

**Gottfried Wilhelm
Leibniz University of Hanover
Faculty of Electrical Engineering and Computer Science
Institute of Practical Computer Science
Department of Software Engineering**

**Architectural analysis and implementation of an
AI component for appointment optimization for
dentists**

**(Architekturanalyse und Implementierung einer KI-Komponente zur
Terminoptimierung für Zahnärzte)**

Bachelor's Thesis

in the field of Computer Science

by

Mohammad Ali Ferdowsi

Examiner: Prof. Dr. Kurt Schneider

Second examiner: Dr. Jil Klünder

Supervisor: M.Sc. Jianwei Shi

Hannover, 15.05.2024

Declaration of Independence

I hereby affirm that I have composed the present bachelor's thesis independently and without any external assistance, and that I have not used any sources or aids other than those stated in the thesis. The thesis has not been submitted to any examination office in the same or a similar form.

Hannover, 15.05.2024

Mohammad Ali Ferdowsi

Content Adjustment

The initial thesis, titled "Architectural analysis and implementation of an AI component for appointment optimization for dentists" aimed to develop a scheduling software with integrated AI functionality focused on optimizing appointments. However, due to the complex nature of AI integration, time constraints, and challenges in acquiring suitable training data, the project's focus was revised. The updated title, "Architecture Analysis and Implementation of Software for Appointment Optimization for Volunteers" reflects this shift, concentrating on software architecture and implementation while ensuring a thorough and feasible thesis completion within the set timeframe. Instead of integrating AI functionality, the thesis now develops an algorithm to optimize appointments.

Contents

Abstract.....	ix
1. Introduction.....	1
1.1 Presentation of Problem.....	2
1.2 Solution approach.....	2
1.3 Structure of thesis	3
2. Foundational Information.....	4
2.1 Software Development Process Models.....	4
2.2 Agile Model	4
2.3 Waterfall Model	5
2.4 Kanban Model.....	5
2.5 Microservice-Architecture.....	6
2.6 Spring.....	6
2.7 Spring Boot	6
2.8 Angular	6
2.9 Kano Model.....	8
2.10 Analyzing Requirements.....	9
2.11 OpenAPI	11
2.12 Contract-First API Development	12
3. Related Work and Market Analysis.....	15
3.1 Related Work	15
3.2 Market Analysis.....	16
4. Requirements Analysis.....	18
4.1 Requirements Elicitation	18
4.2 Interview	18
4.3 Specification	19
4.3.1 Goal Definition.....	19
4.3.2 Functional Requirements	20
4.3.3 Non-Functional Requirements.....	21
4.3.4 Mockups.....	22
4.3.5 Compromises.....	24
5. Implementation.....	25
5.1 Approach.....	25
5.2 Technologies.....	25
5.3 Architecture	26

5.4 Configuration	28
5.5 Best Practices.....	28
5.6 Scheduling Optimization Algorithm	29
6. Testing and Evaluation	32
6.1 Testing	32
6.2 Evaluation.....	32
6.2.1 Study design	33
6.2.2 Participants	34
6.2.3 Study Results.....	35
6.2.4 Threats to Validity and Challenges	38
6.2.5 Summarizing Results.....	39
7. Conclusion	40
8. Future Development	41
8.1 AI Integration	41
8.2 Gamification.....	42
8.3 Roadmap	43
Appendix A - Specification	47
Appendix B - Domain Design	69
Appendix C - Developed Application Screenshots	71
USB Content	76

List of Figures

Figure 1 - Waterfall model life cycle.....	5
Figure 2 - Example of a kanban board	5
Figure 3 - Kano model.....	9
Figure 4 - Overview of the requirements analysis tasks	10
Figure 5 - Example of an OpenAPI Specification	12
Figure 6 - Auto-generated code for backend in Spring from example specification ...	13
Figure 7 - Auto-generated code for frontend in Angular from example specification .	14
Figure 8 - Use case for [FR6]	21
Figure 9 - Mockup for the homepage	23
Figure 10 - Mockup for the calendar page	23
Figure 11 - Application architecture design	26
Figure 12 - Segment of domain design	27
Figure 13 - Simplified example of OnPush usage in the application	29
Figure 14 - Usage scenario for the optimization algorithm	31
Figure 15 - Participant demographic charts.....	35
Figure 16 - SUS Score for each participant and the overall average	36
Figure 17 - Leaderboard page mockup.....	43

List of Tables

Table 1 - Usability goal definition	33
Table 2 - User experience goal definition	33
Table 3 - Feature discovery goal definition.....	34
Table 4 - Recommendations and Observations gathered from the survey	37

List of Codes

Code 1 – Initializing a variable in Typescript	7
Code 2 – Data binding in HTML.....	7
Code 3 - Dependency Injection	7

Abstract

In today's world, the right software is crucial for teams to work together effectively, especially in projects that rely on the goodwill and coordination of volunteers. Good software helps teams communicate better, manage their resources efficiently, and ensure everyone is on the same page. The *Zahnmobil* project, a collaborative non-profit initiative powered by volunteer dentists, assistants, and drivers, along with financial contributions from dedicated individuals, is committed to providing essential dental care to those in need. This thesis emphasizes the development of a user-friendly software solution, particularly tailored to meet the needs of the *Zahnmobil* project. Central to this thesis is a thorough requirements analysis for a detailed understanding of the system's essential needs, followed by an architecture design aimed at creating a maintainable and robust framework. The implementation of the web application used a Contract-First API development approach utilizing OpenAPI. Additionally, enhancing usability was a primary objective of the application. The final stages include testing and a systematic evaluation to assess performance and user satisfaction, ensuring that the software not only meets the immediate operational needs but also sets a foundation for future enhancements. In the evaluation phase, a System Usability Scale (SUS) survey was conducted with 12 participants, resulting in an average SUS score of 96.67, indicating excellent usability.

Chapter 1

1. Introduction

Effective collaboration is key in any team endeavor, and this is particularly true for projects that depend on volunteer participation. The right technological tools can make all the difference, enabling clear communication, efficient organization, and cohesive teamwork. Managing volunteer-driven projects effectively calls for straightforward and intuitive scheduling and management systems. These tools simplify team coordination and improve workflow efficiency [19]. The objective is to create an intuitive and user-friendly web application that enhances overall efficiency, facilitates coordination processes, and guarantees straightforward maintainability. Complex procedures for straightforward tasks, the lack of essential administrative capabilities, and systemic inconsistencies can significantly undermine user experience. Such obstacles not only diminish user satisfaction but can also deter their willingness to participate. Moreover, the high support costs associated with the current *Zahnmobil* system further complicate matters, as *Zahnmobil* is a nonprofit organization, making efficient cost management crucial. Recognizing these challenges, the necessity for a simplified, efficient software solution becomes clear. This thesis delves into software development, focusing on requirement analysis, architectural design, implementation, testing and evaluation. The initial phase involves extensive requirement analysis research to comprehensively understand user needs. This important step informs subsequent stages of the project, ensuring that the software aligns with the actual requirements of the *Zahnmobil* project. The thesis then progresses into the architectural design phase, aiming to establish a flexible and maintainable system architecture. Transitioning into the implementation phase, the focus shifts to crafting a user-friendly software application, utilizing several best practices throughout this phase. Finally, the project highlights thorough testing and evaluation, focusing on assessing the usability of developed application and gathering feedback from the end-users of the *Zahnmobil* project through a survey. This evaluation ensures the software effectively supports the scheduling and operational needs of the *Zahnmobil*. This thesis is conducted in partnership with *adesso*, a leading IT service provider in Germany known for its expansive network and strong emphasis on future-oriented technology solutions.

1.1 Presentation of Problem

Confronting the complexities of application development within a constrained four-month period presents a formidable challenge, particularly when the aim is to accommodate a user base with a wide spectrum of technological expertise. Despite these variances, it is imperative for the application to uphold an intuitive and user-friendly approach. This ensures every volunteer, irrespective of their comfort with technology, can effectively engage with the system. Additionally, it is important that the application not only fulfills the requirements but also embodies a scalable structure and maintainable framework, allowing for future enhancements and development.

1.2 Solution approach

The aim is to develop a tool that is not only efficient and effective but also user-friendly for all, regardless of IT skill or age. A planned communication strategy aligns the project with the *Zahnmobil* project's vision. Regular weekly meetings with the stakeholder, ensures continuous updates, idea exchanges, and feedback incorporation. The thesis begins with a research phase that focuses on requirement analysis methodologies and best practices in requirements engineering. This groundwork ensures a thorough understanding of the *Zahnmobil* project's needs. With the requirements defined, the project emphasizes developing a maintainable and scalable architecture for the application. By adhering to standards that accommodate future changes and growth, the architecture is ensured to be practical and adaptable for long-term use. Additionally, feedback on the developed architecture was sought from a senior architect designer at *adesso*, enhancing its effectiveness and alignment with professional standards. During the implementation phase, the plans are brought to life with a focus on developing an intuitive and user-friendly interface. The design leverages principles of Human-Computer Interaction (HCI) to meet high usability standards. Best practices, such as Contract-First API Development, are applied throughout the implementation to enhance both quality and effectiveness. Furthermore, testing and refinement are essential to ensure the product not only meets the initial requirements but also stands up to real-world use. This testing strategy employs both automated and exploratory testing to ensure all components work seamlessly and meet the required specifications. Finally, the developed application is assessed with the end users of the *Zahnmobil* project through a semi-structured interview process. This involves a practical session where users test the application, followed by a survey where the Standard Usability Scale (SUS) is utilized.

1.3 Structure of thesis

Chapter 2 provides the foundational information necessary to understand the thesis, while Chapter 3 discusses related works and includes a brief market analysis. Chapter 4 describes the process of conducting the requirements analysis. Chapter 5 outlines the system's architectural design and the implementation of the application. Chapter 6 discusses the testing strategies deployed and explores the evaluation of the application through end-user feedback analysis. Chapter 7 concludes the thesis, summarizing key findings. Lastly, Chapter 8 discusses potential future developments, outlining opportunities for further development.

Chapter 2

2. Foundational Information

This Chapter covers essential definitions of theoretical and technological foundational information's that form the basis of this thesis. Understanding these foundational elements is important for following the discussions in later chapters.

2.1 Software Development Process Models

The Software Development Process Model, also known as the Software Development Life Cycle (SDLC), outlines the various activities required for software evolution throughout its lifecycle. It recommends a specific methodology for conducting these activities across the lifecycle and advises on the documents and artifacts that should be generated at the end of each phase. The primary advantage of employing a development process is that it offers a systematic and disciplined framework for development [11]. There are currently many different SDLC models available, such as the Waterfall Model, Kanban Model, and Agile. Each offers its own unique advantages and disadvantages. These models provide structured approaches to software development, catering to various project requirements, team sizes, and goals. Choosing the right model is crucial for effective project management and successful software delivery, as it influences how tasks are approached, executed, and completed throughout the development process [1].

2.2 Agile Model

Agile methodology focuses on adaptability to changing requirements and emphasizes customer satisfaction through the quick delivery of useful software. It encourages welcoming changes at any stage of development and ensures the frequent release of working software, often within weeks. The central principle of Agile is to continually meet customer needs by delivering small, practical increments of software rapidly [4].

2.3 Waterfall Model

The Waterfall model is a linear and sequential approach to software development, where each phase is completed fully before moving on to the next, without overlap or parallelization. This model offers several benefits: it is easy to implement because of its linear structure, it clarifies requirements before development starts, and it requires minimal resources for execution [4].

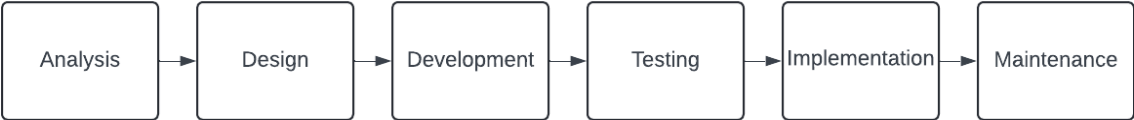


Figure 1 - Waterfall model life cycle

2.4 Kanban Model

The Kanban Model, first developed in the 1950s for Toyota's manufacturing process, has become a widely adopted agile methodology across various industries, including software engineering. Its core principles focus on enhancing team efficiency in daily operations through several key practices. These practices include visualizing the workflow on a Kanban board, limiting the amount of work in progress, and managing the flow of tasks. This approach helps teams to streamline processes and improve productivity [6].

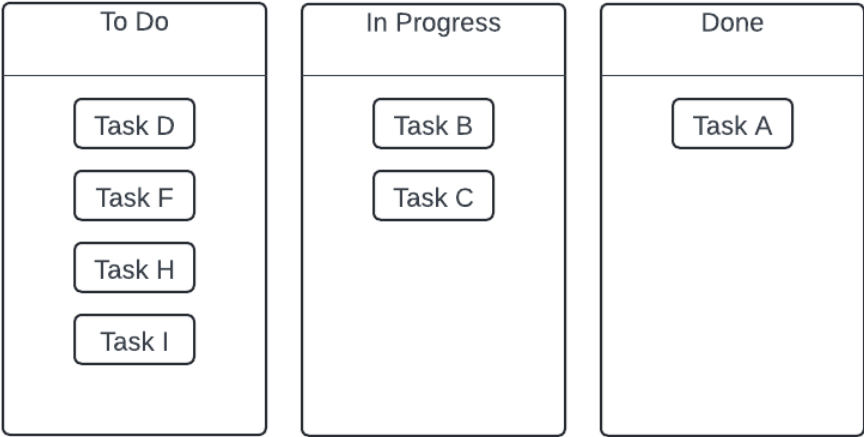


Figure 2 - Example of a kanban board

2.5 Microservice-Architecture

Microservice architecture adopts the strategy of breaking down a software application into a multitude of small, independently functioning services, each loosely connected to the others. This approach moves away from the monolithic architecture model, where all functionalities are embedded within a single, large application. By enabling each service to run its own process and interact through lightweight communication methods, this architecture facilitates a design that is significantly more scalable and easier to maintain [10].

2.6 Spring

Spring is a Java framework designed for building web applications. It provides a robust suite of data persistence options, a comprehensive security framework, and extensive microservices support. Essentially, Spring operates through a core container known as the Spring Application Context, which orchestrates the creation and management of application components, also called beans. These beans are intricately interconnected within the Spring Application Context to assemble a fully functional application. This process is analogous to constructing a jigsaw puzzle, where each piece represents a distinct component that, when connected, forms a complete picture—in this case, a cohesive application [22].

2.7 Spring Boot

Spring Boot is an extension of the Spring framework that simplifies the process of creating standalone, production-grade applications based on Spring. It achieves this by offering default configurations, thus eliminating the need for extensive Spring setup. Additionally, Spring Boot enhances applications with non-functional features such as security and externalized configuration, making development faster and more efficient [23].

2.8 Angular

Angular is a framework designed for building applications, offering a standardized structure that enhances maintainability and scalability for large projects. Key features of Angular include:

1. **Custom Components:** Angular enables the creation of custom, declarative components that encapsulate both functionality and rendering logic, facilitating reuse across applications.
2. **Data Binding:** It simplifies the process of displaying data from the TypeScript code in the view. For instance, when a string variable named 'title' in the TypeScript code (see Code 1) changes, its updated value is automatically reflected in the HTML view (see Code 2), thanks to Angular's data binding mechanism.

```
1 export class AppComponent {  
2   title: string = 'Hello World!';  
3 }
```

Code 1 – Initializing a variable in Typescript

```
1 <div>{{title}}</div>
```

Code 2 – Data binding in HTML

3. **Dependency Injection:** This feature supports the modular creation of services that can be injected as needed, enhancing both the testability and reusability of these services. An example of this is injecting a 'LoginService' into the constructor of a login component, which then manages the logic for user authentication.

```
1 export class LoginComponent {  
2   constructor(private logInService: LoginService) {}  
3 }
```

Code 3 - Dependency injection example

4. **Testability:** Angular has been engineered with a focus on making every component of the application easily testable.
5. **Comprehensiveness:** Angular is a comprehensive framework, offering a wide array of features and tools for server communication, routing, and more, making it a robust solution for web application development [20].

2.9 Kano Model

Introduced by Dr. Noriaki Kano in 1978, the Kano Model classifies system requirements into three categories:

1. Basic attributes: Basic needs are features that are automatically expected and assumed.
2. Performance attributes: Performance needs are the special features that are explicitly requested.
3. Excitement attributes: Excitement attributes are product features that the stakeholder is initially unaware of and discovers as pleasant surprises during use.

Among stakeholders, a process of habituation occurs where what once excited them gradually becomes expected and then basic over time. Consequently, this habituation necessitates a commitment to consistent innovation and the introduction of creative ideas. Furthermore, Understanding the correlation between customer satisfaction and the degree of fulfillment of different types of requirements is crucial. Basic attributes are essential and must be implemented; their absence leads to significant dissatisfaction, while their presence alone does not guarantee full satisfaction. Performance attributes, when fulfilled, enhance customer satisfaction; however, their absence might still be tolerated by customers, though it increases dissatisfaction with each missing element. Excitement attributes can significantly boost customer satisfaction if implemented, yet their absence doesn't necessarily lead to dissatisfaction [17].

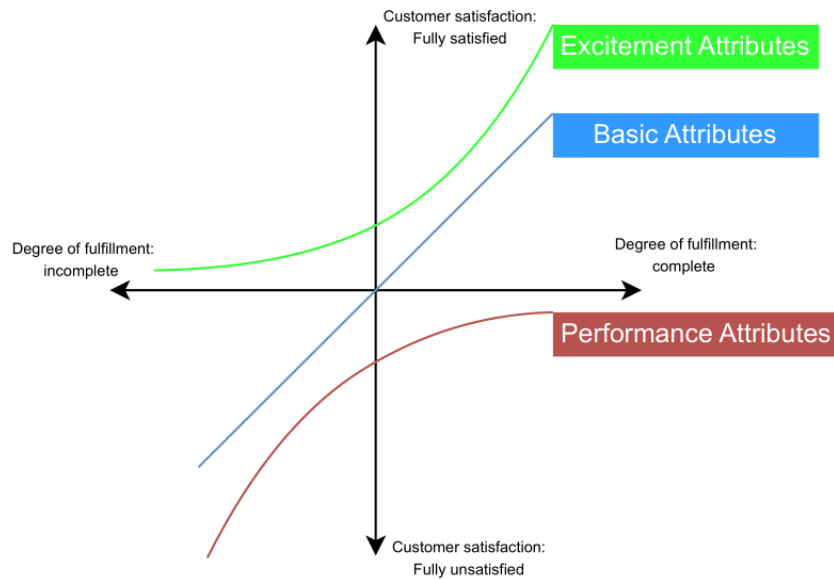


Figure 3 - Kano model

2.10 Analyzing Requirements

In Requirements analysis, after gathering the initial requirements through requirements elicitation, the next step is to analyze the requirements. The goal of analyzing initial requirements is to develop solid and dependable requirements efficiently, using minimal resources. This process helps ensure the collection of requirements meets quality standards and assists in determining where further improvements are needed. It's not necessary to document every new requirement that emerges from this analysis. Instead, these new requirements mainly serve as a basis for deeper analysis. Analyzing the requirements involves the following steps:

1. **Define Original Requirements:** Define the initial requirements through requirements elicitation.
2. **Separate Requirements:** Separate the original requirements into multiple, distinct requirements if necessary. This approach allows each component to be individually assessed in subsequent stages. For example, a requirement to create operations and for an administrator to enroll users in these operations can be divided into two parts: one focusing on the ability to create operations, and another on the administrator's ability to enroll users in the created operations.

3. **Extract Necessary Requirements:** This step ensures that the requirements accurately reflect what the system should deliver, neither exceeding nor falling short of its intended capabilities. It is crucial that the requirements are realistic and achievable within existing constraints, such as development timelines. For example, the requirement for a responsive design for mobile users was evaluated against the time constraints and identified as a potential area for compromise.
4. **Abstract Requirements:** Abstraction aids in forming a comprehensive, preliminary view of the system's requirements. It serves as a common foundation for the identified requirements. For example, the specific ability for administration to edit usernames can be abstracted to a broader capability where administration can edit user profiles.
5. **Supplement Missing Requirements:** From the tasks defined earlier, we initially identified requirements that directly stemmed from the original specifications. Now, we are expanding these to include aspects that were not explicitly mentioned by stakeholders but are somewhat implied by the previously identified requirements. For instance, allowing administrators to edit user profiles raises further questions: Should administrators be able to modify a user's email address or role? Can an administrator grant or remove administrative rights from another user? These additional requirements must be discussed and clarified with the stakeholders.
6. **Refine Requirements:** For each identified requirement, the question is asked whether it should be described in greater detail.
7. **Improve Requirements:** In this final task, we review and potentially improve each previously identified requirement to ensure its quality. This involves verifying that the conditions within each requirement are complete and correct. For example, if a requirement states that users can sign up for an operation, it should also specify that a user can enroll if their corresponding role in the operation is available [17].



Figure 4 - Overview of the requirements analysis tasks

2.11 OpenAPI

OpenAPI is a specification for describing HTTP-based APIs using either YAML or JSON formats. This specification, which outlines an API's inputs and outputs, can be manually crafted or auto-generated from existing code. Once prepared, the specification facilitates the creation of user-friendly documentation and even generation of server stubs for API implementation. Some advantages of using OpenAPI include:

1. **Tooling Support:** OpenAPI specifications are supported by various tools like Swagger Codegen and OpenAPI Generator, which aid in creating applications that communicate with the API. These tools can generate necessary code automatically, offering a significant jumpstart in the application creation process. Other tools, such as Swagger UI, can also be used for visualizing the API definitions.
2. **Customization:** The OpenAPI definition can be customized to meet specific needs, which helps in creating templates that are specific to your requirements.
3. **Speed and Consistency:** OpenAPI can speed up the development process and ensure consistency across different APIs.
4. **Standardization:** OpenAPI assists in standardizing all APIs in consistent patterns, which is particularly beneficial when managing more than one API. This helps in measuring those patterns and improving the overall design and consumption of APIs [15].

This example outlines an OpenAPI specification with an endpoint for fetching the data of a logged-in user (see Figure 5). Line 6 specifies the API's local access point as port 8080. Line 8 introduces an endpoint, `/api/v1/whoami`, and the subsequent lines describe its method—GET—and provide a summary. This level of detail aids in the documentation of APIs. Then, at line 14 till 22, the expected successful (200 OK) and error (400 Something went wrong) responses are detailed. The successful response returns the `whoAmIModel` schema. At line 26 till 47, the `whoAmIModel` schema is defined which is a Data Transfer Object (DTO).

```

1  openapi: 3.0.0
2  info:
3    title: Example Specification
4    version: 0.0.1
5  servers:
6    - url: http://localhost:8080
7  paths:
8    /api/v1/whoami: # Endpoint for getting logged in User data.
9      get: # HTTP Method
10       summary: Gets Logged in User info
11       operationId: getWhoAmI
12       tags:
13         - whoAmI
14       responses:
15         200:
16           description: "OK"
17           content: # Response of endpoint being the whoAmIModel
18             application/json:
19               schema:
20                 $ref: "#/components/schemas/whoAmIModel"
21         400:
22           description: "Something went wrong"
23
24 components:
25   schemas:
26     whoAmIModel: # Creating DTO models
27       type: object
28       required: ["id", "email", "firstName", "lastName"]
29       properties:
30         userId:
31           type: string
32           example: abcd-123
33         userProfileId:
34           type: string
35           example: abcd-123
36         email:
37           type: string
38           example: test@test.de
39         firstName:
40           type: string
41           example: Max
42         lastName:
43           type: string
44           example: Mustermann
45         isAdmin:
46           type: boolean
47           default: false

```

Figure 5 - Example of an OpenAPI Specification

2.12 Contract-First API Development

In API development, two primary approaches are utilized: Code First and Contract First (Design First). The Code First approach involves initially writing the API code and then generating the documentation afterwards with generative tools. On the other hand, the Contract First approach begins with the creation of a detailed API specification such as an OpenAPI Specification. This specification is not only more maintainable but can also be used to generate API stub code for both the frontend and backend simultaneously. This approach enables parallel development across different development teams and in many cases the result is a faster time-to-market for multiple implementations and more consistent documentation [12]. Figures 6 and 7 illustrate

the generated code for the backend in Spring and the frontend in Angular, respectively, from the specification at Figure 5.

```
1  /**
2   * GET /api/v1/whoami : Gets Logged in User info
3   *
4   * @return OK (status code 200)
5   *         or Something went wrong (status code 400)
6   */
7  @ApiOperation(value = "Gets Logged in User info", nickname = "getWhoAmI",
8   notes = "", response = WhoAmIModel.class, authorizations = {
9
10     @Authorization(value = "cookieAuth")
11     }, tags={ "whoAmI", })
12  @ApiResponses(value = {
13     @ApiResponse(code = 200, message = "OK", response = WhoAmIModel.class),
14     @ApiResponse(code = 400, message = "Something went wrong") })
15  @GetMapping(
16     value = "/api/v1/whoami",
17     produces = { "application/json" }
18  )
19  default ResponseEntity<WhoAmIModel> getWhoAmI() {
20     return getDelegate().getWhoAmI();
21 }
```

Figure 6 - Auto-generated code for backend in Spring from example specification

```

96  /**
97   * Gets Logged in User info
98   * @param observe set whether or not to return the data Observable as the body, response or events.
99   * defaults to returning the body.
100  * @param reportProgress flag to report request and response progress.
101  */
102  public getWhoAmI(observe?: 'body', reportProgress?: boolean, options?:
103    {httpHeaderAccept?: 'application/json', context?: HttpContext}): Observable<WhoAmIModel>;
104  public getWhoAmI(observe?: 'response', reportProgress?: boolean, options?:
105    {httpHeaderAccept?: 'application/json', context?: HttpContext}): Observable<HttpResponse<WhoAmIModel>>;
106  public getWhoAmI(observe?: 'events', reportProgress?: boolean, options?:
107    {httpHeaderAccept?: 'application/json', context?: HttpContext}): Observable<HttpEvent<WhoAmIModel>>;
108  public getWhoAmI(observe: any = 'body', reportProgress: boolean = false, options?:
109    {httpHeaderAccept?: 'application/json', context?: HttpContext}): Observable<any> {
110
111    let localVarHeaders = this.defaultHeaders;
112
113    let localVarCredential: string | undefined;
114    // authentication (cookieAuth) required
115    localVarCredential = this.configuration.lookupCredential('cookieAuth');
116    if (localVarCredential) {
117    }
118
119    let localVarHttpHeaderAcceptSelected: string | undefined = options && options.httpHeaderAccept;
120    if (localVarHttpHeaderAcceptSelected === undefined) {
121      // to determine the Accept header
122      const httpHeaderAccepts: string[] = [
123        'application/json'
124      ];
125      localVarHttpHeaderAcceptSelected = this.configuration.selectHeaderAccept(httpHeaderAccepts);
126    }
127    if (localVarHttpHeaderAcceptSelected !== undefined) {
128      localVarHeaders = localVarHeaders.set('Accept', localVarHttpHeaderAcceptSelected);
129    }
130
131    let localVarHttpContext: HttpContext | undefined = options && options.context;
132    if (localVarHttpContext === undefined) {
133      localVarHttpContext = new HttpContext();
134    }
135
136    let responseType_: 'text' | 'json' | 'blob' = 'json';
137    if (localVarHttpHeaderAcceptSelected) {
138      if (localVarHttpHeaderAcceptSelected.startsWith('text')) {
139        responseType_ = 'text';
140      } else if (this.configuration.isJsonMime(localVarHttpHeaderAcceptSelected)) {
141        responseType_ = 'json';
142      } else {
143        responseType_ = 'blob';
144      }
145    }
146    let localVarPath = `/api/v1/whoami`;
147    return this.httpClient.request<WhoAmIModel>('get', `${this.configuration.basePath}${localVarPath}`,
148      {
149        context: localVarHttpContext,
150        responseType: <any>responseType_,
151        withCredentials: this.configuration.withCredentials,
152        headers: localVarHeaders,
153        observe: observe,
154        reportProgress: reportProgress
155      }
156    );
157  }

```

Figure 7 - Auto-generated code for frontend in Angular from example specification

Chapter 3

3. Related Work and Market Analysis

The following chapter examines related works and conducts a market analysis to identify existing products that could be utilized for this project.

3.1 Related Work

In the book "Requirements-Engineering und -Management" by Rupp et al. [17], essential practices and methodologies of requirements engineering are explored in depth. This comprehensive guide covers the critical steps of gathering, analyzing, and managing requirements within various project settings, including both agile and traditional frameworks. In the thesis, Chapters 8 (Requirement Elicitation), 11 (Deriving Good Requirements), 12 (Analyzing Requirements), 16 (Documenting and Communicating Requirements), 17 (Communicating Requirements through Storytelling and User Stories), and 18 (Modeling Requirements) from the book guided the process presented in Chapter 4, from initial requirement gathering to finalizing the specification document.

Muhammad Fazril Bin Mohd Amin's dissertation focuses on the development of a Volunteer Management System (VMS) that partially automates work scheduling and volunteer hour tracking. The current system, as described in the dissertation, involves a weekly sign-up by volunteers to confirm availability, which is then used by human resources to create duty rosters. This method is time-consuming for both volunteers and staff. The proposed VMS aims to streamline this by combining the sign-up process and roster generation into a single, user-friendly application. Due to the similarities in the objectives of developing a VMS and addressing issues in existing systems, this dissertation was reviewed to gain insights into the approaches used and the topics discussed for developing the volunteer management system [13].

The study "Online Students' Appointment System for University Administration" by Zurah Abu et al. focuses on transitioning the appointment scheduling process at Universiti Teknologi MARA from manual to an online system. This transformation

addresses inefficiencies such as delayed confirmations and poor scheduling prioritization. The study reveals a high demand for a more organized appointment process among students and staff, with a majority expressing dissatisfaction with the existing manual system. The proposed solution involved developing a web-based system that allowed for streamlined scheduling, automated reminders, and easier access to appointment statuses. Like our project, both studies concentrate on thoroughly analyzing existing challenges to create effective scheduling systems. Additionally, both works utilized the waterfall methodology and included an evaluation phase, where a Standard Usability Scale was utilized to assess the effectiveness of the solutions [2].

The work "Online Scheduling System for Doctors and Patients in a Hospital" by *De Guzman et al.* developed an online system aimed at reducing patient wait times and improving the scheduling process for outpatient services in hospitals, using web-based technologies to enable appointment bookings and doctor availability checks. This study provides valuable insights into the application of scheduling systems in healthcare settings. Both studies included an evaluation phase where the effectiveness of the systems was assessed using the Standard Usability Scale (SUS), emphasizing the importance of user feedback in the development process. While *De Guzman et al.* focus on developing an online appointment scheduling system to reduce patient wait times and improve the efficiency of the outpatient department, our project centers on developing a system for volunteer management and scheduling within nonprofits, prioritizing usability to accommodate volunteers varied technical skills. This distinction underscores the adaptability of scheduling technologies to meet diverse operational needs across different sectors [7].

One notable difference between our thesis and the works by *Muhammad Fazril Bin Mohd Amin*, *Zurah Abu et al.*, and *De Guzman et al.*, is the depth of our requirements analysis, implementation processes, and testing. In our thesis, these processes were focused on and discussed in a more comprehensive fashion. Additionally, several best practices were discussed and employed in the implementation of our system.

3.2 Market Analysis

A market analysis was conducted to determine the feasibility of using existing Customer Relationship Management (CRM) systems for the *Zahnmobil* project. Various CRM products were examined, including Zoho, which offers extensive features such as analytics, process management, contact management, and calendar appointments. However, for the specific needs of *Zahnmobil* users, these features

introduce unnecessary complexity. The primary requirements of the *Zahnmobil* project do not align with the comprehensive capabilities provided by these systems.

Additionally, calendar components from libraries like FullCalendar and DayPilot were evaluated for their functionality, which includes multiple views (month, week, day) and high customizability. Despite these advantages, the stakeholder of *Zahnmobil* expressed a preference for an improved version of the existing *Zahnmobil* application's calendar. They indicated that while an advanced calendar component could be beneficial, it is considered a non-essential enhancement and was therefore not adopted in the project.

Chapter 4

4. Requirements Analysis

This chapter discusses the requirements analysis conducted with the stakeholder of *Zahnmobil*, guided by the methodologies and principles outlined in [17]. The goal of this chapter is to create a specification that precisely describes the system's services and constraints.

4.1 Requirements Elicitation

Before conducting the requirements elicitation, research was conducted on the *Zahnmobil* project to better understand the context of the project. Subsequently, a discussion with the stakeholder was held to gain an understanding of their desires. This included a contextual inquiry to assess the old system, aiming to reuse existing requirements while identifying and avoiding the replication of previous problems. During this discussion, the product owner's approach to conceptualizing ideas was assessed—whether they thought in concrete terms or abstracted from real-life experiences to more general concepts. This understanding is essential because if the product owner tends to use real-life examples, it becomes the interviewer's task to abstract these into broader requirements and validate them. Conversely, if the product owner speaks in abstract terms, introducing concrete examples is necessary to clarify and confirm the requirements. In this case, the product owner demonstrated the ability to strike a balance between abstraction and concrete examples, providing each as needed. Given the product owner's availability, an interview-based elicitation approach was selected for this thesis.

4.2 Interview

The interview process is divided into three stages: preparation, execution, and follow-up. During the preparation phase, an interview protocol is developed to guide the interview (complete interview protocol in USB Content). This protocol is essential for distinguishing between critical and non-critical topics and providing a structured

approach. It includes bullet points that remind the interviewer of key points, such as summarizing each topic before jumping into a new one with clear examples for mutual understanding, maintaining a focus on priority requirements, and asking thorough questions to eliminate any ambiguities. The interview protocol included questions about functional and non-functional requirements, user interface preferences, technological preferences, and the data managed by the existing system. Understanding this data is important for later developing an optimization algorithm for scheduling. With the interview protocol in place, a semi-structured interview was conducted with the stakeholder, for which recording consent was also obtained. The recorded interview enabled the capture of detailed responses that might have been overlooked during note-taking. This ensured a comprehensive documentation of the interview. After the interview, the responses were reviewed and validated against the notes taken during the interview and the recording. The answers were refined as needed to ensure clarity. The finalized interview protocol was then promptly sent to the stakeholder within two days for approval, ensuring that the information remained current and accurately reflected in the project details. With the initial requirements defined, the Kano Model was utilized to prioritize and further discuss the requirements with the stakeholder. Furthermore, the process of analyzing the requirements was conducted and is detailed in Chapter 2.10 of this thesis, which includes relevant examples.

4.3 Specification

Documenting requirements is crucial for maintaining an understanding of project requirements over time. For documenting these requirements, a traditional requirements specification was selected due to familiarity and comfort with this method. To minimize errors in the layout and content of the documentation, the template from the *Software Engineering Group at Leibniz University Hannover* was utilized. The following subsections discuss key parts of the developed specification. The full specification is included in Appendix A and USB Content.

4.3.1 Goal Definition

The specification outlines the project's goal to develop a web application named *Zahnmobil*. This application is designed to assist volunteers in coordinating their operations and to enable administrators to oversee the project more effectively. Users will be able to log in, manage their profiles, and enroll in operations. The application will allow administrators to create and modify operations. Each operation requires three users, each with distinct roles: *dentist*, *driver*, and *assistant*. Additionally,

administrators will have the capabilities to add or edit operation locations and manage user profiles. The application will also be equipped to send various types of confirmations or invitations via email. Overall, *Zahnmobil* is designed to enhance operational efficiency by focusing on usability as a primary goal, offering an intuitive and user-friendly interface that simplifies work processes.

4.3.2 Functional Requirements

The functional requirements outlined below, based on the template provided in work [16], detail the essential capabilities of the application to be developed.

[FR1] The system shall provide users the ability to log in with verified credentials.

[FR2] The system shall provide users the ability to reset their password using a link sent to their email.

[FR3] The system shall provide users the ability to edit their personal information after login.

[FR4] The system shall provide administrator the ability to add new operation locations to the system.

[FR5] The system shall provide administrator the ability to add new operations to the system.

[FR6] The system shall provide users with specific roles the ability to sign up for operations and receive confirmation via email and calendar entry.

[FR7] The system shall provide administrator the ability to assign users to specific roles in operations.

[FR8] The system shall provide the ability to automatically email users to sign up for unfilled roles one week before the operation.

[FR9] The system will provide administrator the ability to send emails to selected users from the central administration.

Functional requirements in the specification are defined using use cases, which provide a structured way to identify and refine the requirements of a system by mapping out the individual steps needed to perform each use case. Figure 8 illustrates the use case for [FR6].

Use Case 6	Der User trägt sich für einen Einsatz ein.
Erläuterung	Der User mit der Rolle X (X = Fahrer, Zahnarzt, Zahnarzthelferin) möchte sich für einen Einsatz in der spezifischen Rolle X eintragen.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Der User muss eingeloggt sein und die Rolle X muss für den Einsatz frei sein.
Mindestgarantie	Der User mit der Rolle X kann sich nur für Einsätze in dieser spezifischen Rolle eintragen.
Erfolgsgarantie	Bei erfolgreicher Durchführung wird der User in dem Einsatz eingetragen. Der User bekommt auch ein Hinweis („Erfolgreich eingetragen“) und eine E-Mail mit einem Kalendereintrag.
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Mitarbeiter von Zahnmobil
Auslöser	Der User möchte sich für einen freien Einsatz eintragen.
Hauptzenario	<ol style="list-style-type: none"> 1. Der User besucht die Kalenderseite 2. Der User mit der Rolle X klickt auf den „Sich eintragen“-Button von einem Einsatz. 3. Das System speichert den User für die Rolle X in diesem Einsatz 4. System zeigt Hinweistext: „Erfolgreich eingetragen“
Erweiterungen	<p>2- Sich eintragen wird nur gezeigt, wenn der User die Rolle dafür hat</p> <p>4- Der User kann sich auch durch Klicken auf den „Sich austragen“-Button von einem Einsatz austragen. Dieser Button wird dem User angezeigt, sobald er für den Einsatz eingetragen ist. Der User und</p>
Priorität	unverzichtbar
Verwendungshäufigkeit	Gelegentlich

Figure 8 - Use case for [FR6]

4.3.3 Non-Functional Requirements

The non-functional requirements of the application were discussed with the stakeholder and prioritized as follows:

1. Usability
2. Maintainability
3. Portability

The most important non-functional requirement for this project is usability. The user interface should enhance workflows with its intuitive and user-friendly design, catering to users of all ages and varying levels of IT expertise. Furthermore, the application must be well-organized to facilitate easy updates and expansions, which will help minimize errors and adapt quickly to new requirements. Additionally, if time permits,

the application should be accessible on mobile devices, requiring a responsive design for optimal display across different devices. To achieve these quality goals:

- Usability is ensured through the development of mock-ups and continuous customer feedback. Human-centered design principles are applied, along with end-user survey to refine the application.
- Maintainability is achieved by ensuring high cohesion and low coupling in the codebase, adhering to best practices for a well-structured and clearly defined codebase.
- Portability involves implementing a responsive design to make the application functional on various mobile devices. This includes testing the design across multiple dimensions.

4.3.4 Mockups

The design for the high-fidelity mockups was created by drawing insights from the current official *Zahnmobil* website. This approach ensured alignment with the organization's established color scheme and design style. The choice of primary and secondary colors, as well as those indicating success and errors, was made with a focus on branding consistency, aesthetics, and readability. An overall goal was to accommodate the varied IT expertise of the end users, including many older individuals, by developing an intuitive and user-friendly interface. The layout emphasizes visual hierarchy, using distinct colors and font sizes to clarify different levels of information. The homepage is organized into well-defined areas, helping users quickly identify where to focus their attention. Furthermore, some previous issues with the old application included the need for unnecessary clicks for certain tasks, such as creating an operation. The new design simplifies interactions by reducing the number of clicks needed. Subsequently, dialogs were used to help users focus on subtasks when needed, reducing cognitive load. A digital calendar was designed as a metaphor for a physical one, ensuring that online user actions mirror those performed with a physical calendar.

Willkommen Herr Max Mustermann (Zahnarzt)

Gesamtanzahl in diesem Jahr behandelte Einsätze: 10

Gesamtanzahl im vergangenen behandelte Einsätze: 20

meine nächsten 5 Einsätze:

02.03.2024 - 10:30-11:00 Uhr - Kontaktladen Mecki (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr

KALENDER

kommende offene Termine

Alle Fahrer Zahnarzt Helfer/in

Fahrer

02.03.2024 - 10:30-11:00 Uhr - Kontaktladen Mecki (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr

KALENDER

Zahnarzt

02.03.2024 - 10:30-11:00 Uhr - Kontaktladen Mecki (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr

KALENDER

Helfer/in

02.03.2024 - 10:30-11:00 Uhr - Kontaktladen Mecki (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr

KALENDER

kommende 5 Veranstaltungen:

01.04.2024 - 0:00 - 0:00 Uhr - Urlaub - Kein Einsatz

KALENDER

01.04.2024 - 0:00 - 0:00 Uhr - Urlaub - Kein Einsatz

KALENDER

Figure 9 - Mockup for the homepage

Kalender

<< 2023

2025 >>

JAN FEB MAR APR MAI JUN JUL AUG SEP OKT NOV DEZ

KW13

Montag KW 13
01.04.2024
0:00-0:00 Uhr

Sonstige: Sommerfest

Montag KW 13
01.04.2024
0:00-0:00 Uhr

Einsatz: Kontaktladen Mecki
(vormittags)
Raschplatz 8c

Zahnarzt: frei
Helfer: Max Mustermann
Fahrer: eintragen

KW14

Montag KW 13
01.04.2024
0:00-0:00 Uhr

Einsatz: Kontaktladen Mecki
(vormittags)
Raschplatz 8c

Zahnarzt: frei
Helfer: Max Mustermann
Fahrer: eintragen

Figure 10 - Mockup for the calendar page

4.3.5 Compromises

Due to time constraints, it's crucial to discuss potential compromises with the stakeholder and clearly outline them in the specification. The following requirements have been identified as possible compromises and will only be implemented if time permits:

- Exporting a calendar/list of users in PDF format
- Responsive design
- Mailing
- User profile data (excluding essential information like last name, first name, and role)

Chapter 5

5. Implementation

Chapter 5 outlines the implementation of the application. It starts by explaining the selected approach and then explores the application's architecture. Rather than discussing specific technical details, the chapter emphasizes the best practices and critical decisions that shaped the implementation. Screenshots of the developed application are available in Appendix C.

5.1 Approach

A hybrid software development life cycle was adopted to develop the application, combining elements from Waterfall, Kanban, and Agile methodologies to leverage their strengths. The implementation was structured around the Waterfall model's linear and sequential phases, as outlined in Chapter 2.3. This approach ensured simplicity and clarity, with each stage being completed fully before proceeding to the next and testing following the completion of development. To manage the flow of tasks and limit work-in progress effectively, a digital Kanban board was utilized, as detailed in Figure 2. The Kanban board simplified the visual management of tasks and helped maintain focus on current priorities. Furthermore, the Agile methodology's iterative approach was integrated to maintain continuous alignment with the customer's vision. This involved splitting the development into iterations, which allowed for regular customer feedback and facilitated ongoing analysis and improvement. Weekly meetings were scheduled throughout the development phase to ensure consistent communication and adaptation to evolving project needs.

5.2 Technologies

The web application uses Spring Boot for backend operations and MySQL as the database. Angular provides the frontend interface, while OpenAPI is used to efficiently generate code, including server stubs for API implementation and the creation of Data Transfer Objects. Additionally, OpenAPI aids in documenting the API. Maven, which is an open-source, standards-based project management framework that simplifies the

building, testing, reporting, and packaging of projects, is utilized to manage package handling and project organization [21]. The source code is hosted on Atlassian Bitbucket, provided by adesso.

5.3 Architecture

The application architecture consists of a backend built with Spring, which connects to a MySQL database for data exchange. Communication between this backend and the Angular frontend is facilitated by OpenAPI, which generates the necessary skeletons for API implementations. Figure 11 depicts the design of the application architecture.

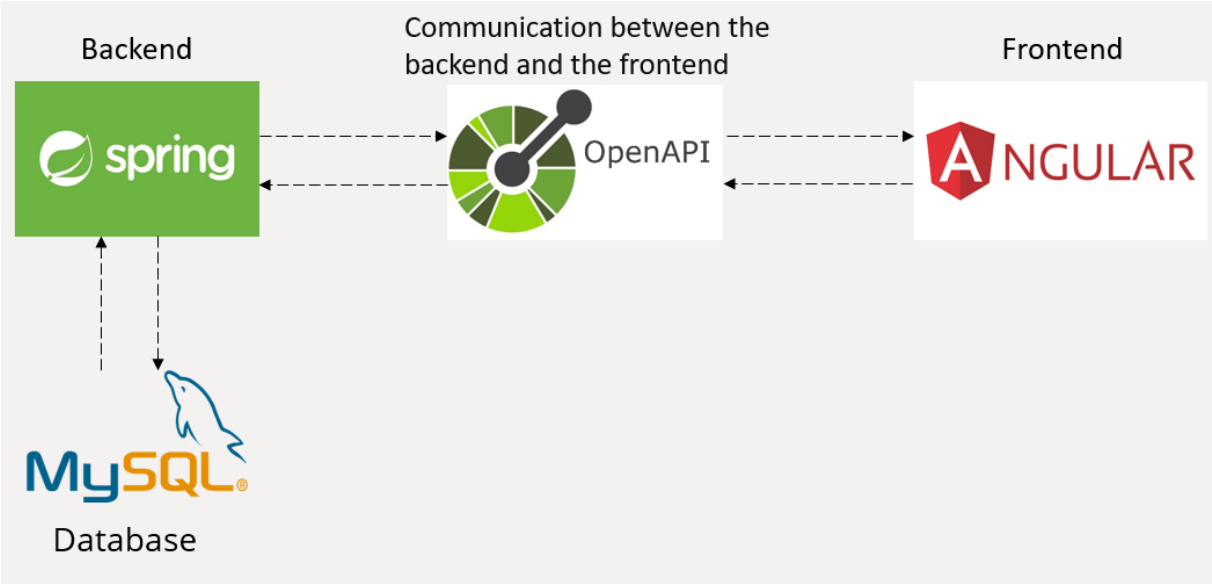


Figure 11 - Application architecture design

An architectural domain design was created to provide a clear overview and guide the development of the system. Different colors distinguished various stereotypes, including classes, enums, interfaces, and entities. Each entity is clearly defined with its attributes, specifying their types and visibility modifiers (public or private). The entities are supported by Java Persistence API (JPA) Repository Interfaces, which persist the entities and handle their Object-Relational Mapping. Essential methods for these interfaces have also been included. The system uses a microservice architecture, which divides the software into small, independent services. This structure enhances maintainability and scalability, with each entity handled by its dedicated service. Relationships between entities are defined by cardinalities, which assist in selecting appropriate Hibernate annotations in Spring Boot for defining entity relationships.

Data Transfer Objects (DTOs) are designed to specify the precise data for communication of backend and frontend. The domain design was refined following a consultation with a senior software architect at adesso. Recommendations regarding the early creation of DTO objects, adherence to naming conventions, and corrections of type errors were integrated into the design. Figure 12 illustrates a segment of the complete domain design. The full domain design is available in Appendix B and USB Content.

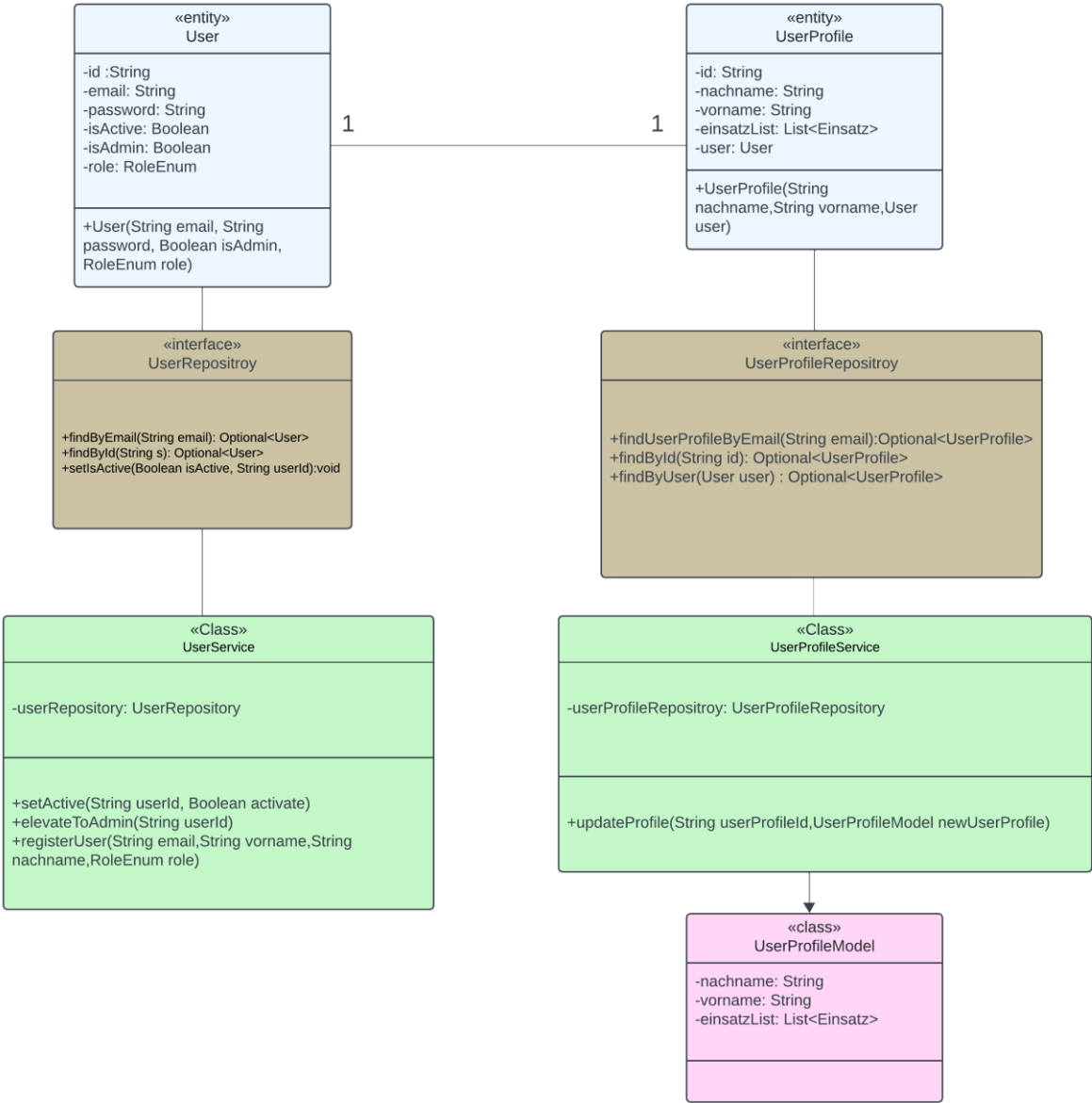


Figure 12 - Segment of domain design

5.4 Configuration

The application implementation begins by establishing a connection between the frontend and backend, as illustrated in Figure 11 application architecture design. The application's structure and dependencies are managed using Maven. Once the structure is in place, initial configurations are made for both Angular and Spring. For Spring, this includes setting up Spring Security for authentication and Cross-Origin Resource Sharing (CORS) and establishing a connection to the MySQL database. Angular requires minimal initial configuration.

5.5 Best Practices

Several best practices were deployed in the development of the application, which includes:

1. **Secure Authentication:** Utilizing Spring Security to ensure a secure authentication process.
2. **Microservice Architecture:** Enhancing modularity by creating distinct services for each entity with low coupling and high cohesion.
3. **Data Transfer Optimization:** Mapping entities to Data Transfer Objects (DTOs) to ensure only necessary information is sent to the frontend.
4. **Error Handling:** Integrating comprehensive error handling during database access to improve system robustness.
5. **Contract-First API Development:** This approach was adopted to improve efficiency in the API development process. Chapter 2.12 details its benefits, including detailed documentation of APIs, the facilitation of parallel frontend and backend API development, and saving time. The flexibility of this method also allows for easy adjustments throughout the development cycle
6. **Component Reusability:** Designing reusable frontend components to save development time and maintain consistency across the application. For instance, a green field component, which signals a successful process, is designed once, and reused throughout the application, with adjustments made only to the text as necessary.

7. **README Guide:** The README file accompanying the project provides a guide that outlines the essential technologies used, setup instructions, and testing protocols, serving as a manual for future developers to understand and contribute to the *Zahnmobil* application.
8. **UI Performance Optimization:** Angular's OnPush change detection strategy was implemented to enhance UI responsiveness and performance by manually managing update trigger. Figure 13 illustrates a simplified example of using OnPush in a component. The change detection strategy is configured at line 23, switching from Default to OnPush. This adjustment means any UI updates need to be triggered manually. Line 28 injects the base class that provides change detection functionality, enabling the developer to initiate change detection processes manually. Later in the code, at line 34, a request is made to the backend to fetch the current user count. Once the data is received and the user count is set at line 36, the developer uses the injected class to trigger a change detection at line 37. This instructs Angular to re-evaluate the component state and update the UI if necessary, reflecting the new user count.

```

19 @Component({
20   selector: 'app-team',
21   templateUrl: './team.component.html',
22   styleUrls: ['./team.component.scss'],
23   changeDetection: ChangeDetectionStrategy.OnPush, // Changing from Default to OnPush ChangeDetectionStrategy
24 })
25 export class TeamComponent {
26   constructor(
27     private userService: UserService,
28     private cdf: ChangeDetectorRef // Base class that provides change detection functionality
29   ) {}
30
31   userCount!: number;
32
33   ngOnInit() {
34     this.userService.getCountOfUsers().subscribe({ // HTTP request to backend for getting user count
35       next: (x) => {
36         this.userCount = x; // updating user count
37         this.cdf.markForCheck(); // Schedules the component for a change detection check.
38       },
39       error(err) {
40         console.log(err);
41       },
42     });
43   }

```

Figure 13 - Simplified example of OnPush usage in the application

5.6 Scheduling Optimization Algorithm

An algorithm has been developed to assist administrators in finding the best location for operations based on which location has had the most patients and additional

services handled in the past. The stakeholder provided the *Zahnmobil* project's database detailing operations from the past twelve years. This database, formatted as an Excel file, includes data on operation locations, dates, patient counts, and additional services counts. "Additional services" encompass all activities other than operating on a patient, such as providing recommendations to patients. The initial step involved cleaning and restructuring the database to facilitate smoother integration into the application. This process included eliminating irrelevant columns and rows, merging cells for consistency, and dividing the data into separate files of the last five years. Prior to that, no additional services were recorded. Following database cleaning, a parser was developed to transfer the operational data from the Excel file into the application database. With all the operations now available in the application database, an aggregation and filtering algorithm was designed, as detailed in the steps below:

1. Map *Operation* Entities to *OperationMapped* Classes:

- Each operation's date is mapped to the corresponding day of the week and month number.
- A value is calculated for each operation by combining the number of patients with one-tenth of the additional services (this ratio is specified by the stakeholder).

2. Group *OperationMapped* Classes by Location:

- Operations are grouped by location using a HashMap, where each key represents a location, and the value is a list of *OperationMapped* classes.

3. Collect Admin Input:

- Inputs include the desired date for an operation and a list of locations.

4. Filter Data for each Location:

- The data is filtered to include only entries that match the day of the week and month specified in the administrator's desired date input.

5. Calculate Average value for Each Location:

- An average value is calculated from the filtered dataset for each location.

6. Recommend the Location with the Maximum Average Value:

- The location with the maximum average value is recommended as the optimal choice.

Figure 14 depicts a usage scenario involving an administrator who wants to schedule an operation on May 15, 2024, but does not know which location to choose from among Steintor, Kröpcke, or Messe. The administrator wants the operation to yield the highest

value, considering the number of patients and the number of additional services provided. The algorithm would recommend the location expected to have the most value for the selected date. In this case, the location recommended is Messe.



Figure 14 - Usage scenario for the optimization algorithm

Chapter 6

6. Testing and Evaluation

This chapter explores the testing and evaluation of the application developed in this thesis.

6.1 Testing

Following the development of the applications, the testing phase was initiated to verify the quality of the application. This phase involved creating automated unit tests to evaluate the core functionalities of both the backend and frontend. The Spring framework uses Mockito, a powerful open-source testing framework for Java, to create mock objects, stub method calls, and verify interactions between objects for unit testing [3]. Similarly, the Angular framework employs Jasmine, a testing framework for JavaScript, to implement unit tests [9]. Additionally, automated end-to-end tests were conducted to ensure that the system functions correctly as a whole. The most critical usage scenarios within the applications were tested using Cypress, an end-to-end test automation framework built and engineered for modern web applications [14]. Following this, exploratory testing was performed throughout the entire application to identify and document current issues. For each section of the application, a charter was documented detailing the objectives, scope, and duration of the exploratory testing. Each identified issue was prioritized based on its importance and impact on the system's functionality. Due to time constraints, only the most critical problems were addressed, though remaining issues will be resolved prior to deploying the application in production mode. The complete exploratory testing documentation and a video showcasing the automated end-to-end tests are available in the USB Content.

6.2 Evaluation

This chapter describes the conduct of a study aimed at evaluating the application developed as part of this thesis.

6.2.1 Study design

The study is designed to evaluate the usability and user experience of the application, with a primary focus on its usability aspects, as it was the central goal of the application. It also aims to collect suggestions for improvements to enhance the application further. A semi-structured interview was designed to gather feedback, aiming to keep the duration between 25 to 30 minutes to respect participants' time and avoid overburdening them. The interview begins with an introduction to the study's objectives, incorporating a brief warm-up conversation to familiarize participants with the study's conduct and purpose.

Participants then engage in practical tasks that simulate real-life interactions with the application, differentiated for normal users and administrators. For instance, A typical user might search for available operations and enroll in it, while an administrator could set up a location for an operation. The tasks are designed to be comprehensive and cover core functionalities of the application for both users and administrators. Upon completing these tasks, participants fill out a survey developed using the Goal-Question-Metric (GQM) approach. Furthermore, for a more structured approach to conducting the study, a study guide was created detailing the structure of the study, the GQM framework, and the survey questions (Full Survey Guide in USB Content). Table 1-3 depicts the formulation of the goals according to the template in work [24].

Analyze	The responses from the survey
for the purpose of	Evaluating the web applications usability
with respect to	Efficiency, effectiveness and ease of use
point of view	Application's end-users
in the context of	Everyday tasks within the <i>Zahnmobil</i> application

Table 1 - Usability goal definition

Analyze	The responses from the survey
for the purpose of	Evaluating the web applications user experience
with respect to	Aesthetic, navigational clarity and user hurdles
point of view	Application's end-users
in the context of	Everyday tasks within the <i>Zahnmobil</i> application

Table 2 - User experience goal definition

Analyze	The responses from the survey
for the purpose of	Identifying potential new features and improvements
with respect to	user satisfaction
point of view	Application's end-users

in the context of	Everyday tasks within the <i>Zahnmobil</i> application
--------------------------	--

Table 3 - Feature discovery goal definition

Subsequently, questions were designed to support the goals, particularly focusing on the application's usability. The Standard Usability Scale (SUS) was employed to evaluate the usability of the application, providing reliable, easy-to-analyze results that facilitate comparisons with similar products [8]. The survey also included questions about the application's core features for both users and administrators to specifically assess their usability. To avoid overburdening participants, only three questions were included addressing user experience aspects such as design, navigation intuitiveness, and challenges encountered during application use. An open-ended question was also added to gather suggestions for desired features or improvements. The survey used a Likert scale ranging from 1 to 5, where 1 indicates 'strongly disagree' and 5 'strongly agree'. A 5-point scale was preferred over a 7-point scale to simplify participant decision-making. A neutral option, rated as 3, allowed participants to express neither a strongly positive nor negative view on specific application aspects. Finally, with the goals and questions formulated, the metrics for assessing the results of the questions were finalized. For the SUS-related questions, the SUS score for each participant was computed, along with the average SUS score across all participants, to assess overall usability. For other questions targeting specific usability features and user experience, the median score was analyzed, and the percentage of agreement or disagreement among responses was calculated. For the open-ended question regarding suggested features, the frequency of specific feature requests was quantified to identify prominent user demands. Furthermore, separate online survey forms were created for administrators and users, each tailored to their specific interactions with the application. These surveys were accompanied by a consent form that outlines the purpose of the study. It assures participants of confidentiality and voluntary participation. By checking a box, participants give their informed consent, acknowledging they understand the survey's details (Full survey in USB Content).

6.2.2 Participants

The recruitment process for the study was initiated by the stakeholder of *Zahnmobil*, who invited all members to partake in the interview. Interested individuals were given the choice to conduct their interviews either online or on-site, with a preference stated for on-site interviews to benefit from smoother procedures and direct communication. Interviews were scheduled based on each participant's preference.

6.2.3 Study Results

A total of 12 participants were interviewed, consisting of 7 dentists, 3 drivers and 2 assistants. Among them, 9 were regular users and 3 were administrators. For the 9 regular users, 3 interviews were conducted online, and all other interviews took place at locations preferred by the participants, either at their homes or in public spaces. Figure 15 illustrates the demographics: there were 8 males and 4 females, with an average age of 60 years. The ages ranged from 29 to 70.

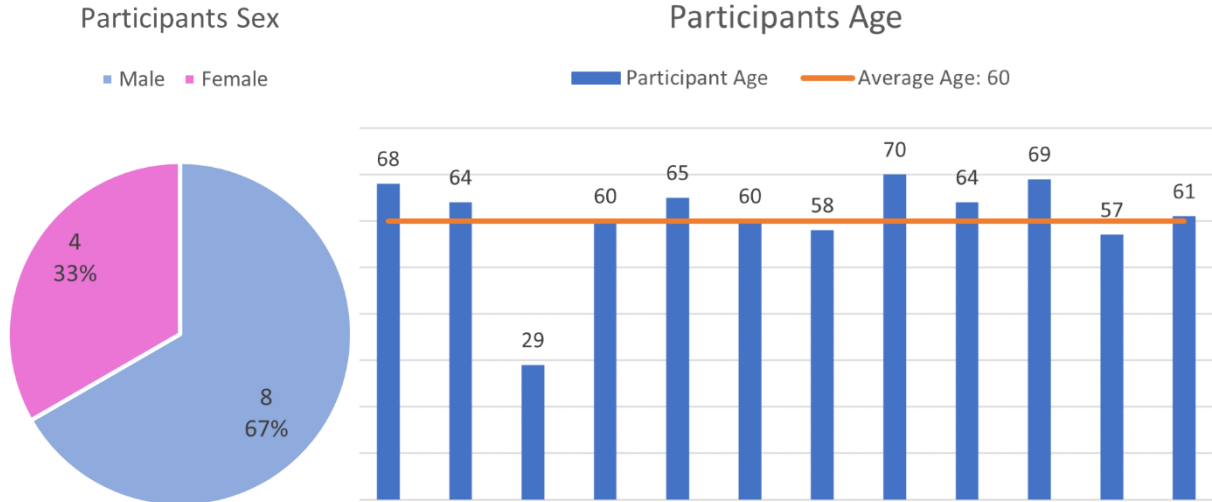


Figure 15 - Participant demographic charts

The SUS-Score for each participant was calculated and analyzed based on the methodology outlined in work [8]. Every participant received an A+ grade, which is the highest possible. The average SUS-Score was 96.67, with 3 users and 1 administrator giving the application a perfect score of 100. The lowest score recorded was 87.5, which still qualifies as an A+ grade.

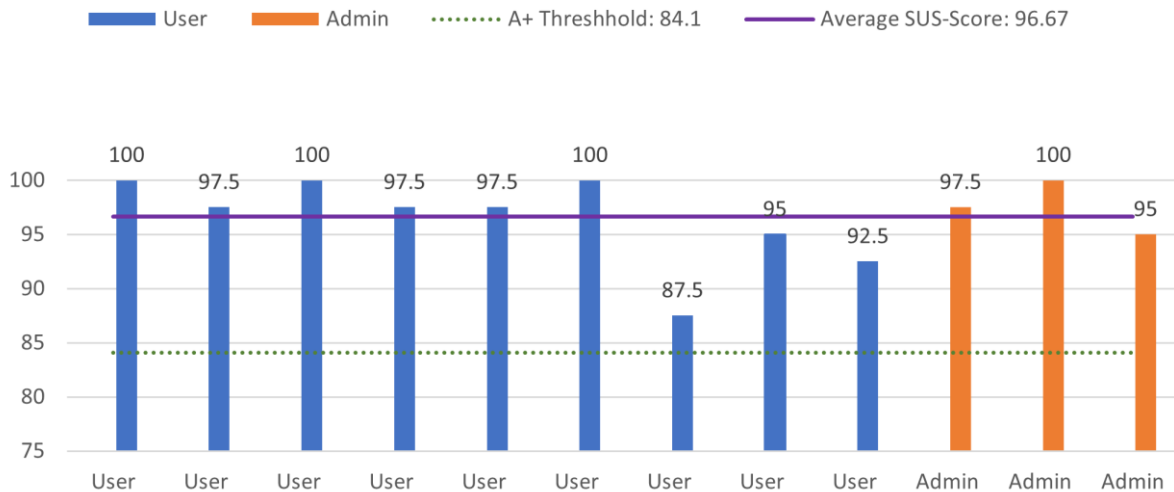


Figure 16 - SUS Score for each participant and the overall average

In analyzing usability-related questions about specific features, such as searching for operations in the calendar, understanding emails from the system, and navigating the application, the responses from participants reflect a uniformly high level of satisfaction with the application’s functionality. Positively phrased questions about the features received a median score of 5, signaling strong agreement with the application's effectiveness. Conversely, negatively formulated questions resulted in a median score of 1, showing strong disagreement with negative statements about the application.

The responses from the administrators regarding administrator-specific features also indicated uniformly high satisfaction levels, except for one question concerning the usefulness of the scheduling optimization algorithm, as discussed in Chapter 5.6. Here, two administrators noted that due to the fixed nature of recurring operations at the same locations, the algorithm's utility was limited. However, one administrator believed it could become beneficial as the organization grows.

Another point of discussion was the attractiveness of the user interface design. While the majority (67%) strongly agreed that the UI design was attractive, there was a notable dissent, with one participant neutral, two disagreeing, and one strongly disagreeing, totaling 25% in disagreement.

Table 4 summarizes the feedback and observations collected from the survey. The first column lists how many participants mentioned each issue, and the second column details the specific recommendations or observations.

#Participants	Recommendations and observations
5	Bigger text size
4	Responsive design

3	Confusion between the actions and states of the 'Eintragen' and 'Austragen' buttons
3	Clicking the buttons more than once due to not recognizing the successful feedback that appeared
3	Integration of operation in external calendars
1	A reminder E-Mail from system for reminding users of signed up operations
1	Sorting of operation locations alphabetically

Table 4 - Recommendations and Observations gathered from the survey

- The most common issue, reported by five participants (41.7%), was difficulty reading the application's text due to its small size.
- Four participants (33.3%) inquired about the application's appearance on mobile phones, noting that they often use the app on their devices.
- Another common confusion involved the button used for enrolling in operations, with three participants (25%) unsure whether it indicated an action or a state. Two participants later clarified that their confusion stemmed from the button's text size.
- Three participants (25%) did not immediately recognize the feedback after confirming an action, leading to errors such as adding the same operation twice or mistakenly enrolling and deregistering from an operation.
- Three participants (25%) appreciated the option to add operations to an external calendar, finding it helpful for tracking their enrolled operations.
- One participant (8.3%) suggested sending a reminder email one day before the operation to help users keep track of their schedules.
- One administrator out of the 3 administrator (33.3%) suggested sorting operation locations alphabetically to improve searchability.

Two recommendations from participants—responsive design and the ability to integrate operations into external calendars—were initially identified as requirements by stakeholders. However, due to time constraints, these features were listed as possible compromises as mentioned in Chapter 4.3.5 and were not implemented in this thesis.

6.2.4 Threats to Validity and Challenges

To mitigate external validity concerns, the study specifically targeted end-users who were already familiar with the current *Zahnmobil* application. This focus aimed to ensure that feedback was relevant and grounded in actual user experiences. However, this approach may also limit the generalizability of the results, as new users completely unfamiliar with the current *Zahnmobil* applications might have different responses and insights. Many user functionalities of the current *Zahnmobil* application were integrated to the developed application with improvements, making it easier for users familiar with the old system to adapt and find it less complex. The new features and changes primarily catered to administrative functions, and since 9 of the 12 survey participants were regular users, this may have impacted the results.

For internal validity, consistency was maintained through a semi-structured interview format, where each participant received the same introduction and completed similar tasks tailored to their role as a user or administrator. During the practical task section of the interview, minimal assistance was provided to replicate a real-life usage scenario and observe any difficulties encountered, ensuring feedback reflected typical application use. Despite these measures, several issues could still impact the study's internal validity:

1. **Remote Task Execution:** Initially, the plan for the online survey allowed the three participants who wished to conduct the survey online to remotely control the shared screen and perform the tasks. However, due to the participants' varying levels of IT expertise, this approach was modified. Instead, the application and tasks were demonstrated in detail, and participants were periodically asked where they would click, effectively simulating a usage scenario. This change could influence participants' feedback, as they did not engage directly with the application to perform the tasks, potentially affecting the internal validity of the study.
2. **Question Format:** The survey's use of alternating positive and negative questions confused some participants, leading to occasional incorrect responses despite clarifications provided during the interview.

Furthermore, to ensure strong construct validity, the study employed several key strategies:

1. **Standard Usability Scale (SUS):** The SUS was used to evaluate the application's usability. This widely recognized and validated tool is known for its excellent reliability, enhancing the accuracy of the usability assessment.
2. **Task-Based Evaluation:** Participants engaged in practical tasks simulating real-life interactions with the application. Tasks were tailored for both regular users and administrators, ensuring that the evaluation was relevant and comprehensive .
3. **GQM Approach:** The survey was developed using the Goal-Question-Metric (GQM) approach, providing a structured and systematic framework. This approach ensured that survey questions were directly linked to the study's goals, thereby improving construct validity by collecting relevant data.

6.2.5 Summarizing Results

The study achieved an impressive average SUS score of 96.67, indicating high satisfaction among the 12 participants with the application's usability. This suggests that the *Zahnmobil* Project could effectively utilize this application for their purposes [8]. However, the results also identified areas for improvement, such as increasing text size and preventing multiple buttons clicks by users. Additionally, the study highlighted the most desired features among the participants, which include responsive design, integration with external calendars, and email reminders for scheduled operations.

Despite these positives, 25% of participants found the application's design unattractive. This suggests that there is room for aesthetic enhancement. Moreover, the feedback revealed that the optimization algorithm, which selects locations with the highest number of patients based on operation history, does not align with *Zahnmobil* project's goal in choosing locations. However, according to the stakeholder of *Zahnmobil*, as operations expand, this algorithm might become more applicable.

Chapter 7

7. Conclusion

This thesis has effectively navigated the complexities involved in developing a software application tailored for the *Zahnmobil* Project. The primary objective was to create a user-friendly, efficient software system that enhances operational efficiency and facilitates the management of volunteer-driven dental services. Throughout this project, a comprehensive analysis of requirements was conducted to ensure that the developed software met the specific needs of the *Zahnmobil* Project. The architectural design aimed at creating a robust and maintainable framework was implemented successfully. The implementation phase adhered closely to several best practices in software development, ensuring the system was not only functional but also scalable and secure. A notable strategy was the adoption of a Contract-First API development approach using OpenAPI, which facilitated parallel development streams, enhanced maintainability, and significantly reduced development time. The testing phase highlighted the application's reliability and performance through systematic automated and exploratory testing. Subsequently, the evaluation phase, conducted through user interviews, provided critical insights into the application's usability, confirming high satisfaction levels among users as evidenced by the average SUS score of 96.67 among 12 participants. Feedback from this phase also pinpointed areas for future enhancement. In conclusion, this thesis not only achieved its goal of improving volunteer coordination and operational efficiency but also laid a solid foundation for future technological enhancements within the *Zahnmobil* Project.

Chapter 8

8. Future Development

This chapter explores the potential directions for the future development of the application.

8.1 AI Integration

In the future development of this project, one innovative idea is incorporating an Artificial Intelligence Scheduler to optimize operation scheduling. This AI Scheduler would align operations with user availability. Administrators would specify the dates for operations scheduled in the upcoming month, while users would input their available time slots for that same period. The AI Scheduler would then match users to operations, ensuring each operation has a dentist, driver, and assistant. This setup is a classic example of a Constraint Satisfaction Problem (CSP), which seeks solutions that satisfy a specific set of constraints [5]. In this context, the variables represent the roles for each operation, with the domain of these variables being the entire set of users in the system.

Key constraints for this system include:

1. Each operation requires one and only one dentist, driver, and assistant.
2. The scheduling of operations must align with user availability.
3. Users cannot be assigned multiple roles simultaneously.
4. Users cannot be scheduled for overlapping operations.

Additionally, the CSP framework allows for optimizing initial solutions to achieve the most effective solution. For example, an objective function could be introduced to distribute the workload evenly among users, preventing burnout by avoiding over-scheduling users in too many operations. Additionally, another objective function could aim to minimize staff changes. By keeping staff assignments consistent, especially in operations that recur at the same locations and times, we can enhance comfort and potentially improve performance. To effectively implement the AI Scheduler, the system's database should be updated to save user availability. Additionally, the user

interface for inputting user availability should be designed to be user-friendly for those with limited IT skills. A calendar component would simplify the process, enabling users to easily indicate their availability by selecting days and specifying available times. However, challenges such as user adoption and flexibility in handling unforeseen changes need careful consideration. Encouraging regular updates of user availability and accommodating emergencies are critical for the system's effectiveness.

Another proposal for improving the application is introducing a recommendation system to enhance the user experience. This system would offer personalized operation recommendations based on users' historical interactions with the application, diverging from the current system that only suggests upcoming operations on the homepage. A machine learning technique utilizing collaborative item-based filtering would be appropriate for this enhancement. The backend already stores all the relevant information of operations. For the technical implementation, the Apache Mahout framework could be utilized. Mahout supports a collaborative filtering approach using an ItemBasedRecommender and various measures to compute item similarities, such as Pearson correlation, cosine similarity, Jaccard coefficient, and log-likelihood ratio. In our scenario, the items are the operations, and similarities can be assessed based on aspects including day of the week, month, time frame of deployment, and operation location. These similarities can be precomputed and stored, enabling efficient data retrieval from a relational database, and facilitating scalable and effective machine learning deployment [18].

8.2 Gamification

To improve operations and enable scalability for the *Zahnmobil* Project, a motivational strategy involving gamification could be introduced. This approach would include a point system where employees earn points for completing operations, with additional points awarded for consistent participation, such as weekly or monthly streaks. A leaderboard could display the top five employees in each role, showcasing those with the highest points (see Figure 18 for a mockup). Employees who prefer privacy can choose to hide their standings on the leaderboard. Additionally, a personalized homepage could feature a leveling system that reflects the points employees have accumulated. Although this concept received positive feedback from stakeholders, it was not implemented in the bachelor thesis due to time constraints.






	Zahnarzt	Fahrer	Helfer/in
	Max Mustermann	Max Mustermann	Max Mustermann
	Max Mustermann	Max Mustermann	Max Mustermann
	Max Mustermann	Max Mustermann	Max Mustermann

Figure 17 - Leaderboard page mockup

8.3 Roadmap

The primary goal of the roadmap is to prepare the application for deployment, prioritizing the following key steps:

1. **Implementing Compromised Requirements:** Implementing the compromised Requirements such as Responsive Design and Export functionalities.
2. **Incorporating User Feedback:** Implement changes based on survey result (see Chapter 6.2.5), such as increasing text size.
3. **Bug Fixes:** Address issues identified during exploratory testing.

4. **Enhanced Automated Testing:** Develop comprehensive automated tests for all features.
5. **Integration and Server Deployment:** The final steps involve deploying the application on a server to provide user access and integrating the application with the *Zahnmobil* website to ensure that operations created internally are visible externally.

References

- [1] A. Haraty, R. and Hu, G. 2018. Software process models: a review and analysis. *IJET* 7, 2.29, 325.
- [2] Abu, Z., Jasmisham, N. H., and Mangshor, N. N. A. 2022. ONLINE STUDENTS' APPOINTMENT SYSTEM FOR UNIVERSITY ADMINISTRATION. *Journal of Islamic* 7, 46.
- [3] Acharya, S. 2014. *Mastering unit testing using Mockito and JUnit*. Packt Publishing Ltd.
- [4] Balaji, S. and Murugaiyan, M. S. 2012. Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. *International Journal of Information Technology and Business Management* 2, 1, 26–30.
- [5] Brailsford, S. C., Potts, C. N., and Smith, B. M. 1999. Constraint satisfaction problems: Algorithms and applications. *European journal of operational research* 119, 3, 557–581.
- [6] Damij, N. and Damij, T. 2024. An Approach to Optimizing Kanban Board Workflow and Shortening the Project Management Plan. *IEEE Trans. Eng. Manage.*, 1–8.
- [7] Guzman, M. R. Q. de, Ordoñez, J. L. N., Somocierra, R. O., and Fuentes, G. S. 2021. Online Scheduling System for Doctors and Patients in a Hospital. In *Proceedings of the International Conference on Industrial Engineering and Operations Management, Monterrey, Mexico*.
- [8] Klug, B. 2017. An overview of the system usability scale in library website and system usability testing. *Weave: Journal of Library User Experience* 1, 6.
- [9] Kozłowski, P. and Darwin, P. B. 2013. *Mastering Web Application Development with AngularJS*. Packt Pub.
- [10] Lauretis, L. de. 2019. From Monolithic Architecture to Microservices Architecture. In *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE. DOI=10.1109/issrew.2019.00050.
- [11] MALL, R. 2018. *Fundamentals of software engineering*. Eastern Economy Edition. PHI Learning Private Limited, Delhi.
- [12] McKenzie Tucci. 2024. *Code-First vs. Design-First: Eliminate Friction with API Exploration*. <https://swagger.io/blog/code-first-vs-design-first-api/>. Accessed 10 May 2024.
- [13] Muhammad Fazril Bin Mohd Amin. 2012. *Volunteer Management System*. Bachelor of Technology (Hons), Universiti Teknologi PETRONAS.

- [14] Mwaura, W. 2021. *End-to-End Web Testing with Cypress: Explore techniques for automated frontend web testing with Cypress and JavaScript*. Packt Publishing Ltd.
- [15] Ponelat, J. S., Tam, T., and Rosenstock, L. 2022. *Designing APIs with Swagger and OpenAPI*. Manning Publications, Shelter Island.
- [16] Rupp, C. 2004. Requirements templates-the blueprint of your requirement. *Requirements Engineering* 366.
- [17] Rupp, C. and SOPHISTen, d. 2020. *Requirements-Engineering und -Management*. Carl Hanser Verlag GmbH & Co. KG, München.
- [18] Schelter, S. and Owen, S. 2012. Collaborative filtering with apache mahout. *Proc. of ACM RecSys challenge*.
- [19] Schönböck, J., Raab, M., Altmann, J., Kapsammer, E., Kusel, A., Pröll, B., Retschitzegger, W., and Schwinger, W. 2016. A survey on volunteer management systems. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, 767–776.
- [20] Seshadri, S. 2018. *Angular: Up and running: Learning angular, step by step*. O'Reilly Media, Inc.
- [21] Varanasi, B. 2019. *Introducing Maven: A Build Tool for Today's Java Developers*. Apress.
- [22] Walls, C. 2022. *Spring in action*. Manning, Shelter Island, NY.
- [23] Webb, P., Syer, D., Long, J., Nicoll, S., Winch, R., Wilkinson, A., Overdijk, M., Dupuis, C., and Deleuze, S. 2013. Spring boot reference guide. *Part IV. Spring Boot features* 24.
- [24] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A. 2012. *Experimentation in software engineering*. Springer Science & Business Media.

Appendix A - Specification

1 Mission des Projekts

1.1 Erläuterung des zu lösenden Problems

In diesem Projekt soll eine Webanwendung mit dem Namen "Zahnmobil" entwickelt werden, die den Freiwilligen bei der Koordinierung ihrer Einsätze hilft und gleichzeitig dem Management ermöglicht, das Projekt besser zu verwalten. Die Benutzer der Anwendung sollen sich anmelden und ihr Profil verwalten können. Die Anwendung soll dem Management die Möglichkeit bieten, Einsätze zu erstellen und zu bearbeiten, damit die Benutzer sich dafür anmelden können. Jeder Einsatz erfordert 3 Benutzer mit unterschiedlichen Rollen: Fahrer, Zahnarzt und Helfer. Das Management muss in der Lage sein, neue Einsatzorte hinzuzufügen oder zu bearbeiten und Benutzerprofile zu verwalten. Die Anwendung soll auch in der Lage sein, verschiedene Arten von Bestätigungen oder Einladungen per E-Mail an die Benutzer zu senden.

Die Anwendung sollte die Arbeitsprozesse nicht komplizierter machen, sondern durch eine intuitive und benutzerfreundliche Gestaltung zur Vereinfachung der Arbeitsprozesse beitragen.

1.2 Domänenbeschreibung

Dieses Projekt soll das bestehende System der Zahnmobil-Gruppe verbessern. Es wird den Freiwilligen ermöglichen, sich einfacher für Einsätze anzumelden, und die Verwaltung für das Management vereinfachen. Das Ziel ist es, eine effiziente und strukturierte Koordination innerhalb der Zahnmobil-Gruppe durch diese Anwendung zu ermöglichen.

1.3 Maßnahmen zur Anforderungsanalyse

Für die Anforderungsanalyse werden Mockups, Use Cases, User Stories und UML Klassen Diagramm erstellt.

2 Rahmenbedingungen und Umfeld

2.1 Einschränkungen und Vorgaben

Das Frontend der Anwendung wird mit Angular gebaut. Für das Backend wird Java benutzt, zusammen mit dem Spring Framework, Spring Boot und Spring Data. Spring Data JPA sorgt für die Verbindung zwischen Backend und Datenbank. Backend und Frontend kommunizieren über eine REST-Schnittstelle, die mit Swagger klar und nach dem OpenAPI-Standard dokumentiert wird.

2.2 Anwender

Diese Anwendung wird von den freiwilligen Helfern des Zahnmobil-Projekts genutzt. Die Nutzer kommen aus verschiedenen Altersgruppen und haben unterschiedliche IT-Erfahrungen.

2.3 Schnittstellen und angrenzende Systeme

Die API soll den REST-Prinzipien folgen, welche sich in der Softwareentwicklung als Standard für die Kommunikation zwischen Frontend und Backend etabliert haben. Spring Boot ist besonders darauf ausgerichtet, eine REST-API zu implementieren. Datenobjekte, die zwischen Front- und Backend ausgetauscht werden, nutzen das JSON-Format, was die Deserialisierung in native Java-Objekte mittels verbreiteter Frameworks wie Jackson vereinfacht. Dies fördert zudem die Lesbarkeit der Daten für Entwickler.

Für die Speicherung von Datenobjekten wie User, Einsätze und ähnlichem wird eine relationale Datenbank verwendet. Die Verwendung von Spring Data oder Hibernate als

ORM-Mapper erleichtert die Kommunikation mit der Datenbank und unterstützt die effiziente Verwaltung der Daten. Es soll auch eine Schnittstelle zu der aktuellen Webseite vorhanden sein.

3 Funktionale Anforderungen

3.2 Use Case-Beschreibungen

Use Case 1	User loggt sich ein
Erläuterung	Der User, der bereits ein Konto erstellt hat, sollte sich einloggen können, wenn die eingegebenen Daten stimmen.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Die Verwaltung hat schon den User in der Datenbank registriert und der User hat schon sein Passwort geändert.
Mindestgarantie	sichere Authentifizierung und klare Rückmeldung bei fehlerhaften Eingaben
Erfolgsgarantie	Nach Übersichtseite umgeleitet
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Mitarbeiter von Zahnmobil
Auslöser	Der Benutzer startet die Anmeldung, um Zugang zu erhalten, oder wird nach dem Versuch, einen geschützten Link aufzurufen, zur Anmeldeseite umgeleitet.
Hauptscenario	<ol style="list-style-type: none"> 1. User besucht die Login Seite 2. Login Maske wird angezeigt 3. User gibt seine E-Mail und Passwort 4. System prüft die eingegebenen Daten 5. System leitet User an das Übersichtseite weiter
Erweiterungen	4: WENN die vom User eingegebenen Daten nicht richtig sind, DANN gibt das System eine Fehlermeldung. Es folgt wieder der 2. Schritt
Priorität	unverzichtbar
Verwendungshäufigkeit	regelmäßig

Use Case 2	User setzt das Passwort zurück
Erläuterung	Beim erstmaligen Einloggen muss der Nutzer sein eigenes Passwort festlegen, oder es kann vorkommen, dass der Nutzer sein Passwort vergessen hat.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Die Verwaltung hat schon den User in der Datenbank registriert.
Mindestgarantie	Eine E-Mail wird verschickt an seine Emailadresse, wenn es die eingegebene E-Mail in dem System gibt.
Erfolgsgarantie	Der Nutzer kann nun mit dem erhaltenen Link sein Passwort zurücksetzen.
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Mitarbeiter von Zahnmobil
Auslöser	Wenn der User sein Passwort zurücksetzen will.

Hauptszenario	<ol style="list-style-type: none"> 1. Der User besucht die Login Seite 2. Der User klickt auf "Passwort vergessen" und gibt seine E-Mail-Adresse ein. 3. System zeigt einen Hinweistext, dass, falls diese E-Mail-Adresse existiert, ein Link zum Zurücksetzen des Passworts versendet wird. 4. Der User klickt auf den erhaltenen Link in seiner E-Mail. 5. System leitet den User zur Seite für das Zurücksetzen des Passworts. 6. Der User gibt sein neues Passwort in zwei Eingabefeldern ein. 7. Das System speichert das neue Passwort entsprechend der Eingabe. 8. Das System zeigt den Hinweistext „Dein Passwort wurde erfolgreich zurückgesetzt“ an. 9. Das System leitet den User zur Login-Seite weiter.
Erweiterungen	6a. WENN der User unterschiedliche Eingaben hat in diesen 2 Inputfelder, DANN gibt System eine Fehlermeldung und der User kann nochmal sein Passwort eingeben.
Priorität	unverzichtbar
Verwendungshäufigkeit	regelmäßig

Use Case 3	User bearbeitet seine Stammdaten
Erläuterung	Der User muss in der Lage sein, seine Stammdaten zu bearbeiten.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Der User muss eingeloggt sein.
Mindestgarantie	Datenaktualisierungen werden entweder vollständig oder gar nicht erfolgen und der Zugriff auf aktuelle Stammdaten erhalten bleibt und klare Rückmeldungen bei Bearbeitungsfehlern erfolgen.
Erfolgsgarantie	Bei erfolgreicher Bearbeitung werden die Stammdaten korrekt aktualisiert
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Mitarbeiter von Zahnmobil
Auslöser	Der User startet den Bearbeitungsvorgang seiner Stammdaten
Hauptszenario	<ol style="list-style-type: none"> 1. Der User wählt aus, ob er Anrede, Nachname, Vorname, Titel oder seine privaten und geschäftlichen Kontaktdaten, einschließlich Straße, PLZ, Ort, Telefon, Mobilnummer und E-Mail verwalten. 2. System zeigt den User seine Eingabemöglichkeiten, entweder ein Dropdown-Menü oder ein offenes Textfeld 3. Der User klickt auf „Speichern“. 4. Die Änderungen werden in der Datenbank aktualisiert.
Erweiterungen	1a. Bei Eingabefeldern validiert das System die Daten, zeigt bei ungültigen Eingaben (z.B. falsches Format, unzulässige Zeichen)

	eine spezifische Fehlermeldung (z.B. "Ungültige Mobilnummer ") an und fordert den User zur Korrektur auf.
Priorität	unverzichtbar
Verwendungshäufigkeit	Gelegentlich

Use Case 4	Die Verwaltung fügt einen neuen Einsatzort hinzu.
Erläuterung	Die Verwaltung möchte einen neuen Einsatzort hinzufügen.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Die Verwaltung muss mit seinem Account mit Admin Fähigkeiten eingeloggt sein
Mindestgarantie	Bei Fehlern im Hinzufügeprozess bleibt die bestehende Liste der Einsatzorte unverändert und konsistent.
Erfolgsgarantie	Bei erfolgreicher Durchführung wird der neue Einsatzort korrekt zur Einsatzortliste hinzugefügt und ist im System für Erstellung einen Einsatz sichtbar und verwendbar
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Verwaltung
Auslöser	Die Verwaltung identifiziert die Notwendigkeit, einen neuen Einsatzort im System zu erfassen, und startet den Prozess zur Dateneingabe
Hauptszenario	<ol style="list-style-type: none"> 1. Die Verwaltung klickt auf den „Einsatzorte“-Button in Übersicht Seite. 2. Die Verwaltung wählt „Einsatzort hinzufügen“ aus. 3. System zeigt die Eingabe Möglichkeiten wie Name Einsatzort, Straße, PLZ, Ort, Hinweis Stellplatz und Hinweis öffentlich, Ansprechpartner vor Ort, Einsatzzeit vor Ort, Einsatzzeit für Zahnarzt, Fahrer und Arzthelferin der User gibt seine Daten ein. 4. Die Verwaltung bestätigt die Eingaben durch Klicken auf „Einsatzort Hinzufügen“. 5. System speichert den Einsatzort in dem System und zeigt Erfolg Hinweistext.
Erweiterungen	3a.Bei Eingabefehlern validiert das System die Daten, zeigt bei ungültigen Eingaben (z.B. falsches Format, unzulässige Zeichen) eine spezifische Fehlermeldung (z.B. "PLZ") an und fordert den User zur Korrektur auf.
Priorität	unverzichtbar
Verwendungshäufigkeit	Häufig

Use Case 5	Die Verwaltung fügt einen neuen Einsatz hinzu.
Erläuterung	Die Verwaltung möchte einen neuen Einsatz hinzufügen.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Die Verwaltung muss mit seinem Account mit Admin Fähigkeiten eingeloggt sein
Mindestgarantie	Bei Fehlern im Hinzufügeprozess bleibt die bestehende Liste der Einsätze unverändert und konsistent.

Erfolgsgarantie	Bei erfolgreicher Durchführung wird der neue Einsatz korrekt zur Einsatzliste hinzugefügt und ist im System für Eintragen für die User sichtbar und verwendbar
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Verwaltung
Auslöser	Die Verwaltung möchte einen neuen Einsatz hinzufügen und klickt dafür auf „Planung“ auf der Übersichtseite
Hauptscenario	<ol style="list-style-type: none"> 1. Die Verwaltung klickt auf den „Planung“-Button auf der Übersichtseite. 2. System zeigt die Eingabe Möglichkeiten wie die Art des Einsatzes (normaler Einsatz, Veranstaltung, Urlaub, Sonstige) und Einsatzort von der Einsatzortliste und die Verwaltung gibt seine Daten ein. 3. Die Verwaltung wählt aus einem Kalenderübersicht das Datum für den Einsatz 4. Die Verwaltung bestätigt die Eingaben durch Klicken auf „Einsatz Hinzufügen“. 5. System speichert den Einsatz in dem System und zeigt Erfolg Hinweistext
Erweiterungen	4a: Es können mehrere Termine ausgewählt werden, und für jedes ausgewählte Datum wird ein Einsatz in der Datenbank erstellt.
Priorität	unverzichtbar
Verwendungshäufigkeit	Bedarfsweise

Use Case 6	Der User trägt sich für einen Einsatz ein.
Erläuterung	Der User mit der Rolle X (X = Fahrer, Zahnarzt, Zahnarzthelferin) möchte sich für einen Einsatz in der spezifischen Rolle X eintragen.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Der User muss eingeloggt sein und die Rolle X muss für den Einsatz frei sein.
Mindestgarantie	Der User mit der Rolle X kann sich nur für Einsätze in dieser spezifischen Rolle eintragen.
Erfolgsgarantie	Bei erfolgreicher Durchführung wird der User in dem Einsatz eingetragen. Der User bekommt auch ein Hinweis („Erfolgreich eingetragen“) und eine E-Mail mit einem Kalendereintrag.
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Mitarbeiter von Zahnmobil
Auslöser	Der User möchte sich für einen freien Einsatz eintragen.
Hauptscenario	<ol style="list-style-type: none"> 1. Der User besucht die Kalenderseite 2. Der User mit der Rolle X klickt auf den „Sich eintragen“-Button von einem Einsatz. 3. Das System speichert den User für die Rolle X in diesem Einsatz 4. System zeigt Hinweistext: „Erfolgreich eingetragen“
Erweiterungen	2a: Sich eintragen wird nur gezeigt, wenn der User die Rolle dafür hat

	4- Der User kann sich auch durch Klicken auf den „Sich austragen“-Button von einem Einsatz austragen. Dieser Button wird dem User angezeigt, sobald er für den Einsatz eingetragen ist. Der User und
Priorität	unverzichtbar
Verwendungshäufigkeit	Gelegentlich

Use Case 7	Die Verwaltung fügt einen User zu einem Einsatz hinzu
Erläuterung	Die Verwaltung möchte einen User mit der Rolle X = (Fahrer, Zahnarzt, Zahnarthelferin) für einen Einsatz in der entsprechenden Rolle X eintragen.
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Die Verwaltung muss mit seinem Account mit Admin Fähigkeiten eingeloggt sein
Mindestgarantie	Bei Fehlern im Hinzufügeprozess bleibt der Einsatz unverändert und konsistent
Erfolgsgarantie	Bei erfolgreicher Durchführung wird der User in den Einsatz eingetragen und erhält eine Bestätigungs-E-Mail mit einem Kalendereintrag. Die Verwaltung bekommt ebenfalls einen Hinweis („Erfolgreich eingetragen“)
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Verwaltung
Auslöser	Die Verwaltung möchte einen User für eine freie Rolle im Einsatz eintragen.
Hauptszenario	<ol style="list-style-type: none"> 1. Die Verwaltung besucht die Teamseite. 2. Die Verwaltung klickt auf das Aktionen-Icon. 3. Das System zeigt die möglichen Aktionen in einem Dropdown-Menü an. 4. Die Verwaltung wählt die Option „User in Einsatz eintragen“ aus. 5. Das System öffnet das Fenster „User in Einsatz eintragen“. 6. Die Verwaltung sucht nach dem User und dem Einsatz, wählt diese aus und klickt auf „eintragen“. 7. Das System trägt den User in den Einsatz ein, und der User erhält eine E-Mail mit einem Kalendereintrag.
Erweiterungen	6a: Nur Einsätze mit offener Rolle X werden gezeigt.
Priorität	unverzichtbar
Verwendungshäufigkeit	Gelegentlich

Use Case 8	Das System lädt die User zum Eintragen in einen Einsatz ein
Erläuterung	Das System erkennt, wenn eine Woche vor Beginn eines Einsatzes noch eine Rolle X (X = Fahrer, Zahnarzt, Zahnarthelferin) offen ist, und sendet dann eine Empfehlungs-E-Mail an die User mit der gleichen Rolle. Zusätzlich wird eine Hinweis-E-Mail an die Verwaltung geschickt
Systemgrenzen (Scope)	Gesamtsystem

Ebene	Hauptfunktion
Vorbedingung	System muss in Betrieb sein
Mindestgarantie	Es werden keine unnötigen E-Mails geschickt
Erfolgsgarantie	Wenn eine Rolle X bei einem Einsatz eine Woche vor dem Einsatztermin noch unbesetzt ist, schickt das System eine E-Mail an alle User mit der Rolle X und fragt nach Teilnahme Zusätzlich wird eine Hinweis-E-Mail an die Verwaltung geschickt
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	System
Auslöser	Ein Einsatz hat eine Woche vor Beginn noch keinen eingetragenen Rolle X
Hauptszenario	<ol style="list-style-type: none"> 1. Das System detektiert, dass der Einsatz in einer Woche stattfindet und die Rolle X noch unbesetzt ist 2. Das System schickt eine E-Mail an alle User mit der Rolle X und fordert zur Teilnahme auf. Zusätzlich wird eine Hinweis-E-Mail an die Verwaltung geschickt
Erweiterungen	2a: In der E-Mail wird ein Link bereitgestellt, über den man zum Einsatz weitergeleitet wird.
Priorität	unverzichtbar
Verwendungshäufigkeit	Gelegentlich

Use Case 9	Die Verwaltung schickt Mail an die User
Erläuterung	Die Verwaltung möchte mit der Zentrale E-Mail-Adresse von Zahnmobil, Mail an die User schicken
Systemgrenzen (Scope)	Gesamtsystem
Ebene	Hauptfunktion
Vorbedingung	Die Verwaltung muss mit seinem Account mit Admin Fähigkeiten eingeloggt sein
Mindestgarantie	Die Verwaltung kann alle alte gesendete Mail ansehen. Und nur Verwaltung kann Mail schicken.
Erfolgsgarantie	Bei erfolgreicher Durchführung wird eine Mail an gewählte User mit der Zentrale E-Mail-Adresse von Zahnmobil geschickt
Stakeholder	Mitarbeiter von Zahnmobil
Hauptakteur	Die Verwaltung
Auslöser	Die Verwaltung klickt auf „Mailing“
Hauptszenario	<ol style="list-style-type: none"> 1. Die Verwaltung klickt auf „Mailing“. 2. Die Verwaltung sucht nach einem User in der Benutzerliste oder in einem Suchfeld und fügt diesen zur Liste der ausgewählten Benutzer hinzu. 3. Die Verwaltung gibt ihre Nachricht in ein Eingabefeld ein und klickt auf „Mail senden“. 4. Das System versendet eine E-Mail an die ausgewählten Benutzer, und die E-Mail wird im System gespeichert.
Erweiterungen	2a: Die Verwaltung kann auch User aus dem List entfernen 3a: Die Verwaltung ist auch in der Lage, Dateien anzuhängen.
Priorität	Niedrig
Verwendungshäufigkeit	Gelegentlich

4 Qualitätsanforderungen

4.1 Qualitätsziele des Projekts

Die wichtigste nicht-funktionale Anforderung in diesem Projekt ist die Usability. Es ist wichtig, dass das User Interface die Arbeit nicht unnötig erschwert, sondern durch ein intuitives und benutzerfreundliches Design die Arbeitsabläufe erleichtert, insbesondere da die Anwendung von einer breiten Altersgruppe und Personen mit unterschiedlichen IT-Kenntnissen genutzt wird. Ein weiteres bedeutendes nicht-funktionales Anforderung ist die Wartbarkeit. Es ist von großer Bedeutung, dass die Anwendung klar strukturiert ist, um zukünftige Aktualisierungen oder Erweiterungen zu vereinfachen. Eine gut gewartete Anwendung reduziert das Risiko von Fehlern und erleichtert die schnelle Anpassung an veränderte Anforderungen. Weiterhin soll die Anwendung auch auf mobilen Geräten genutzt werden und deswegen ist es wünschenswert, dass sie über ein responsives Design verfügt, um eine optimale Darstellung auf verschiedenen Endgeräten sicherzustellen.

4.2 Qualitäts-Prioritäten des Kunden

Die Qualitätsziele sind wie folgt priorisiert:

1. Usability
2. Maintainability
3. Portability

4.3 Wie Qualitätsziele erreicht werden sollen

Die Usability wird durch die Erstellung von Mock-ups sichergestellt, und es wird aktiv Feedback von den Kunden eingeholt. Prinzipien der menschenzentrierten Interaktion werden eingesetzt, um die Benutzerfreundlichkeit zu gewährleisten. Zudem werden Umfragen und Tests mit den Kunden durchgeführt, um eine klare und effektive Gestaltung der Anwendung zu garantieren.

Die Wartbarkeit wird gewährleistet, indem darauf geachtet wird, dass der Code der Anwendung eine hohe Kohäsion und geringe Kopplung aufweist. Es wird gezielt, Best-Practice-Methoden zu verfolgen, um eine strukturierte und klar definierte Codebasis zu schaffen

Für die Portabilität wird angestrebt, bei ausreichender Zeit, ein responsives Design zu implementieren, sodass die Anwendung auch auf verschiedenen Mobilgeräten nutzbar ist. Dies wird erreicht, indem das responsive Design in verschiedenen Dimensionen getestet und Feedback von den Kunden eingeholt wird.

5 Hinweise zur Umsetzung

5.1 User Stories

User Story 1:

Als Freiwilliger der Zahnmobil-Gruppe möchte ich mich schnell und unkompliziert für einen Einsatz mit der Rolle Fahrer anmelden können, um die Leute, die zahnmedizinische Versorgung benötigen, zu helfen.

Akzeptanzkriterien:

- Einfacher Zugriff: Ich kann die Anwendungsplattform über mein bevorzugtes Gerät (Smartphone, Tablet, PC) erreichen.
- Übersichtliche Darstellung: Ich sehe eine klare Auflistung der verfügbaren Einsätze mit relevanten Details wie Datum, Ort und benötigte Rollen.
- Einfache Anmeldung: Mit wenigen Klicks kann ich mich für einen Einsatz anmelden.
- Bestätigung: Nach der Anmeldung erhalte ich eine Bestätigung per E-Mail, die alle wichtigen Informationen zum Einsatz enthält.
- Rollenbasierte Anmeldung: Die Plattform berücksichtigt meine Qualifikationen und erlaubt mir nur, mich für passende Rollen (Fahrer, Zahnarzt, Zahnarthelferin) anzumelden.

User Story 2:

Als Verwaltung der Zahnmobil-Gruppe möchte ich schnell und unkompliziert einen Einsatzort in dem System hinzufügen, sodass später Einsätze mit diesem neuen Einsatzort in dem System hinzugefügt werden kann.

Akzeptanzkriterien:

- Intuitive Einsatz-Erstellung: Ich kann als Verwaltungsperson über eine benutzerfreundliche Schnittstelle schnell und einfach einen neuen Einsatzort hinzufügen.
- Detaillierte Informationseingabe: Ich habe die Möglichkeit, alle relevanten Einsatzdetails (Name Einsatzort, Straße, PLZ, Ort, Hinweis Stellplatz und Hinweis öffentlich, Ansprechpartner vor Ort, Einsatzzeit vor Ort, Einsatzzeit für Arzt, Fahrer und Arzthelferin.) übersichtlich einzutragen.
- Sofortige Aktualisierung: Nach dem Hinzufügen eines Einsatzort wird dieser unmittelbar in der Übersicht aller Einsatzorte angezeigt und ist für die Verwaltung um neue Einsätze zu erstellen verfügbar.
- Überprüfung der Eingaben: Vor der endgültigen Bestätigung des neuen Einsatzes erfolgt eine Überprüfung auf Vollständigkeit und Korrektheit der Informationen.

- Rollenbasierte Zugriffsrechte: Nur berechnigte Verwaltungspersonen können Einsätze hinzufügen, um die Sicherheit und Genauigkeit der Daten zu gewährleisten.

User Story 3:

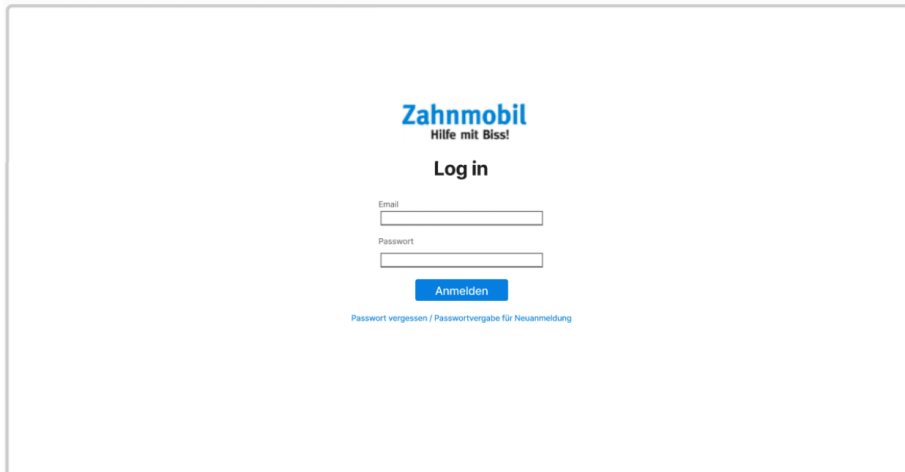
Als Verwaltung der Zahnmobil-Gruppe möchte ich schnell und unkompliziert einen Einsatz hinzufügen zu können, sodass die Mitarbeiter der Zahnmobil-Gruppe, sich dafür eintragen.

Akzeptanzkriterien:

- Intuitive Einsatz-Erstellung: Ich kann als Verwaltungsperson über eine benutzerfreundliche Schnittstelle schnell und einfach einen neuen Einsatz hinzufügen.
- Detaillierte Informationeingabe: Ich habe die Möglichkeit, alle relevanten Einsatzdetails (Einsatzort, Einsatztyp, Datum) einzutragen. Für die Eintragung von Datum benutze ich gerne eine Kalenderübersicht.
- Sofortige Aktualisierung: Nach dem Hinzufügen eines Einsatzes wird dieser unmittelbar in der Übersicht aller Einsätze angezeigt und ist für die Mitarbeiter sichtbar.
- Rollenbasierte Zugriffsrechte: Nur berechnigte Verwaltungspersonen können Einsätze hinzufügen, um die Sicherheit und Genauigkeit der Daten zu gewährleisten.

5.2 Mockups

Log in Seite



Zahnmobil
Hilfe mit Biss!

Log in

Email

Passwort

[Anmelden](#)

[Passwort vergessen / Passwortvergabe für Neuanmeldung](#)

Password vergessen Seite

Zahnmobil
Hilfe mit Biss!

Passwort vergessen

Haben sie ihr Passwort vergessen ?
Tragen sie bitte ihre E-Mail Adresse ein.

Email

[Ersetze Passwort](#)

Password Confirmation Seite

Zahnmobil
Hilfe mit Biss!

Neues Passwort erstellen

Neues Passwort

Neues Passwort wiederholen

[Ersetze Passwort](#)

Übersicht Seite

Zahnmobil
Hilfe mit Biss!

Ausmelden

ÜBERSICHT
STAMMDATEN
KALENDER
MAILING
PLANUNG
TEAM
ENSATZORTE

Willkommen Herr Max Mustermann (Zahnarzt)

Gesamtanzahl in diesem Jahr behandelte Einsätze: 10
Gesamtanzahl im vergangenen behandelte Einsätze: 20

meine nächsten 5 Einsätze: KALENDER

02.03.2024 - 10:30-11:00 Hr - Kontaktladen Meckl (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr

kommende offene Termine

Aktiv
Fahrer
Arzt
Helferin

Fahrer

02.03.2024 - 10:30-11:00 Hr - Kontaktladen Meckl (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr KALENDER

Arzt

02.03.2024 - 10:30-11:00 Hr - Kontaktladen Meckl (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr KALENDER

Helferin

02.03.2024 - 10:30-11:00 Hr - Kontaktladen Meckl (vormittags)
Einsatz Gruppe fahrer: 8:15 - 12:30 Uhr KALENDER

kommende 5 Veranstaltungen:

01.04.2024 - 0:00 - 0:00 Uhr - Urlaub - Kein Einsatz KALENDER

01.04.2024 - 0:00 - 0:00 Uhr - Urlaub - Kein Einsatz KALENDER

Stammdaten Seite

Zahnmobil
Hilfe mit Biss!

Ausmelden

ÜBERSICHT
STAMMDATEN
KALENDER
MAILING
PLANUNG
TEAM
ENSATZORTE

Stammdatensatz

Anrede:

Nachname*:

Vorname:

Titel:

Kontakt geschäftlich

Firma:

Strasse (geschäftl.):

PLZ (geschäftl.):

Ort (geschäftl.):

Telefon (geschäftl.):

Email-Adresse (geschäftl.):

Administration

Funktion: ▼

Benutzerlevel: ▼

Kontakt privat

Strasse (privat)

PLZ (privat):

Ort (privat):

Telefon (privat):

Email-Adresse (privat)*:

Email-Adresse (alternativ):

Geburtsdatum:

Speichern

Kalender

Zahnmobil
Hilfe mit Biss!

ÜBERSICHT STAMMDATEN **KALENDER** MAILING PLANUNG TEAM ENGATZORTE

Aussehen

Kalender

<< 2023 2025 >>

JAN

FEB

MAR

APR

MAI

JUN

JUL

AUG

SEP

OKT

NOV

DEZ

KW13

Montag KW 13
01.04.2024
0:00-0:00 Uhr

Urlaub - Kein Einsatz

Montag KW 13
01.04.2024
0:00-0:00 Uhr

Einsatz: Kontakladen Mecki
(vormittags)
Raschplatz 8c

Zahnarzt: **frei**
Helfer: **Max Mustermann**
Fahrer: eintragen

KW14

Montag KW 13
01.04.2024
0:00-0:00 Uhr

Einsatz: Kontakladen Mecki
(vormittags)
Raschplatz 8c

Zahnarzt: **frei**
Helfer: **Max Mustermann**
Fahrer: eintragen

Mailing 1/2

Zahnmobil
Hilfe mit Biss!

ÜBERSICHT STAMMDATEN KALENDER **MAILING** PLANUNG TEAM ENGATZORTE

Aussehen

Mailing

[neues Mail hinzufügen](#)

Lfd. Nummer	Betreff	angelegt	geändert	Letzter Versand	Aktionen
1	Hallo World 1	2023.02.31 10:33:10	geändert	2023.02.31 10:33:10	
2	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
3	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
4	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
5	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
6	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
7	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
8	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	

<
>

Mailing 2/2 (neues Mail schicken)

Zahnmobil
Hilfe mit Biss!

ÜBERSICHT STAMMDATEN KALENDER MAILING PLANUNG TEAM EINSATZORTE

Mailing neues Mail schicken

Lfd. Nummer	Betreff	angelegt	geändert	Letzter Versand	Aktionen
1	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
2	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
3	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
4	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
5	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
6	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
7	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	
8	Hallo World 1	2023.02.31 10:33:10		2023.02.31 10:33:10	

< >

Planung

Zahnmobil
Hilfe mit Biss!

ÜBERSICHT STAMMDATEN KALENDER MAILING PLANUNG TEAM EINSATZORTE

Einsatz Hinzufügen

Einsatzort auswählen:

Einsatz Typ:

April 2021 < >

Mo	Tu	We	Th	Fr	Sa	Su
				1	2	3 4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Hinzugefügte Daten:
06.4.2021

Einsatz Hinzufügen

Team 1/3

Zahnmobil
Hilfe mit Biss!

ÜBERSICHT STAMMGATEN KALENDER MAILING PLANUNG **TEAM** EINSATZORTE

Benutzerverwaltung

Benutzer suchen: Funktion:

Nachname	Vorname	Funktion	Telefon privat	E-Mail	Mobil	Aktionen
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	
Ferdowski	Mohammad Ali	Fahrer	01752331231	alferdos@xasd.com	01752331231	

Team 2/3 (User Dateien bearbeiten)

Zahnmobil
Hilfe mit Biss!

ÜBERSICHT STAMMGATEN KALENDER MAILING PLANUNG **TEAM** EINSATZORTE

Benutzer bearbeiten

Nachname:

Vorname:

Funktion:

Telefon privat:

E-Mail:

Mobil:

Speichern

Aktionen:

Team 3/3

The screenshot shows the 'Zahnmobil' web application interface. At the top, the logo 'Zahnmobil' and the slogan 'Hilfe mit Biss!' are visible. A navigation bar contains the following tabs: ÜBERSICHT, STAMMDATEN, KALENDER, MAHLING, PLANUNG, TEAM, and EINSATZORTE. The 'TEAM' tab is currently selected. A modal window titled 'User in Einsatz eintragen' is open, allowing for user assignment. The modal contains the following fields and options:

- Benutzer auswählen:** A text input field containing 'Max Mustermann'.
- Funktion:** A dropdown menu with 'Fahrer' selected.
- Einsatz auswählen:** A text input field containing 'Kontakladen Mecki (vormittags) - 12.04.2024'.
- Einsatzort:** A dropdown menu with 'Kontakladen Mecki' selected.

Below these fields, the system displays the selected user and assignment details:

- Benutzer:** Max Mustermann (Fahrer)
- Einsatz:** Kontakladen Mecki (vormittags) Raschplatz 8c

An 'eintragen' button is located at the bottom right of the modal. On the left side of the main application, a sidebar shows a list of users, each with a 'Ferdow' button. On the right side, there are several rows of three asterisks (***) representing a list of items.

Einsatzorte 1/3

The screenshot shows the 'Einsatzorte' (Deployment Locations) management page in the 'Zahnmobil' application. The navigation bar is the same as in the previous screenshot, but the 'EINSATZORTE' tab is selected. The page features a search bar labeled 'Einsatzort suchen:' with a search icon. Below the search bar, there is a list of deployment locations, each with a plus sign icon and a text label:

- + Neuen Einsatz Hinzufügen
- > NUK Burgweg 5 Vormittags
- > NUK Burgweg 5 Vormittags
- > NUK Burgweg 5 Vormittags
- > NUK Burgweg 5 Vormittags
- > NUK Burgweg 5 Vormittags

Einsatzorte 2/3

Zahnmobil
Hilfe mit Biss!

Anmelden
ÜBERSICHT STAMMDATEN KALENDER MAHLING PLANUNG TEAM **EINSAATZORTE**

Einsatzorte

Einsatzort suchen:

▼ **NJK Burgweg 5 Vormittags**

<p>Name Einsatzort: <input style="width: 100%;" type="text"/></p> <p>Straße: <input style="width: 100%;" type="text"/></p> <p>PLZ: <input style="width: 100%;" type="text"/></p> <p>Hinweis Stelplatz: <input style="width: 100%; height: 20px;" type="text"/></p> <p>Hinweis Öffentlich: <input style="width: 100%; height: 20px;" type="text"/></p> <p>Ansprechpartner vor Ort: <input style="width: 100%;" type="text"/></p> <p>Einsatzzeit vor Ort: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/></p> <p>Einsatzzeit Uhrzeit für Arzt: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/></p> <p>Einsatzzeit Uhrzeit für Fahrer: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/></p> <p>Einsatzzeit Uhrzeit für Helfer: <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/></p>	<p>kommende 5 Termine gemäß Planung:</p> <table style="width: 100%; border-collapse: collapse; font-size: 8px;"> <thead> <tr> <th style="text-align: left;">Wechertag</th> <th style="text-align: left;">Tag</th> <th style="text-align: left;">Zeit</th> </tr> </thead> <tbody> <tr><td>Samstag</td><td>20.01.2024</td><td>9:00-11:00</td></tr> <tr><td>Samstag</td><td>20.01.2024</td><td>9:00-11:00</td></tr> <tr><td>Samstag</td><td>20.01.2024</td><td>9:00-11:00</td></tr> <tr><td>Samstag</td><td>20.01.2024</td><td>9:00-11:00</td></tr> <tr><td>Samstag</td><td>20.01.2024</td><td>9:00-11:00</td></tr> </tbody> </table>	Wechertag	Tag	Zeit	Samstag	20.01.2024	9:00-11:00	Samstag	20.01.2024	9:00-11:00	Samstag	20.01.2024	9:00-11:00	Samstag	20.01.2024	9:00-11:00	Samstag	20.01.2024	9:00-11:00
Wechertag	Tag	Zeit																	
Samstag	20.01.2024	9:00-11:00																	
Samstag	20.01.2024	9:00-11:00																	
Samstag	20.01.2024	9:00-11:00																	
Samstag	20.01.2024	9:00-11:00																	
Samstag	20.01.2024	9:00-11:00																	

Einsatzorte 3/3

Zahnmobil
Hilfe mit Biss!

Anmelden
ÜBERSICHT STAMMDATEN KALENDER MAHLING PLANUNG TEAM **EINSAATZORTE**

Einsatzort hinzufügen

Name Einsatzort:

Straße:

PLZ:

Hinweis Stelplatz:

Hinweis Öffentlich:

Ansprechpartner vor Ort:

Einsatzzeit vor Ort:

Einsatzzeit Uhrzeit für Arzt:

Einsatzzeit Uhrzeit für Fahrer:

Einsatzzeit Uhrzeit für Helfer:

6 Probleme und Risiken

WENN keine Differenzierung der Berechtigungen zwischen den User besteht, **DANN** hat jeder die Möglichkeit, einen Einsatz zu erstellen.

Abhilfe: Die Implementierung eines Rechtemanagement ist erforderlich.

WENN ein User sich für einen Einsatz mit einer anderen Rolle als seiner eigenen registrieren kann, **DANN** führt dies zu erheblichen Problemen für den Einsatz.

Abhilfe: Es ist sicherzustellen, dass die Registrierung für Einsätze nur entsprechend der tatsächlichen Rolle des Users möglich ist.

WENN die Daten der Einsätze nicht konsistent sind, nicht aktualisiert werden oder aus irgendeinem Grund den Usern nicht angezeigt werden, **DANN führt** dies zu erheblichen Problemen für den Einsatz.

Abhilfe: Es ist erforderlich, die stetige Konsistenz, Aktualisierung und Verfügbarkeit der Einsatzdaten für alle User sicherzustellen.

7 Optionen zur Aufwandsreduktion

7.1 Mögliche Abstriche

Folgende Funktionen sind als wünschenswert gekennzeichnet und können bei Zeitmangel gestrichen werden:

- Kalender/Liste von Usern auszudrücken in PDF Format
- Responsive Design
- Mailing
- Stammdaten (Außer Wichtige Daten wie Nachname, Vorname, Funktion)

7.2 Inkrementelle Arbeit

Für die Umsetzung dieses Projekts verfolgen wir einen Contract-First-Ansatz, wobei unser primärer Fokus Am Anfang darauf liegt, eine robuste Schnittstelle zwischen Frontend und Backend zu erstellen. Die OpenAPI/Swagger-Spezifikation spielt hier eine zentrale Rolle, indem sie die Strukturen und Dienste definiert. Dieser Ansatz sorgt nicht nur für die Einhaltung von Standards in der Entwicklung, sondern erleichtert auch die Dokumentation von Endpoints und Wartung von dem Projekt.

Iteration 1 - Fundament und Kommunikation:

In dieser ersten Phase wird das Fundament für Frontend und Backend erstellt. Deren Zusammenspiel wird mittels der Contract-First-Methode und OpenAPI geschaffen. Dabei legen wir besonderen Wert auf die genaue Definition der wichtigen Endpunkte und Datentransferobjekte (DTO).

Iteration 2 - Hauptfunktionen:

In diesem Abschnitt konzentrieren wir uns auf die Umsetzung der wesentlichen Anforderungen. Sollte die Zeit es erlauben, werden auch zusätzliche, wünschenswerte Funktionen integriert.

Iteration 3 - Polishing und Tests:

In dieser Phase werden die implementierten Funktionalitäten verfeinert und entsprechend getestet, um ihre einwandfreie Funktion sicherzustellen.

Iteration 4 - Bewertung und Feedback:

In der abschließenden Phase bewerten wir die Leistung der Applikation. Unser Ziel ist es, die Anwendung auf einem Server zu installieren und ein realistisches Testumfeld zu schaffen. Mitarbeiter von Zahnmobil werden die Anwendung testen und anschließend über einen online Fragebogen ihre Rückmeldung geben. Dieses Feedback ist von großer Bedeutung für die Weiterentwicklung und Optimierung der Anwendung sowie für die Behebung von Bugs.

8 Glossar

Komponente	Java-Klasse, mit der sich das Resultat eines Teams starten lässt. Muss Instanz einer Unterklasse der hier erstellten Klasse Component sein.
OpenAPI / Swagger	OpenAPI, auch bekannt als Swagger, ist eine Spezifikation zur Beschreibung von RESTful APIs in einem standardisierten Format. Sie erleichtert das Verstehen, Nutzen und Dokumentieren von Webdiensten, indem sie klare Anweisungen über Struktur und Funktionen der API bereitstellt.
DTO	Ein DTO ist ein einfaches Objekt, das dazu dient, Daten zwischen verschiedenen Schichten oder Teilen einer Anwendung zu übertragen, ohne unnötige Komplexität oder Abhängigkeiten einzuführen.
Endpunkte	"Endpunkte" in der Welt der Webdienste und APIs bezeichnen spezifische Adressen (URLs) im Netzwerk, an denen bestimmte Ressourcen oder Dienste zugänglich sind.
Java	Programmiersprache, was das Entwickeln von Anwendungen in der ersten Linie ermöglicht
REST API	Eine Schnittstelle, die die vier vorgegebenen Kriterien für die API einhalten
Front End	Der Teil der Anwendung, der sich auf das Aussehen der Anwendung konzentriert
Back End	Das Backend einer Anwendung befasst sich hauptsächlich mit dem Abrufen, Speichern und Darstellen von Daten, die von spezifischen

	URL-Links abhängen. Es umfasst auch Funktionen wie Authentifizierung, Datenvalidierung und weitere zentrale Verarbeitungsaufgaben.
SQL	Eine Sprache, die es ermöglicht, Abfragen an eine Datenbank zu richten.
Angular	Frontend Framework von Google
Spring Boot	Backend Framework

9 Abnahme-Testfälle

Abnahmetestfall: Login mit validen Userdaten.

Setup: Applikation wurde in Chrome geöffnet.

Eingabe	Ausgabe
User gibt gültige Email musterman@gmail.com und Passwort pass12345 ein	User wird erfolgreich eingeloggt und wird nach Übersichtseite geleitet

Abnahmetestfall: User bearbeitet seine Stammdaten.

Setup: Applikation wurde in Chrome geöffnet und User ist eingeloggt.

Eingabe	Ausgabe
User klickt auf Stammdaten bei Nav Bar.	User wird nach Stammdaten Seite geleitet
User schreibt in Nachname Feld: „Amigo“ und klickt auf speichern	System speichert die neuen Stammdaten von dem User

Abnahmetestfall: User trägt sich für einen Einsatz ein.

Setup: Applikation wurde in Chrome geöffnet und User ist eingeloggt and auf Kalender Seite.

Eingabe	Ausgabe
User klickt auf SEP	System zeigt alle vorhandenen Einsätze in September
User klickt auf eintragen und trägt sich als Fahrer ein für den Einsatz Kontaktladen Mecki	System speichert den User als Fahrer für den Einsatz

Abnahmetestfall: User trägt sich von einem Einsatz aus.

Setup: Applikation wurde in Chrome geöffnet und User ist eingeloggt und auf Kalender Seite.

Eingabe	Ausgabe
User klickt auf SEP	System zeigt alle vorhandenen Einsätze in September
User klickt auf Austragen und trägt sich als Fahrer aus für den Einsatz Kontaktladen Mecki	System aktualisiert den Einsatz und wird nochmal eintragen zeigen.

Abnahmetestfall: Verwaltung möchte eine Mail schicken

Setup: Applikation wurde in Chrome geöffnet und User ist eingeloggt und auf Mail Seite.

Eingabe	Ausgabe
Die Verwaltung schreibt für den Betreff „Einladung zum Zahnarztveranstaltung“ und Nachricht „Sie sind herzlich eingeladen zu Zahnarztveranstaltung in Hannover Messe“ und wählt alle User mit Zahnarzt Funktion und klickt auf „Mail Schicken“	System schickt mit der zentrale Email-Adresse eine Mail an Die E-Mail-Adresse von allen ausgewählten Usern

Abnahmetestfall: Verwaltung möchte ein Einsatzort hinzufügen

Setup: Applikation wurde in Chrome geöffnet und User ist eingeloggt und auf Einsatzorte Seite.

Eingabe	Ausgabe
Die Verwaltung klickt auf „neuen Einsatz hinzufügen	System leitet der User nach neuen Einsatz Hinzufügen Seite
Die Verwaltung gibt folgende Daten ein: Name Einsatzort: Kontaktladen Mecki, Straße: Kopernickusstrasse PLZ: 30149, Hinweisstellplatz: - , Hinweisöffentlich: -, Ansprechpartner vor Ort: -, EinsatzzeitvorOrt: 9 bis 13, EinsatzzeitfürArzt:9 bis 11, EinsatzzeitfürFahrer: 9 bis 12, EinsatzzeitfürHelfer: 9 bis 13 und klickt auf Einsatzhinzufügen	System speichert den Einsatzort in dem System und zeigt eine Erfolg Hinweistext. Das neue Einsatzort ist jetzt bei Einsatz hinzufügen sichtbar

Abnahmetestfall: Verwaltung möchte einen Einsatz hinzufügen

Setup: Applikation wurde in Chrome geöffnet und User ist eingeloggt und auf Planung Seite.

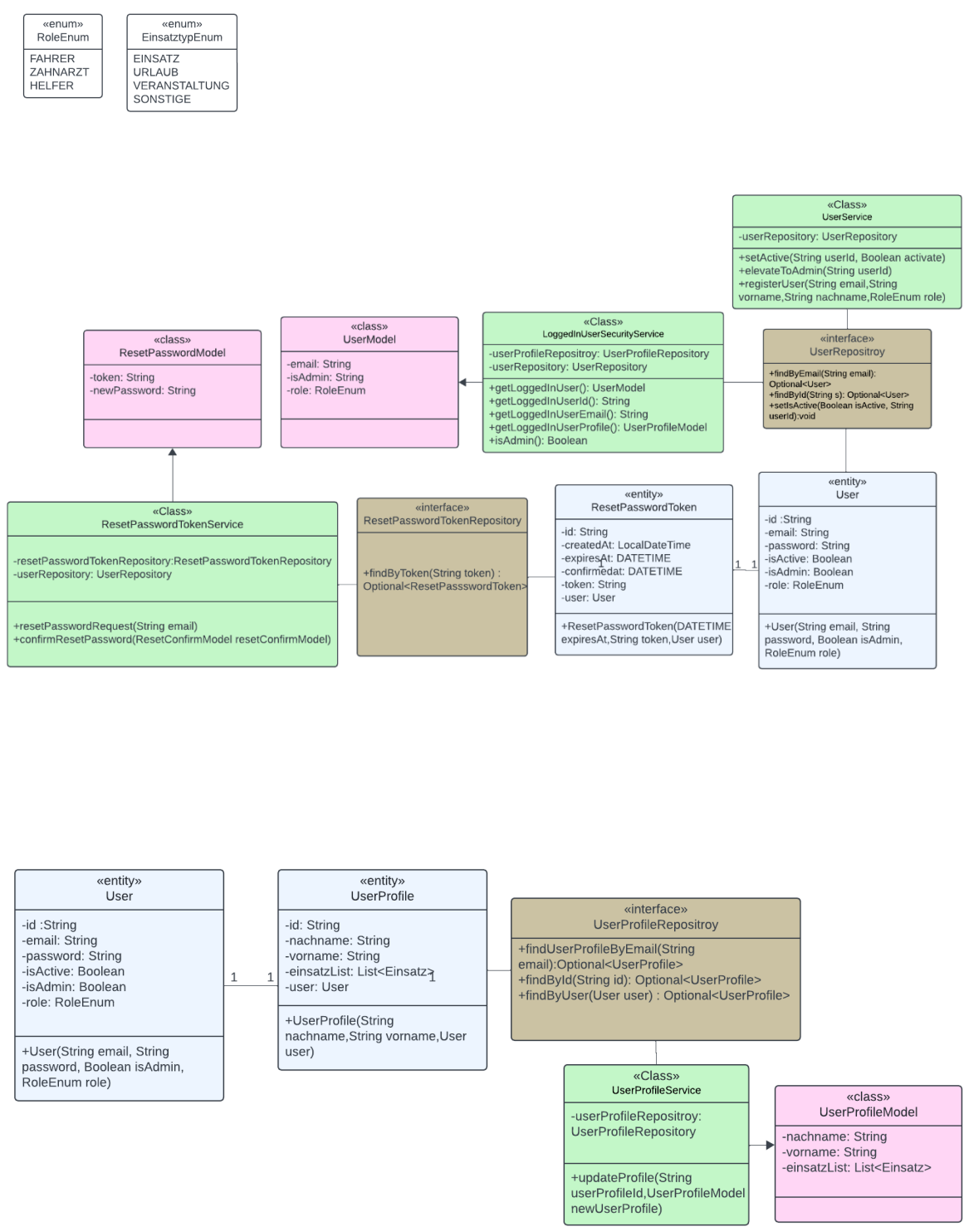
Eingabe	Ausgabe
Die Verwaltung gibt Kontaktladen Mecki (Vormittags) als Einsatzort und Einsatz als Einsatztyp ein und wählt 6te und 8te September 2024 und klickt auf Einsatz hinzufügen	System Speichert zwei Einsätze mit Kontaktladen Mecki (Vormittags) als Einsatzort und 06.Sep.2024 und 8.Sep.2024 als Datum. Der Einsatz ist jetzt auf Kalender für die User auch sichtbar und eintragbar.

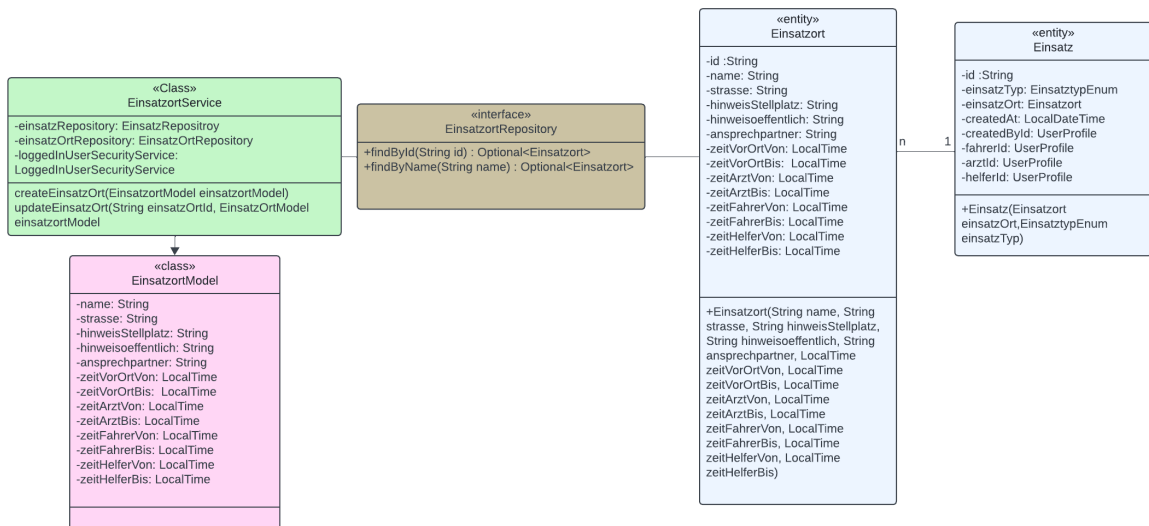
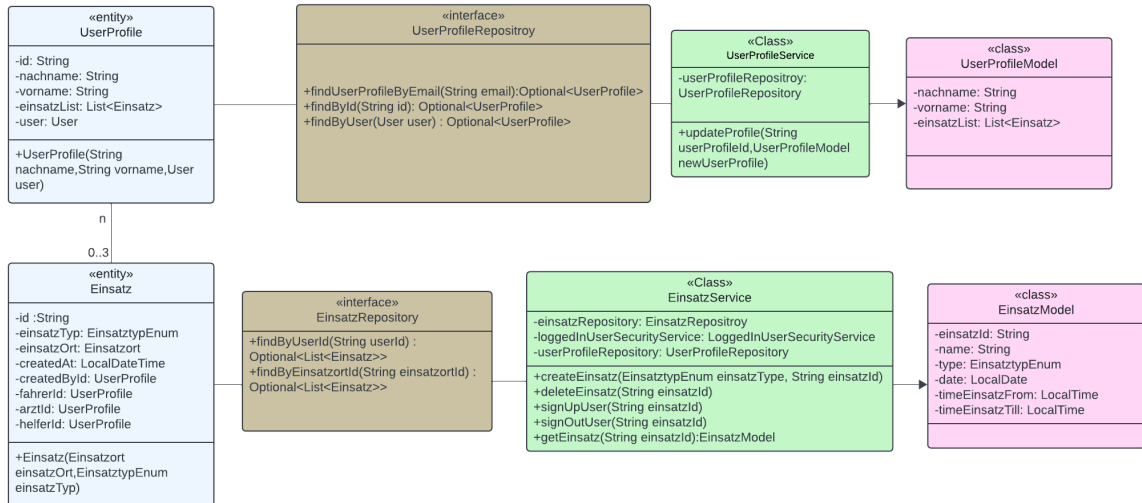
Abnahmetestfall: Verwaltung möchte Funktion von einem User bearbeiten

Setup: Applikation wurde in Chrome geöffnet und User ist eingeloggt und auf Team Seite.

Eingabe	Ausgabe
Die Verwaltung schreibt „Mustermann“ Nachname und aktiviert den Filter Funktion „Fahrer“	System zeigt alle User mit Nachname „Mustermann“ und Funktion „Fahrer“
Die Verwaltung klickt auf Edit Aktion	System zeigt ein Fenster zu Bearbeiten von User
Die Verwaltung verändert Die Funktion von User von Fahrer zu Helfer und klickt auf „Speichern“	System speichert die neue Funktion von Musterman und jetzt kann Musterman sich als Helfer für Einsätze eintragen

Appendix B - Domain Design





Appendix C - Developed Application Screenshots

Zahnmobil
Hilfe mit Biss!

Log in

Email

Password

Anmelden

[Passwort vergessen / Passwortvergabe für Neuanmeldung](#)

Willkommen Herr Max Mustermann (FAHRER)

Gesamtanzahl in diesem Jahr behandelte Einsätze: 5

Gesamtanzahl im vergangenen behandelte Einsätze: 5

meine nächsten 5 Einsätze:

2024-05-11 - 00:00-12:00 - Christuskirche
Einsatz Gruppe Fahrer: 14:30-16:00

[Calendar](#)

2024-05-14 - 00:00-12:00 - Christuskirche
Einsatz Gruppe Fahrer: 14:30-16:00

[Calendar](#)

2024-05-16 - 00:00-12:00 - Christuskirche
Einsatz Gruppe Fahrer: 14:30-16:00

[Calendar](#)

2024-05-24 - 00:00-12:00 - Christuskirche
Einsatz Gruppe Fahrer: 14:30-16:00

[Calendar](#)

2024-05-25 - 00:00-12:00 - Christuskirche
Einsatz Gruppe Fahrer: 14:30-16:00

[Calendar](#)

kommende offene Termine

Fahrer

3000-04-23 - 00:00-12:00 - Christuskirche
Einsatz Gruppe Fahrer: 14:30-16:00

[Calendar](#)

3000-05-23 - 00:00-12:00 - Christuskirche
Einsatz Gruppe Fahrer: 14:30-16:00

[Calendar](#)

kommende 5 Veranstaltungen:

Stammdatensatz

Titel:

Anrede:

Nachname*: Vorname*:

Administration

Email:

Funktion: Benutzerlevel:

[Speichern](#)

Kalender

<< 2023

2025 >>

- JAN
- FEB
- MAR
- APR
- MAI
- JUN
- JUL
- AUG
- SEP
- OKT
- NOV
- DEZ

KW19

Saturday
2024-05-11
00:00-12:00

Einsatz: Christuskirche
Lutherkirche Str 16

Zahnarzt: frei
Faher: eintragen
Helfer: frei

KW20

Tuesday
2024-05-14
00:00-12:00

Einsatz: Christuskirche
Lutherkirche Str 16

Zahnarzt: frei
Faher: austragen
Helfer: frei

Thursday
2024-05-16
00:00-12:00

Einsatz: Christuskirche
Lutherkirche Str 16

Zahnarzt: frei
Faher: eintragen
Helfer: frei

KW21

Friday
2024-05-24
00:00-12:00

Einsatz: Christuskirche
Lutherkirche Str 16

Zahnarzt: frei
Faher: austragen
Helfer: frei

Saturday
2024-05-25
00:00-12:00

Einsatz: Christuskirche
Am Steintor 30169 Hannover

Zahnarzt: frei
Faher: austragen
Helfer: frei

Einsatz Typ:

Einsatz ?

Einsatzort auswählen:

Suche Einsatzort

MAY 2024 < >

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Ausgewählte Tagen:

Einsatz hinzufügen

19 20 21 22 23 24 25

Einsatzort Empfehlung Feature

✕

Basierend auf den ausgewählte Einsatzorten und dem gewünschten Datum wird der beste Einsatzort für das entsprechende Datum vorgeschlagen

Einsatzort auswählen: +

Ausgewählte Einsatzorte:

- Caritas - Tagestreff Clemenskirche
- Ukrainische Flüchtlinge Messe

Datum auswählen: Choose a date +

5/25/2024

MM/DD/YYYY

Ukrainische Flüchtlinge Messe
✕

Einsatzort vorgeschlagen

Zahnmobil

Hilfe mit Biss!

ÜBERSICHT
STAMMDATEN
KALENDER
PLANUNG
TEAM
EINSATZORTE

Benutzerverwaltung

Benutzer suchen: Funktion: +

Nachname	Vorname	Funktion	E-Mail	Aktion
Mosavi	Muster	HELFER	test3@test.de	...
Muster	Muster	FAHRER	lawkorridor@gmail.com	...
Muster	John	ZAHNARZT	test2@test.de	...
Musterrmann	Max	FAHRER	test@test.de	...



Einsatzorte



Einsatzort auswählen:

Suche Einsatzort

- Männerwohnheim Schulenburger Landstraße - Männerwohnheim Schulenburger Landstraße
- Caritas - Kindersprechstunde - Caritas - Kindersprechstunde
- Christuskirche - Lutherkirche Str 16
- Charcoal Street BBQ e.V. - Charcoal Street BBQ e.V.
- Kopernickus - Am Kopernickus 13 30229 Hannover
- Tagestreff Nordstadt - Tagestreff Nordstadt
- Gesundheitsamt - Gesundheitsamt
- MESSE Hannover - Messegelände 30521 Hannover

Charcoal Street BBQ e.V. - Charcoal Street BBQ e.V.

Einsatzort hinzufügen

✕

Einsatzort Name:

Straße:

Hinweis Stellplatz:

Hinweis öffentlich:

Ansprechpartner vor Ort:

Einsatzzeit vor Ort: HH : MM bis HH : MM

Einsatzzeit Uhrzeit für Zahnarzt: HH : MM bis HH : MM

Einsatzzeit Uhrzeit für Fahrer: HH : MM bis HH : MM

Einsatzzeit Uhrzeit für Helfer: HH : MM bis HH : MM

Einsatzort hinzufügen

Frauenwohnheim Langensalza-Straße - Frauenwohnheim Langensalza-Straße

USB Content

Interview Protocol: Protocol developed to guide the interview process in Chapter 4.2.

1) Interview/Interview-Protocol.pdf

Specification: Developed Specification in Chapter 4.3.

1) Specification/Specification.pdf

Architecture Design: Domain and Application Design developed in Chapter 5.3.

1) Architecture/Domain-design.pdf

2) Architecture/Application-design.pdf

Testing: Exploratory testing documentation and a video showcasing the end-to-end test conducted by Cypress from Chapter 6.1.

1) Testing/Exploratory-Testing.pdf

2) Testing/End-To-End Testing.mp4

Survey: The survey guide and the survey for both the user and administrator, which were discussed in Chapter 6.2.1.

1) Survey/Survey Guide.pdf

2) Survey/Survey for User.pdf

3) Survey/Survey for Admin.pdf

Source Code:

1) zahnmobil

