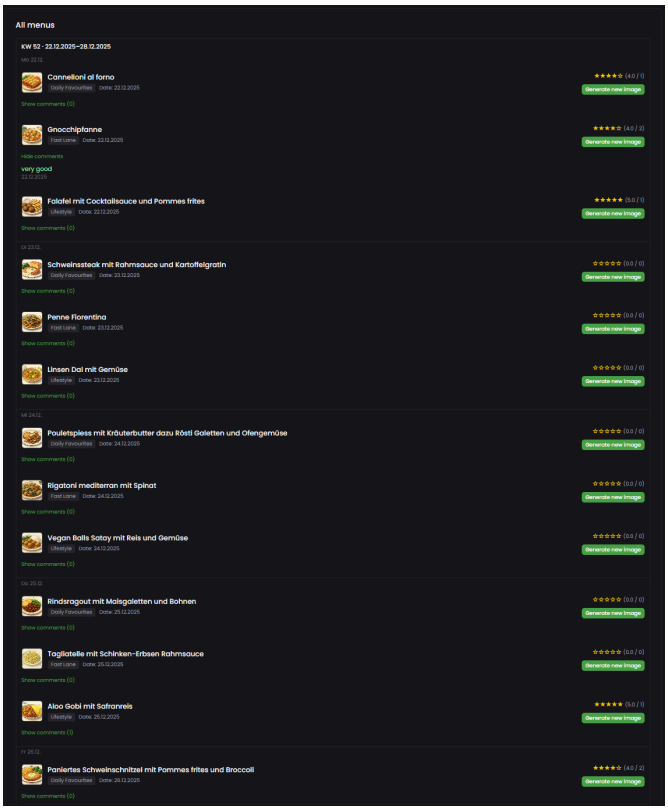

Figure 54: Canteen Dashboard - Menus List View

The menus and their images are fetched using the same backend endpoints as in the student view. However, unlike the student-facing menu plan, menus in the dashboard are not loaded by date. Instead, they are retrieved using pagination with offset and limit, as it cannot be assumed which dates already contain imported menus.

The list loads 15 menus (1 week) at a time, with the most recently created menus displayed at the top. Additional menus are fetched dynamically as the user scrolls. This incremental loading approach improves performance and ensures fast initial page load times, even when a large number of menus is stored in the system.

5.4.5 Authentication and Security

The canteen user with the email address mensa@ost.ch authenticates using the same login mechanism as regular users. This approach allows the canteen user to access and verify the application from the same perspective as students, including the regular menu views. After a successful login, canteen users are automatically redirected to the canteen Dashboard.

All sensitive endpoints used by the dashboard are protected against unauthorized access. BiteWise-specific endpoints required for dashboard functionality are explicitly restricted to the user mensa@ost.ch. In addition, selected Smart Eating endpoints were adapted to enforce the same access constraints. The UploadRecipeImage functionality is further secured using the internal API key mechanism to prevent external access. For certain endpoints, the canteen user is also required to have the Role.Admin role.

At the moment, the canteen user is created during the application setup process. In a production-ready environment, user credentials would need to be removed from the codebase.

5.4.6 Sequence Diagram Canteen Dashboard

The sequence diagram illustrates the data loading process of the canteen Dashboard. After the initial page load, the first set of menus is retrieved using pagination. Additional menus are then loaded incrementally as the user scrolls through the dashboard, with each menu being enriched with recipe details, ratings, and images.

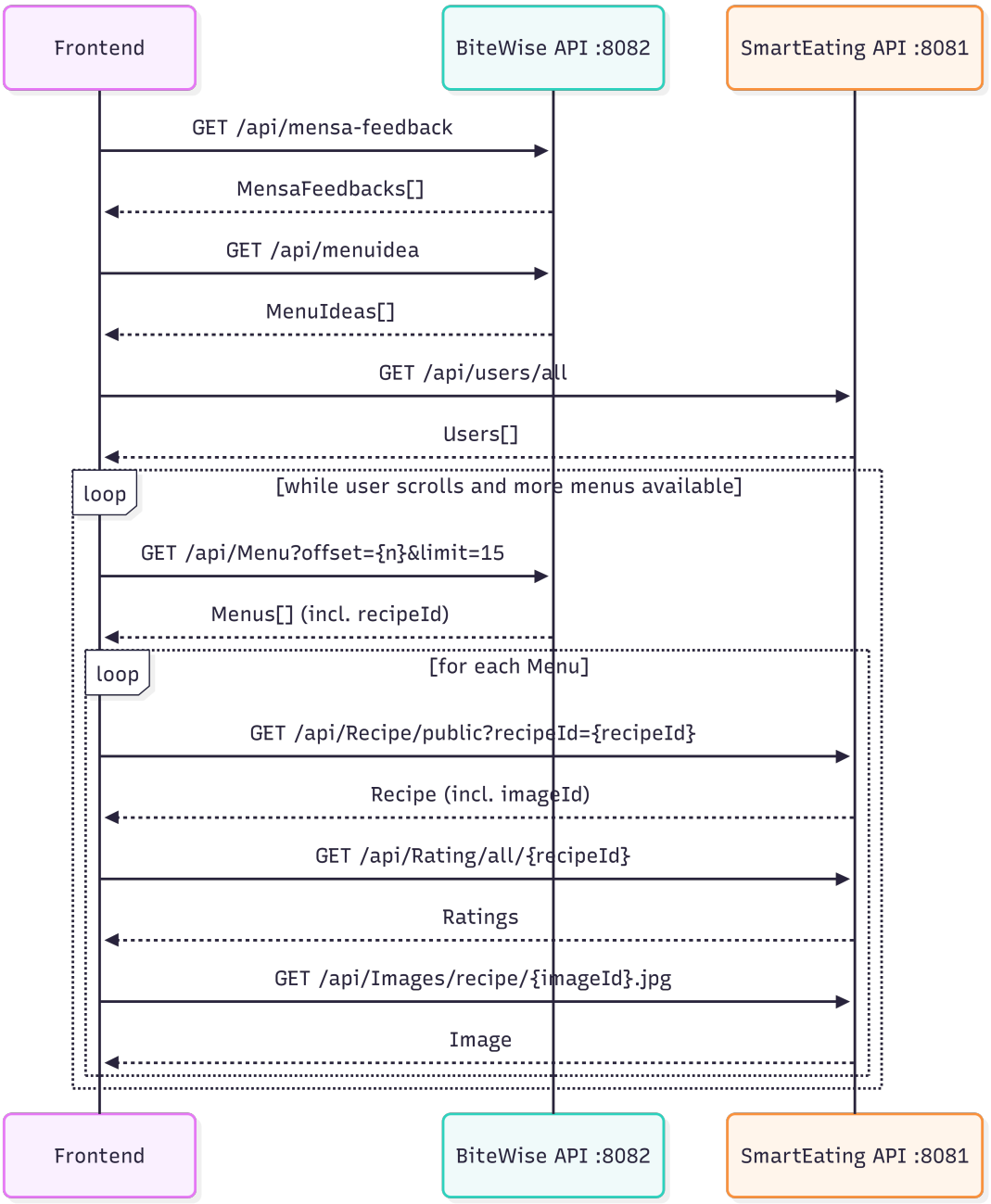

Figure 55: Sequence Diagram Canteen Dashboard

5.4.7 Covered Use Cases

Use Case ID	Use Case	Priority	Deviations
UC-S01	Menu Import	*High Prio*	Originally planned as a fully automated scheduler-based PDF import from the canteen website. In the final implementation, menus are imported manually via the dashboard using a canteen-provided Excel file, because the enriched data source (including products/ingredients) is not publicly retrievable.
UC-S02	Nutrition Data Generation	*High Prio*	Nutrition data generation is triggered automatically as part of the same backend workflow after the manual menu import (within the same request flow), rather than being executed as an independent scheduled task. In addition, the generated nutrition values are not stored directly on the menu entity, but are persisted in Smart Eating as part of the created recipe and referenced from the menu via the stored recipe ID.
UC-M01	Mensa Login	*High Prio*	The login procedure for the canteen remains unchanged as described in the use case. However, the originally planned distinction between Admin and canteen Employee roles was simplified. In BiteWise, only the dedicated canteen account has access to the dashboard, reducing complexity for this project.
UC-M02	Feedback Insights	*High Prio*	-
UC-S03	Menu Image Generation	*Mid Prio*	Originally planned as a scheduler-driven background job with fallback images. In the final implementation, images are generated immediately after a manual menu import and can also be triggered or regenerated per menu via the UI. Generation is performed through a dedicated internal imagegeneration service using AI.

Table 34: Covered Use Cases Canteen Dashboard

5.5 Nutrition Insights Page

When clicking on the see details button on the menu plan page in daily menu view, the student gets nutritional information of each menu. The displayed nutritions for each menu are: Calories, protein, carbs, fat and fiber. The decision to display these nutritions directly come from our students questionnaire from the beginning of this SA. As already mentioned in chapter 5.2.1.3 , the nutritional values are not very precise, which is a thing that has to improve on the Smart Eating project before a production release.

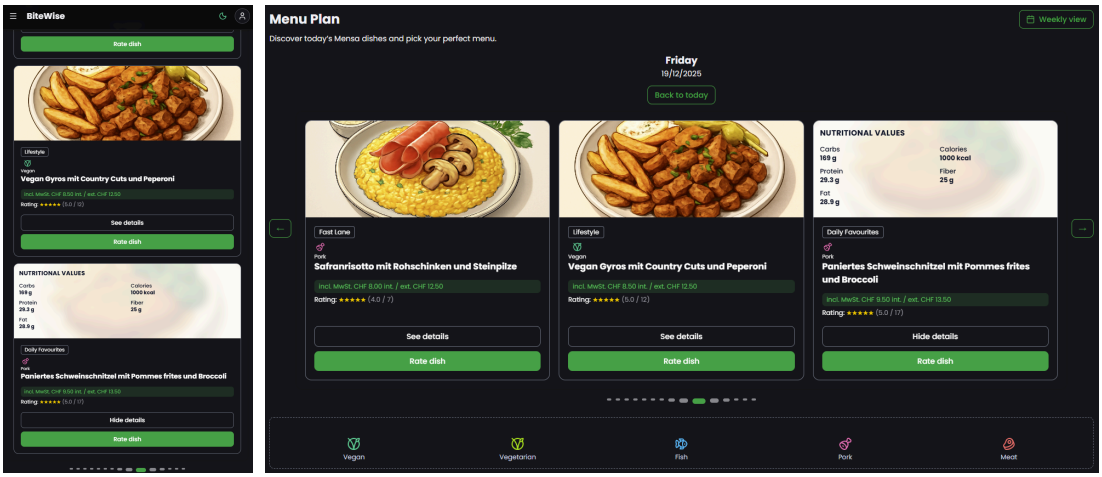

Figure 56: Nutrition Insights Page

5.5.1 Covered Use Cases

Use Case ID	Use Case	Priority	Deviations
UC-U05	Nutrition Insights	*High Prio*	-

Table 35: Covered Use Cases Nutrition Insights Page

5.6 Personal Dietary Matching Page

A logged in student can set their personal dietary preferences to get personalised recommendations on what menu is best suited for them or not.

5.6.1 Meal Preferences Settings

On the top right corner of the application, where the user icon is, the student can set their personal meal preferences under the navigation item with the same name. There the student can first set their preferred eating habits. They can choose from: Vegan, Vegetarian and no pork. Here as well, due to health concerns, the settings for lactosefree and glutenfree have been removed.

In a second step, students can change their amount of intake for each of these nutritions: Calories, protein, carbs, fat and fiber. They can choose with a slider from low, average and high intake.

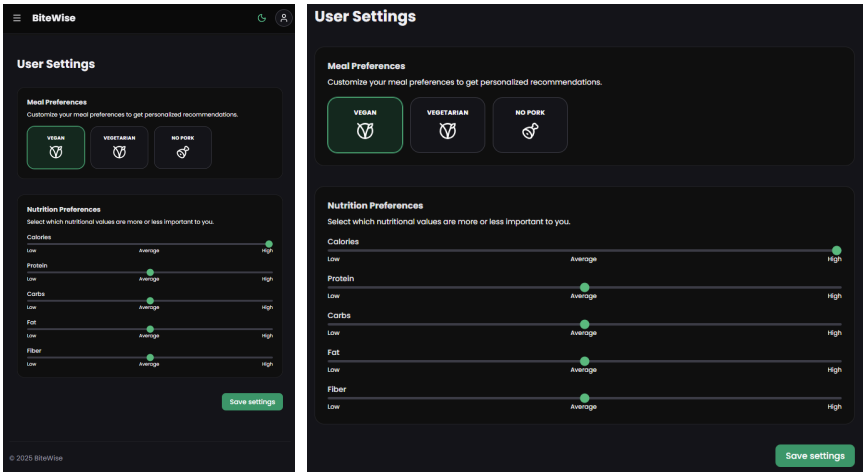

Figure 57: Dietary Matching Page

5.6.2 Personalised Menu Plan Page

Now with their personal meal preferences set, students immediately see which menus are recommended for them (marked with a green border and the “for you” tag) and which menus definitely do not match with their preferences (greyed out and tagged with “Not suitable for you”).

The matching logic is defined as follows:

1st Priority: Match the preferred eating habits (tags) and filter out menus after strict rules:
- If vegan: Only recommend vegan menus, grey out all others.
- If vegetarian: Recommend vegan or vegetarian menus, grey out everything with meat and fish.
- If no pork: No recommendation for now, grey out everything with pork.

2nd Priority: For all still eligible menus after step 1, go through each nutrition and keep track of a matching score after these rules:
- If nutrition is set on average: No influence.
- If nutrition is set on high: Menu with highest amounts of this nutrition gets increased score by 1.
- If nutrition is set on low: Menu with lowest amounts of this nutrition gets increased score by 1.
- Do this for each nutrition, compare scores of the three menus and recommend the menu with the highest score.
- If multiple menus have the same highest score, recommend all of them.

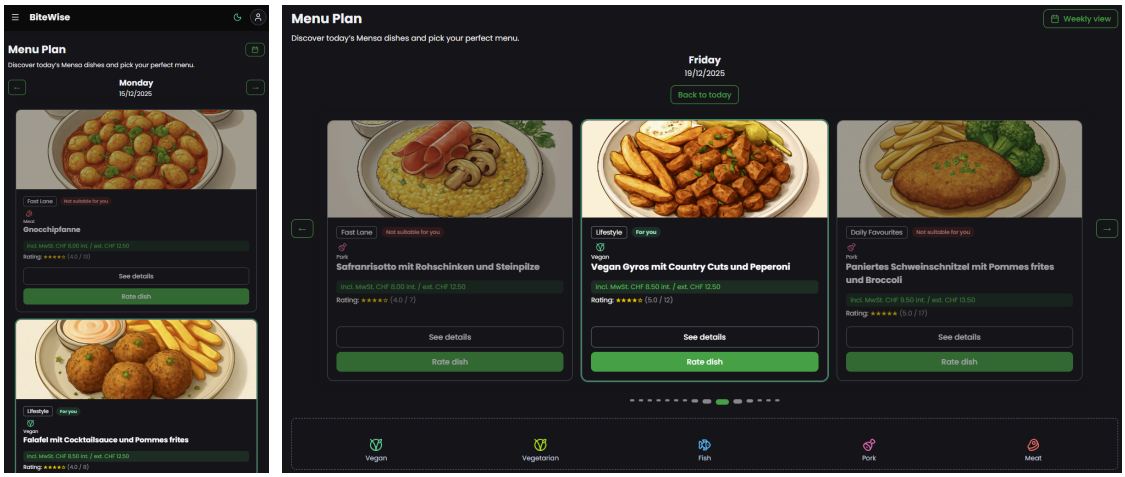

Figure 58: Personalised Daily Menu Plan Page

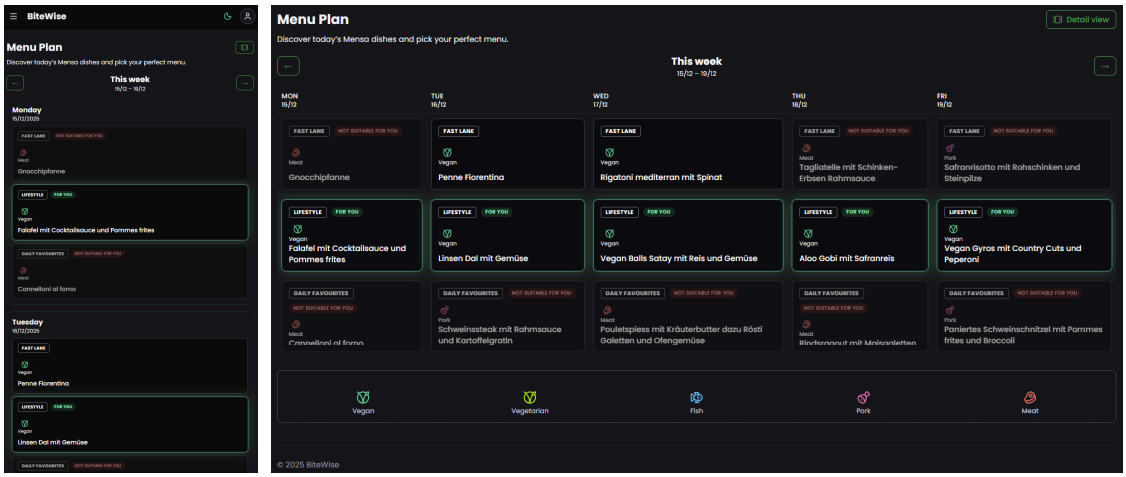

Figure 59: Personalised Weekly Menu Page

5.6.3 Activity Diagram

The activity diagram documents the two-stage matching logic used for personalised recommendations.

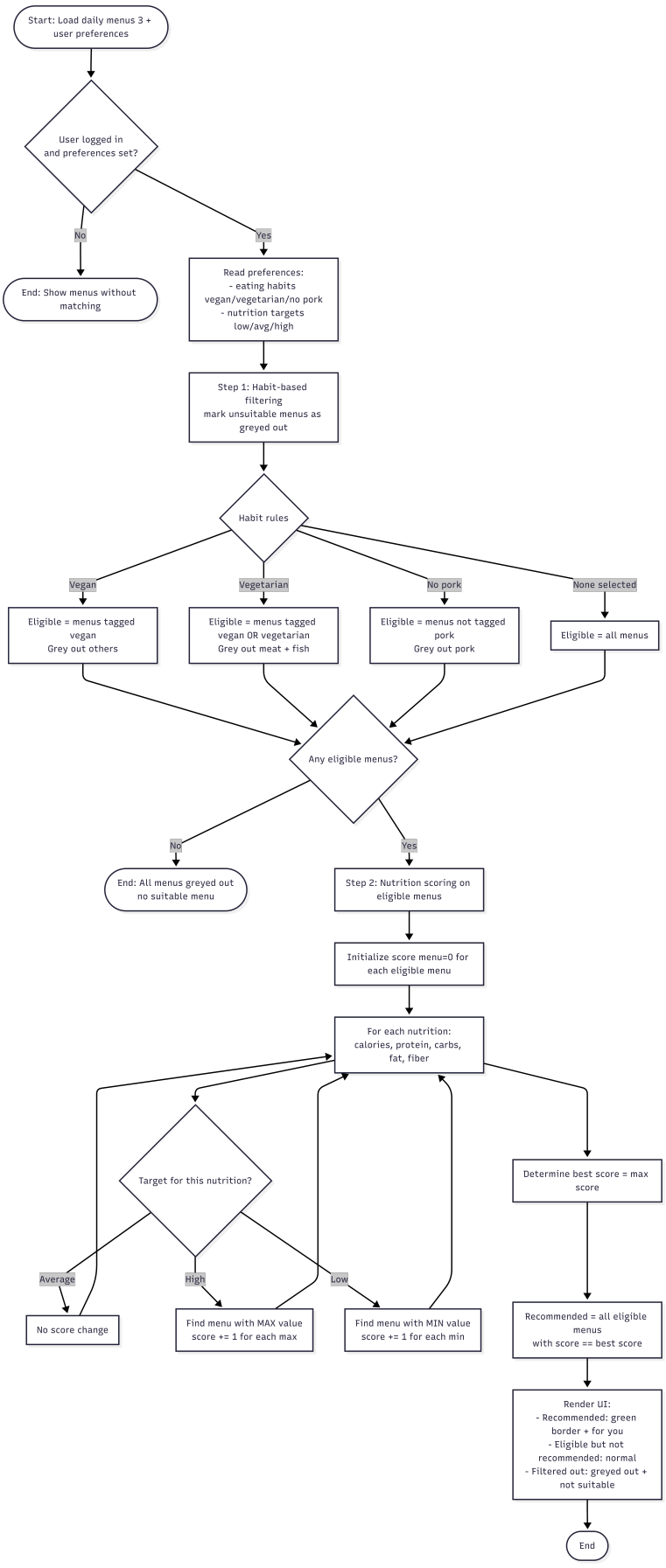

Figure 60: Activity Diagram Personal Dietary Matching

5.6.4 Covered Use Cases

Use Case ID	Use Case	Priority	Deviations
UC-U04	Personal Dietary Matching	*High Prio*	The recommendations do not show why the menu has been highlighted as best suited or not compatible.

Table 36: Covered Use Cases Personal Dietary Matching Page

5.7 Dark & Light Mode

A light and dark mode were implemented, following the browser's system settings by default, which ensures a good user experience. However, the manual toggle button for switching between modes is currently not functional.

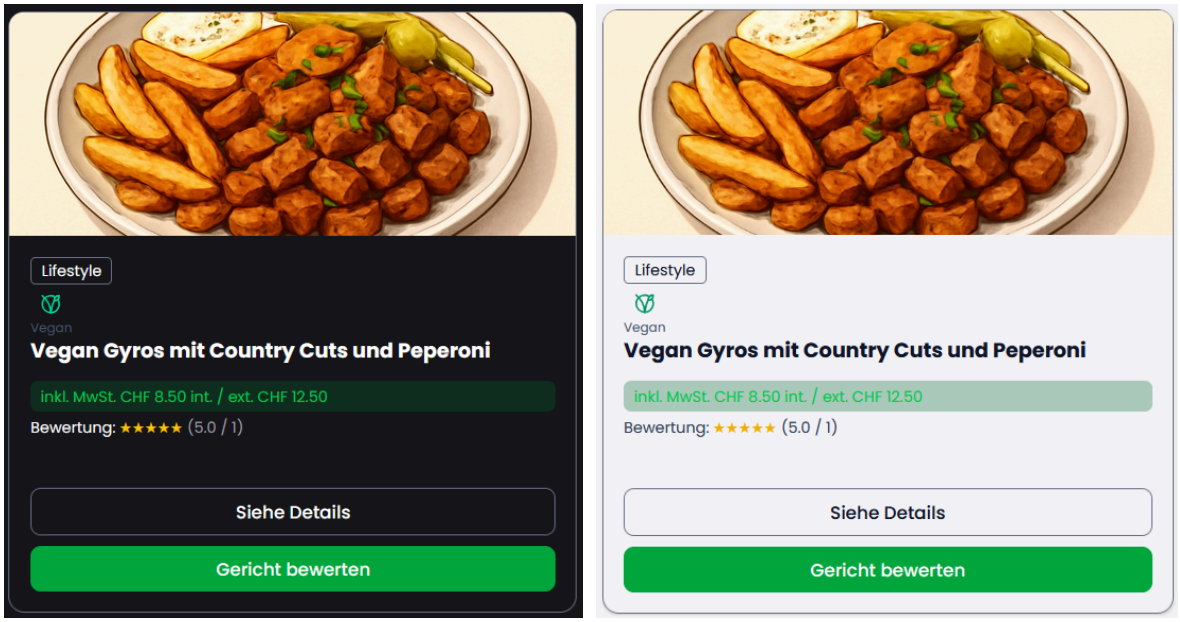

Figure 61: Example Dark / Light Mode

The following code example illustrates how Tailwind CSS can be used to apply different styles for light and dark mode.

```
<h1 className="text-slate-600 dark:text-gray-500">Example</h1>
```

5.8 Backend API Specification

5.8.1 BiteWise API

The following endpoints represent the BiteWise API endpoints that were newly introduced and implemented as part of this project. They provide the BiteWise-specific backend functionality required for this project and are documented using the automatically generated OpenAPI specification.

5.8.1.1 Canteen Feedback

This endpoint allows the canteen user to retrieve all submitted canteen feedback entries for review and analysis.

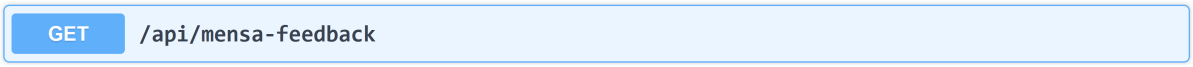

Figure 62: Canteen Feedback GET Endpoint

This endpoint allows students to submit new general feedback about the canteen, including ratings and optional comments.

Figure 63: Canteen Feedback POST Endpoint

5.8.1.2 Menu

The menu endpoint is used to retrieve menu data for both student and canteen views.

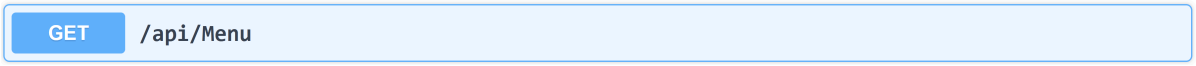

Figure 64: Menu GET Endpoint

The generate image endpoint triggers AI-based image generation for a specific menu entry and is restricted to the dedicated canteen user.

Figure 65: Generate Image Endpoint

5.8.1.3 MenuIdea

This endpoint allows the canteen user to retrieve all submitted menu ideas for evaluation.

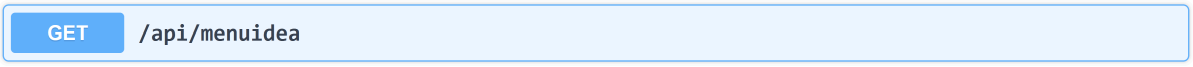

Figure 66: Menu Idea GET Endpoint

This endpoint allows logged-in students to submit new menu ideas for future dishes.

Figure 67: Menu Idea POST Endpoint

5.8.1.4 MenuPlanImport

This endpoint enables the canteen user to import weekly menus from an Excel file into the BiteWise system.

Figure 68: Menuplan Import Endpoint

This endpoint enables the canteen user to import and update ingredient data from an Excel file used during menu planning.

POST /api/menuplan-import/ingredients

Figure 69: Ingredients Import Endpoint

5.8.2 SmartEating API

5.8.2.1 New created Endpoints

This endpoint was introduced to allow BiteWise to upload generated recipe images to the Smart Eating system. It is used as part of the AI-based image generation workflow and is protected by an internal API key to ensure that only trusted backend services can access it.

POST /api/Images/recipe/{recipeId}

Figure 70: Create Recipe Image Endpoint

This endpoint was added to support the ingredient import process from BiteWise. It allows the creation of ingredient entities in the Smart Eating domain and is restricted to users with the “Role.Admin” role.

POST /api/Ingredient

Figure 71: Create Recipe Ingredient Endpoint

5.8.2.2 Adapted Endpoints

This endpoint was adapted to remove user-specific authentication constraints. The change was required so that recipe ratings can be retrieved in unauthenticated scenarios (e.g. Daily Menu View) and by the canteen user within the canteen Dashboard.

GET /api/Rating/all/{recipeId}

Figure 72: GET Recipe Ratings Endpoint

5.8.3 Image Generation API

To automate the creation of menu images, a dedicated Image Generation API was introduced as an internal Smart Eating service running on port 8031. The service provides a minimal HTTP interface that generates images based on a text prompt and returns the result as PNG bytes. Access is restricted via an internal API key to prevent external usage.

5.8.3.1 Generate Image

POST http://localhost:8031/generate Generates an image from a given prompt and returns the PNG image as raw bytes (image/png).

Request body (JSON):

- prompt (string, required): Text prompt describing the desired image
- size (string, optional): 1024x1024 | 1024x1536 | 1536x1024 (default: 1536x1024)
- quality (string, optional): low | medium | high (default: low)
- model (string, optional): gpt-image-1-mini | gpt-image-1 (default: gpt-image-1-mini)

Required header:

- X-Internal-Api-Key: Internal key configured via environment variables

Response:

- 200 OK: PNG bytes (image/png)
- 401 Unauthorized: Missing or invalid internal API key
- 502 Bad Gateway: Image generation failed or no valid image data returned

6 Quality Measures

6.1 Code Quality

6.1.1 Working Environment

We define some quality measures and guidelines to ensure a high quality of software and documentation and consistency across the team.

6.1.1.1 Documentation

- Document language is English.
- The documentation is written in Typst.
- A folder is created for each main chapter (e.g. 2), with a Typst file labeled “00_<ChapterName>.typ”.
  - Within these folders, for each sub-chapter (e.g. 2.1), a new Typst file is created labeled “XX_<SubChapterName>.typ” the number corresponds with the chapter number.
- To make collaboration easier in Git, each new sentence is written on a new line.

6.1.1.2 Code

- To ensure consistency across the code, linters are used by each developer locally in their IDE.
- The code is commented wherever necessary.
- Classes and methods are named according to the standard naming conventions.

6.1.1.3 Jira

- Jira is used to track the progress of the project.
- Each task is tracked by a ticket.
- Tickets in the TO DO column have a description of the task with a definition of done.
- When a developer works on a ticket, it is moved to the IN PROGRESS column and assigned to the developer.
- When the task is completed, the developer creates a pull request and moves the ticket to the IN REVIEW column.
- After the pull request is approved and merged, the ticket is moved to the DONE column and the developer logs the time spent on the task.

6.1.1.4 Version Control

- A polyrepo in GitLab is used to manage the code.
  - One repo is the forked project of the original Smart Eating project to push the new code of the sub-platform.
  - The second repo is for the documentation.
- Commits and branches are done in English.

6.1.1.4.1 Branching

- Each task is worked on in a separate branch.
- The branch names are structured as follows:
  - For the forked coding repo: The branch name is the prefix, followed by the ticket number followed by a short description of the task. Separated by dashes (e.g. “feat/bw-12-add-new-feature”).
  - For the documentation repo: The branch name is the ticket number followed by a short description of the task. Separated by dashes (e.g. “bw-22-write-documentation”).
- The main branch is protected and can only be modified via pull requests.

Prefix	Usage
feat/	For developing new features
bugfix/	To fix bugs in the code

Table 37: Branch Prefixes

6.1.1.4.2 Commits

- Commits are done regularly and in a structured manner.

- Commit messages are the ticket number followed by a short message describing the changes (e.g. “BW-12: Changed method to be more efficient”).

6.1.1.4.3 Merge Requests

- Merge requests are created for each branch.
- The name of the merge request is the same as the branch name.
- The merge request is created from the branch to the main branch.
- The merge request is reviewed by a second developer and needs an approval.
- Commits are squashed and the feature branch is deleted after merge.

6.1.2 Test Concept

This test concept describes how the quality of the project will be ensured. Although the overall scope of testing is intentionally limited, the selected tests demonstrate a structured and maintainable approach to validating the system. The goal is to test the central functional behaviour of the application in both backend and frontend, while keeping the effort in a reasonable balance with the project size.

6.1.2.1 Functional Requirements

6.1.2.1.1 Unit Tests

Backend

The backend follows a modular architecture consisting of controllers, services and a context abstraction. This structure enables isolated testing of individual components without requiring the full application to run.

A small set of unit tests will be implemented in the backend, focusing on the most relevant parts of the business logic:

- Service tests
- Controller tests using mocks for the service layer

The purpose of these tests is to verify:

- correct handling of valid and invalid inputs
- correct execution of processing logic
- proper interaction with the service layer
- correct error handling behaviour

These tests will be executed locally during development.

Frontend

Frontend unit tests will be implemented with Jest and React Testing Library, both of which are part of the smart eating project setup. The focus lies on components with logic rather than visual appearance:

- React components that contain conditional rendering or interactive behaviour
- Redux Toolkit slices (reducers, actions, selectors)

These tests ensure that UI behaviour is predictable and that state transitions remain stable as the application evolves.

6.1.2.1.2 Integration & System Tests

Due to the limited project scope and available time, we focused our testing efforts on areas where they provide the highest value with reasonable effort. The backend architecture was explicitly designed to be modular and therefore well suited for unit testing, allowing us to validate core logic without requiring a full application environment. In contrast, integration and system tests would involve complex external dependencies such as authentication, external APIs and database interactions, significantly increasing setup and maintenance overhead. For the defined MVP, manual end-to-end validation is sufficient to ensure that the most important user flows function correctly, making a fully automated integration or system test suite unnecessary for this project.

6.1.2.1.3 Summative Usability and Acceptance Tests

In the second to last project week, we will conduct a final summative usability and user acceptance test of the current MVP with representative end users (e.g. students and canteen employees). The goal is to validate the MVP from an end user perspective and to collect qualitative feedback for future improvements.

The test will be carried out with a small group of 3 participants. Each participant will perform predefined tasks, such as viewing the menu plan for a specific day. The sessions will be moderated and will take approximately 30 minutes per person. During the test, we will observe user behaviour, ask follow-up questions and use a short questionnaire to capture perceived usability and satisfaction.

The results of this final user test will be documented and used as input for potential future development of the system. Due to the limited time frame of the project, identified improvements will not be implemented within this thesis.

6.1.2.2 Non-Functional Requirements

No dedicated automated tests are implemented for non-functional requirements (NFRs). Instead, verification is performed through manual performance evaluations, code inspections or behavioural checks during development and deployment. The verification methods for each NFR are defined directly within the NFR section of this documentation.

6.1.3 Test Protocols

6.1.3.1 Backend Tests

The successfully completed backend tests are shown here.

6.1.3.1.1 Controller Tests

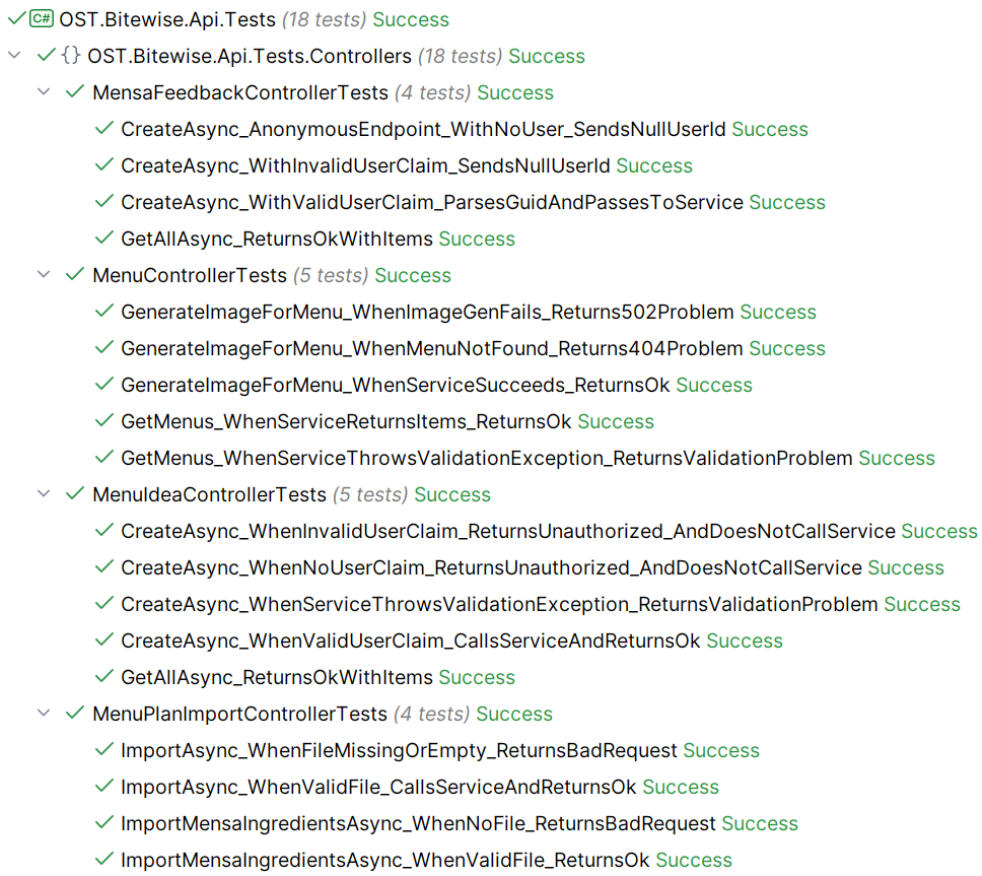

Figure 73: Backend Controller Tests

6.1.3.1.2 Service Tests

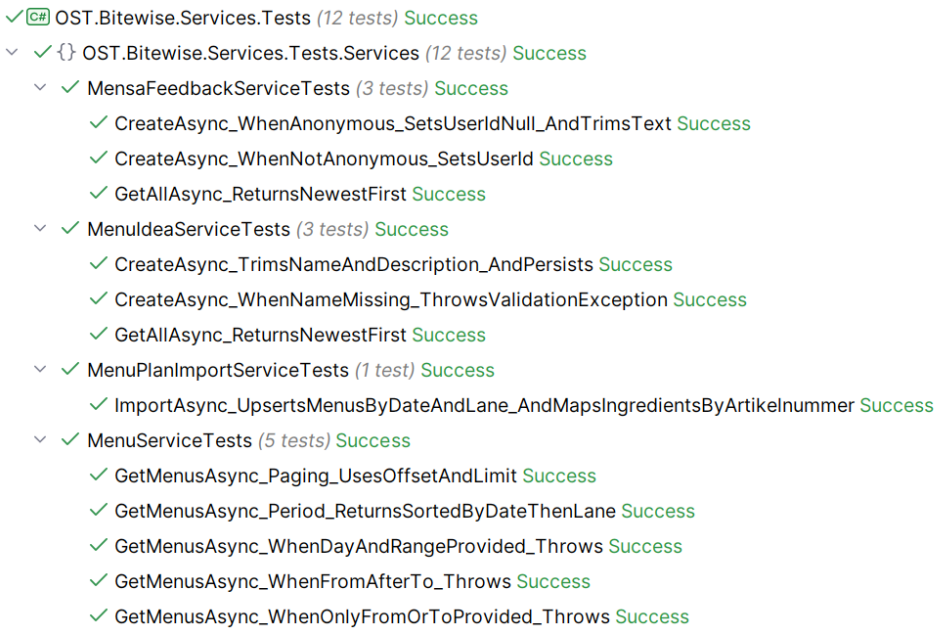

Figure 74: Backend Service Tests

6.1.3.1.3 Code Coverage

All controllers have 100% code coverage because tests were written for all of them.

– OST.Bitewise.Api	100	196	296	586	33.7%	18	50	36%
OST.Bitewise.Api.Auth.WebAppAnyAuthenticationHandler	0	28	28	61	0%	0	8	0%
OST.Bitewise.Api.Auth.WebAppAuthenticationHandler	0	36	36	77	0%	0	12	0%
OST.Bitewise.Api.Auth.WebAppMachineLogin	0	29	29	55	0%	0	4	0%
OST.Bitewise.Api.Controllers.MensaFeedbackController	18	0	18	46	100%	4	4	100%
OST.Bitewise.Api.Controllers.MenuController	32	0	32	77	100%	0	0	
OST.Bitewise.Api.Controllers.MenuIdeaController	20	0	20	54	100%	4	4	100%
OST.Bitewise.Api.Controllers.MenuPlanImportController	30	0	30	69	100%	10	12	83.3%
Program	0	103	103	147	0%	0	6	0%

Figure 75: Backend Api Tests

Tests were created for the four services for which it was deemed appropriate. MenuImageService and RecipeCreationService were not tested because they mainly orchestrate multiple external systems and APIs, and isolating them with extensive mocking would add high maintenance overhead while providing limited additional test value. For the MenuPlanImportService, only one of the two functions was tested, which is why the coverage there is 50%.

– OST.Bitewise.Services	260	427	687	1186	37.8%	44	188	23.4%
OST.Bitewise.Services.DTOs.DayDto	3	0	3	9	100%	0	0	
OST.Bitewise.Services.DTOs.ImageUploadResultDto	0	1	1	4	0%	0	0	
OST.Bitewise.Services.DTOs.IngredientImportRequest	0	1	1	9	0%	0	0	
OST.Bitewise.Services.DTOs.IngredientImportResult	0	1	1	4	0%	0	0	
OST.Bitewise.Services.DTOs.MensaFeedbackCreateDto	11	0	11	38	100%	0	0	
OST.Bitewise.Services.DTOs.MensaFeedbackMapping	18	0	18	27	100%	0	0	
OST.Bitewise.Services.DTOs.MensaFeedbackReadDto	5	11	16	38	31.2%	0	0	
OST.Bitewise.Services.DTOs.MenuCreateDto	0	1	1	16	0%	0	0	
OST.Bitewise.Services.DTOs.MenuDto	3	0	3	11	100%	0	0	
OST.Bitewise.Services.DTOs.MenuIdeaCreateDto	2	0	2	19	100%	0	0	
OST.Bitewise.Services.DTOs.MenuIdeaMapping	3	0	3	12	100%	0	0	
OST.Bitewise.Services.DTOs.MenuIdeaReadDto	5	2	7	19	71.4%	0	0	
OST.Bitewise.Services.DTOs.MenuItemDto	1	1	2	8	50%	0	0	
OST.Bitewise.Services.DTOs.MenuMapping	3	0	3	12	100%	0	0	
OST.Bitewise.Services.DTOs.MenuPlanImportRequest	0	3	3	11	0%	0	0	
OST.Bitewise.Services.DTOs.MenuReadDto	4	3	7	16	57.1%	0	0	
OST.Bitewise.Services.DTOs.MenuUpdateDto	0	1	1	16	0%	0	0	
OST.Bitewise.Services.DTOs.RecipeIngredientRefDto	0	3	3	9	0%	0	0	
OST.Bitewise.Services.Parsing.MensaIngredientSheetParser	0	49	49	70	0%	0	14	0%
OST.Bitewise.Services.Parsing.MenuPlanSheetParser	0	109	109	164	0%	0	68	0%
OST.Bitewise.Services.Services.MensaFeedbackService	33	0	33	54	100%	4	4	100%
OST.Bitewise.Services.Services.MenuIdeaService	27	0	27	50	100%	3	4	75%
OST.Bitewise.Services.Services.MenuImageService	0	90	90	136	0%	0	18	0%
OST.Bitewise.Services.Services.MenuPlanImportService	98	97	195	264	50.2%	10	42	23.8%
OST.Bitewise.Services.Services.MenuService	44	3	47	85	93.6%	27	30	90%
OST.Bitewise.Services.Services.RecipeCreationService	0	51	51	85	0%	0	8	0%

Figure 76: Backend Services Code Coverage

6.1.3.2 Frontend Tests

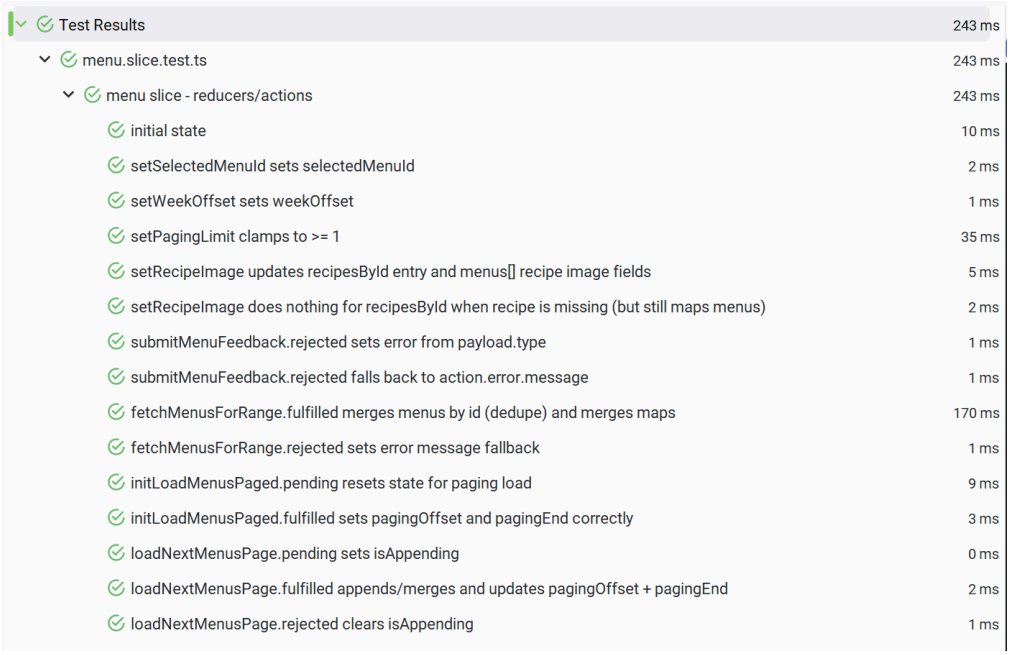

Figure 77: Frontend Menu State Tests

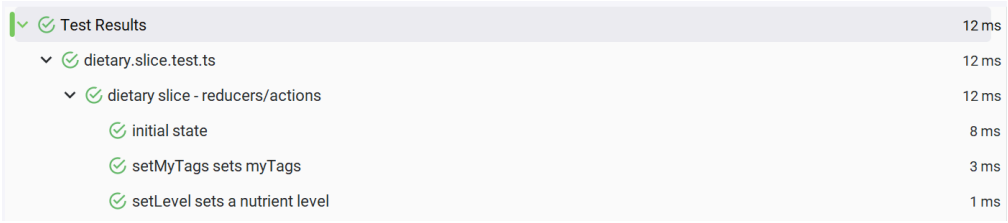

Figure 78: Frontend Dietary State Tests

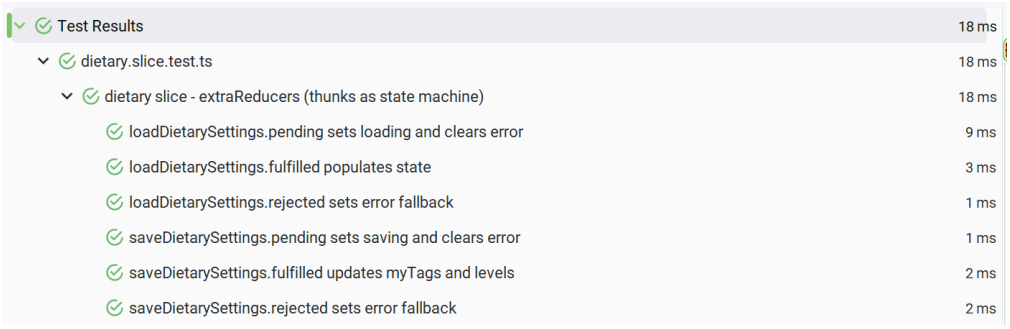

Figure 79: Frontend Dietary State Thunk Tests

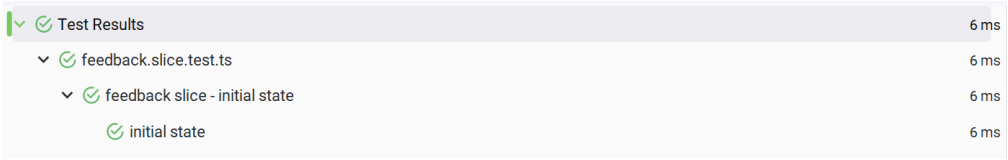

Figure 80: Frontend Feedback State Tests

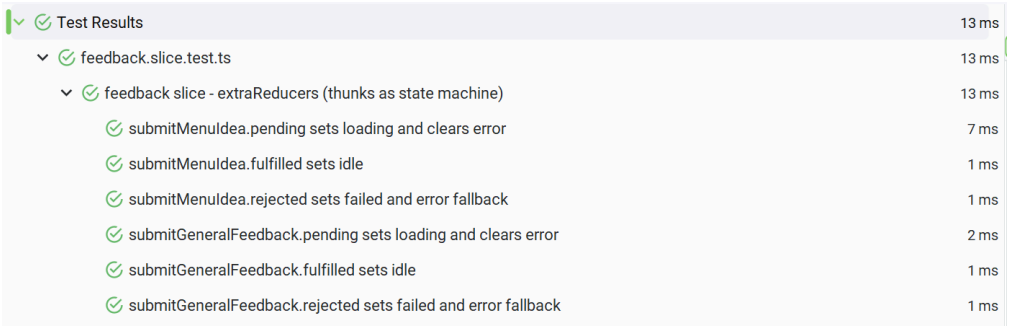

Figure 81: Frontend Feedback State Thunk Tests

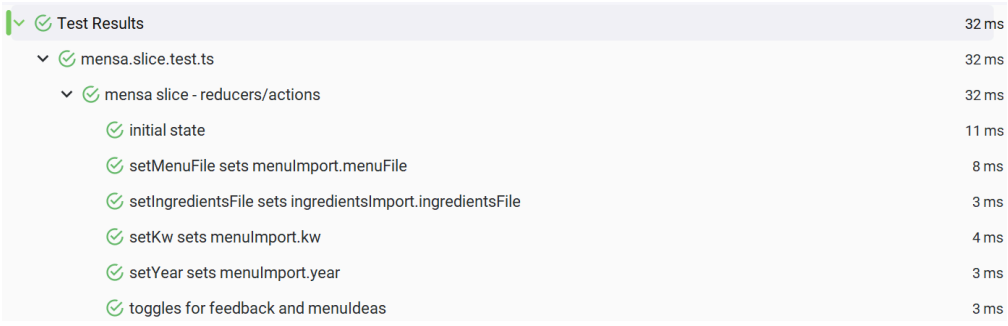

Figure 82: Frontend Canteen State Tests

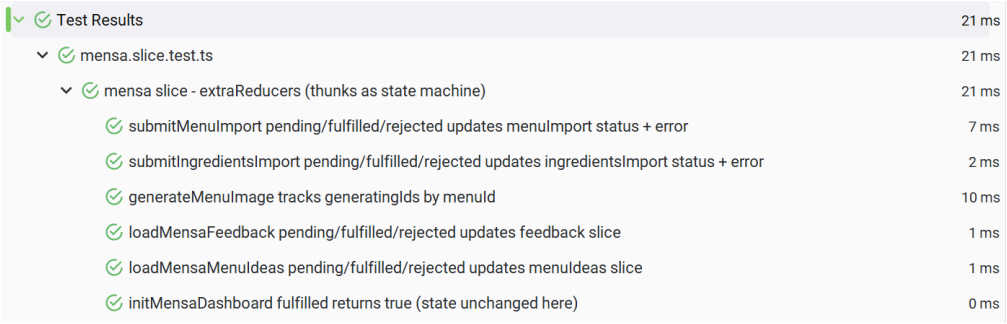

Figure 83: Frontend Canteen State Thunk Tests

6.1.4 Code Reviews

During the span of this SA, our advisor Clemens Meier gave us the opportunity to request two formal code reviews. These reviews helped identify potential weaknesses or code smells and ensure that best practices and appropriate design patterns were applied throughout the implementation. They provided targeted, personalised feedback on architectural, design and implementation decisions.

In the following two chapters, we describe in detail how the feedback was implemented. Small and very specific changes to a certain part of a code fragment are not listed separately. Instead, related comments are grouped by topic and the resulting improvements are explained collectively.

6.1.4.1 Code Review 1 - 14.11.2025

6.1.4.1.1 Backend

Nr	Problem	Improvement
1	Instead of AppDbContext, it is better to inject DbSet.	Implemented with IBiteWiseContext
2	Microsoft recommends await RunAsync(); since dotnet 9	This has been implemented.

Table 38: Code Review 1 Improvements Backend

6.1.4.1.2 Frontend

Nr	Problem	Improvement
1	Nested ternary operations make code harder to read.	Extract these nested ternary operations into separate functions and use if/else statements with returns.
2	Using array index as react keys can cause rendering and performance issues.	Instead of using array indexes, replace them with keys from reliant information like IDs that won't change.
3	React hooks are called conditionally or after early returns, which violates the Rules of Hooks.	Moved all hook calls to the beginning of their component.
4	Unsafe handling of falsy values.	Usage of optional chaining (?.) to better handle null and undefined values.
5	Use nullish coalescing operator (??) instead of logical OR (||), to not include falsy values.	Replace logical ORs with nullish coalescing operator.
6	Don't mix union types and strings (e.g: lane: 'FastLane' | 'Lifestyle' | 'DailyFavourites' | string;)	Prefer using union types and remove the string parts.
7	Redundant react fragments	Removed unnecessary react fragments where not needed.
8	Create components to reduce copying same tailwind styles over and over again.	Created some new components to reduce complexity and improve readability of some bigger components and to reduce copying the same tailwind classes, like: NavArrowButtons, IconLegends, Stars, …
9	Use generated NSwag clients for API calls.	Delete the self created HttpClients to do API calls and use the generated ones by NSwag.
10	Use redux states, instead of handling them in component	Created new slices for menu and user and store states in Redux.
11	Reduce complexity of DailyMenu component	Solved by creating new sub components for the carousel and menu cards and handling the states in redux.

Table 39: Code Review 1 Improvements Frontend

A lot of these problems could easily be fixed with linters. Linters are also part of our quality measures to ensure consistent code, but they were often forgotten to be used before finishing a new feature. Thanks to the code review, this will now be remembered to make sure not as much code smells are left in the code anymore.

6.1.4.2 Code Review 2 - 08.12.2025

6.1.4.2.1 Backend

Nr	Problem	Improvement
1	Clean Architecture	The biggest part of the review revision was the change in architecture. Previously, there was only one .NET project, 'OST.Bitewise.API', with corresponding subfolders. Now, a clean architecture approach has been implemented, with additional projects created for the domain, infrastructure and services. This is described in more detail in the chapter on **architecture** .
2	Single Responsibility Principle (SRP)	The Single Responsibility Principle violations were addressed by splitting previously big components into smaller, focused units. In particular, the original *MenuPlanImportService* and *MenuController* no longer handle Excel reading, parsing, external WebApp calls and database persistence in one place. Instead, responsibilities are now separated into *MenuPlanExcelReader*, *MenuPlanSheetParser*, *MensaIngredientSheetParser*, *RecipeCreationService*, *MenuImageService* and the thin controller layer, each with a clear and narrowly defined purpose.
3	Magic Strings	All configuration entries have been replaced by strongly typed settings classes (e.g. WebAppSettings, ImageGenSettings, MensaImportSettings). Controller and service logic now consume structured objects instead of arbitrary string keys. This removes magic strings and improves discoverability.
4	Type the configuration and inject it via IOptions	Configuration is now provided through typed settings classes injected via IOptions, replacing direct string-based access. This enables central validation at startup, removes configuration logic from services, and makes dependencies explicit and strongly typed.
5	Minimise NSwag on client per controller and inject via interface	Instead of using one large generated WebAppApiClient, the solution now injects the specific NSwag-generated interfaces such as IEditRecipeClient or IUserClient. To enable this, the NSwag configuration was adjusted to generate separate client interfaces per controller. Exceptions have been commented in the code.
6	ExcelReaderFactory does not belong in Services	This has been outsourced.
7	use Linq more often, not too many nested for loops	Linq was used more. This improved the readability of the code in these places, compared to the many nested for loops.
8	no unhandled catches	The unhandled catches were removed or better handled.
9	foreach instead of unnecessary classic for loops	Where possible, classic for loops were replaced by foreach loops.

Table 40: Code Review 2 Backend Improvements Part 1

Nr	Problem	Improvement
10	improve WebAppSession.cs	Machine authentication is no longer triggered from within application services but is executed during application startup in Program.cs. This aligns with proper dependency injection boundaries.
11	API: Return typed DTOs instead of just IActionResult	The API return values have now all been neatly typed.
12	Do not pass an IFormFile to the services	Now stream is being transferred
13	BaseUrl for NSwag Client	BaseURL comes from the settings and is set in the HTTP client and passed to the generated client.
14	Authenticate new CreateIngredient Endpoint in SmartEating API with User Role	*Not implemented/potential for improvement* After trying to implement this, it gave us some errors and we decided due to time constraints, that we do not further investigate in this issue.
15	It has German prompts in the code	*Not implemented/potential for improvement* Dotnet has resx files for this purpose. However, in consultation with the supervisor, it was left as it was. This could be improved in the future.
16	Replace ILoggerFactory and obsolete ISystemClock	*Not implemented/potential for improvement* Was not implemented due to problems with the base constructor.

Table 41: Code Review 2 Backend Improvements Part 2

6.1.4.2.2 Frontend

Nr	Problem	Improvement
1	Separate logic handling from views	In all of the feature components a bigger rework was done to decouple the views from their logic. For each component with state handling, a separate slice was created. The logic handling for each component then was moved into their respective slice, where the computation and service calls are implemented via thunks and actions. Clients are then fetched via extra arguments in thunks. States are changed via reducers and selectors were created to get the state into the needed shape.
2	Remove unused translations	For i18n, some leftover unused translations were still in the project from Smart Eating. Every translation was checked and if not used, it was removed.
3	Get clients via context	API clients are now fetched via “clients-context”, where “useApiClients” is used to only return the needed client.
4	Implement Snackbar outside of component	Moved the Snackbar initialization outside of the component.
5	Use toasts instead of snackbar	Replaced an instance where the snackbar was used to toasts, since the logic then can be handled by ToastConsumer and can be triggered in the toast slice by redux thunks.
6	Split big components into smaller sub components	Bigger components were split up to use smaller sub components. Like DailyMenuCarousel and WeekDay.
7	Create components for reused elements	Some elements that were reused multiple times were moved into own components, like buttons.
8	Use Luxon DateTime instead of JS Dates	Replaced all instances of built-in JS Date occurrences with Luxon to simplify usage and make code more robust.

Table 42: Code Review 2 Improvements Frontend

6.2 User Evaluation

6.2.1 Summative Usability and Acceptance Tests

The summative usability and acceptance tests were conducted in the second to last project week with the goal of validating the implemented MVP from an end-user perspective. In contrast to the earlier usability tests conducted on a Figma prototype, these tests were performed on the fully implemented web application.

The focus was not on UI exploration or concept validation, but on confirming that the central user flows work as intended, are understandable without guidance, and are generally accepted by the target audience. The results serve as a qualitative validation of the MVP and as a foundation for future development beyond the scope of this semester project.

6.2.1.1 Test Participants

The test was conducted with three OST students, all of whom regularly use the canteen. To avoid bias and learning effects, none of the participants had taken part in the earlier Figma usability tests. The participant requirements were otherwise aligned with the previously defined screener criteria used during the prototype phase.

Due to time constraints and the current maturity level of the canteen dashboard, no tests were conducted with canteen employees. The focus of this summative test was intentionally placed on the student-facing part of the application, which represents the primary user group of the MVP.

6.2.1.2 Test Setup and Procedure

The test setting closely followed the setup used during the earlier usability tests .

Each session was moderated by one test leader, while the second team member observed user behaviour and took notes. During the tasks, no assistance or guidance was given, and participants were encouraged to verbalize their thoughts if possible.

Each session lasted approximately 20 minutes and followed this structure:
- Short introduction and explanation of the test procedure
- Execution of predefined test scenarios
- Post-test interview including Likert-scale questions

Written observation protocols were created for each session. The detailed observation notes are provided in Appendix F .

6.2.1.3 Test Scenarios

The test scenarios were based on the same core scenarios used in the earlier usability tests (Appendix C), but slightly adapted to reflect the actually implemented functionality. Use cases that were not implemented in the MVP were removed from the test cases accordingly.

The goal was to validate whether users could complete these tasks intuitively and without prior explanation.

6.2.1.4 Interviews and Questionnaires

In contrast to the prototype usability tests, no pre-interview was conducted for this summative test, as the participants already matched the previously defined target group criteria.

After completing the tasks, a structured post-interview was conducted with each participant. Besides general questions, the interview included specific follow-up questions targeting the implemented MVP features.

In addition, the post-interview was extended with a short Likert-scale questionnaire (5-point scale from “Yes” to “No”) to capture a more standardized assessment of usability and acceptance (Likert, 1932) . The items covered perceived user-friendliness, navigation clarity, visual appeal, ease of use, time efficiency, as well as intention to use the application in everyday OST context and willingness to recommend it. The detailed questions and individual responses are documented in Appendix G .

6.2.2 Overall Test Results

Overall, the summative usability and acceptance tests showed that the MVP meets its primary goals from a student perspective. All participants were able to complete the core tasks without external help and expressed a generally positive impression of the application.

Minor usability issues and improvement ideas were identified, but none of them prevented task completion or led to significant confusion. These findings confirm that the implemented solution is suitable as a functional MVP and provide clear input for future iterations.

6.2.2.1 Identified Issues and Improvement Potential

Based on the observations, interviews, and questionnaire responses, several minor usability issues and improvement opportunities were identified. These findings are discussed in detail in the following section, where individual aspects of the application are analysed more closely.

6.2.2.1.1 General

During a running of a test with one test person, the application seemed to be a bit buggy, meaning that menus were randomly not found anymore and the menu plan pages were shown as empty. We identified the problem as slow endpoints from the Smart Eating API, that some calls would load for a long time and occasionally time out. This only happened with the first test person, for the others the application was running fine and bug-free.

This issue is something that would need more investigating to find out the exact cause.

6.2.2.1.2 Daily Menu View & Menu Feedback

One test person tried to click on the vegan icon for the dietary preferences, to filter for only vegan menus. Filtering and sorting menus was also a discussion point to add as a use case, but got quickly removed due to lack of options that a student has in the canteen with only three menus.

Nevertheless, such a feature could be implemented into BiteWise in future iterations.

An issue that was pointed out by two test persons was, that they expected more from the “See details” button and not only to see the nutritions. They thought it would show more information to the menu like a description.

This is something that will automatically be fixed in future iterations, when the use case for sustainability information will be implemented, which would show more info next to the nutritions. Also, showing a description for the menu can be implemented in future improvements.

For the menu feedback, test persons were confused about the “Saved in meal plan” message after sending the feedback. This feature does not have a use in the current version of BiteWise, since the meal plan from Smart Eating cannot be accessed in the current state.

To avoid confusion, some info text could be added to explain what the meal plan is and what it would do.

6.2.2.1.3 Nutrition Data

A big pain point for the nutrition data for the menus were, that the displayed information was not accurate.

This is a known issue as described in earlier chapters, which has to be investigated further on the Smart Eating side.

6.2.2.1.4 Personal Diet Matching

All of the test persons had issues to find the correct menu point in the user navigation. The two options presented there, Menu preferences and User settings, were off putting and most of the test persons said, they expected that the configuration would be in the menu point that is named settings.

This could be fixed by either renaming the menu points or merge the two options together in one settings page.

6.2.2.1.5 Feedback and Menu ideas

Two test persons did not find the feedback page immediately, they knew about where to give feedback for an individual menu, but had to search a bit for the general feedback page. Also, one test person said they did not expect the menu idea form to be in the feedback page as well.

As a fix, the feedback page could be moved elsewhere to make it more present and separate the two forms for canteen feedback and menu ideas.

6.2.2.2 Limitations and Outlook

The results of the summative usability and acceptance tests must be interpreted in light of the project constraints. The small number of participants and the focus on students limit the generalizability of the findings.

Nevertheless, the test results provide valuable qualitative insights and confirm the overall direction of the solution. Due to the limited timeframe of this semester project, the identified improvements will not be implemented, but they form a solid basis for future development of the BiteWise platform.

7 Future Development

This chapter summarises the necessary improvements and enhancement opportunities identified during implementation as well as those revealed by the user evaluation It consolidates both technical and user-centered findings into a structured overview and serves as a feature request list for future developers. The listed items capture limitations of the current MVP and outline concrete directions for further development beyond the scope of this semester project.

7.1 Improvements to existing features

Name	Related Use Cases	Description
Recommendation transparency	UC-U04	Explain why a menu is recommended or greyed out (e.g. matched tags, nutrient score breakdown). Add a short explanation per menu.
Nutrition data quality	UC-S02 / UC-U05	The calculated nutrition values and tags are often inaccurate and sometimes implausible. During implementation, the usefulness of the current Smart Eating Analyze workflow for canteen menus was questioned, as it is optimised for classic recipes and cannot reliably infer portions and ingredient composition. As a future improvement, the Smart Eating approach could be refined for the canteen context. Alternatively, it may be more practical to introduce a dedicated nutrition endpoint tailored to canteen data, potentially using also an AI-based approach that accepts structured ingredient inputs and produces more consistent, explainable results, without relying on the USDA-based calculation pipeline.
Image quality and control	UC-S03	The visual style and realism of generated menu images should be evaluated more carefully to understand which types of images appeal to students and encourage them to visit the canteen, without creating misleading expectations about the actual dishes. In addition, it should be clarified together with the canteen whether AI-generated images are desirable at all from their perspective. An alternative would be to allow the canteen to upload their own images, but this is likely impractical in daily operations and therefore only a theoretical option.
Student menu filtering	UC-U01	For students, simple and intuitive filtering could be added by allowing interaction with existing UI elements, for example clicking on dietary icons to filter menus by tag. This would enable quick exploration without introducing complex filter controls.
Menu details depth	UC-U05	Users expected the “See details” action to provide more comprehensive information than nutrition values alone. Future improvements could include a short menu description and additional context such as sustainability information.
Navigation clarity for settings	UC-U04 / UC-E02	The separation between “Menu preferences” and “User settings” was unclear to users. A better approach would be to merge these into a single settings page and remove unnecessary or unused options (e.g. weight), resulting in a simpler and more intuitive configuration flow.

Table 43: Potential improvements for existing features Part 1

Name	Related Use Cases	Description
Performance and reliability	-	During development and user tests, some calls to the BiteWise API sometimes showed noticeable performance issues. While acceptable for the scope of this project, response times and stability should be improved before a production release, for example through better backend optimisation, more consistent error handling, and clearer frontend feedback in case of delays or failures.
Theme handling and accessibility	-	Dark mode and light mode are implemented, but currently always follow the browser or system setting. The theme toggle in the UI does not override this behaviour and is therefore functionally ineffective. This should be improved so that users can explicitly select and persist their preferred theme, while still ensuring sufficient contrast and readability in both modes.

Table 44: Potential improvements for existing features Part 2

7.2 Extensions / New Features

Name	Related Use Cases	Description
Canteen Menu Filter and Search	UC-M03	Implement UC-M03
Canteen Dashboard CRUD Menus	UC-M04	Implement UC-M04
Sustainability Insights	UC-U08	Implement UC-U08
Feedback UX clarity	UC-U03	The “Saved in meal plan” message caused confusion, as the Smart Eating meal plan is not visible or usable within BiteWise. This feature would either require better integration and visibility of the Smart Eating meal plan or a clearer explanation within BiteWise. Otherwise, the message should be removed to avoid misleading users.
Feedback analytics dashboard	UC-M02	Add trend charts over time (e.g. ratings per week), drill-down by lane/menu, and export (CSV).
Automated imports with audit trail	UC-S01	Reintroduce automation (scheduler/cron) where feasible: detect new source files, run imports, and provide an audit log with diffs, validation issues, and rollback options.
Faster imports	UC-S01	Optimise the menu import to shorten waiting times for the mensa employee.
SSE events	-	Use Server-Sent Events (SSE) to stream import and processing status updates in real time to the frontend, improving transparency and user feedback during long-running operations.

Table 45: Potential extensions and future features

8 Project Plan

8.1 Project Management

We use an agile method with a lot of similarities to Scrum. The notable differences are:

- Sprint length of 1 week, starting on Mondays and ending on Sundays.
- No dailies, since we won't be working on the project every day. Instead we update each other on the progress of each task on a frequent basis.
- No Sprint Plannings and Retrospectives. Instead, we discuss the tasks for the next sprint on Monday mornings, after the previous sprint has completed.
- Towards the end of each sprint on Fridays, we have a weekly review with the supervisors to discuss the current state of the project.

8.2 Long Term Planning

8.2.1 Epics

The initial rough plan was created in Jira using epics. It shows the different project phases and serves as a guide. It can be refined later on.

In Jira, epics are also used to categorise tickets.

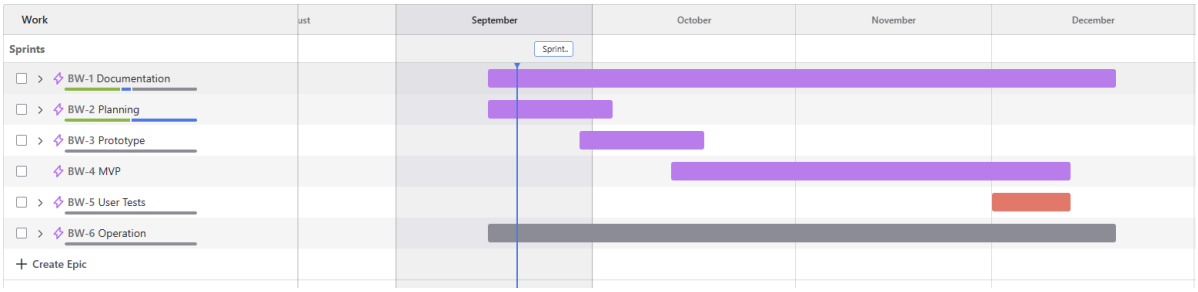

Figure 84: Long Term Planning in Jira

The epics are described and defined in more detail below. For each epic, the originally planned time period is presented first, followed by the actual time period as realised during the project.

Epic Name	Planned Period	Actual Period	Description
Documentation	15.09.2025 - 19.12.2025	15.09.2025 - 19.12.2025	Includes the writing of the entire documentation up to submission.
Planning	15.09.2025 - 03.10.2025	15.09.2025 - 05.10.2025	Includes basic planning, interviews and surveys, and requirements analysis.
Prototype	29.09.2025 - 17.10.2025	29.09.2025 - 26.10.2025	Includes designing, completing a prototype and doing user tests.
MVP	13.10.2025 - 12.12.2025	20.10.2025 - 15.12.2025	Includes the complete development of our end product, the so-called minimum viable product.
User Tests	01.12.2025 - 12.12.2025	08.12.2025 - 12.12.2025	Includes the final user tests, possibly with a few last improvements.
Operation	15.09.2025 - 19.12.2025	15.09.2025 - 19.12.2025	Includes recurring activities throughout the project, such as meetings, minute-taking or reviews.

Table 46: Epics

8.2.2 Milestones

The epics are usually completed with a milestone. These can be seen in the following table.

Planned Date	Actual Date	Milestone
19.12.2025	19.12.2025	Final submission
03.10.2025	05.10.2025	End of planning phase
17.10.2025	26.10.2025	Completion of first prototype
12.12.2025	15.12.2025	Completion of MVP

Table 47: Milestones

8.3 Short Term Planning

8.3.1 Planning (Target State)

For short-term planning, only the high-priority use cases were scheduled into concrete implementation sprints. The use cases were not only classified by priority but also ordered within their priority level to define a clear implementation sequence. This decision was made intentionally, as the actual development pace was difficult to estimate at the beginning of the project. By focusing on essential features, the planning remained realistic and flexible. Mid- and low-priority use cases were left unscheduled and were implemented dynamically, depending on the remaining time and overall progress.

8.3.2 Actual Outcome

In practice, the initially planned implementation pace could not be met, especially during the early phases of the project. This was mainly due to the required onboarding into new technologies and the existing Smart Eating system. In particular, the nutrition data generation and import proved to be more complex than expected, as several approaches had to be revised or discarded. Additionally, issues in the existing Smart Eating system were identified during this phase, which required fixes and, in some cases, the creation of new backend endpoints.

Towards the later stages of the project, development efficiency improved significantly. As a result, not only were all planned “Must Have” features completed, but several “Could Have” features were also successfully implemented.

A detailed overview of the planned versus implemented use cases can be found in the Use Cases chapter .

8.4 Time Tracking Report

Time tracking was done in Jira by logging the time directly on the worked on ticket. In the following sections the total work log is visualized in different graphics.

8.4.1 Time per person

This graph depicts the total time worked on the project and how the time is distributed on each team member.

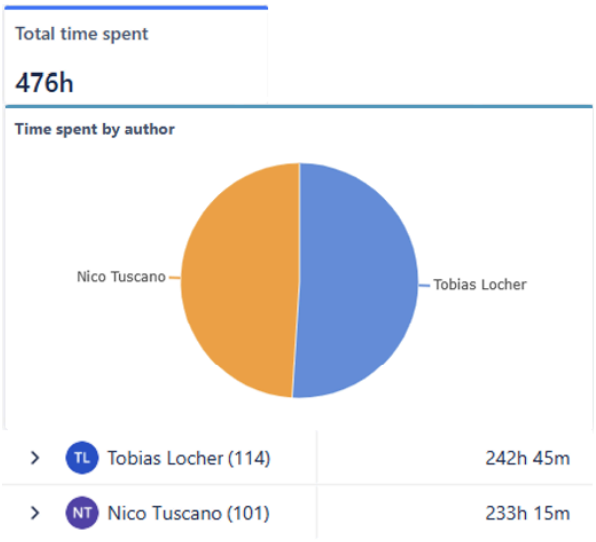

Figure 85: Time per Person Graph

8.4.2 Time per Epic

This graph shows how much time on each epic has been worked on.

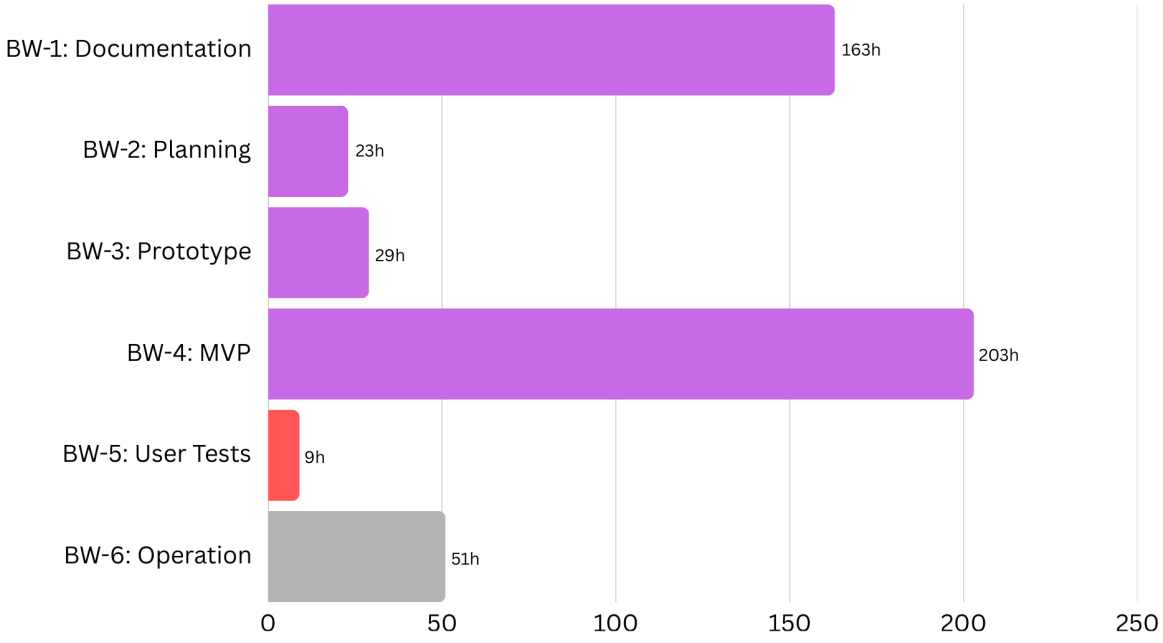

Figure 86: Time per Epic Graph

8.5 Risk Management

This chapter describes potential risks in the project, assesses them using a risk matrix and presents measures for risk mitigation. This allows problems to be identified early on and dealt with effectively.

It is a mixture of general project risks but also very specific risks associated with our work.

The positioning of risks in the risk matrix is based on a careful assessment of the probability and severity of each risk. This assessment takes into account both the potential impacts described and the risk mitigation measures already established. As a result, the matrix reflects not only the pure risk potential, but also the expected effectiveness of the precautions taken.

Risk analysis is a continuous process and is not carried out on a one-off basis. Risks are regularly reviewed, since new ones may arise, existing ones may change and measures are adapted as needed.

8.5.1 Risks

No.	Risk	Description & Possible Impact	Measures for Risk Mitigation
		Technical	
RT01	Difficulties in development and integration	The existing Smart Eating project is already very advanced. There is a risk that the existing structure is too complicated for us to develop our subplatform. This could cost us valuable development time.	We hold weekly meetings with the supervisors. It is important to ask questions early on, both during meetings and outside of them, that might hinder our progress.
RT02	Tech stack complicates implementation	We decided early on to stick to the existing tech stack as much as possible. In addition to the many advantages, this also carries the risk that we will work more slowly due to a lack of routine or even realise late in the project that the tech stack is not sufficient for our solution.	Even though we still have little practical experience with the technologies, we have already learned a lot about them in courses. The existing tech stack has been critically reviewed and is well suited for web apps like ours.

Table 48: Risk List Technical

No.	Risk	Description & Possible Impact	Measures for Risk Mitigation
		Organizational	
RO01	Organisational problems	It is possible that we may get stuck due to a lack of knowledge or other problems and lose valuable time.	If we have any questions that prevent us from continuing to work efficiently, we try to ask the relevant people early on. We also hold a weekly meeting with our supervisors, where any open questions can be clarified. This also serves to confirm that we are still on the right track. In addition, we also talk every week within the team so that we both always know what needs to be done.

Table 49: Risk List Organizational

No.	Risk	Description & Possible Impact	Measures for Risk Mitigation
		Human	
RH01	Unexpected long absence of a team member	It is always possible that a team member may be absent for a longer period of time for various reasons.	It is difficult to minimise this risk. The only thing you can do is communicate as early as possible when a personal absence is imminent. We will also prioritise our requirements so that the less important features can be removed in the event of an absence.

Table 50: Risk List Human

No.	Risk	Description & Possible Impact	Measures for Risk Mitigation
		External	
RE01	Collaboration with other students	Since we also need to identify the needs of students and incorporate them into our work, we are conducting a survey. There is a risk that too few students will participate or that the results will be of little use to us. We will also need students to test our prototype. There is a risk that we will not find enough of them.	We included the opinion of the canteen management to ensure the quality of the survey and critically examined what information we needed. We also asked the supervisors for feedback before sending out the survey. In order to reach as many people as possible, we asked for permission early on to distribute the survey through various channels. We are also trying to find small goodies for the students who will later carry out tests with us.
RE02	Collaboration with the canteen	Since the canteen ultimately has to be able to use our product and evaluate the results, their opinion is also very important. We therefore need to maintain close communication, require some information, and may also need test subjects. These are all critical points that could cause our project to stall.	Already in the first week, we had a productive meeting with Florian Wunderlich, the canteen manager. This makes us optimistic that future cooperation will also be positive and straightforward. Nevertheless, we must continually critically examine what information we need and ask for it early on so as not to waste any time.
RE03	Insufficient menu data	The canteen menus must not only be displayed, but also evaluated. For example, it must be possible to match them to one's own diet plan. This requires additional data such as nutritional values. One risk is that these cannot be specified with sufficient accuracy. It could also be that the canteen does not provide enough information to specify them accurately.	After our initial discussion with the canteen, we now have a better understanding of how the canteen obtains information about ingredients. In this project, we must remain aware of this risk and request the necessary information at an early stage.

Table 51: Risk List External

Legend

Color	Meaning
	Very high
	High
	Medium
	Low
	Eliminated

Table 52: Risks Legend

8.5.2 Initial Risk Matrix - Sprint 1

Probability	Severity			
	Negligible	Marginal	Critical	Catastrophic
Certain				
Likely				
Possible			RT01, RE01, RE02	
Unlikely			RT02, RO01	RE03
Rare				RH01
None				

Table 53: Risk Matrix - Sprint 1

8.5.3 Risk Matrix - Sprint 3

The risk of collaborating with other students (**RE01**) has decreased since we successfully completed the survey. However, it remains, as we will still be doing usability tests.

Due to the rather limited information provided by the canteen regarding nutritional values, it is difficult to calculate the nutritional values per menu. However, in our weekly review meeting, we decided that we could use the existing AI for the recipes. The risk that the nutritional values based on our information will be too inaccurate (**RE03**) remains, but the consequences would no longer be catastrophic because we can at least complete our project with values.

A new risk has also emerged:

No.	Risk	Description & Possible Impact	Measures for Risk Mitigation
Technical			
RT03	API authentication	Although our platform is a separate sub-platform, we still need to access many existing API endpoints, such as authentication or recipes. For certain endpoints, you have to be logged in and therefore authenticate yourself. As we are not yet technically advanced enough, we cannot yet estimate how complex this will be. We could lose time on this or not be able to do it.	We have already made some contacts who are involved in developing Smart Eating. We will contact these people if we have any questions. If we get stuck, we will get in touch early on so that as little time as possible is lost.

Table 54: New Risk Sprint 3

Probability	Severity			
	Negligible	Marginal	Critical	Catastrophic
Certain				
Likely				
Possible			RT01, RE02	
Unlikely			RT02, RO01, **RE01**, **RE03**, **RT03**	
Rare				RH01
None				

Table 55: Risk Matrix - Sprint 3

8.5.4 Risk Matrix - Sprint 6

The risk of collaborating with other students (**RE01**) has decreased further, as we have successfully completed the prototype usability tests. This means that only end user tests remain at the end of the project.

In the meantime, we have received more data on the menus and nutritional values from the canteen. As a result, the probabilities of working with the canteen (**RE02**) and insufficient menu data (**RE03**) have decreased further.

Probability	Severity			
	Negligible	Marginal	Critical	Catastrophic
Certain				
Likely				
Possible			RT01	
Unlikely			RT02, RO01, **RE02**, RT03	
Rare		**RE01**	**RE03**	RH01
None				

Table 56: Risk Matrix - Sprint 6

9 Conclusion

9.1 Reflection

The goal of this project was to implement a functional MVP of the Smart Eating sub-platform BiteWise that brings the OST canteen and students closer together.

To achieve this, a comprehensive requirements analysis was conducted, including a survey and usability tests with a large number of students. This process allowed many relevant requirements to be identified early on.

The complexity of the existing Smart Eating platform required careful coordination and, at times, difficult decisions regarding scope and prioritization. Nevertheless, all defined must-have use cases were successfully implemented, along with several additional use cases, and most non-functional requirements were met.

Students can now view canteen menus in a modern UI, including nutritional information. Individual menus can be rated, and users can submit general feedback as well as menu ideas. For the canteen, the dashboard provides an efficient way to import menus and review feedback. The integration of image generation further enhances the overall presentation of the menus.

The project also proved to be a valuable learning experience. Working with real user feedback, integrating into an existing system, and continuously improving the code through code reviews from our supervisor, helped deepen both technical and methodological skills. Software tests further contributed to ensuring the correctness and reliability of the implemented logic.

The final user tests showed that the web application was generally well received and confirmed that the chosen direction was appropriate. Additional improvement points and detailed feature requests were identified, providing a clear foundation for future development.

Based on the evaluations and tests conducted with students, the project can be considered a success. It delivers a solid technical and conceptual foundation that can be further extended and, with future improvements, transitioned into live operation for the canteen.

9.2 Outlook

Although the web application already provides a solid foundation, several improvements are still required before it can be put into live operation for the canteen. The identified feature requests have been carefully documented in the Future Development chapter and serve as a clear roadmap for further development.

We believe that the application could be brought to a production-ready state through a follow-up semester or bachelor thesis. By implementing and refining the described features, and potentially adding further enhancements, the platform could be prepared for real-world use. In this context, a more detailed performance analysis would also be necessary, particularly regarding the SmartEating API, to ensure that the system meets performance and scalability requirements under live operating conditions. At the same time, the accuracy and reliability of the nutritional value calculation represents one of the most critical aspects for a productive deployment. Since nutritional information is a core feature of the platform, further validation, refinement of the calculation logic, and closer alignment with the canteen's ingredient data would be essential to ensure trustworthy results.

An additional key factor for a successful rollout is the acceptance by the canteen staff. The long-term success of the platform depends on whether the solution integrates smoothly into existing workflows and provides clear value for daily operations. Continued collaboration with the canteen, early feedback, and possible training or onboarding measures would therefore be essential to ensure sustainable adoption.

We look forward to following the future development of BiteWise and hope to see the platform deployed in daily use at the canteen in the near future.

Acknowledgments

Clemens Meier is thanked for his competent supervision, constructive input during the weekly meetings, and thorough code reviews, which significantly improved the code quality and provided valuable learning opportunities.

Prof. Dr. Mitra Purandare is thanked for the competent supervision and constructive input.

Jeriel Frei, **Simeon Rhyner** and **Michelle Tuscano** are thanked for the careful proofreading of the documentation and the constructive feedback.

Florian Wunderlich is thanked for the constructive input from the mensa and for providing goodies for the test participants.

All students who participated in our survey or in the usability tests of our app are gratefully acknowledged.

Bibliography

Brown, S. (2025,). *C4 Model*. https://c4model.com/

Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development* (3rd ed.). Prentice Hall PTR.

Likert, R. (1932). A Technique for the Measurement of Attitudes. *Archives of Psychology*, *22*(140).

Martin, R. C. (2017). *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Prentice Hall PressOne Lake Street Upper Saddle River, NJUnited States.

OpenAI. (2025a, December). *OpenAI Image Generation Documentation*. https://platform.openai.com/docs/guides/image-generation

OpenAI. (2025c, December). *OpenAI Pricing*. https://openai.com/api/pricing/

OpenAI. (2025b, December). *OpenAI Tokenizer*. https://platform.openai.com/tokenizer

OST. (2025, September). *Smart Eating Project*. https://www.ost.ch/de/projekt/smart-eating-new6866421eaaeff444850398

List of Figures

List of Tables

Glossary

AI	Artificial Intelligence
API	Application programming interface
Boilerplate	Standardised code or configuration that provides a basic starting structure for a project and is reused with minimal changes.
C4	A software architecture model that describes a system using four hierarchical diagram levels: Context, Container, Component, and Code.
DB	Database
DTO	Database transfer object
ECTS	European Credit Transfer and Accumulation System
EF Core	Entity Framework Core
ERD	Entity relationship diagram
Figma	Web-based design and prototyping tool for creating, collaborating on, and testing user interfaces in real time.
FR	Functional Requirements
HTTP	Hypertext transfer protocol
IFS	Institut für Software (En: Institute for Software)
Lane	Term used by canteen to categorize their menus.
Mensa	German word for canteen, interchangeable use
Mobile-first	An approach where a user interface is designed for mobile devices first and then progressively adapted for larger screens.
MVP	Minimum Viable Product
NFR	Non-Functional Requirements
N/A	Not applicable
OST	Eastern Switzerland University of Applied Sciences, in German: Ostschweizer Fachhochschule

SA	Semester project, in German: Semesterarbeit/Studienarbeit
Thunks	A programming pattern where functions are used to delay or control the execution of logic, commonly used in state management to handle asynchronous operations.
Toast	A short, non-intrusive notification used to provide quick feedback to users about an action or event in the application.
UC	Use Case
UI	User Interface
UML	Unified modelling language
USDA	U.S. Department of Agriculture
UX	User Experience

List of Tools

Purpose	Tools
Data Analysis and Visualization	ChatGPT, Excel, Microsoft Forms
Idea Generation	ChatGPT, Figma, Figma AI
Translation	ChatGPT, DeepL
Prototyping	Figma, Figma AI
Coding	ChatGPT, JetBrains Rider, JetBrains IntelliJ
Text Creation, Editing, and Proofreading	ChatGPT, DeepL
Project Report	Typst, Visual Studio Code
Collaboration and Project Management	Jira, Teams, Outlook, GitLab
DevOps	Docker, GitLab

Appendix

A Personal Reports

A.1 Tobias Locher

From a technical perspective, I was mainly responsible for developing the .NET backend, while Nico implemented most of the frontend. Although I had prior experience building backends, this was my first project using .NET. I was already familiar with the technology from an OST module, but the project still required a significant learning effort, which is exactly the purpose of such academic projects.

The code reviews by Clemens Meier were particularly valuable. They were detailed, well explained during meetings, and provided clear guidance. Even though addressing them took considerable time, it was important for me to implement most of the feedback properly. Especially in the area of Clean Architecture, I was able to deepen my understanding significantly.

Integrating our work into the existing Smart Eating system was sometimes challenging. Some required endpoints were missing and had to be added, and parts of the existing recipe and nutrition AI logic did not always fit perfectly with our menu-focused application. This required additional effort, but it also provided valuable experience in understanding and extending a large, unfamiliar codebase.

In addition to the backend, I also implemented the canteen dashboard. This allowed me to gain some hands-on frontend experience and further broaden my skill set.

We are very satisfied with the overall scope of the project. We were able to implement a solid set of features. It was particularly interesting to go through the entire process, from target group analysis and UI design to prototype testing and final user tests, which goes beyond a purely programming-focused project.

Team collaboration worked well and communication with our supervisor was effective due to the weekly meetings. This helped clarify questions early and ensured we stayed on the right track. One weakness was the middle phase of the project, where we focused heavily on implementation and somewhat neglected documentation. As a result, a large amount of documentation had to be completed at the end.

For my bachelor thesis, I take away the lesson that documenting features continuously during development is highly beneficial. Despite the late effort, creating clean and structured documentation was important to me, and I was able to contribute my part to a final result that we could confidently submit.

A.2 Nico Tuscano

I am very pleased with how the whole SA and the BiteWise application have turned out in the end. We were able to implement all the must-have features that we defined in the beginning based on the requirements and even had time to implement some extra features like the canteen dashboard and image generation.

For me personally, I was able to learn a lot of new things, primarily in frontend development, since I mainly worked on the new React frontend for the BiteWise application while Tobias did most of the backend work. As someone who wants to specialize in frontend development, this was a perfect opportunity for me to broaden my skill set in different frameworks, learn more about frontend architecture and design patterns, and experience the whole process from requirements analysis to a finished MVP myself.

For this, the code reviews by Clemens Meier were perfect to learn more about best practices in frontend development and how specific patterns, such as Redux, work correctly.

The work dynamic in the team was good. Tobias and I had already worked on several projects together during our time at OST, so we had no starting problems and could begin working immediately without any issues. Our division of tasks aligned perfectly with our personal interests, so each of us could work efficiently and with good motivation on each task. This also helped with the learning effect of the SA, which was very high for me.

Regarding time management, we started strongly with the requirements analysis and the prototyping, but after that, during the implementation phase, we started to slow down. This was especially noticeable in the documentation part, which was somewhat left aside while we were working on the code.

My takeaway from this for future projects, and especially for the bachelor thesis, is that documentation is also an integral part of the project process and should be done at every stage. Even while programming, when it might sometimes feel like a burden instead of something useful, in the end everyone benefits from documentation being done in a timely manner.

All in all, I enjoyed working on this project for our SA, even when some challenges occurred during it, which I think is normal for any project. In the end, I can say without a doubt that I am proud of the quality of the final project and the documentation as well, and I am pleased with what we submitted.

B Figma Usability Tests Interview Questions

B.1 Pre-interview

- Was studierst du?

- Wie oft gehst du in die Mensa?
- Weisst du, wo man der Mensa Feedback geben kann?
- Hast du der Mensa bereits Feedback gegeben? Über welchen Kanal?
- Wo schaust du das tägliche Mensa Menu aktuell nach?
  - Auf dem Handy oder Notebook?
- Kannst du dich einfach entscheiden?

- Achtest du speziell auf etwas in deiner Ernährung? Auf was?
- Trackst du aktuell deine Ernährung?
  - Mit welchem Tool?
  - Auf welchem Gerät?
- Welche Werte sind dir dabei speziell wichtig?

B.2 Post-interview

- Wie ist dein allgemeiner Eindruck?
- Was hat Dir gut gefallen?
- Was war nicht so gut?
- Hast du etwas vermisst?

- Ist dir die Feedback-Funktion direkt aufgefallen?
- Konntest du schnell und einfach Feedback geben?
- Hast du eine Feedback-Art vermisst?

- War dir direkt klar, welches Menu dir empfohlen wird?
  - Würdest du diese Funktion wünschen?
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
  - Fehlen dir Konfigurationsmöglichkeiten?
- Wie findest du die Wahl und Darstellung der Icons?
- War dir klar, wo die Nährwerte zu finden sind?
  - Wie fandest du die Darstellung der Nährwerte?
  - Fehlen Werte, welche dir wichtig wären?
- War es einfach zu sehen, welche Menus am nachhaltigsten sind?
  - Hilft dir diese Info?
  - Wie findest du die Darstellung?
- Konntest du das heutige Menu schnell finden?
- Wie fandest du die Eingabeformulare beim Feedback?
- Welche Bilder würden dir am besten gefallen?
  - Reale Bilder von der Mensa (aber eben schwer umsetzbar)
  - AI Bilder Clip-Art, Comic mässig
  - AI Bilder Clip-Art, Comic mässig aber realistischer
  - AI möglichst realistisch

C Figma Usability Test Scenarios

C.1 Test Scenario A (Web)

Du hast von deinem Studienkollegen Tobias auf Teams einen Link zu einer neuen Webseite namens «BiteWise» erhalten. Er sagt du sollst diese Seite unbedingt mal ausprobieren, da man viele Infos zur Mensa dort erhält. Es ist ohnehin bald Mittag und du wolltest mal schauen, was es heute in der Mensa gibt, wie die Preise sind und was die restlichen Studierenden an der OST zu den Menüs so sagen.

C.2 Test Scenario A (Mobile)

Du hast von deinem Studienkollegen Tobias von einer neuen Webseite namens «BiteWise» gehört. Er sagt du sollst diese Seite unbedingt mal ausprobieren, da man viele Infos zur Mensa dort erhält. Es ist ohnehin bald Mittag und du wolltest mal schauen, was es heute in der Mensa gibt, wie die Preise sind und was die restlichen Studierenden an der OST zu den Menüs so sagen.

C.3 Test Scenario B

Da du dich Vegan ernährst und Laktoseintolerant bist, möchtest du wissen, welches der Menüs für dich überhaupt in Frage stehen.

C.4 Test Scenario C

Da du sehr auf deine Ernährung und auch auf die Umwelt achtest, möchtest du herausfinden was in den jeweiligen Menüs von heute steckt und wie sehr sie die Umwelt belasten (oder auch nicht).

C.5 Test Scenario D

Mittlerweile ist es nach der Mittagspause und du hast deine Quinoa Bowl in der Mensa sehr genossen. Nach deinem Geschmack war das Gemüse aber ein bisschen zu wenig gesalzen. Du entscheidest dich dafür dies der Mensa mitzuteilen, damit sie dies verbessern können.

C.6 Test Scenario E

In der nächsten Lektion unterhältst du dich mit einer Mitstudentin über die Mensa. Sie liebt Fisch und behauptet, dass es in der Mensa schon Ewigkeiten kein Fisch mehr gab. Du erinnerst dich jedoch daran, dass dies noch gar nicht so lange her ist. Überprüfe dies in BiteWise und zeige ihr, wann es zuletzt Fisch in der Mensa gab.

C.7 Test Scenario F

Nach ein paar Tagen Nutzung der Webseite, hast du dir mittlerweile auch ein Konto erstellt (als Sarah Müller). Du hast gehört, dass du damit einiges an deinen persönlichen Esspräferenzen anpassen kannst. Dies möchtest du jetzt tun. Du ernährst dich Vegan und bist Laktoseintolerant. Ebenso möchtest du darauf achten, dass du möglichst viele Kalorien zu dir nimmst. Siehst du nun besser welche Gerichte zu dir passen und welche nicht?

C.8 Test Scenario G

Dir ist zu Ohren gekommen, dass die Mensa wenig Rückmeldungen von Studierenden bekommt, um sich verbessern zu können. Du möchtest dies ändern und der Mensa nun mitteilen, wie du über sie denkst. Aber lieber ohne, dass sie wissen von wem.

C.9 Test Scenario H

Deine Mama macht den besten Flammkuchen der Welt. Sie macht immer Blattgold und Zucchini drauf. Dieses wundersame Rezept würdest du auch gerne in der Mensa essen. Teile ihnen dies mit.

C.10 Test Scenario I

Ein langer anstrengender Tag im Studium geht zu Ende. Am nächsten Montag geht es aber schon wieder weiter für dich. Gerne möchtest du schon jetzt wissen, was es dann in der Mensa für dich gibt.

D Figma Usability Test Protocols

D.1 Test Scenario A

Testszenario A								
Schritt	Erwartung	P01	P02	P03	P04	P05		Bemerkungen
1	Öffnet die BiteWise Webseite / Lesezeichen	Ja	Ja	Ja	Ja	Ja		P03 möchte keine weiteren Apps installieren (BiteWise als Lesezeichen auf iPad)
2	User sieht alle 3 Menüs	Ja	Ja	Ja	Ja	Ja		P01 Bilder passen nicht zu den Menüs P02 gleiche Bilder verschiedene Menüs P04 Achtet auf Bilder, welche nicht passen zum Gericht P04 erwähnt Gegessen Button, versteht ihn noch nicht ganz
3	User sieht den Preis	Ja	Ja	Ja	Ja	Ja		P05 Preise und Menütitel stechen zu wenig heraus
4	User versteht die Bewertung	Ja	Nein	Ja	Nein	Nein		P02 hat Kommentare gesucht P03 hat Kommentare gesucht P04 weiss nicht ob die Leute auch noch was zu den Bewertungen schreiben oder wie die Sterne zu Stande kommen
5	User sagt was zu den Kategorien	Nein	Ja	Nein	Ja	Ja		P04 vermisst Icon-Beschreibung/Legende (auf Mobile) (Hover Effekt fehlt auf Mobile)
								P05 findet Dark mode
								Notiz: Nach P01 wurden Laktose und Glutenfrei Icons geändert um Verwirrung zu vermeiden.

Figure 87: Test Protocol Scenario A

D.2 Test Scenario B

Testszenario B								
Schritt	Erwartung	P01	P02	P03	P04	P05		Bemerkungen
1	User versteht das Vegan Icon	Ja	Ja	Ja	Ja	Ja		
2	User versteht das Laktosefrei Icon	Nein	Ja	Ja	Ja	Ja		P01 versteht Laktosefrei Icon als es beinhaltet Laktose P02 hätte gerne bei den Details noch Text, also eben, dass es laktosefrei ist z.B., zusätzlich zu den Icons
3	User wählt das richtige Menü aus (Quinoa Bowl)	Nein	Ja	Ja	Ja	Ja		

Figure 88: Test Protocol Scenario B

D.3 Test Scenario C

Testszenario C								
Schritt	Erwartung	P01	P02	P03	P04	P05		Bemerkungen
1	User klickt auf "Siehe Details" Button	Ja	Ja	Ja	Ja	Ja		
2	User bemerkt die Nährwert Daten und versteht sie	Ja	Ja	Ja	Ja	Ja		
3	User sieht die Nachhaltigkeits Bewertung und versteht sie	Ja	Nein	Ja/Nein	Ja	Ja		P02 Sterne sind zu wenig klar, nichts aussagend, lieber CO2 oder ähnliches P03 Sterne Score zu unklar, was bedeuted es genau und wieso bzw. wer hat das bestimmt P04 Wie berechnet wäre noch interessant P05 fragt sich, was die Nachhaltigkeit genau bietet, mehr Erklärung/Details P05 fragt sich, ob die Kalorien pro Portion gerechnet sind

Figure 89: Test Protocol Scenario C

D.4 Test Scenario D

Testszenario D								
Schritt	Erwartung	P01	P02	P03	P04	P05		Bemerkungen
1	User klickt auf "In der Mensa gegessen" Button	Ja	Ja	Ja	Nein	Nein		P03 denkt, man könnte nicht verstehen, dass dahinter dann das Feedback noch liegt.
2	User klickt auf "Ja, fortfahren"	Ja	Ja	Ja	Nein	Nein		
3	User gibt Sternebewertung ab	Ja	Ja	Ja	Nein	Nein		
4	User gibt Kommentar ab	Ja	Ja	Ja	Nein	Nein		P04 denkt es ist schon gut, dass die Kommentare nicht öffentlich sind.
5	User sendet das Feedback	Ja	Ja	Ja	Nein	Nein		
								P05 macht generelles Feedback, da sie Feedback nicht hinter gegessen Button erwartet.

Figure 90: Test Protocol Scenario D

D.5 Test Scenario E

Testszenario E								
Schritt	Erwartung	P01	P02	P03	P04	P05		Bemerkungen
1	User swiped / klickt im Karussel nach links	Ja	Ja	Ja	Ja	Ja		P01 klickt, anstatt swiped P03 sucht andere Ansicht, als Tagesansicht. Wochenansicht wie bei einem Kalender P04 vermisst Wochenansicht
2	User findet das letzte Menü mit Fisch	Ja	Ja	Nein	Ja	Ja		P03 nach dem suchen der Wochenansicht trotzdem dann durch Tage geklickt, aber Fisch Icon nicht gefunden. P04 hat gefunden, aber fand das einzel Klicken mühsam. Besser: Wochenansicht oder sogar mit Filter

Figure 91: Test Protocol Scenario E

D.6 Test Scenario F

Testszenario F								
Schritt	**Erwartung**	**P01**	**P02**	**P03**	**P04**	**P05**		**Bemerkungen**
1	User klickt auf das Profil	Ja	Ja	Ja	Nein	Ja		P02 überrascht, dass es zwei Optionen gibt P04 kommentiert mehr, aber kommt nicht von alleine drauf, dass man es beim Profil einstellen kann.
2	User klickt auf Profilseite	Ja	Ja	Ja	Nein	Ja		
3	User wählt Vegan und Laktosefrei bei Ernährungsweise an	Ja	Ja	Ja	Nein	Ja		P02 klickt automatisch auch "Kein schweinefleisch" und "vegetarisch" an, weil er das ja dann auch ist
4	User macht Kalorien auf Hoch bei Fokus	Ja	Ja	Nein	Nein	Nein		P01 passt weitere sachen an (protein) P02 möchte gerne scrollen
5	User speichert die Änderungen	Ja	Ja	Ja	Nein	Ja		P05 scrollen geht nicht, kann nicht speichern darum verwirrt ob es gespeichert hat
6	User sieht die grüne Umrandung und "Für dich" Tag	Ja	Ja	Ja	Nein	Ja		P02 findet ausgegraut zu undeutlich

Figure 92: Test Protocol Scenario F

D.7 Test Scenario G

Testszenario G								
Schritt	**Erwartung**	**P01**	**P02**	**P03**	**P04**	**P05**		**Bemerkungen**
1	User klickt in Navbar auf "Feedback"	Ja	Ja	Ja	Ja	Ja		P04 braucht einen Hint, dass er es auch effektiv in der App machen kann
2	User klickt auf "Generelles Feedback"	Ja	Ja	Ja	Ja	Ja		P03 zögert bisschen vor feedback geben, da anonym button nicht gefunden
3	User füllt Feedback Formular aus	Ja	Ja	Ja	Ja	Ja		
4	User klickt "Anonym senden" Checkbox an	Ja	Ja	Ja	Nein	Ja		P03 hat Anonym Checkbox spät gefunden. P04 beschreibt mehr, was er vom Formular denkt, schickt es dann aber nicht ab.
5	User sendet das Feedback ab	Ja	Ja	Ja	Nein	Ja		P01senden ist verwirrend, da Feedbackformular nicht resettet und keine Response kommt P02 möchte auch Feedback wie: "Feedback wurde gesendet"

Figure 93: Test Protocol Scenario G

D.8 Test Scenario H

Testszenario H								
Schritt	**Erwartung**	**P01**	**P02**	**P03**	**P04**	**P05**		**Bemerkungen**
1	User klickt in Navbar auf "Feedback"	Ja	Ja	Ja	Ja	Ja		P02 vorher beim Feedback schon gesehen.
2	User klickt auf "Menüvorschlag"	Ja	Ja	Ja	Ja	Ja		
3	Menü gibt Flammkuchen als Gericht ein	Ja	Ja	Ja	Nein	Ja		P04 beschreibt mehr, was er vom Formular denkt, schickt es dann aber nicht ab.
4	User gibt eine Beschreibung ein	Ja	Ja	Ja	Nein	Ja		P02 würde vielleicht gerne noch einen Link dazu senden
5	User sendet Vorschlag ein	Ja	Ja	Ja	Nein	Ja		P01 wieder kein response bei versendung P02 wieder kein response bei versendung

Figure 94: Test Protocol Scenario H

D.9 Test Scenario I

Testszenario I								
Schritt	**Erwartung**	**P01**	**P02**	**P03**	**P04**	**P05**		**Bemerkungen**
1	User geht auf Menüplan	Ja	Ja	Ja	Ja	Ja		
2	User swiped / klickt im Karussel nach rechts bis Montag 20.10.2025	Ja	Ja	Ja	Ja	Ja		P01 klickt wieder anstatt swipen P05 klickt anstatt swiped, da sie kein Karusel bei so einer simplen App erwartet
3	User findet das passende Gericht für ihn/sie	Nein	Ja	Ja	Ja	Ja		P01 versteht Icons falsch

Figure 95: Test Protocol Scenario I

D.10 Test Protocol Legend

Legend		Web	Mobile
P01		☐	☑
P02		☑	☐
P03		☐	☑
P04		☐	☑
P05		☐	☑

Figure 96: Test Protocol Legend

E Figma Usability Test Interview Protocols

E.1 Usability Test 1 - Gioele (P01)

E.1.1 Pre-interview

- Was studierst du?
  - Informatik
- Wie oft gehst du in die Mensa?
  - Selten
- Weisst du, wo man der Mensa Feedback geben kann?
  - Ja, kenne QR codes
- Hast du der Mensa bereits Feedback gegeben? Über welchen Kanal?
  - Ja, weil es etwas gratis gab, online Formular
- Wo schaust du das tägliche Mensa Menu aktuell nach?
  - OST App, auch in Mensa manchmal
  - Auf dem Handy oder Notebook?
    - Immer handy
- Kannst du dich einfach entscheiden?
  - Manchmal ja, entscheide meist für Menü da Buffet teller teuer, vermisse Forschungsbar
- Achtest du speziell auf etwas in deiner Ernährung? Auf was?
  - Nein, auch wenn wofür in Mensa?
- Trackst du aktuell deine Ernährung?
  - Habe schonmal, vergesse es aber oft
  - Mit welchem Tool?
    - Selbst programmierte App aus Schule und Health App
  - Auf welchem Gerät?
    - Handy
- Welche Werte sind dir dabei speziell wichtig?
  - Kalorien, Proteine

E.1.2 Post-interview

- Wie ist dein allgemeiner Eindruck?
  - sehr gut
- Was hat Dir gut gefallen?
  - App interaktiv
  - Präferenzen auswählen
- Was war nicht so gut?
  - Icons verwirrend
- Hast du etwas vermisst?
  - Bei Details, Icons mit Inhaltsstoffen und Beschreibung. Also das es klar ist.
- Ist dir die Feedback-Funktion direkt aufgefallen?
  - Ja im Menu gesehen
- Konntest du schnell und einfach Feedback geben?
  - Ja, Bestätigung, dass erfolgreich abgeschickt wäre noch gut
- Hast du eine Feedback-Art vermisst?
  - Ja, Kommentare, welche andere Menschen sehen können
- War dir direkt klar, welches Menu dir empfohlen wird?
  - Ja grüner Rand erkannt.
  - Würdest du diese Funktion wünschen?
    - Ja
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
    - Ja beim Profil macht Sinn. Grösse und Gewicht sollte auch bei den anderen Ernährungsinfos sein.
  - Fehlen dir Konfigurationsmöglichkeiten?
    - Eingabelogik, z.B. entweder Vegan oder vegetarisch -> vegi heisst auch vegane werden vorgeschlagen.

- Wie findest du die Wahl und Darstellung der Icons?
  - Milch Icon heisst laktosefrei, das ist verwirrend. Besser durchgestrichen. Das gleiche bei Glutenfrei
  - Vegan, vegetarisch schwer zu unterscheiden, evl. Beschriftung
- War dir klar, wo die Nährwerte zu finden sind?
  - Ja, sinnvoll
  - Wie fandest du die Darstellung der Nährwerte?
    - gut
  - Fehlen Werte, welche dir wichtig wären?
    - Genauere Inhaltsstoffe wie z.B. Milch, alles was drin ist
- War es einfach zu sehen, welche Menus am nachhaltigsten sind?
  - Ja
  - Hilft dir diese Info?
    - Ja noch interessant.
  - Wie findest du die Darstellung?
    - Lieber grüne Punkte oder Planeten/Fussabdrücke als gelbe Sterne, da Sterne schon bei Bewertung
- Konntest du das heutige Menu schnell finden?
  - Ja
- Wie fandest du die Eingabeformulare beim Feedback?
  - PDF mit Rezept hochladen wäre gut.

E.2 Usability Test 2 - Dominik (P02)

E.2.1 Pre-interview

- Was studierst du?
  - Digital Design
- Wie oft gehst du in die Mensa?
  - 1- 2 mal pro Woche
- Weisst du, wo man der Mensa Feedback geben kann?
  - QR Codes
- Hast du der Mensa bereits Feedback gegeben? Über welchen Kanal?
  - Nein
- Wo schaust du das tägliche Mensa Menu aktuell nach?
  - Direkt in der Mensa, manchmal Website (PDF runterladen nervt aber)
  - Auf dem Handy oder Notebook?
    - Laptop
- Kannst du dich einfach entscheiden?
  - Ja eher
- Achtest du speziell auf etwas in deiner Ernährung? Auf was?
  - Nein
- Trackst du aktuell deine Ernährung?
  - Nein
  - Mit welchem Tool?
    –
  - Auf welchem Gerät?
    –
- Welche Werte sind dir dabei speziell wichtig?
  -

E.2.2 Post-interview

- Wie ist dein allgemeiner Eindruck?
  - Gut
- Was hat Dir gut gefallen?
  -
- Was war nicht so gut?
  - Preis und Menutitel präsenter
- Hast du etwas vermisst?
  - Kommentare anderer Studierende würden mehr helfen als Sterne
- Ist dir die Feedback-Funktion direkt aufgefallen?
  - Ist aufgefallen in Navbar
- Konntest du schnell und einfach Feedback geben?
  - Beim allgemeinen Feedback, die Punkte klarer beschreiben
- Hast du eine Feedback-Art vermisst?
  - Nein
- Würdest du Bilder wünschen pro Menu?
  - Symbolbilder zu verwirrend
- War dir direkt klar, welches Menu dir empfohlen wird?
  - Grüne Umrandung bemerkt, das Für dich aber nicht. Ausgegraute Kacheln sind zu undeutlich
  - Würdest du diese Funktion wünschen?
    –
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
    – Überrascht über zwei Optionen
  - Fehlen dir Konfigurationsmöglichkeiten?
    –
- Wie findest du die Wahl und Darstellung der Icons?
  - Icons könnten eindeutiger sein

- War dir klar, wo die Nährwerte zu finden sind?
  - Ja
  - Wie fandest du die Darstellung der Nährwerte?
    - Gut
  - Fehlen Werte, welche dir wichtig wären?
    - Laien meinung, sagt fehlt nichts
- War es einfach zu sehen, welche Menus am nachhaltigsten sind?
  - Sterne sind zu nichts sagend, lieber mehr beschreiben mit CO2 austoss oder ähnlichem
  - Hilft dir diese Info?
    - Ja
  - Wie findest du die Darstellung?
    -
- Konntest du das heutige Menu schnell finden?
  - Ja
- Wie fandest du die Eingabeformulare beim Feedback?
  - Möglichkeit zum Link senden wäre noch gut, möchte nicht so viel schreiben bei einem Rezept

E.3 Usability Test 3 - Elia (P03)

E.3.1 Pre-interview

- Was studierst du?
  - EEU
- Wie oft gehst du in die Mensa?
  - 4 Tage in der Woche
- Weisst du, wo man der Mensa Feedback geben kann?
  - Mit Personen sprechen, bei Erika
- Hast du der Mensa bereits Feedback gegeben? Über welchen Kanal?
  - Menu Vorschlag
- Wo schaust du das tägliche Mensa Menu aktuell nach?
  - Drei Teller vor Mensa
  - Auf dem Handy oder Notebook?
    –
- Kannst du dich einfach entscheiden?
  - Ja, oft was vom Selbstwahlbuffet
- Achtest du speziell auf etwas in deiner Ernährung? Auf was?
  - Laktosefrei, steht aktuell nicht mehr bei Mensa
- Trackst du aktuell deine Ernährung?
  -
  - Mit welchem Tool?
    –
  - Auf welchem Gerät?
    –
- Welche Werte sind dir dabei speziell wichtig?
  - Laktosefrei

E.3.2 Post-interview

- Wie ist dein allgemeiner Eindruck?
  - Gute Idee
- Was hat Dir gut gefallen?
  - Gut aufgebaut
- Was war nicht so gut?
  - Nachhaltigkeitsscore nicht so aussagekräftig
- Hast du etwas vermisst?
  - Kommentare der anderen Personen
  - Wochenansicht, mit weniger Infos und ohne Bilder. Z.B. nur Name und Icons
- Ist dir die Feedback-Funktion direkt aufgefallen?
  -
- Konntest du schnell und einfach Feedback geben?
  - Anonym Knopf gesucht, vielleicht eher zu oberst.
- Hast du eine Feedback-Art vermisst?
  -
- War dir direkt klar, welches Menu dir empfohlen wird?
  -
  - Würdest du diese Funktion wünschen?
    –
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
    – Ja.
  - Fehlen dir Konfigurationsmöglichkeiten?
    – Vielleicht noch mehr wie z.B. Fisch. Oder Filter?
- Wie findest du die Wahl und Darstellung der Icons?
  - Gut. Fisch andere Farbe?
- War dir klar, wo die Nährwerte zu finden sind?

  - Ja
  - Wie fandest du die Darstellung der Nährwerte?
    - Macht Sinn
  - Fehlen Werte, welche dir wichtig wären?
    - Eventuell könnte Salz interessant sein
- War es einfach zu sehen, welche Menus am nachhaltigsten sind?
  -
  - Hilft dir diese Info?
    -
  - Wie findest du die Darstellung?
    -
- Konntest du das heutige Menu schnell finden?
  - Ja
- Wie fandest du die Eingabeformulare beim Feedback?
  -
- Welche Bilder würden dir am besten gefallen?
  - Reale Bilder von der Mensa (aber eben schwer umsetzbar)
  - **AI Bilder Clip-Art, Comic mässig**
  - AI Bilder Clip-Art, Comic mässig aber realistischer
  - AI möglichst realistisch

E.4 Usability Test 4 - Cedric (P04)

E.4.1 Pre-interview

- Was studierst du?
  - EEU
- Wie oft gehst du in die Mensa?
  - 2-3x pro Woche
- Weisst du, wo man der Mensa Feedback geben kann?
  - Ja
- Hast du der Mensa bereits Feedback gegeben? Über welchen Kanal?
  - QR-Code Formular, Menuvorschlag, mehr Vegetarisches
- Wo schaust du das tägliche Mensa Menu aktuell nach?
  - OST-App
  - Auf dem Handy oder Notebook?
    - Handy
- Kannst du dich einfach entscheiden?
  - Aus Menuplan oder teilweise spontan noch nach den drei Tellern
- Achtest du speziell auf etwas in deiner Ernährung? Auf was?
  - vegetarisch, etwas ausgewogen nach Gefühl aber nicht speziell
- Trackst du aktuell deine Ernährung?
  - Nein
  - Mit welchem Tool?
    - –
  - Auf welchem Gerät?
    - –
- Welche Werte sind dir dabei speziell wichtig?
  -

E.4.2 Post-interview

- Wie ist dein allgemeiner Eindruck?
  - Cool die Darstellung und Infos zu sehen.
- Was hat Dir gut gefallen?
  - Menus vergleichen cool. Als Vegi ist die Auswahl halt eh schon kleiner.
- Was war nicht so gut?
  -
- Hast du etwas vermisst?
  - Kommentare von anderen?
  - Wochenansicht
- Ist dir die Feedback-Funktion direkt aufgefallen?
  - Ja
- Konntest du schnell und einfach Feedback geben?
  - Ja macht Sinn.
- Hast du eine Feedback-Art vermisst?
  - Ausgewogenheit noch bewerten
- War dir direkt klar, welches Menu dir empfohlen wird?
  -
  - Würdest du diese Funktion wünschen?
    - –
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
    - Es war nicht klar, dass man in so einer App tatsächlich Esspräferenzen angeben kann, deshalb gar nie soweit gekommen

  - Fehlen dir Konfigurationsmöglichkeiten?
    –

- Wie findest du die Wahl und Darstellung der Icons?
  - gut, aber noch besser beschreiben (auch mobile kein Hover)
- War dir klar, wo die Nährwerte zu finden sind?
  - Ja
  - Wie fandest du die Darstellung der Nährwerte?
    – Könnte interessant sein, für Personen die tracken
  - Fehlen Werte, welche dir wichtig wären?
    – Nein
- War es einfach zu sehen, welche Menus am nachhaltigsten sind?
  - Ja
  - Hilft dir diese Info?
    – interessant
  - Wie findest du die Darstellung?
    – Begründung fehlt
- Konntest du das heutige Menu schnell finden?
  - Ja
- Wie fandest du die Eingabeformulare beim Feedback?
  -
- Welche Bilder würden dir am besten gefallen?
  - Reale Bilder von der Mensa (aber eben schwer umsetzbar)
  - AI Bilder Clip-Art, Comic mässig
  - **AI Bilder Clip-Art, Comic mässig aber realistischer**
  - AI möglichst realistisch

E.5 Usability Test 5 - Vivien (P05)

E.5.1 Pre-interview

- Was studierst du?
  - Digital Design
- Wie oft gehst du in die Mensa?
  - 1x pro Woche
- Weisst du, wo man der Mensa Feedback geben kann?
  - Nein
- Hast du der Mensa bereits Feedback gegeben? Über welchen Kanal?
  - Ja wegen eigener Recherche in einem Studiumsmodul, sonst nicht
- Wo schaust du das tägliche Mensa Menu aktuell nach?
  - OST App
  - Auf dem Handy oder Notebook?
    - Handy
- Kannst du dich einfach entscheiden?
  - Nein, wegen Laktoseintoleranz
- Achtest du speziell auf etwas in deiner Ernährung? Auf was?
  - Laktose, Gesund (Ballaststoffe)
- Trackst du aktuell deine Ernährung?
  - Ja
  - Mit welchem Tool?
    –
  - Auf welchem Gerät?
    –
- Welche Werte sind dir dabei speziell wichtig?
  - Laktose, Gesund (Ballaststoffe)

E.5.2 Post-interview

- Wie ist dein allgemeiner Eindruck?
  - Gut bedienbar, einfach, Karoussel nicht erwartet, das es eher einfach aussieht. Schrift & Farben gut
- Was hat Dir gut gefallen?
  - Gut bedienbar
- Was war nicht so gut?
  -
- Hast du etwas vermisst?
  - evtl. zusätzliches PDF für Wochenansicht.
- Ist dir die Feedback-Funktion direkt aufgefallen?
  - Allgemeines Feedback ja. Gegessen Button: nicht klar, dass dahinter Feedback ist.
- Konntest du schnell und einfach Feedback geben?
  -
- Hast du eine Feedback-Art vermisst?
  - Nein
- War dir direkt klar, welches Menu dir empfohlen wird?
  - Ja aber eher Dezent. Könnte auch übersehen werden. Vor allem das Ausgegraute. Lieber nicht Rot wegen Achtung. Vielleicht blau.
  - Würdest du diese Funktion wünschen?
    –
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
    –
  - Fehlen dir Konfigurationsmöglichkeiten?
    –
- Wie findest du die Wahl und Darstellung der Icons?
  - Klar
- War dir klar, wo die Nährwerte zu finden sind?

  - Ja
  - Wie fandest du die Darstellung der Nährwerte?
    - Gut
  - Fehlen Werte, welche dir wichtig wären?
    -
- War es einfach zu sehen, welche Menus am nachhaltigsten sind?
  - Ja, aber ohne eine Info (z.B. kleines i) verwirrend
  - Hilft dir diese Info?
    -
  - Wie findest du die Darstellung?
    -
- Konntest du das heutige Menu schnell finden?
  - Ja
- Wie fandest du die Eingabeformulare beim Feedback?
  - Klar. Evtl auch noch Icons für Vegan/ Vegetarisch usw.
- Welche Bilder würden dir am besten gefallen?
  - Reale Bilder von der Mensa (aber eben schwer umsetzbar)
  - AI Bilder Clip-Art, Comic mässig
  - **AI Bilder Clip-Art, Comic mässig aber realistischer**
  - AI möglichst realistisch

F Summative Usability and Acceptance Test Observations

F.1 Test Observations - Jeriel (P01)

F.1.1 Test Scenario A

- Sieht heutiges Menu
- Titel wird nicht ganz angezeigt, weil zu lang
- Denkt es gibt mehr Infos in Details als nur Nutritions
- Sieht Sternebewertung
- Sieht Preis

F.1.2 Test Scenario B

- Versucht zu filtern mit Icon-Legende
- Findet Veganes Menu anhand von Icon beim Menu

F.1.3 Test Scenario D

- Klickt auf Gericht bewerten
- Gibt Ratings ab
- Sieht es ist im Tagesmenu Plan gespeichert, weiss aber nicht wo der ist -> gibt es noch nicht

F.1.4 Test Scenario E

- Benutzt Wochenansicht und findet entsprechendes Menu schnell

F.1.5 Test Scenario F

- Weiss grundsätzlich nicht, was das Konto nützt
- Wäre also von alleine nicht darauf gekommen eins zu machen
- Weiss nicht in welchen der beiden Einstellungs-Optionen, klickt zuerst auf das falsche (Benutzereinstellungen)
- Stellt seine Einstellungen ein
- Sieht Menuvorschalg

F.1.6 Test Scenario G

- Findet Feedback
- Gibt Feedback
- Klickt Anonym
- Sendet ab
- Sieht, dass es gesendet wurde

F.1.7 Test Scenario H

- Macht korrekt Menuvorschlag
- Reicht diesen ein
- Fände es cool selbst eingereichte Menuvorschläge angezeigt zu bekommen

F.1.8 Test Scenario I

- Navigiert auf Montag und sieht was es gibt

F.2 Test Observations - Philipp (P02)

F.2.1 Test Scenario A

- Sieht heutige Menus, Preise, Bewertungen, Icons

F.2.2 Test Scenario B

- Findet ein Menu

F.2.3 Test Scenario D

- Klickt auf Gericht bewerten
- Bewertet das Menu
- Schreibt Kommentar

F.2.4 Test Scenario E

- Klickt Tage durch
- Findet korrektes Menu
- Sieht auch Wochenansicht
- Hat Wochenansicht aus Gewohnheit der Mensa PDFs in die falsche Richtung gelesen (Zeilen/Spalten) und dann gemerkt, dass es hier umgekehrt ist

F.2.5 Test Scenario F

- Alte Smart Eating Einstellungen verwirrend (Gewicht etc.)
- Füllt diese aus, anstatt den anderen Menupunkt mit den Nutrition Einstellungen
- Findet richtige Einstellungen verspätet
- Sieht ausgegraute Menus

F.2.6 Test Scenario G

- Findet Feedback Menupunkt
- Füllt allgemeines Feedback aus
- Klickt anonym und sendet
- Sieht Bestätigungs Toast ist aber trotzdem noch skeptisch ob es wirklich geklappt hat

F.2.7 Test Scenario H

- Findet Menuvorschläge
- Füllt Vorschlag aus
- Sendet ab
- Sieht Bestätigungs Toast ist aber trotzdem noch skeptisch ob es wirklich geklappt hat

F.2.8 Test Scenario I

- Findet Menu am Montag

F.3 Test Observations - Simeon (P03)

F.3.1 Test Scenario A

- Kein Rating gesehen, aber Preise, Titel und Infos

F.3.2 Test Scenario C

- Findet das richtige Menü

F.3.3 Test Scenario D

- Hat Feedback Funktion gefunden und gesendet

F.3.4 Test Scenario E

- Hat in den vorherigen Tagen korrekt Schweingericht gefunden

F.3.5 Test Scenario F

- Findet Dietary Settings direkt und stellt korrekt ein, sieht recommendations

F.3.6 Test Scenario G

- Wollte zuerst nochmal Menü Feedback machen danach aber richtiges Feedback gefunden

F.3.7 Test Scenario H

- Findet Menüvorschlag direkt und sendet ein

F.3.8 Test Scenario I

- Klickt vor bis zum nächsten Montag und findet das Gericht

G Summative Usability and Acceptance Test Interview Protocols

G.1 Post Interview - Jeriel (P01)

- Wie ist dein allgemeiner Eindruck?
  - Nicht schlecht
- Was hat Dir gut gefallen?
  - Anzeige und Feedback
- Was war nicht so gut?
  - Ganzer Name beim Gericht, ungenaue Nutritions, Buggy (weil SmartEating API war langsam)
- Hast du etwas vermisst?
  - Menübeschreibung, ganzer Titel anzeigen bei Details
- Konntest du schnell und einfach Feedback geben?
  - Verständlich, Nicht alle Stern Kategorien als Pflichtfelder machen
- War dir direkt klar, welches Menu dir empfohlen wird?
  - Roter Rahmen gut, “Für dich” Tag nicht so präsent
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
    - 2 verschiedene Einstellungspunkte, eines würde reichen, Form sonst gut
- Wie findest du die Wahl und Darstellung der Icons?
  - Verständlich, vegetarisch & vegan gleiches Icon ist etwas verwirrend
- Wie fandest du die Darstellung der Nährwerte?
  - Ungenau, sonst ok
- Konntest du das heutige Menu schnell finden?
  - Ja, Smart Eating im Header verlinken wäre aber noch nützlich
- Wie fandest du die Wochenansicht?
  - Sehr Gut, übersichtlich
- Gefallen die die generierten Bilder?
  - Sieht cool aus, bräuchte es für ihn aber nicht, würde eher Wochenansicht benutzen

- Die Website hat ein nutzerfreundliches Interface.
  - [ ] Ja
  - [x] (Sonst) Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [x] (Buggy API) Nein
- Die Navigation auf der Website ist einfach und verständlich.
  - [ ] Ja
  - [x] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Darstellung der Website inkl. Bilder gefällt mir.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein

- Die Website ist einfach zu bedienen.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Website ist zeiteffizient zu bedienen.
  - [ ] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [x] Eher Nein (PDF schneller)
  - [ ] Nein
- Ich würde die Website im OST-Alltag verwenden.
  - [x] Ja (integriert in OST-App, oder schnell erreichbar)
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Ich würde die Website meinen Mitstudenten weiterempfehlen.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein

G.2 Post Interview - Philipp (P02)

- Wie ist dein allgemeiner Eindruck?
  - Gut, in etwa das was es auch in der Campus App geben könnte, also besser als PDF
- Was hat Dir gut gefallen?
  - Angenehm am Notebook, man kann es anschauen, ohne das PDF herunterzuladen, Menu bewerten cool, weil man spezifisch Feedback geben kann, dass kann man bis jetzt nicht wirklich
- Was war nicht so gut?
  - Einstellungen Unterscheidung
- Hast du etwas vermisst?
  - Nein
- Konntest du schnell und einfach Feedback geben?
  - Ja, aber Menuvorschläge hätte er nicht beim Feedback erwartet.
- War dir direkt klar, welches Menu dir empfohlen wird?
  - Ja, wenn mal alles ausgegraut wäre verwirrend, aber sollte nie der Fall
- Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
  - Nein Einstellungen Unterscheidung war verwirrend, ein Menupunkt besser
- Wie findest du die Wahl und Darstellung der Icons?
  - Gut
- Wie fandest du die Darstellung der Nährwerte?
  - Ok, aber wenn nur Nährwerte, dann nach Nährwerte anzeigen benennen (wäre noch mehr vorgesehen in der Zukunft unter diesem Punkt)
- Konntest du das heutige Menu schnell finden?
  - Ja
- Wie fandest du die Wochenansicht?
  - Verwirrend weil Spalten/Zeilen getauscht im Vergleich zum PDF
- Gefallen die die generierten Bilder?
  - Besser als kein Bild, richtiges Bild wohl unrealistisch, da Mensa diese nicht machen kann, von dem her OK.

- Die Website hat ein nutzerfreundliches Interface.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Navigation auf der Website ist einfach und verständlich.
  - [ ] Ja
  - [x] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Darstellung der Website inkl. Bilder gefällt mir.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Website ist einfach zu bedienen.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein

- [ ] Nein

- Die Website ist zeiteffizient zu bedienen.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Ich würde die Website im OST-Alltag verwenden.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Ich würde die Website meinen Mitstudenten weiterempfehlen.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein

G.3 Post Interview - Simeon (P03)

- Wie ist dein allgemeiner Eindruck?
  - Gute Bilder zum einen Eindruck zu verschaffen, schöne Webseite
- Was hat Dir gut gefallen?
  - Gute Idee mit Dietary matching aber bei 3 Gerichten vielleicht bisschen unnötig, für Leute die es brauchen aber sicher nützlich
- Was war nicht so gut?
  - Bei Weekly View Lanes immer wieder beschriftet, besser ausserhalb der Tabelle nur einmal, Menü Feedback verwirrend dass nicht steht ob Anonym oder nicht
- Hast du etwas vermisst?
  - Nein
- Konntest du schnell und einfach Feedback geben?
  - Ja
- War dir direkt klar, welches Menu dir empfohlen wird?
  - Ja
  - Hast du einfach gefunden, wo du deine Ernährungsinfos anpassen kannst?
    - Ja, aber bei Einstellungen denkt man eher daran, dass es dort zu finden ist
- Wie findest du die Wahl und Darstellung der Icons?
  - Vegan und Vegetarisch Symbol sollten unterschiedlich sein
- Wie fandest du die Darstellung der Nährwerte?
  - Übersichtlich
- Konntest du das heutige Menu schnell finden?
  - Ja
- Wie fandest du die Wochenansicht?
  - Gut aber besser mit Bilder auf daily
- Gefallen die die generierten Bilder?
  - Ja

- Die Website hat ein nutzerfreundliches Interface.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Navigation auf der Website ist einfach und verständlich.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Darstellung der Website inkl. Bilder gefällt mir.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Die Website ist einfach zu bedienen.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein

- [ ] Nein

- Die Website ist zeiteffizient zu bedienen.
  - [ ] Ja
  - [x] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Ich würde die Website im OST-Alltag verwenden.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein
- Ich würde die Website meinen Mitstudenten weiterempfehlen.
  - [x] Ja
  - [ ] Eher Ja
  - [ ] Neutral
  - [ ] Eher Nein
  - [ ] Nein