Republic of Tunisia
The Ministry of Higher Education
and Scientific Research

The University of Carthage

The Higher Institute of Information and
Communication Technologies

END-OF-STUDY PROJECT REPORT

Submitted in Partial Fulfillment of the Requirements for the
BACHELOR DEGREE IN SOFTWARE ENGINEERING AND INFORMATION SYSTEMS

Field of Study : Software Engineering and Information Systems

Development of an Employment Services Web Platform

By
ISLEM BARGAOUI
YUSSEF MRABET

Conducted within ...

Code Cooperation

Academic Year: 2022-2023

Republic of Tunisia
The Ministry of Higher Education and Scientific Research

The University of Carthage

The Higher Institute of Information and Communication Technologies

END-OF-STUDY PROJECT REPORT

Submitted in Partial Fulfillment of the Requirements for the
BACHELOR DEGREE IN SOFTWARE ENGINEERING AND INFORMATION SYSTEMS

Field of Study : Software Engineering and Information Systems

Development of an Employment Services Web Platform

By
ISLEM BARGAOUI
YUSSEF MRABET

Conducted within....

Authorization of graduation project report submission:

Professional Supervisor:
Karim Hamrouni

Issued on :

Signature:

Academic Supervisor:
Fahima Ben Guirat

Issued on :

Signature:

Dedications

To my family **Mounir Bargaoui** and **Sawcen Ben Khayatia**,
thank you for your unwavering support and encouragement throughout my academic journey. Your belief in me has been the foundation upon which I have built my success, and I am forever grateful for your love and guidance,
I would like to express my gratitude to my friends for their unwavering support and encouragement. Your constant inspiration and motivation have been instrumental in my academic journey, and I am thankful for your presence in my life. .

Islem Bargaoui

With great pride, sincerity and affection, I dedicate this work to all the people who have significantly influenced my life. To my parents, who instilled in me the most important values in life and who have been my constant mentors throughout my educational and professional journey. To my sisters, who have supported and encouraged me in all my endeavors, and who have always given me the strength to pursue my dreams. To my closest friends, who have been a constant source of joy, laughter and support throughout my life. I also dedicate this work to all the people who believed in me, trusted me, and allowed me to grow as an individual. Let this modest work be an expression of my gratitude and love to all of you. I thank you for your moral support and encouragement throughout my journey. You have been strong pillars in my life and I am deeply grateful for your presence. I wish you all the happiness, peace and prosperity that you deserve. You are the reason I did this work, and I am honored to be able to share it with you all.

Yussef Mrabet

Thanks

To our Supervisors, **Dr. Fahima Ben Guirat and Karim Hamrouni**,
For believing in us and helping us finish our end of studies project, you have our most grateful thanks.

Contents

General introduction **1**

1 Context of the project **2**
1.1 Introduction . . . 2
1.2 Presentation of the host organization . . . 2
1.2.1 Host organization: . . . 2
1.2.2 Areas of activity . . . 3
1.3 Project presentation . . . 3
1.3.1 Project framework . . . 3
1.3.2 Study of the existing . . . 3
1.3.3 Proposed solution . . . 4
1.4 Project management Methodology . . . 4
1.4.1 Methodology . . . 4
1.5 Technical Choice . . . 6
1.5.1 Development Technologies & Frameworks . . . 6
1.5.2 Collaboration and Developement Tools . . . 9
1.6 General Architecture . . . 11
1.6.1 Deployment Diagram . . . 11
1.6.2 MSC Stack Architecture . . . 12
1.6.3 CI Architecture . . . 12
1.7 Conclusion . . . 13

2 Requirements Specification **14**
2.1 Introduction . . . 14
2.2 Requirements Specification . . . 14
2.2.1 Functional requirements . . . 14
2.2.2 Non-functional needs . . . 15
2.2.3 Identification of the actors . . . 16
2.3 Global Use Cases Diagram . . . 17
2.4 Product Backlog . . . 18
2.5 Conclusion . . . 19

Release 1: User Profile Management **20**
3.1 Introduction . . . 20
3.2 Sprint 0 . . . 20
3.2.1 Identification . . . 21
3.2.2 Conception . . . 24
3.3 Sprint 1 . . . 30
3.3.1 Identification . . . 30

3.3.2 Conception . . . 33
3.4 Sprint 2 . . . 36
3.4.1 Identification . . . 37
3.4.2 Conception . . . 39
3.5 Realization . . . 46
3.5.1 Realization of sprint 0 . . . 46
3.5.2 Realization of sprint 1 . . . 52
3.5.3 Realization of sprint 2 . . . 58
3.6 Conclusion . . . 66

Release 2: Job posts and Payment Integration 67
4.1 Introduction . . . 67
4.2 Sprint 3 . . . 67
4.2.1 Identification . . . 67
4.2.2 Conception . . . 71
4.3 Sprint 4 . . . 77
4.3.1 Identification . . . 77
4.3.2 Conception . . . 78
4.4 Sprint 5 . . . 81
4.4.1 Identification . . . 81
4.4.2 Conception . . . 82
4.5 Realization . . . 85
4.6 Conclusion . . . 92

Release 3: Enhanced User Experience 93
5.1 Introduction . . . 93
5.2 Sprint 6 . . . 93
5.2.1 Identification . . . 93
5.2.2 Conception . . . 95
5.3 Sprint 7 . . . 100
5.3.1 Identification . . . 100
5.3.2 Conception . . . 102
5.3.3 **Global Classes Diagram** . . . 109
5.4 Realization . . . 110
5.5 Conclusion . . . 114

General conclusion 115

List of Figures

1.1 Company Logo . . . 2
1.2 Scrum Structure[11] . . . 5
1.3 Next.JS[8] . . . 6
1.4 Nest.JS[7] . . . 6
1.5 Prisma ORM[10] . . . 6
1.6 MySQL[6] . . . 7
1.7 Material UI[5] . . . 7
1.8 OpenAPI[9] . . . 7
1.9 Sonar[13] . . . 8
1.10 Swagger[14] . . . 8
1.11 Digital Ocean[1] . . . 8
1.12 Visual Studio Code[15] . . . 9
1.13 Github Desktop[4] . . . 9
1.14 Github[3] . . . 10
1.15 Slack [12] . . . 10
1.16 Figma [2] . . . 10
1.17 Deployment Diagram . . . 11
1.18 MSC Diagram . . . 12
1.19 CI Diagram . . . 13
2.20 Global Use Cases Diagram . . . 17
3.1 Use case diagram of "create account" . . . 21
3.2 Use case diagram of "Login" . . . 22
3.3 Use case diagram of "Reset password" . . . 23
3.4 Class diagram of "Create account" . . . 24
3.5 Sequence diagram of "Create account" . . . 25
3.6 Class diagram of "Login" . . . 26
3.7 Sequence diagram of "Login" . . . 27
3.8 Class diagram of "Reset Password" . . . 28
3.9 Sequence diagram of "Reset password" . . . 29
3.10 Use case diagram of "Create profile" . . . 30
3.11 Use case diagram of "Consult Profile" . . . 32
3.12 Class diagram of "Create profile" . . . 33
3.13 Sequence diagram of "Create profile" . . . 34
3.14 Class diagram of "consult profile" . . . 35
3.15 Sequence diagram of "Consult profile" . . . 36
3.16 Use case diagram of "edit profile" . . . 37
3.17 Use case diagram of "edit account" . . . 38
3.18 Use case diagram of "delete account" . . . 39
3.19 Class diagram of "edit profile" . . . 40

3.20 Sequence diagram of "edit profile" 41
3.21 Class diagram of "edit account" 42
3.22 Sequence diagram of "edit account" 43
3.23 Class diagram of "delete account" 44
3.24 Sequence diagram of "delete account" 45
3.25 interface of "Create account" 1 46
3.26 interface of "Create account" 2 47
3.27 account confirmation email 48
3.28 interface of "Login" 49
3.29 interface of "Reset password" email input 49
3.30 email of password reset 50
3.31 interface of "reset password" new password input 51
3.32 on-boarding welcome message 53
3.33 Profile picture upload interface 53
3.34 Activity input interface 54
3.35 Gallery upload interface 54
3.36 Focus input interface 55
3.37 Bio input interface 55
3.38 PDF files upload interface 56
3.39 Personal strengths interface 56
3.40 User welcoming message 57
3.41 Profile interface 58
3.42 edit Profile interface 59
3.43 edit personal data interface 60
3.44 edit access data interface 61
3.45 edit email interface 61
3.46 password confirmation to update email 62
3.47 edit password interface 62
3.48 edit visibility settings interface 63
3.49 toggle newsletter setting interface 64
3.50 delete account interface 65
3.51 confirm account deletion interface 65
4.1 Use case diagram of the user story "Create a job post" 68
4.2 Use case diagram of the User Story "Edit a job post" 69
4.3 Use case diagram of the user Story "Delete a job post" 70
4.4 Class diagram of "Create a job post" 71
4.5 Sequence diagram of "Create a job post" 72
4.6 Class diagram of "Edit a job post" 73
4.7 Sequence diagram of "Edit a job post" 74
4.8 Class diagram of "Delete a job post" 75
4.9 Sequence diagram of "Delete a job post" 76
4.10 Use case diagram of the user story 'Pay for a job post' 77
4.11 Class diagram of "Pay for a job" 79
4.12 Sequence diagram of "Pay for a job post" 80
4.13 Use case diagram of the user story 'Consult users job posts' 81
4.14 Class diagram of "Consult users job posts" 83
4.15 Sequence diagram of "Consult users job posts" 84
4.16 Realization of use case "Create a job post" 1 85

4.17 Realization of use case "Create a job post" 2 85
4.18 Realization of use case "Edit a job post" 1 86
4.19 Realization of use case "Edit a job post" 2 86
4.20 Realization of use case "Delete a job post" 1 87
4.21 Realization of use case "Delete a job post" 2 87
4.22 payment plans interface 88
4.23 payment realisation interface 88
4.24 payment confirmation interface 89
4.25 interface of all posts that are Offering a job 90
4.26 interface of all posts that are Looking for a job 90
4.27 interface of all URGENT posts that are Offering a job 91
4.28 interface of all URGENT posts that are Looking for a job 91
4.29 Details of a job post 92
4.30 Details of an URGENT job post 92
5.1 Use case diagram of the user story "Search for job posts" 94
5.2 Use case diagram of the user story "filter job posts" 94
5.3 Class diagram of "Search for job posts" 96
5.4 Sequence diagram of "Search for job posts" 97
5.5 Class diagram of "Filter job posts" 98
5.6 Sequence diagram of "Filter job posts" 99
5.7 Use case diagram of the user story "Favorite job posts" 100
5.8 Use case diagram of the user story "Consult users profiles" 101
5.9 Use case diagram of the user story "Message users" 102
5.10 Class diagram of "Favorite job posts" 103
5.11 Sequence diagram of "Favorite job posts" 104
5.12 Class diagram of "Consult users profiles" 105
5.13 Sequence diagram of "Consult users profiles" 106
5.14 Class diagram of "Message users" 107
5.15 Sequence diagram of "Message users" 108
5.16 Global Classes Diagram 109
5.17 Realization of use case "Favorite job posts" 1 110
5.18 Realization of use case "Favorite job posts" 2 111
5.19 Realization of use case "Consult users profiles" 1 112
5.20 Realization of use case "Consult users profiles" 2 112
5.21 interface where the user contacts the job author 113
5.22 interface where the user inputs his message 113
5.23 message sent 114
5.24 Interface of the messenger 114

List of Tables

2.1 Product Backlog 18
3.1 Description of use case "Create account" 21
3.2 Description of use case "Login" 22
3.3 Description of use case "Reset password" 23
3.4 Description of use case "Create Profile" 31
3.5 Description of use case "Consult Profile" 32
3.6 Description of use case "Edit Profile" 37
3.7 Description of use case "Edit account" 38
3.8 Description of use case "Delete account" 39
4.1 Description of use case "Create Job Posts" 68
4.2 Description of use case "Edit job posts" 69
4.3 Description of use case "Delete job posts" 70
4.4 Description of use case "Pay for a Job Post" 78
4.5 Description of use case "Consult users job posts" 82
5.1 Description of use case "Search for job posts" 94
5.2 Description of use case "filter job posts" 95
5.3 Description of the use case "Favorite job posts" 100
5.4 Description of the use case "Favorite job posts" 101
5.5 Description of the use case "Message users" 102

General introduction

In today's fast-paced and competitive world, finding employment can be a challenging and daunting task. While job boards and employment agencies exist, they may not always provide the best fit for job seekers or employers.

To address this issue, we present TippJob, a platform that connects individuals or businesses offering jobs with those seeking employment opportunities. TippJob offers a simple and efficient way to post job offers, whether it's for a full-time position or a one-time task, such as mowing a lawn. By creating a user-friendly interface, TippJob aims to make the job search process easier and more accessible for all users.

Our end-of-studies project provides an opportunity to develop this platform further and improve the user experience. Throughout this report, we will discuss the context of the project, the requirements specifications, and the implementation process of TippJob.

During our internship at Code Cooperation, we have followed the scrum Methodology to better organize our work and development, thus making our report consist of these following chapters:

- "**Context of the project**", The initial chapter will serve as an introductory study and provide the project's contextual background as well as the infrastructure and general architecture of the project.
- "**Requirements Specification**", The second chapter of this report will be dedicated to outlining the project requirements, which is considered the first and most significant step in our development process. We will provide a detailed description of these requirements.
- "**Release 1: User Profile Management**", This first release aims to provide users with an intuitive and seamless experience when creating and managing their profiles, enabling them to maintain control over their data..
- "**Release 2: Job Posts and Payment Integration**", the second release adds payment integration and job posting features, improving the platform's user-friendliness and value.
- "**Release 3: Enhanced User Experience**", The third and final release will introduce new search filters and messaging functionalities, designed to enhance user interactions and facilitate communication among users.

Chapter 1

Context of the project

1.1 Introduction

This chapter provides an overview of the host organization, details the project developed, outlines the technology choices made, and explains the methodology followed.

1.2 Presentation of the host organization

1.2.1 Host organization:

Code Cooperation is a startup builder and software development agency co-founded by Mr Taib Ben Dai as a CTO and Mr Mutas Bezari as a CEO. Their mission is to help startups bring their ideas to life through the development and deployment of web and mobile applications. Code Cooperation has worked with notable clients such as GIZ and TRANSLATLY.

What sets Code Cooperation apart from other agencies is their commitment to long-term partnerships with startups. They are interested in becoming a founding partner of the client's startup and being paid by equity, rather than short-term earnings. Code Cooperation's main idea is to offer startups of different stages a scalable CTO and/or development support, ranging from building a complete application to separate modules, components, and/or features. They charge only their low self-costs for this work, demonstrating their dedication to the success of their clients.

Figure 1.1: Company Logo

1.2.2 Areas of activity

The company specializes in providing solutions for businesses and startups, drawing upon their expertise in three primary sectors: software engineering, technical assistance, and consulting. Some examples of their solutions include:

- Ilioa Music
- Translatly
- seniorenplatzfinder
- relokatehr

1.3 Project presentation

1.3.1 Project framework

In today's fast-paced world, finding and offering job opportunities can be a time-consuming and challenging process. To address this issue, we developed Tippjob, a platform that connects job seekers and employers in a simple and efficient way. The platform provides a user-friendly interface for job posting, browsing, and applying.

1.3.2 Study of the existing

There are already several online job platforms available to the public, such as LinkedIn, Glassdoor, and Indeed. However, these platforms have their limitations and drawbacks that may cause inconvenience to job seekers, employers, and recruiters. Some of these limitations include:

- Traditional job platforms typically focus on full-time, long-term positions, while neglecting short-term, one-off, or part-time job opportunities.
- Limitation to certain industries or job types, making the user's choices narrow for a wider range of job offers.
- Candidates have to go through a lengthy application and screening process, which may not be necessary or practical for one-time or short-term jobs.
- Focusing on connecting candidates with employers, and missing the ability for individuals to post job offers for services they need (e.g. lawn mowing, pet care, etc.).
- Requirement for job seekers to have an established professional profile or resume, which could be a barrier for individuals who are just starting out or looking for casual work.

1.3.3 Proposed solution

With the increasing demand for job opportunities and the limitations of current job posting platforms, TippJob offers a comprehensive solution that connects job seekers and job providers in an easy-to-use platform. Using cutting-edge technologies and an intuitive interface, TippJob offers the following features:

- Seamless job posting and search functionalities for both employers and job seekers.
- Having a wide range of job types, from short-term to one-off, limited only by the user's imagination.
- The ability be more general about industries and cover a wider range of job offers.
- The posting and application process is streamlined to be quick and easy.
- This solution is not limited to connecting candidates with employers but could also allow individuals to post job offers for services they need (e.g. lawn mowing, pet care, etc.).

1.4 Project management Methodology

1.4.1 Methodology

The decision has been made that Scrum is the most appropriate and well-structured methodology for this project. However, for those unfamiliar, what exactly is Scrum?
It is a framework that promotes team collaboration in planning, developing, delivering, and sustaining products in different work settings using an agile and organized approach. The framework encourages decision-making based on past experiences and continuous learning from current work, allowing for reflection and improvement. Despite its structured nature, the Scrum framework remains agile and flexible, as depicted in the accompanying figure.

Figure 1.2: Scrum Structure[11]

The figure 1.5 illustrates the main stages involved in utilizing the SCRUM framework, which are as follows:

1. The product owner typically leads the creation of the backlog.
2. Team members plan the order in which to work on features based on their level of importance, defining the sprints.
3. Daily Scrum meetings, attended by the scrum master and team members, are held to provide daily progress reports.

1.5 Technical Choice

1.5.1 Development Technologies & Frameworks

1.5.1.1 Next.JS

Next.js is a React framework that allows for server-side rendering and static site generation. It also includes features like automatic code splitting and optimized performance. It was used for the front-end of the project.

Figure 1.3: Next.JS[8]

1.5.1.2 Nest.js

Nest.js is a Node.js framework that provides a structured approach to building scalable and maintainable server-side applications. It was used for the back-end of the project.

Figure 1.4: Nest.JS[7]

1.5.1.3 Prisma

Prisma is a modern ORM for Node.js and TypeScript that provides a type-safe and easy-to-use database access layer. It was used to connect the Nest.js back-end to the MySQL database.

Figure 1.5: Prisma ORM[10]

1.5.1.4 MySQL

MySQL is a popular open-source relational database management system. It was used as the database for the project.

Figure 1.6: MySQL[6]

1.5.1.5 Material UI

Material UI is a React UI framework that provides pre-built components with a consistent design language based on Google's Material Design. It was used for the front-end of the project to create a modern and responsive user interface.

Figure 1.7: Material UI[5]

1.5.1.6 OpenAPI

OpenAPI is a specification for building APIs that allows for automated generation of documentation, client libraries, and server stubs. It was used to define the API endpoints for the project.

Figure 1.8: OpenAPI[9]

1.5.1.7 Sonar

Sonar is a platform for continuous code quality inspection that provides metrics, code analysis, and feedback on code quality. It was used to ensure code quality throughout the development process.

Figure 1.9: Sonar[13]

1.5.1.8 Swagger

Swagger is a toolset for designing, documenting, and testing APIs. It provides a user-friendly interface for defining API endpoints and parameters, and generates interactive documentation for developers. It also includes testing tools for verifying API behavior.

Figure 1.10: Swagger[14]

1.5.1.9 DigitalOcean

DigitalOcean is a cloud infrastructure provider that offers a range of cloud-based services for deploying and scaling web applications. It provides a simple and cost-effective platform, with a user-friendly interface and strong community support. it was used for the deployment of the project

Figure 1.11: Digital Ocean[1]

1.5.2 Collaboration and Developement Tools

1.5.2.1 Visual Studio Code (VScode)

Visual Studio Code (VScode) is a popular code editor that offers a wide range of features and extensions to enhance productivity and streamline development workflows. It supports multiple programming languages and provides features such as IntelliSense, debugging, version control, and more.

Figure 1.12: Visual Studio Code[15]

1.5.2.2 Github Desktop

Github Desktop is a user-friendly desktop client for Github that makes it easy to manage repositories, collaborate with team members, and perform basic Git operations such as committing, branching, and merging.

Figure 1.13: Github Desktop[4]

1.5.2.3 Github

Github is a web-based platform for version control and collaboration that enables developers to store and manage code repositories, track changes, collaborate with team members, and perform code reviews.

Figure 1.14: Github[3]

1.5.2.4 Slack

Slack is a popular team communication tool that allows teams to collaborate and communicate in real-time via channels, direct messages, and integrations with other tools. It offers features such as file sharing, video calls, and app integrations.

Figure 1.15: Slack [12]

1.5.2.5 Figma

Figma is a web-based design and prototyping tool that enables teams to collaborate on designing user interfaces, graphics, and other visual elements. It offers features such as vector editing, collaboration tools, and real-time design feedback.

Figure 1.16: Figma [2]

1.6 General Architecture

We will use a deployment diagram to illustrate the overall architecture of our project, which includes the MSC stack architecture.

1.6.1 Deployment Diagram

A deployment diagram is a UML diagram that offers a clear and concise way to illustrate the architecture of a system. It shows how the different software components are distributed across hardware and execution environments, as well as how they interact through middleware.
Our project's deployment diagram is displayed in the figure 1.17, offering a comprehensive overview of our system's architecture.

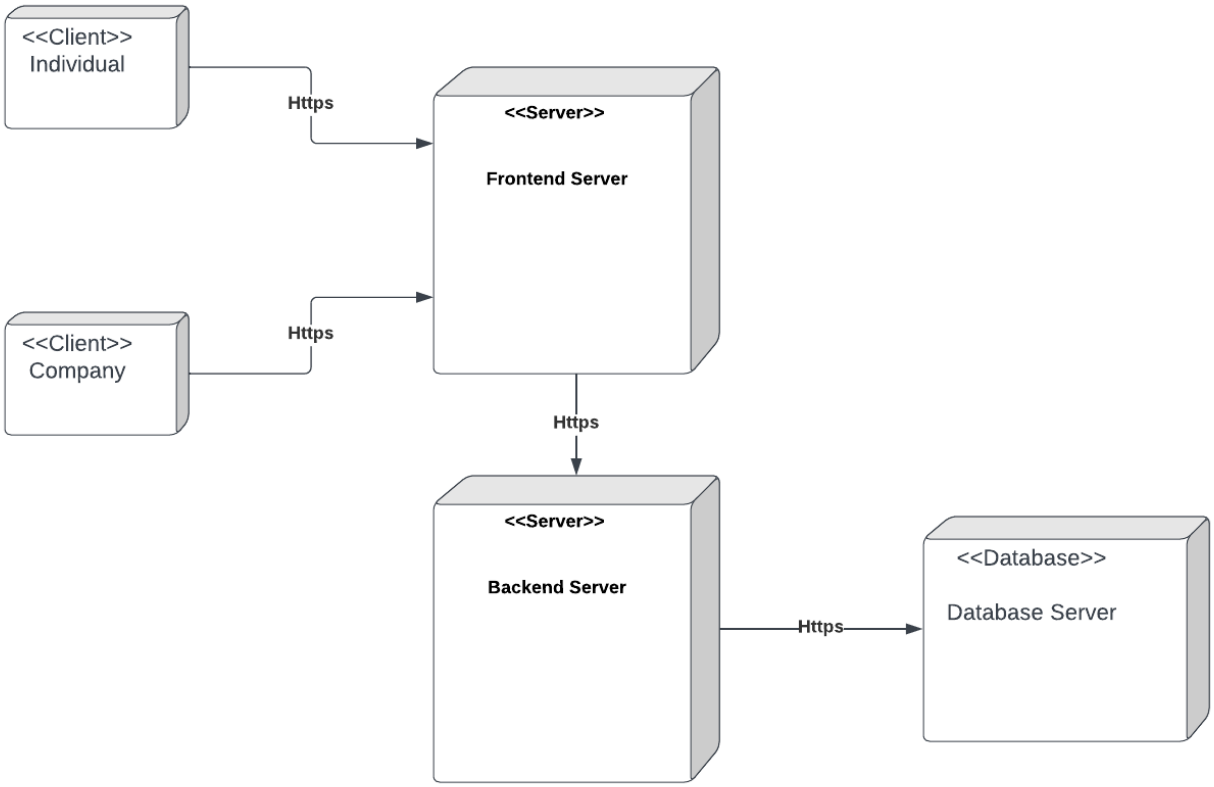

Figure 1.17: Deployment Diagram

1.6.2 MSC Stack Architecture

MSC, or Model-Service-Controller, is a variation of the traditional Model-View-Controller (MVC) architecture commonly used in web application development. In MSC, the View component is replaced by a Service component that encapsulates the business logic of the application. This approach promotes greater separation of concerns and can lead to a more modular and scalable application architecture.

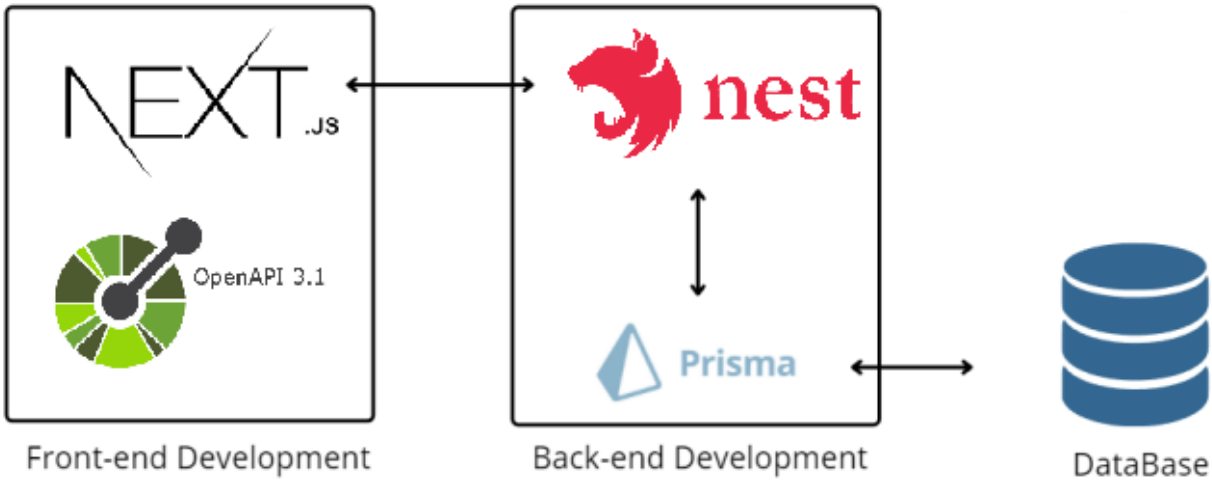

Figure 1.18: MSC Diagram

1.6.3 CI Architecture

Continuous Integration (CI) is a software development practice that involves frequently integrating and testing code changes to catch bugs and ensure that the software remains functional and stable throughout the development process. CI involves automating the build, test, and deployment process of software, allowing developers to quickly identify and fix issues as they arise. By integrating code changes frequently, CI helps to prevent integration problems that can occur when multiple developers work on the same codebase. This results in faster development cycles, higher quality software, and more efficient use of resources. the figure 1.19 represents our CI process.

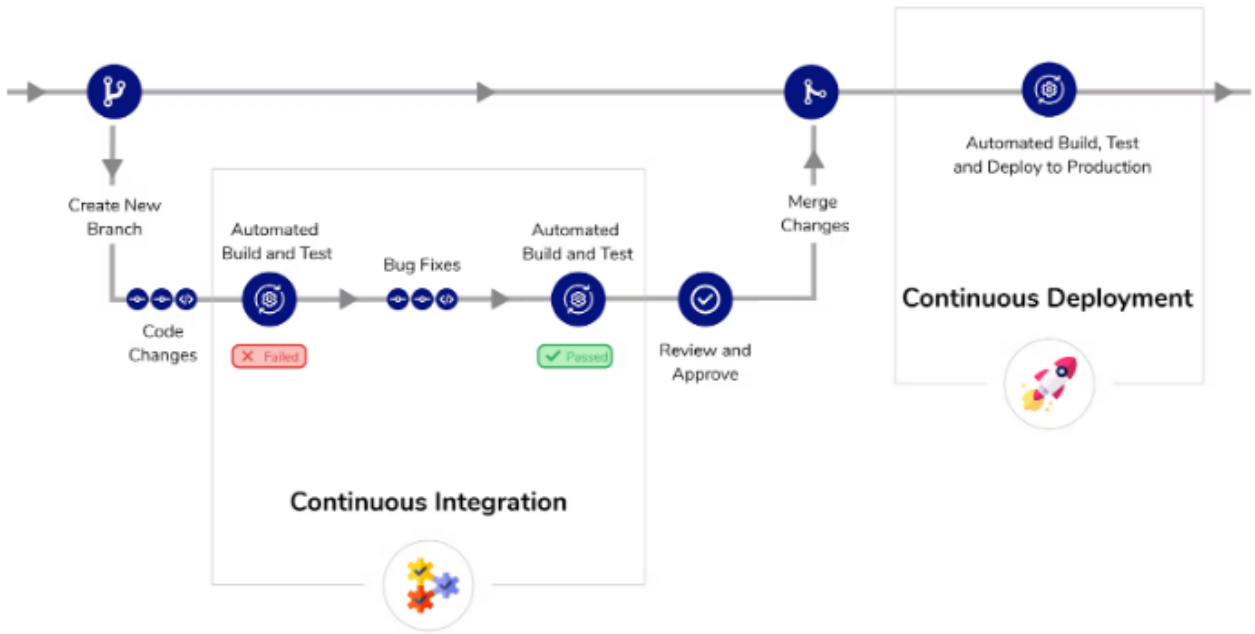

Figure 1.19: CI Diagram

1.7 Conclusion

This chapter outlines the overall framework of the internship program, and serves as a segue to the subsequent phase of needs specification and planning.

Chapter 2

Requirements Specification

2.1 Introduction

In this chapter, we begin our work by defining the project's requirements and identifying the different actors involved. We then create the general use case diagram, which is followed by the development of the product backlog. The product backlog serves as a critical tool in explaining the different aspects of our work.

2.2 Requirements Specification

Our primary objective is to develop and launch a functional and successful product, which necessitates a meticulous and systematic approach to requirement analysis. This involves clearly defining the software's intended functionality, performance expectations, and the necessary features to meet the requirements of our users. These needs can be classified into two distinct categories:

2.2.1 Functional requirements

Functional requirements refer to the specific services and functions that a software must provide. It is important to clearly define them for both the development team and users to ensure that the software meets their expectations.
In our project, we have identified the following functional requirements that our software must adhere to:

- **Register:** Users can create a new account with their personal information.
- **Login:** Users can access their account with their email and password.
- **Reset Password:** Users can reset their password in case they forgot it.
- **Manage profile:** During the onboarding process, users are given the ability to create a personalized profile page which includes a profile gallery, profile picture, and information. Additionally, users can easily update or delete their profile information as needed (account info included).
- **Edit user info and settings:** Users can modify their personal information and settings including visibility to other users.
- **Consult posts:** Users can view posts published by other users.

- **Consult users profiles:** Users can view profiles of other users.
- **Manage Posts:** Users can create, edit, and delete their own posts, they can be urgent or normal posts.
- **Pay for a Job Post:** The payment system integrated into the web app empowers the user to extend the duration of their job offer's visibility beyond the default time frame. This functionality provides the user with greater flexibility and control over their job postings, enabling them to adapt to unforeseen circumstances and maximize their chances of finding suitable candidates.
- **Favorite posts:** Users can mark posts as their favorites to access them easily later.
- **Search for posts:** Users can search for posts by keywords or tags.
- **Filter posts:** Users can filter posts by different criteria such as date, category, and author.
- **Message users:** Users can send private messages to other users.

2.2.2 Non-functional needs

A nonfunctional requirement refers to a limitation or condition imposed on a system that is not related to its specific functionalities. Instead, it pertains to aspects such as the environment, implementation, performance, platform dependencies, user-friendliness, maintenance, scalability, and reliability of the system. These constraints play a significant role in the overall success of the software and must be taken into account during the development process.

- **Performance:** The solution must be designed to load quickly and respond promptly to user requests. This means that the system should be able to handle high traffic without compromising performance.
- **Security:** The solution must be secure and protect user data against unauthorized access, data breaches, and cyber attacks. This includes using encryption for sensitive data, implementing authentication and authorization mechanisms, and following industry best practices for security.
- **Reliability:** The solution must be available and operational 24/7. This means that the system should be designed with redundancy and failover mechanisms to ensure that the application is always accessible and that any disruptions are minimal.
- **Scalability:** The solution must be scalable to accommodate growth and changes in user traffic. This means that the system should be designed to handle an increasing number of users and data volume without experiencing performance degradation.

- **Usability:** The web application should be easy to use and intuitive for all users, regardless of their technical knowledge or background. This includes designing a user interface that is easy to navigate, providing clear and concise instructions.
- **Maintainability:** The solution must be easy to maintain, update, and modify without causing disruptions to its functionality or performance.

2.2.3 Identification of the actors

User

The web app is designed to cater to a single actor, which can either be an INDIVIDUAL or a COMPANY. Both actors have access to the same functionalities; the only difference being that the COMPANY has a designated company name while the INDIVIDUAL does not. The user can seamlessly interact with all aspects and features of the web app, regardless of their actor type.

2.3 Global Use Cases Diagram

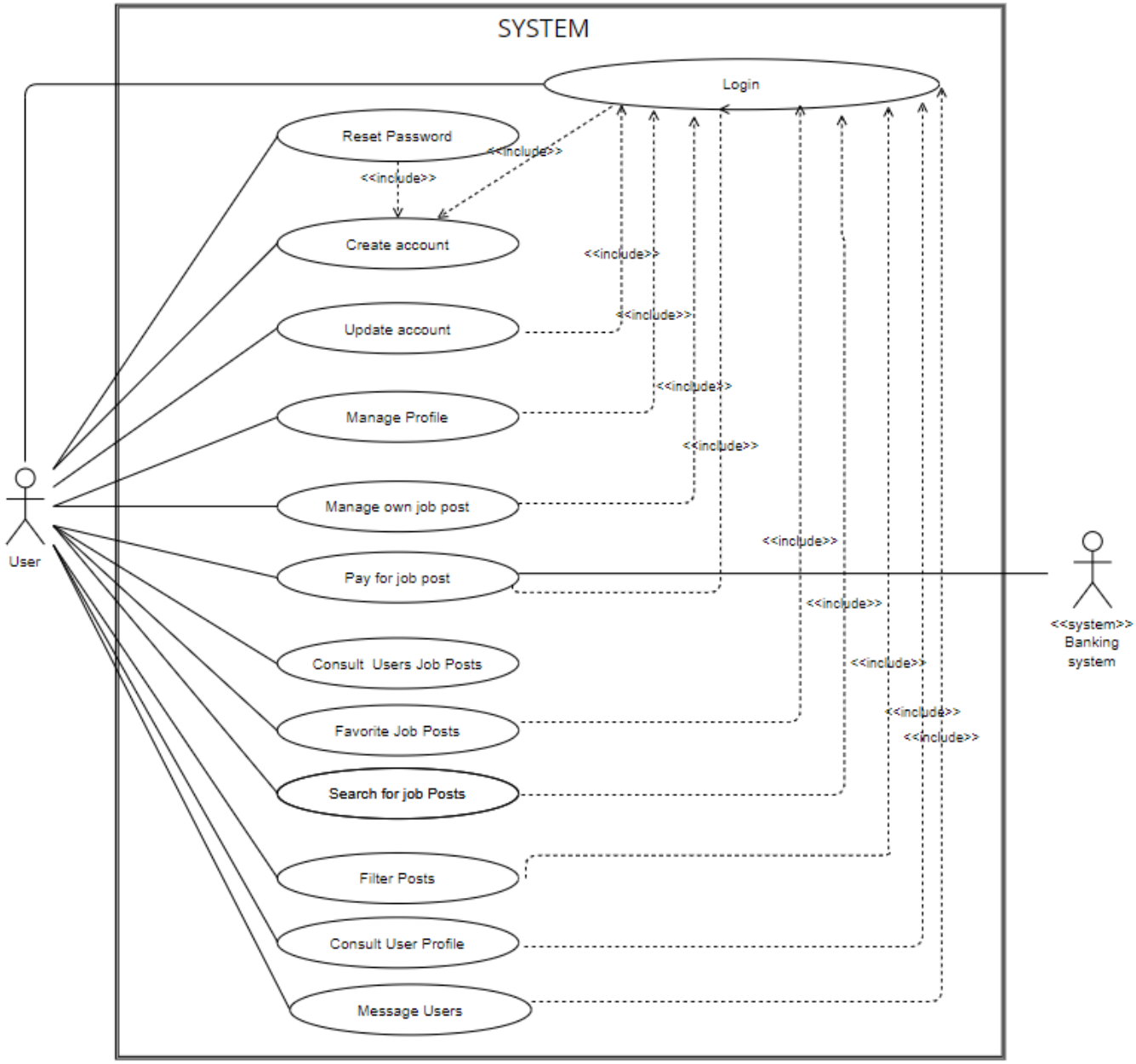

Figure 2.20: Global Use Cases Diagram

2.4 Product Backlog

the following table shows our product backlog :

User Story	Priority	Estimation	Planning	
As a user i can Register	1	High	Sprint 0	**Release 1**
As a user i can Login	1	High	Sprint 0	
As a user i can Reset my password	1	Medium	Sprint 0	
As a user i can create my profile through the onboarding proccess	2	High	Sprint 1	
As a user i can consult my profile	2	Medium	Sprint 1	
As a user i can edit my profile (profile picture, gallery, information)	3	Medium	Sprint 2	
As a user i can edit my account settings (personal information, Visibility, email, password, newsletter)	3	Medium	Sprint 2	
As a user i can delete my account permanently	3	Medium	Sprint 2	
As a user i can post a job offer	4	High	Sprint 3	**Release 2**
As a user i can edit my job offers	4	Medium	Sprint 3	
As a user i can delete my job offer	4	Medium	Sprint 3	
As a user i can pay for my job offers	5	Medium	Sprint 4	
As a user i can consult offers and requests posts	6	Medium	Sprint 5	
As a user i can search for job offers	7	High	Sprint 6	**Release 3**
As a user i can filter job offers	7	Medium	Sprint 6	
As a user i can favorite job offers	8	Medium	Sprint 7	
As a user i can consult other users profiles	8	Medium	Sprint 7	
As a user i can exchange messages with another user	8	High	Sprint 7	

Table 2.1: Product Backlog

2.5 Conclusion

This chapter has allowed us to identify the key Users and their requirements for the project. Subsequently, we established a Product Backlog that served as the foundation for the development of our sprints. This analysis will serve as the basis for our work going forward into Chapter 3. In the upcoming chapter, we will focus on the first release, "User Profile Management".

Chapter 3

Release 1: User Profile Management

3.1 Introduction

This release marks the initial development phase of our project, "User Profile Management". Our focus was on laying the groundwork for the platform's future development, with a particular emphasis on user registration, login, and Account/Profile management. The platform offers users a personalized experience, with a profile page where they can consult and edit their profile information, as well as a settings page where they can edit login details, personal information, and delete their account.

This chapter will detail the 3 sprints related to this release, with a thorough explanation of each use case refined and conceived in each of the sprints. For each use case, we will provide the refinement and conception, including use case diagrams, textual descriptions, class diagrams, and sequence diagrams. While we did not have design decisions for this release, we will provide a comprehensive overview of the development process throughout the 3 sprints.

3.2 Sprint 0

The user stories that follow outline the scope of our first sprint.

- As a user, I can Register.
- As a user, I can Login.
- As a user, I can Reset my password.

3.2.1 Identification

This section provides a collection of use case diagrams for our use cases, accompanied by textual descriptions in the form of tables

3.2.1.1 Refinement of the use case "Create account"

The Figure 3.1 shows the use case diagram for the user story "Create account"

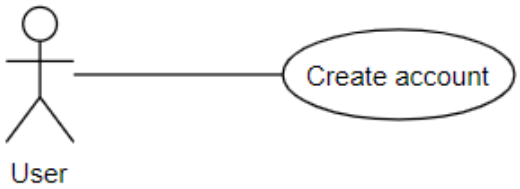

Figure 3.1: Use case diagram of "create account"

The table 3.1 describes the use case "Create account" textually in the form of a table

Use Case Scenario	Create account
Actors	User
Pre-conditions	The actor has to have an E-mail address
Post-conditions	The user account is created and ready for login
Main scenario	1- The actor types in their personal information. 2- The actor clicks "Register/Jetzt registrieren". 3- The server verifies the information. 3- If the information is valid, the server sends a verification email to the actor's E-mail address. 4- The actor clicks on "Anmeldung bestätigen" in the email to confirm their account.
Exception	if the information is not valid, the server alerts the user

Table 3.1: Description of use case "Create account"

3.2.1.2 Refinement of the use case "Login"

The figure 3.2 represents the use case diagram of the user story "Login" which depends on the use case "Create account"

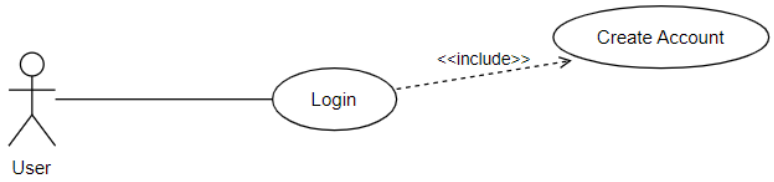

Figure 3.2: Use case diagram of "Login"

The table 3.2 shows the description of the use case "Login" textually in the form of a table.

Use Case Scenario	Login
Actors	User
Pre-conditions	The actor has to have a verified account
Post-conditions	Login tokens are stored in the browser and the user is logged in.
Main scenario	1- The actor types in his credentials 2- The actor clicks "Login/Anmelden" 3- The server verifies the credentials 4- if the credentials are correct, the server then creates a session for that account.
Exception	if the credentials are wrong, the server alerts the user.

Table 3.2: Description of use case "Login"

3.2.1.3 Refinement of the use case "Reset password"

The figure 3.3 presents the use case diagram of the user story "Reset password" which is dependent on the use case "Create account"

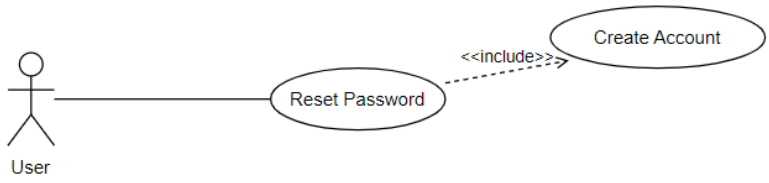

Figure 3.3: Use case diagram of "Reset password"

The table 3.3 describes the use case "Reset password" textually in the form of tables.

Use Case Scenario	Reset password
Actors	User
Pre-conditions	The actor has to have an account The login modal is open
Post-conditions	Password is changed
Main scenario	1- the actor clicks on "Forgot my password?/ Passwort vergessen?". 2- the actor types in his email address. 4- the actor clicks on "send/Senden". 3- the server verifies the email. 3- if the email exists, the server sends an E-mail to the user's address. 4- the actor clicks on "Anmeldung bestätigen" in the email 5- the actor enters a new password and re-enters it again. 6-if the passwords match, the new password is set.
Exception	1-if the passwords do not match, an error is displayed. 2- if the email doesn't exist, the server alerts the user

Table 3.3: Description of use case "Reset password"

3.2.2 Conception

In this step, we establish how the interactions and use cases identified in the previous step should occur. It is a crucial stage that lays the foundation for the development process. To represent our logic in a clear and consistent manner, we utilize Class and Sequence Diagrams. These diagrams help formalize our approach and provide a visual representation of the system's components and their interactions.

3.2.2.1 Class Diagram of the User Story "Create account"

The figure 3.4 represents the class diagram of the use case "Create account"

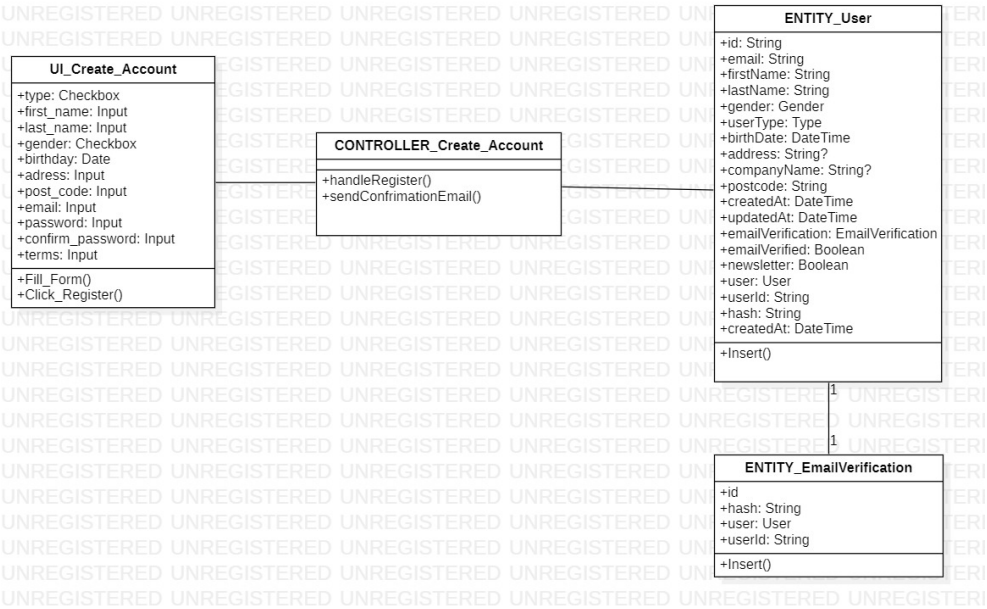

Figure 3.4: Class diagram of "Create account"

3.2.2.2 Sequence Diagram of the User Story "Create account"

Figure 3.5 presents the registration process, allowing users to create their own accounts and access the platform's features.

- The registration process involves the user entering personal information, email verification by the server, and confirmation of the account through a button in the confirmation email.
- if the email already exists in the database, the server sends an error message and prompts the user to try again with a different email address.

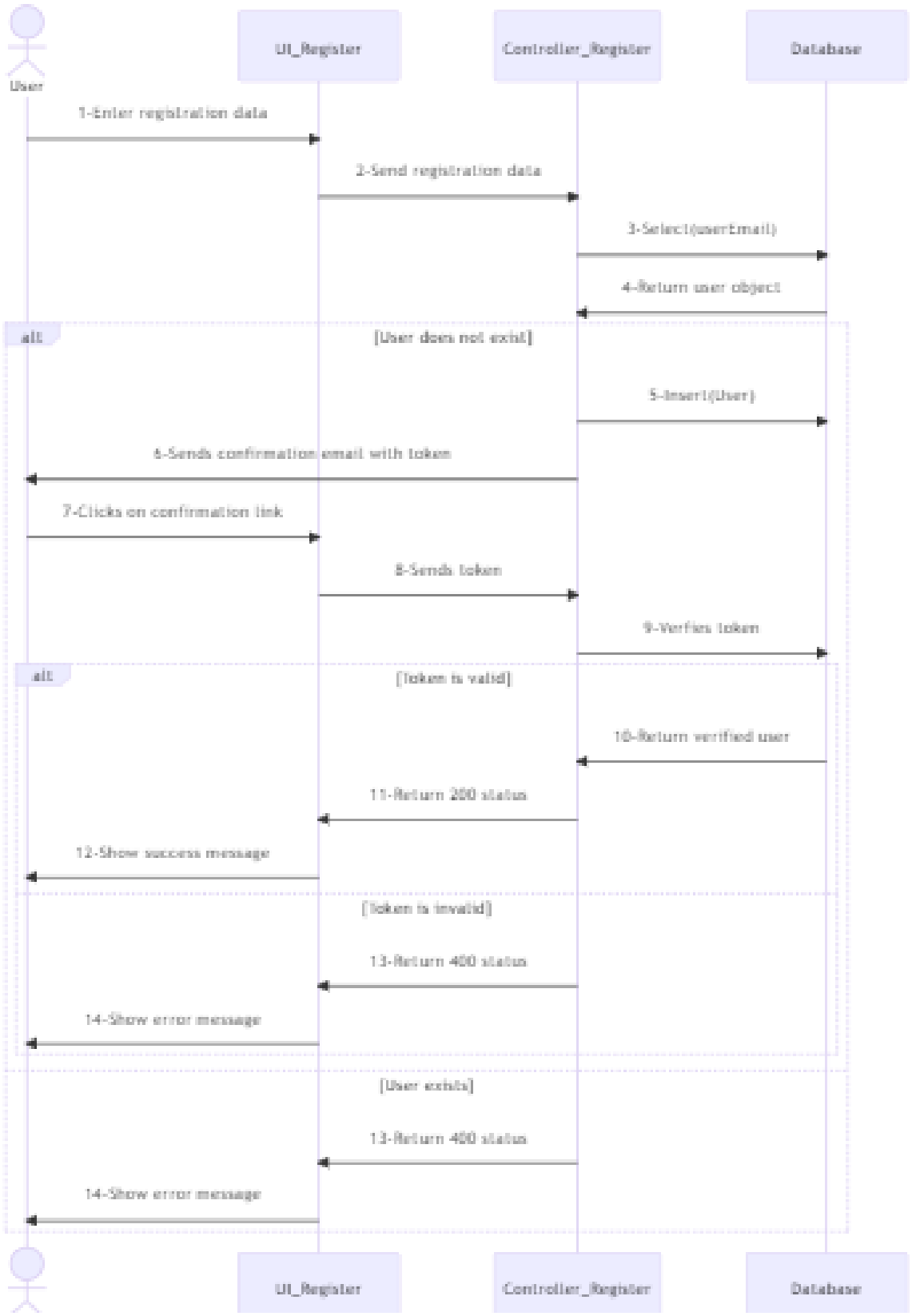

Figure 3.5: Sequence diagram of "Create account"

3.2.2.3 Class Diagram of the User Story "Login"

The figure 3.6 represents the class diagram of the use case "Login"

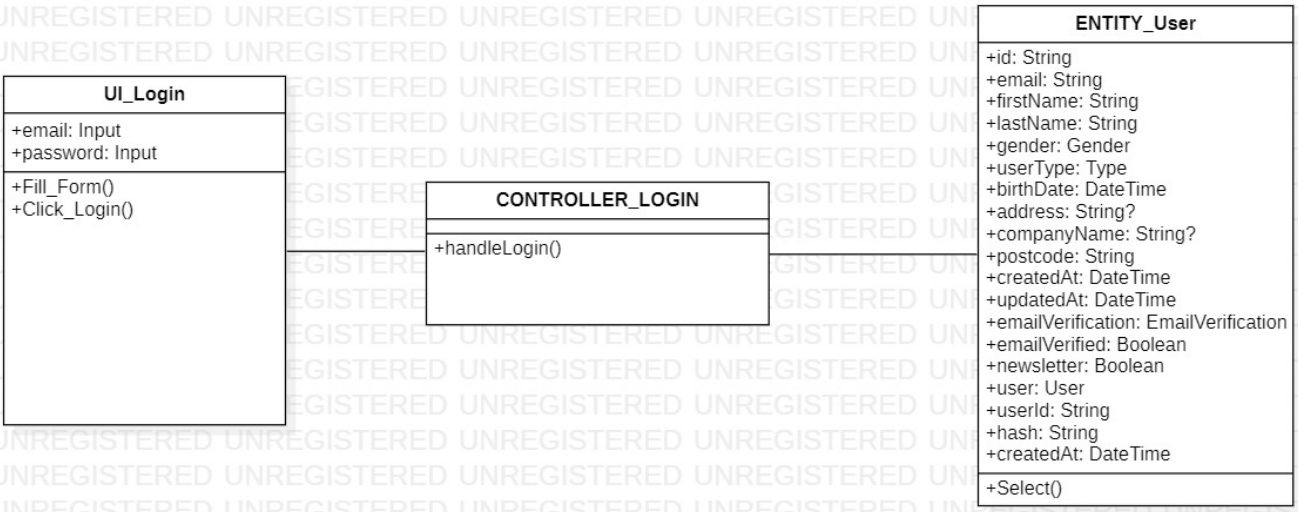

Figure 3.6: Class diagram of "Login"

3.2.2.4 Sequence Diagram of the User Story "Login"

Figure 3.7 presents the Login procedure.

- To initiate the Login process, the user must provide their login credentials in the designated form and subsequently click on the login button. Once submitted, the controller undertakes verification of the user's data, changing the navbar status if authentication is successful.
- In the event of failed authentication, the system displays an alert indicating that the entered credentials are incorrect and prompts the user to try again.

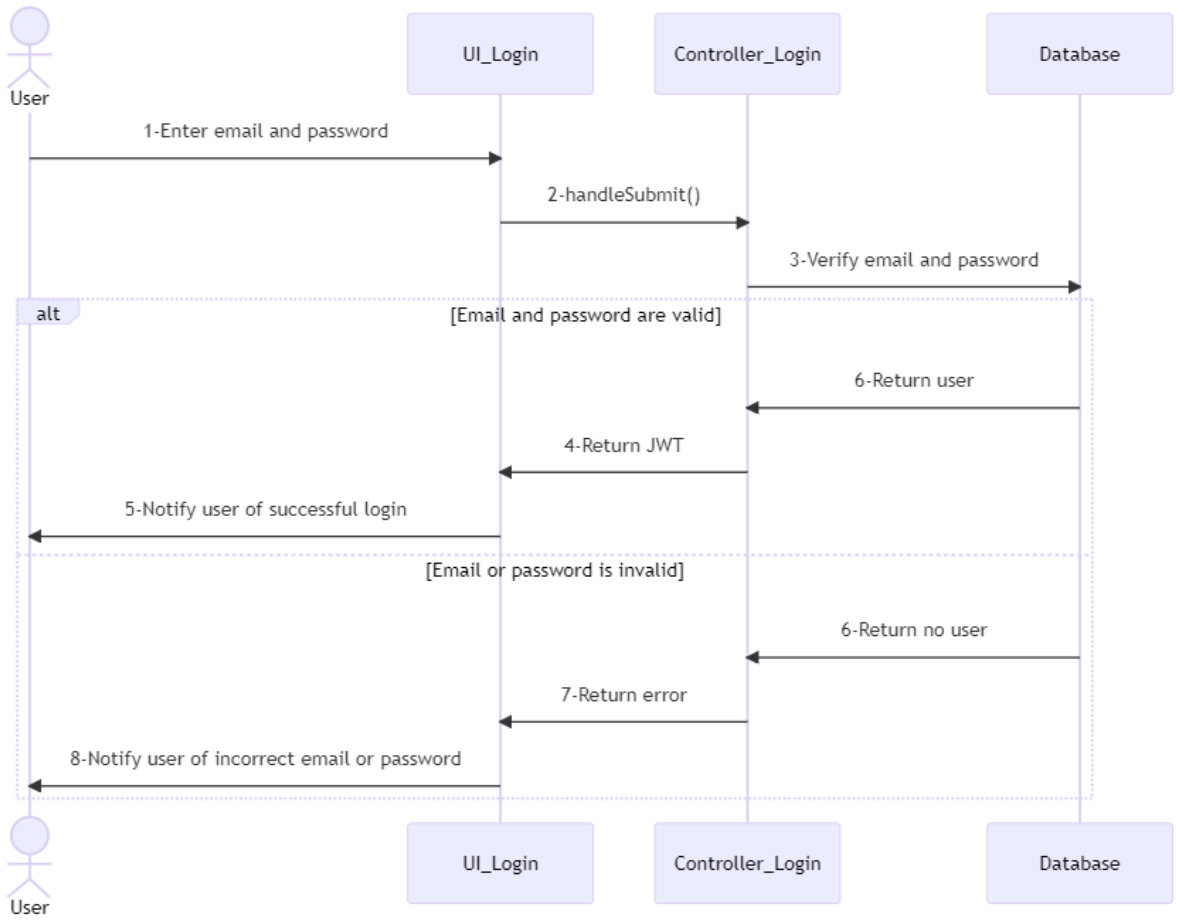

Figure 3.7: Sequence diagram of "Login"

3.2.2.5 Class Diagram of the User Story "Reset password"

The figure 3.8 presents the class diagram of the use case "Reset password".

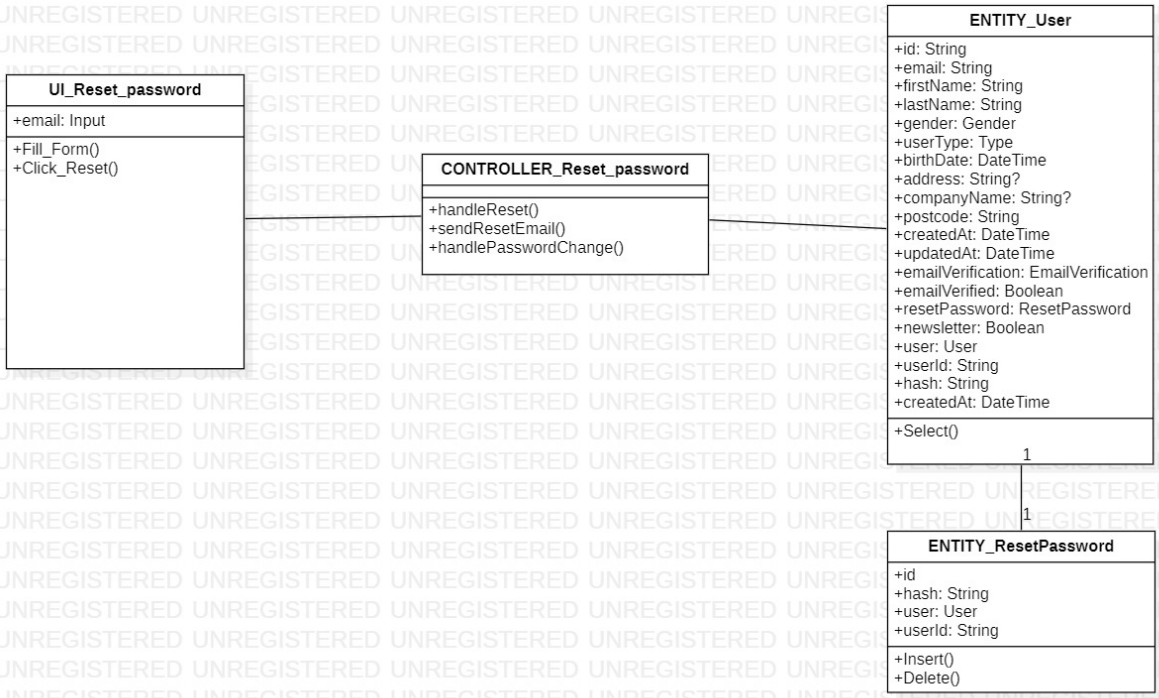

Figure 3.8: Class diagram of "Reset Password"

3.2.2.6 Sequence Diagram of the User Story "Reset password"

Figure 3.9 presents the reset password procedure, the user has to have an account in order to reset his password.

- The reset password feature allows the user to reset their password in case they forget it. To use this feature, the user clicks on the "Forgot my password?" link on the login modal and enters their email address. The server verifies the email and sends a confirmation email to the user's address.
- The user confirms their account by clicking on the button in the email and is prompted to enter a new password. If the new password is entered correctly, it is set and the user can log in with their new password.
- If there is an error in the process, an error message is displayed to the user.

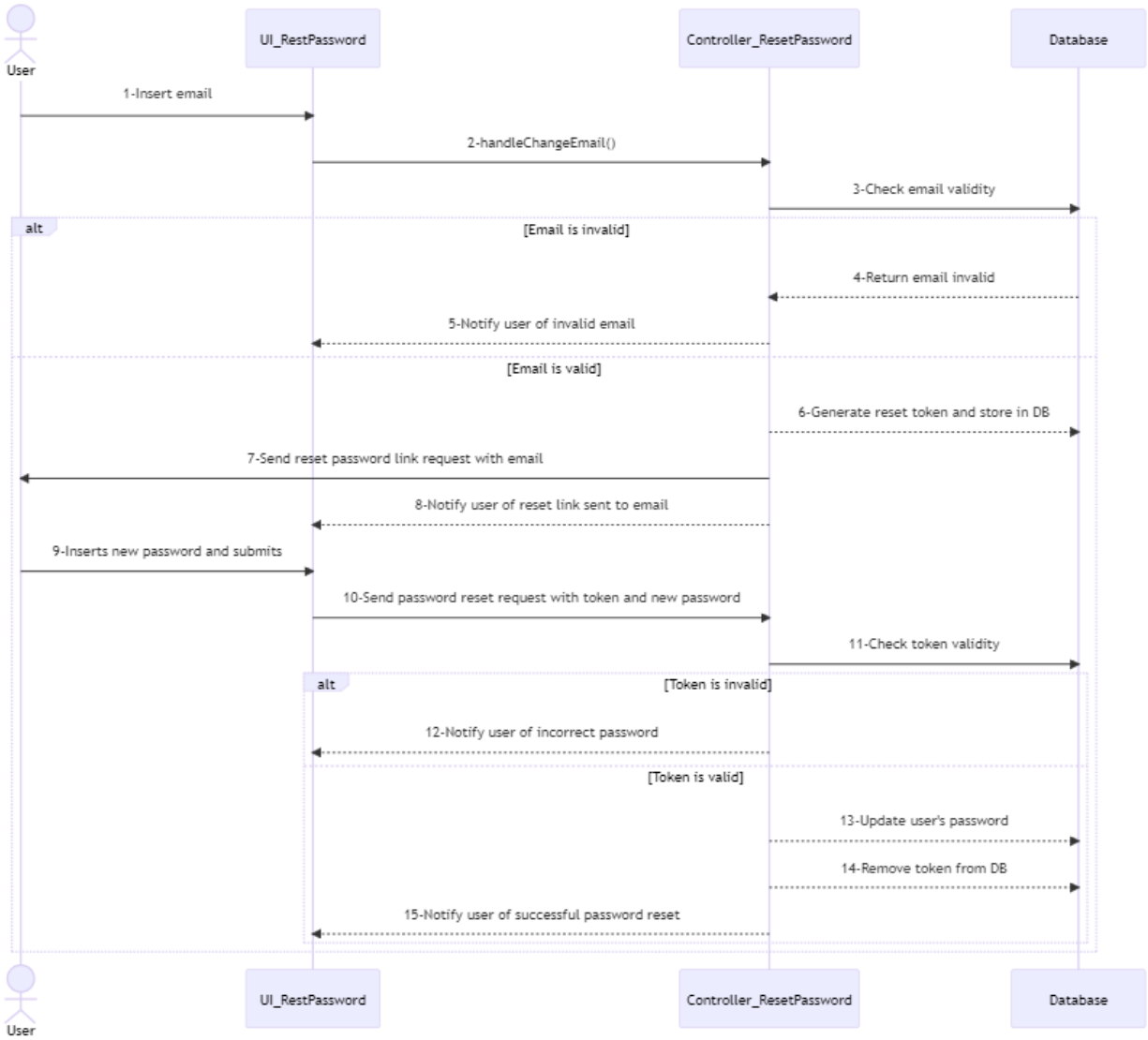

Figure 3.9: Sequence diagram of "Reset password"

3.3 Sprint 1

The set of user stories below outlines the objectives of our second sprint:

- As a user, I can create my profile through the on-boarding process.
- As a user, I can consult my profile page.

3.3.1 Identification

This section provides a collection of use case diagrams for our use cases, accompanied by textual descriptions in the form of tables.

3.3.1.1 Refinement of the use case "Create profile"

The figure 3.10 shows the use case diagram of the user story "Create profile" which depends on the use case "Login"

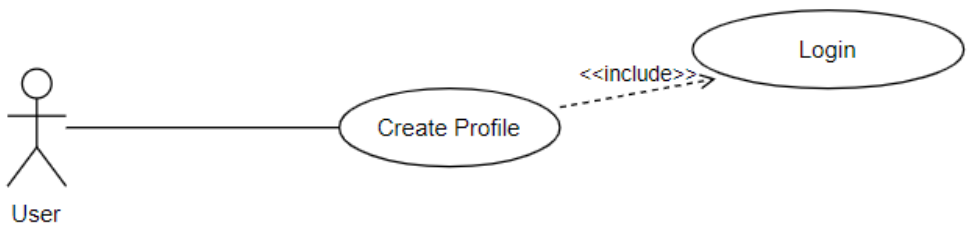

Figure 3.10: Use case diagram of "Create profile"

The table 3.4 describes the use case "Create profile" textually in the form of tables.

Use Case Scenario	Create profile
Actors	User
Pre-conditions	The actor has to be logged in The user needs to not have any profile info set up
Post-conditions	Profile info is created
Main scenario	1- the actor uploads his profile picture. 2- the actor clicks "Save/Speichern" 3-the server uploads the data to the database 4- the actor selects and enters his activity information. 5- the actor clicks "Save/Speichern" 6- the server uploads the data to the database 7- the actor uploads pictures to add to his gallery. 8- the actor clicks "Save/Speichern" 9-the server uploads the data to the database 10-the actor enters his focus information 11-the actor clicks "Save/Speichern" 12-the server uploads the data to the database 13-the actor inputs his bio 14- the actor clicks "Save/Speichern" 15- the server uploads the data to the database 16- the user uploads his pdf files 17- the user clicks "Save/Speichern" 18- the server uploads the data to the database

Table 3.4: Description of use case "Create Profile"

3.3.1.2 Refinement of The use case "Consult Profile"

The figure 3.11 shows the use case diagram of the user story "Consult profile" that depends on the use case "Login" of course.

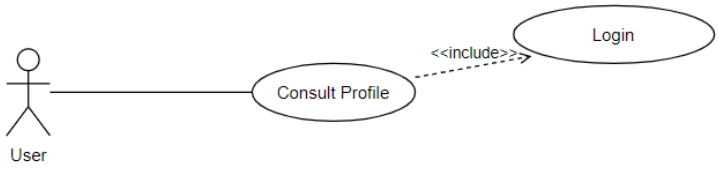

Figure 3.11: Use case diagram of "Consult Profile"

The table 3.5 describes the use case "Consult profile" textually in the form of tables.

Use Case Scenario	Consult profile
Actors	User
Pre-conditions	The actor has to be logged in
Post-conditions	The user's profile page is displayed
Main scenario	1- the actor clicks on his profile picture in the navbar 2- the profile picture menu is displayed 3- the actor clicks on "Profile/Profil" 4- the user's profile page is displayed

Table 3.5: Description of use case "Consult Profile"

3.3.2 Conception

In this step, we use Class and Sequence Diagrams to provide a visual representation of the system's components and their interactions.

3.3.2.1 Class Diagram of the User Story "Create profile"

The figure 3.12 presents the class diagram of the use case "Create profile".

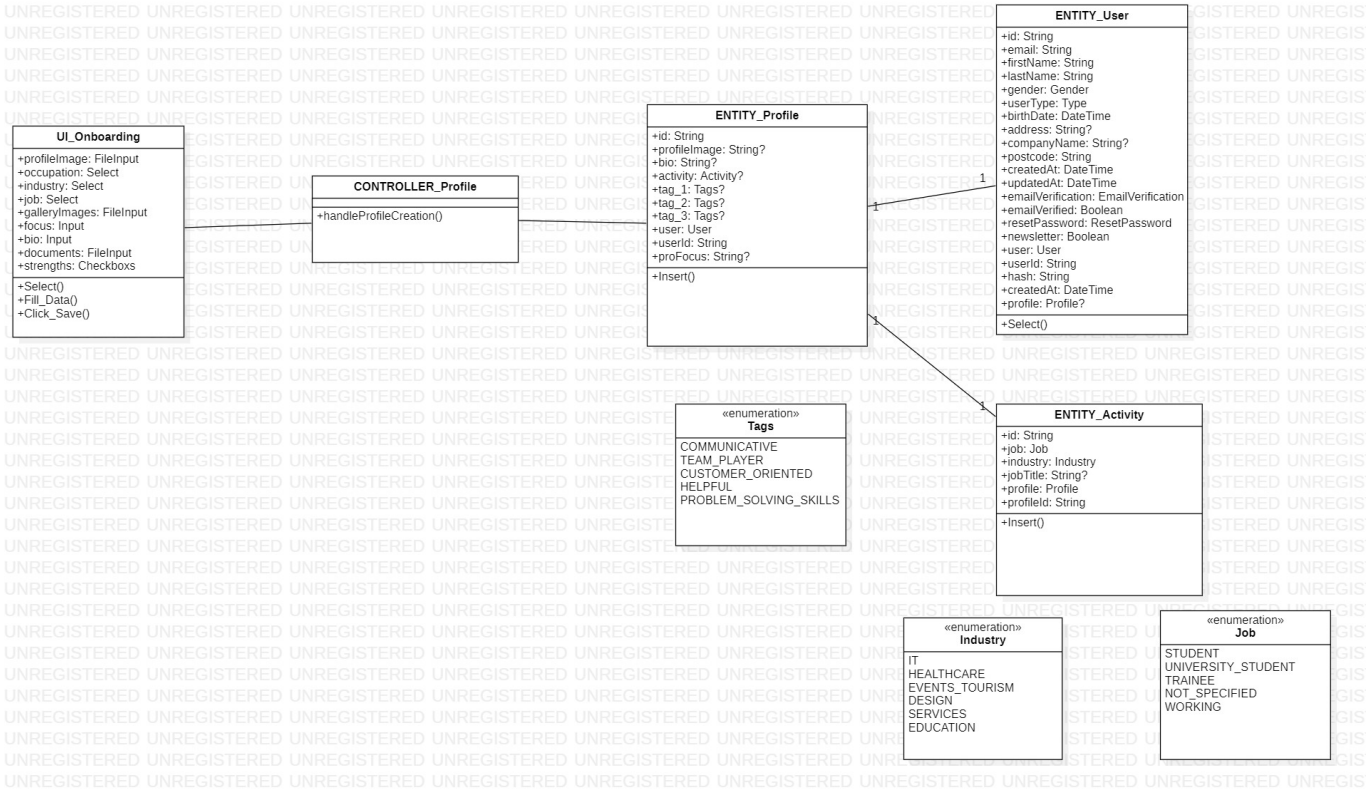

Figure 3.12: Class diagram of "Create profile"

3.3.2.2 Sequence Diagram of the User Story "Create profile"

Figure 3.13 presents the Create profile procedure, the user has to be logged in to access this user-story and he has to have never logged in before, this procedure is called the on-boarding process. It consists of a step by step modal stepper so the user creates his profile information one by one and send them to the database individually.

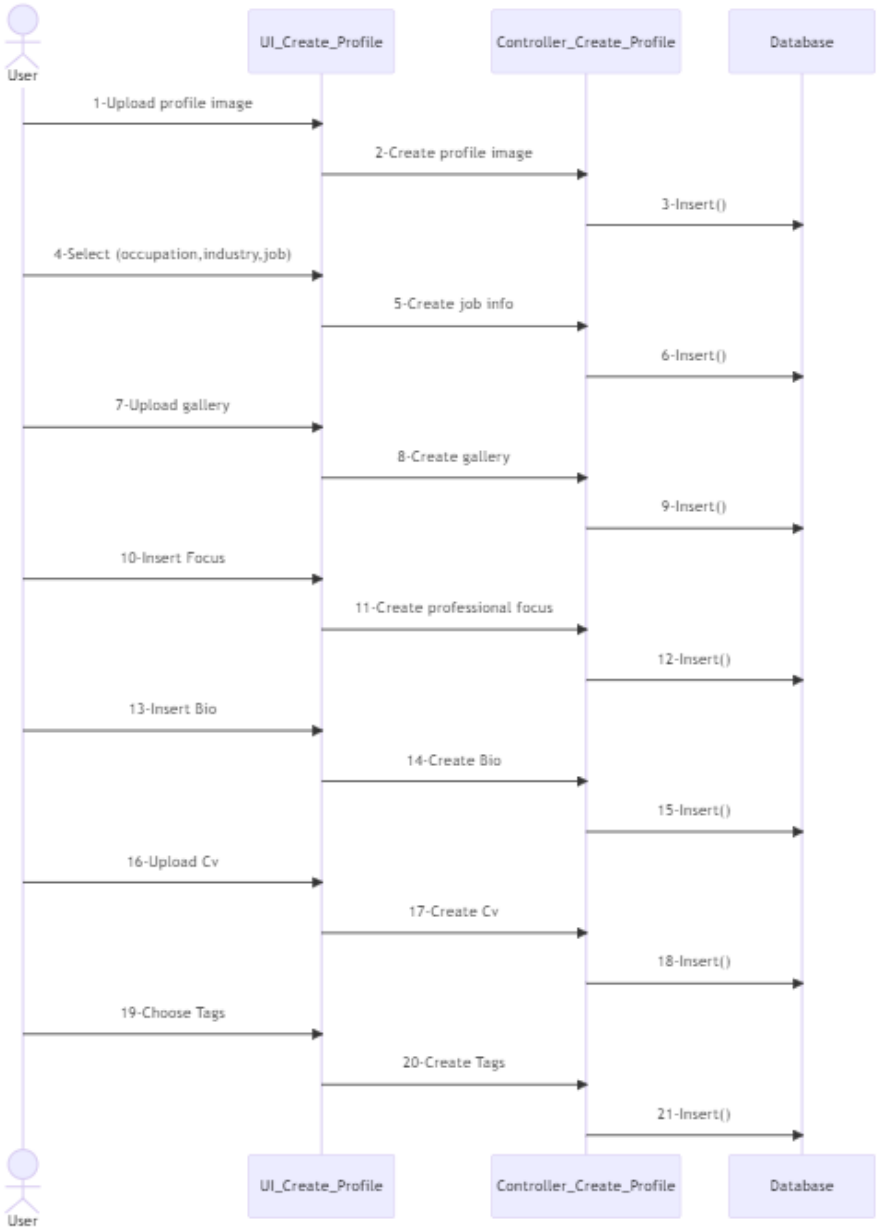

Figure 3.13: Sequence diagram of "Create profile"

3.3.2.3 Class Diagram of the User Story "Consult profile"

The figure 3.14 represents the class diagram of the use case "Consult profile"

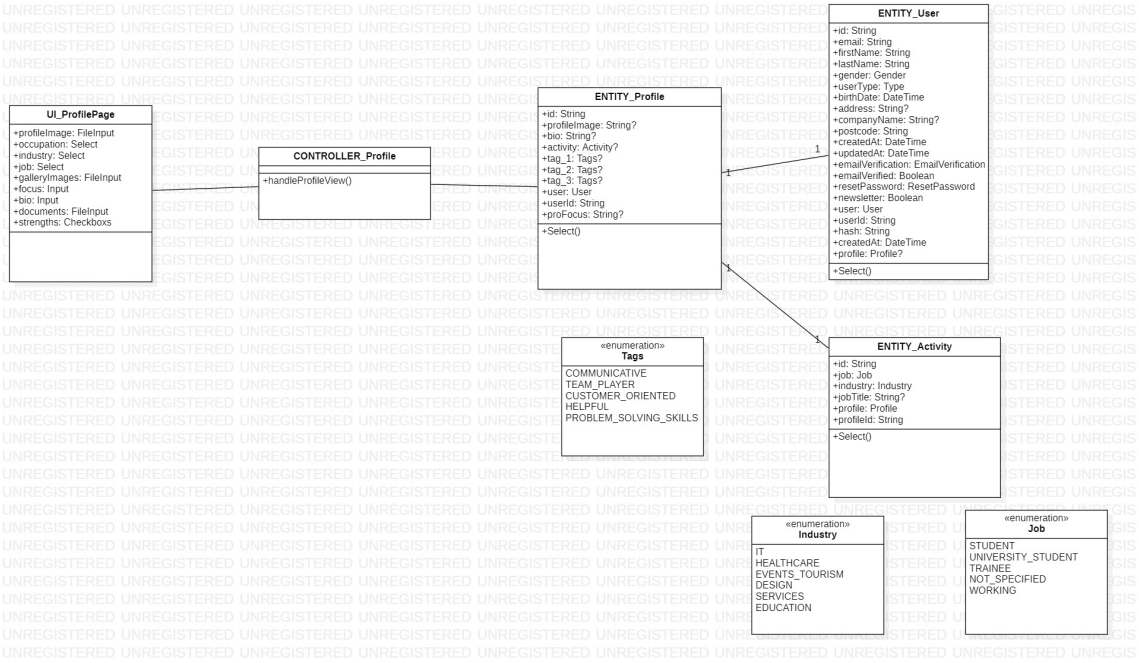

Figure 3.14: Class diagram of "consult profile"

3.3.2.4 Sequence Diagram of the User Story "Consult profile"

Figure 3.15 presents the consult Profile procedure, the user has to be logged in to access this user-story. This is the profile page interaction use Case which will be related to edit profile info in the next sprint.

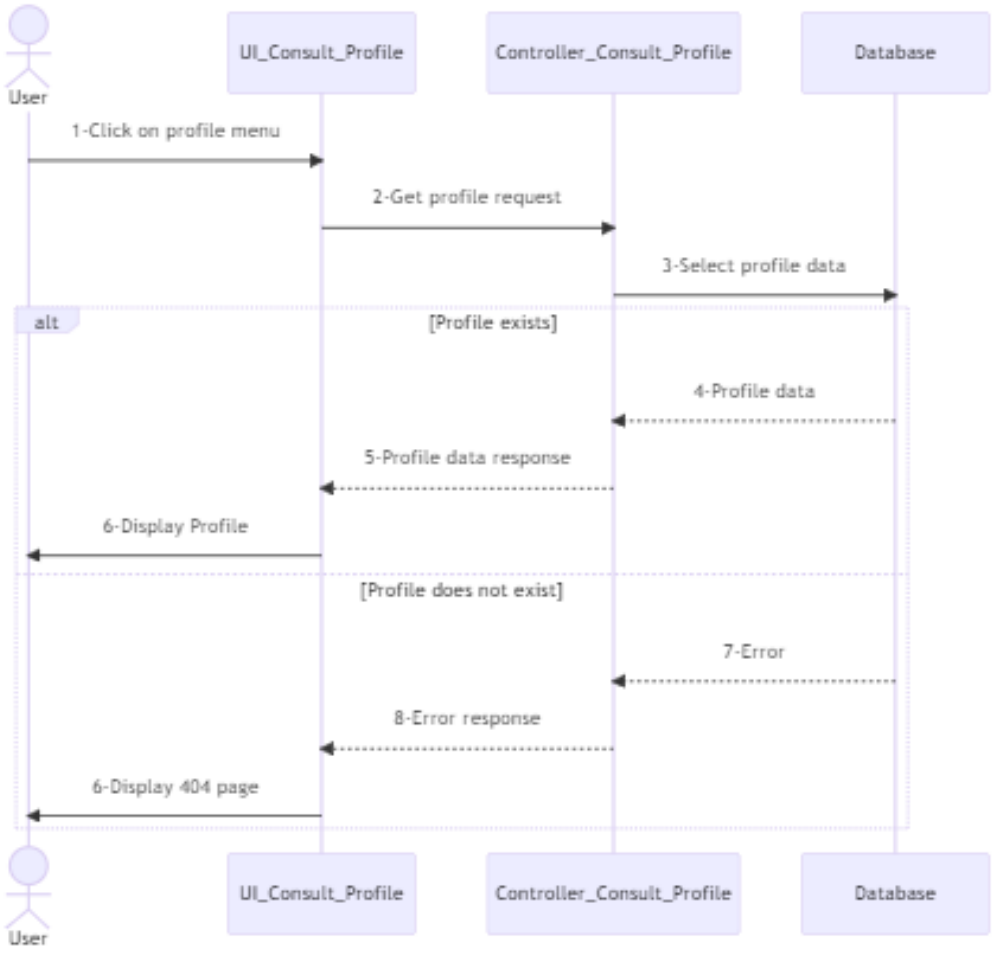

Figure 3.15: Sequence diagram of "Consult profile"

3.4 Sprint 2

The user stories that follow outline the scope of our third sprint.

- As a user, I can edit my profile information.
- As a user, I can edit my account information and settings.
- As a user, I can delete my account.

3.4.1 Identification

This section provides use case diagrams and textual descriptions for our use cases.

3.4.1.1 Refinement of the use case "edit profile"

The figure 3.16 represents the use case diagram of the user story "Edit profile", it's dependent on the use cases "Login" of course and "Create profile".

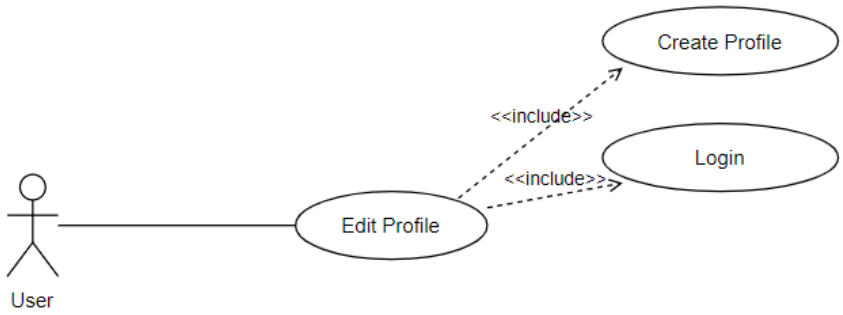

Figure 3.16: Use case diagram of "edit profile"

The table 3.6 describes the use case "Edit profile" textually in the form of tables.

Use Case Scenario	edit profile
Actors	User
Pre-conditions	The actor has to be logged in
Post-conditions	Profile information are updated
Main scenario	1-the user clicks "Edit profile/Profil bearbeiten" 2-edit buttons are displayed throughout the page 3- the user clicks on the designated info's edit button 4- editing modal is displayed 5-the user updates his information 6- the user clicks "Save/Speichern" 7- the data is updated in the database 8- the information is updated in the profile page

Table 3.6: Description of use case "Edit Profile"

3.4.1.2 Refinement of the use case "edit account"

The figure 3.17 shows the use case diagram of the user story "Edit account" which is dependent on the use cases "Create account" and "Login"

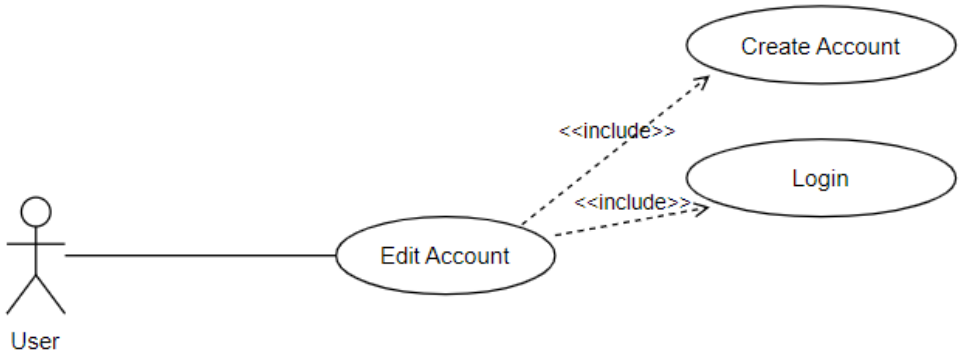

Figure 3.17: Use case diagram of "edit account"

The table 3.7 describes the use case "Edit account" textually in the form of tables.

Use Case Scenario	Edit account
Actors	User
Pre-conditions	The actor has to be logged in
Post-conditions	Account information settings are updated
Main scenario	1- the actor selects a setting tab 2- the actor updates the information 3- the actor clicks "Save/Speichern" 4- the data is updated in the database

Table 3.7: Description of use case "Edit account"

3.4.1.3 Refinement of the use case "delete account"

The figure 3.18 shows the use case diagram of the user story "Delete account" that is dependent on the use cases "Create account" and "Login"

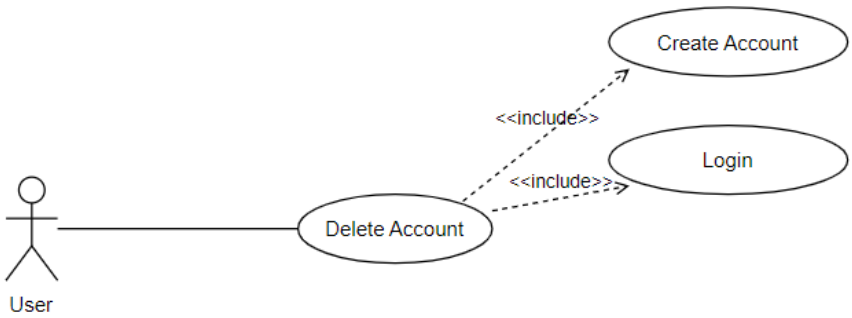

Figure 3.18: Use case diagram of "delete account"

The table 3.8 describes the use case "Delete account" textually in the form of tables.

Use Case Scenario	Delete account
Actors	User
Pre-conditions	The actor has to be logged in
Post-conditions	Account is completely deleted from the database
Main scenario	1- the actor clicks "next/weiter" in the delete account tab of the settings 2- the actor clicks "yes, delete my account/Ja, ich bin einverstanden" 3- the server deletes the account from the database 4- the user is logged out

Table 3.8: Description of use case "Delete account"

3.4.2 Conception

This step establishes the interactions of the use cases and utilizes Class and Sequence Diagrams to formalize our approach visually

3.4.2.1 Class Diagram of the User Story "edit profile"

The figure 3.19 represents the class diagram of the use case "Edit profile".

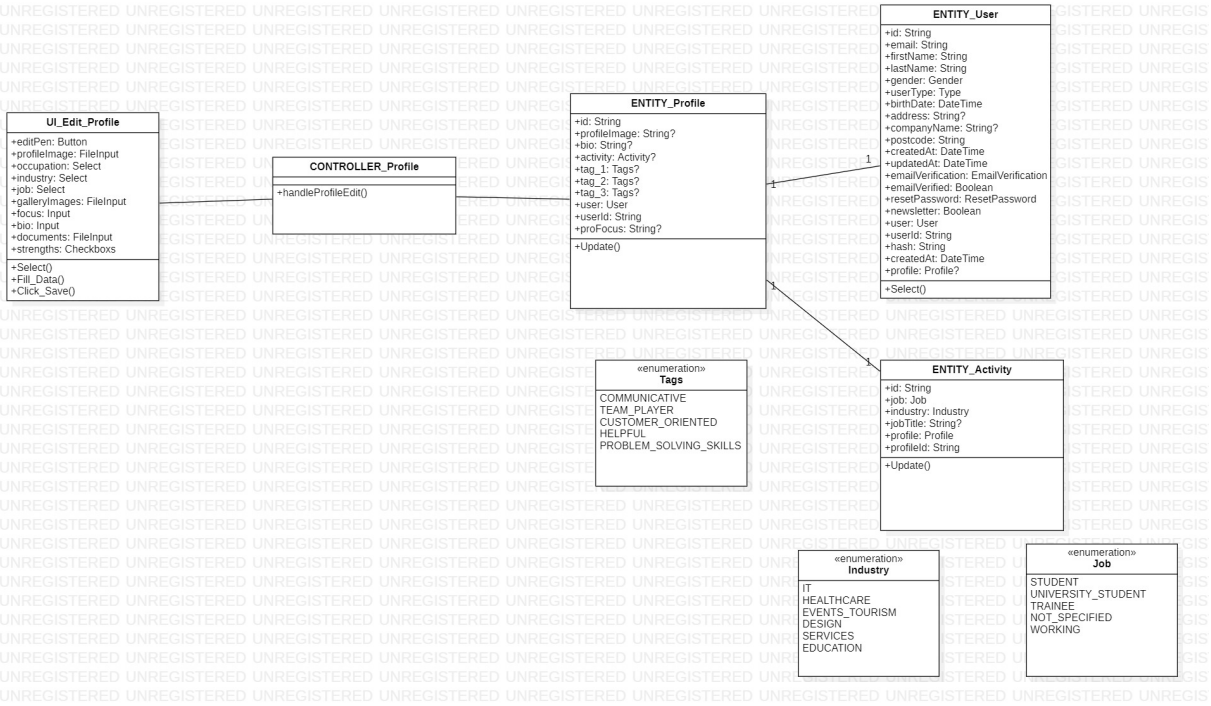

Figure 3.19: Class diagram of "edit profile"

3.4.2.2 Sequence Diagram of the User Story "edit profile"

Figure 3.20 presents the edit profile procedure, the user has to have an account in order to modify his profile.

- the profile page contains a button that toggles the edit mode in the page, when clicked, various edit buttons are displayed throughout the page with each button edits the info near it.
- when the user clicks on one of the edit buttons, a designated modal shows with fields to update the information.

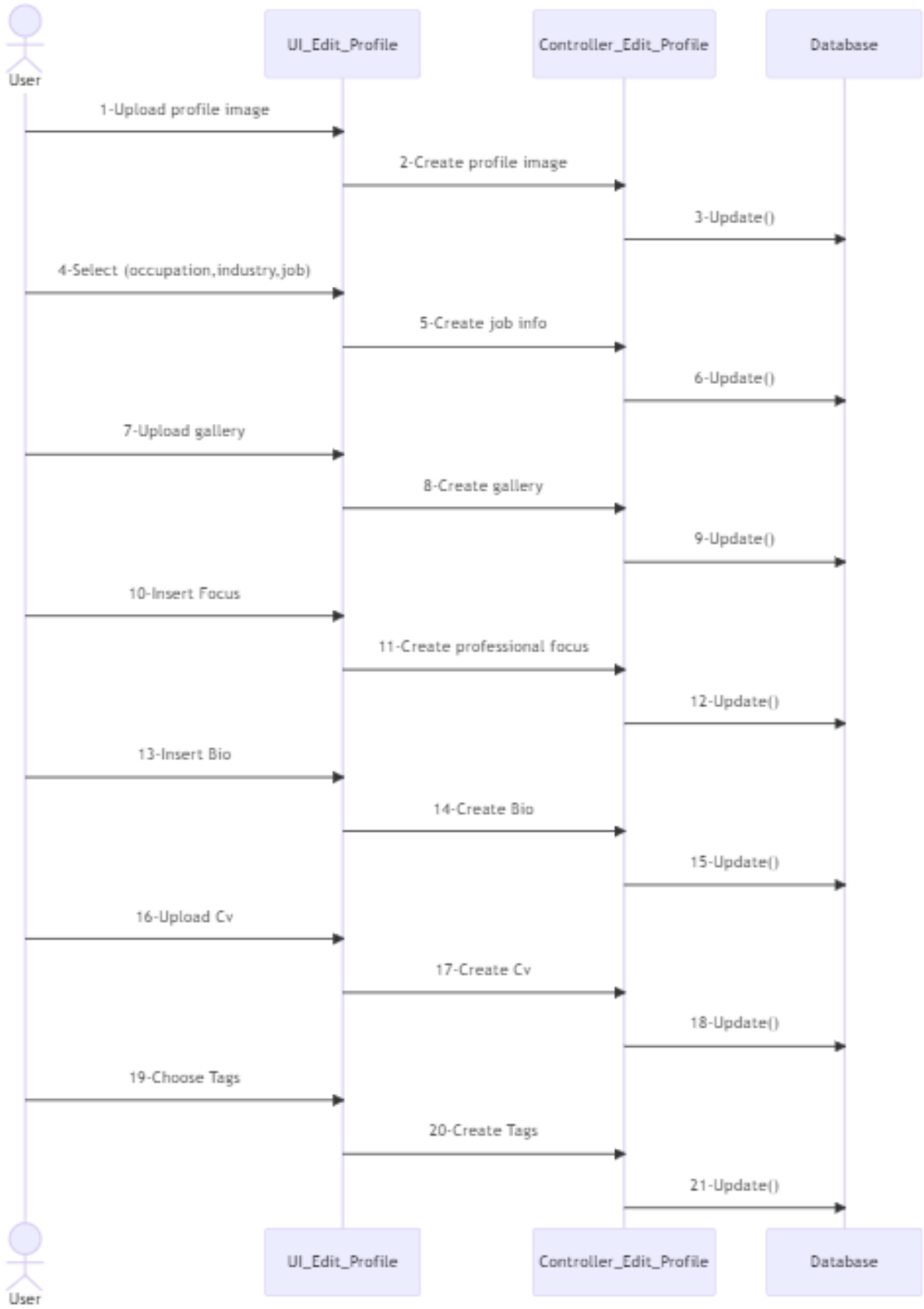

Figure 3.20: Sequence diagram of "edit profile"

3.4.2.3 Class Diagram of the User Story "edit account"

The figure 3.21 presents the class diagram of the use case "Edit account".

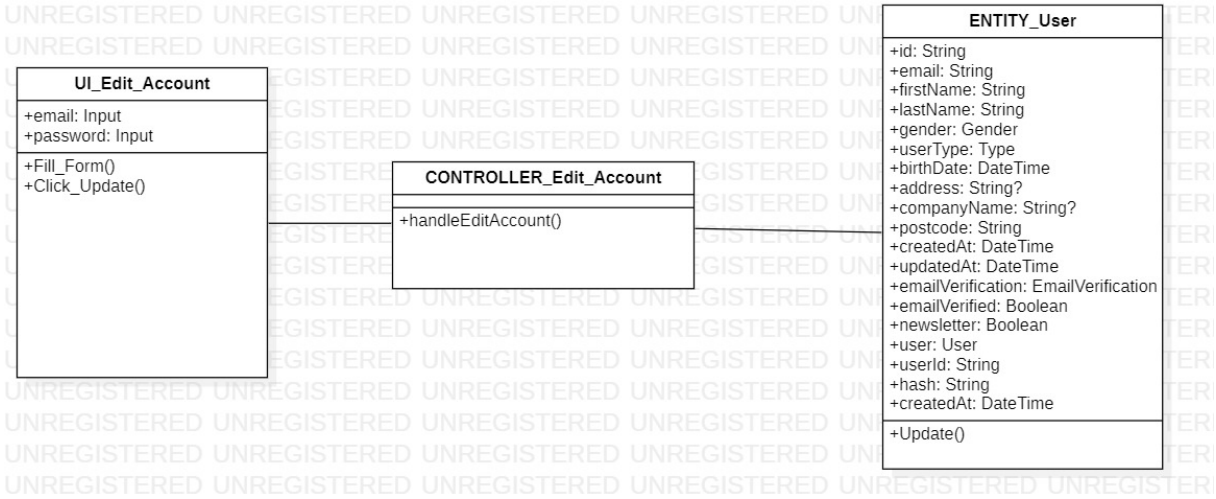

Figure 3.21: Class diagram of "edit account"

3.4.2.4 Sequence Diagram of the User Story "edit account"

Figure 3.22 presents the edit account procedure, the user has to have an account in order to modify his account settings.

- when the settings page is displayed, the user will see multiple settings categories that when clicked, the setting page is displayed with fields to update the information
- the settings include personal information, login data, profile visibility, account deletion and newsletter toggle

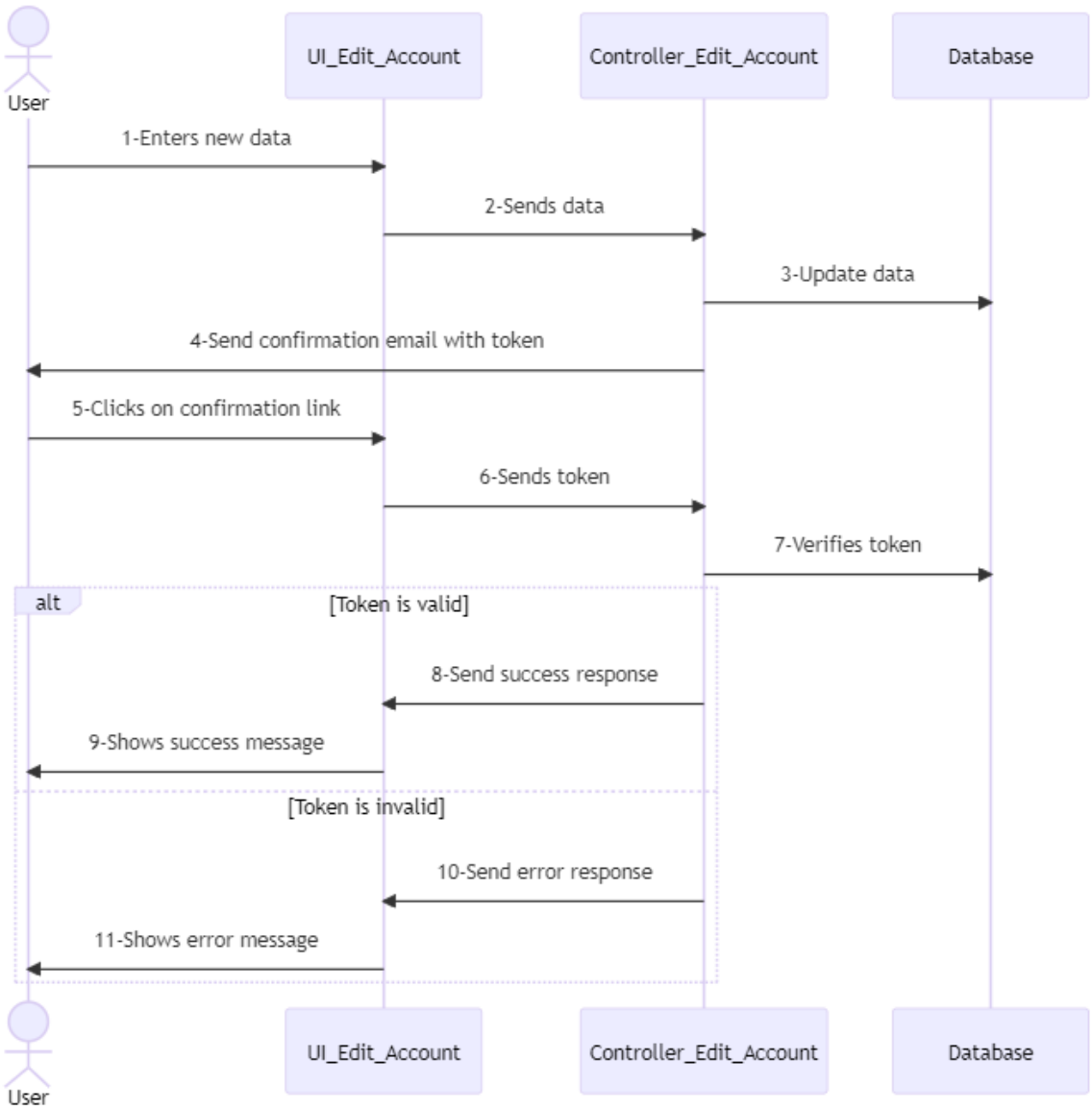

Figure 3.22: Sequence diagram of "edit account"

3.4.2.5 Class Diagram of the User Story "delete account"

The figure 3.23 presents the class diagram of the use case "Delete account".

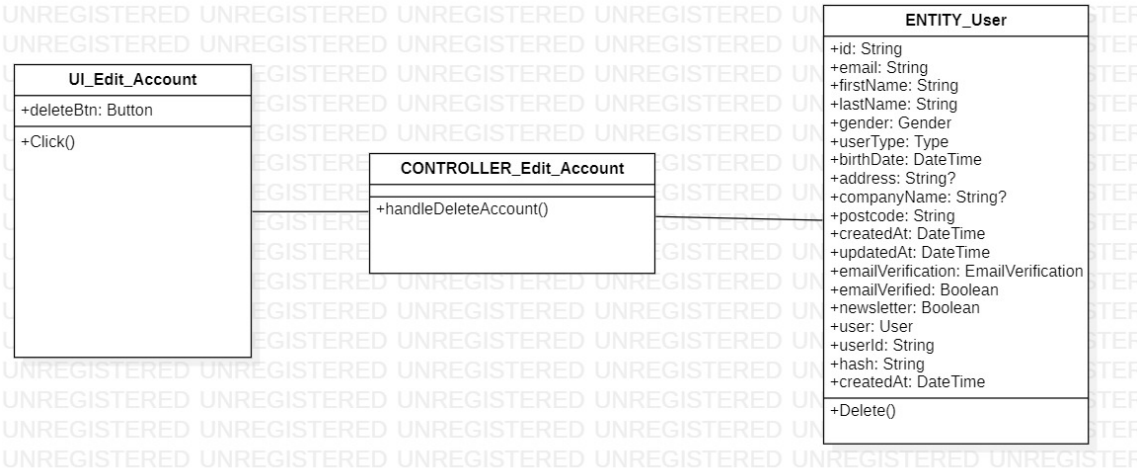

Figure 3.23: Class diagram of "delete account"

3.4.2.6 Sequence Diagram of the User Story "delete account"

Figure 3.24 presents the delete account procedure, the user has to have an account in order to delete his account.

- when the setting tab of "delete account" is displayed the user is provided with a warning saying that the account will be completely deleted and there's no way back once done.
- when the user clicks next a confirmation modal is displayed to the user.
- when the button "yes, delete my account/Ja, ich bin einverstanden" is clicked, the account is completely deleted and the user is logged out.

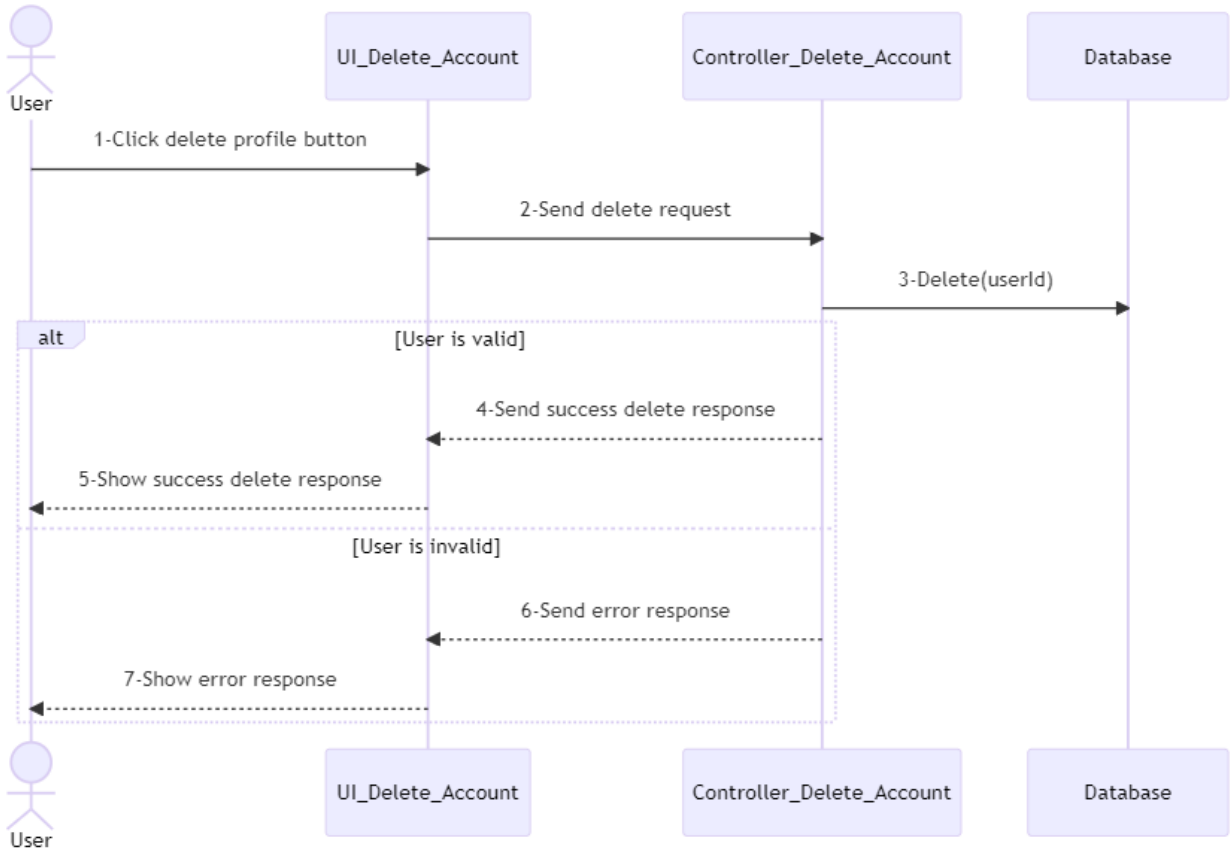

Figure 3.24: Sequence diagram of "delete account"

3.5 Realization

In the following section we will present the realised interfaces for each of the 3 sprints, for each use Case we will have a UI for the user to fulfill his need.

3.5.1 Realization of sprint 0

In this Realization we will focus on the interfaces of "Login", "Register" and reset password.

3.5.1.1 Realization of the use Case "Create account"

The figure 3.25, 3.26 and 3.27 show where a new user can register and confirm via email address to create his own account.

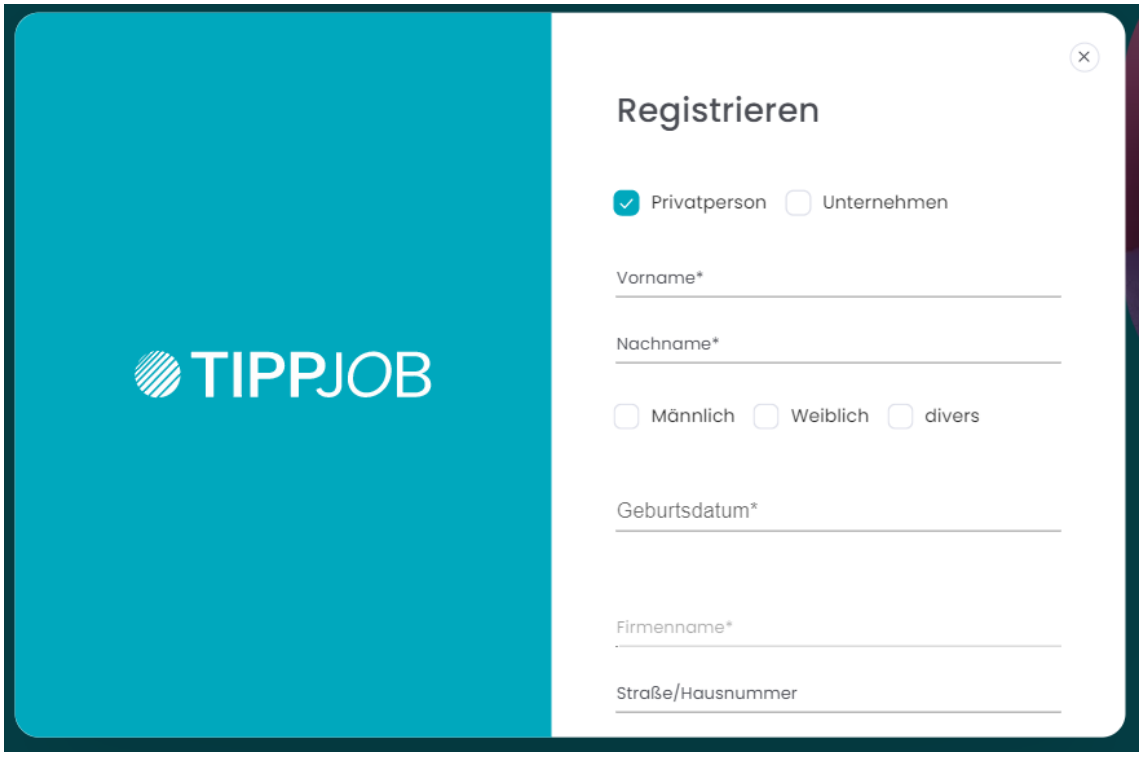

Figure 3.25: interface of "Create account" 1

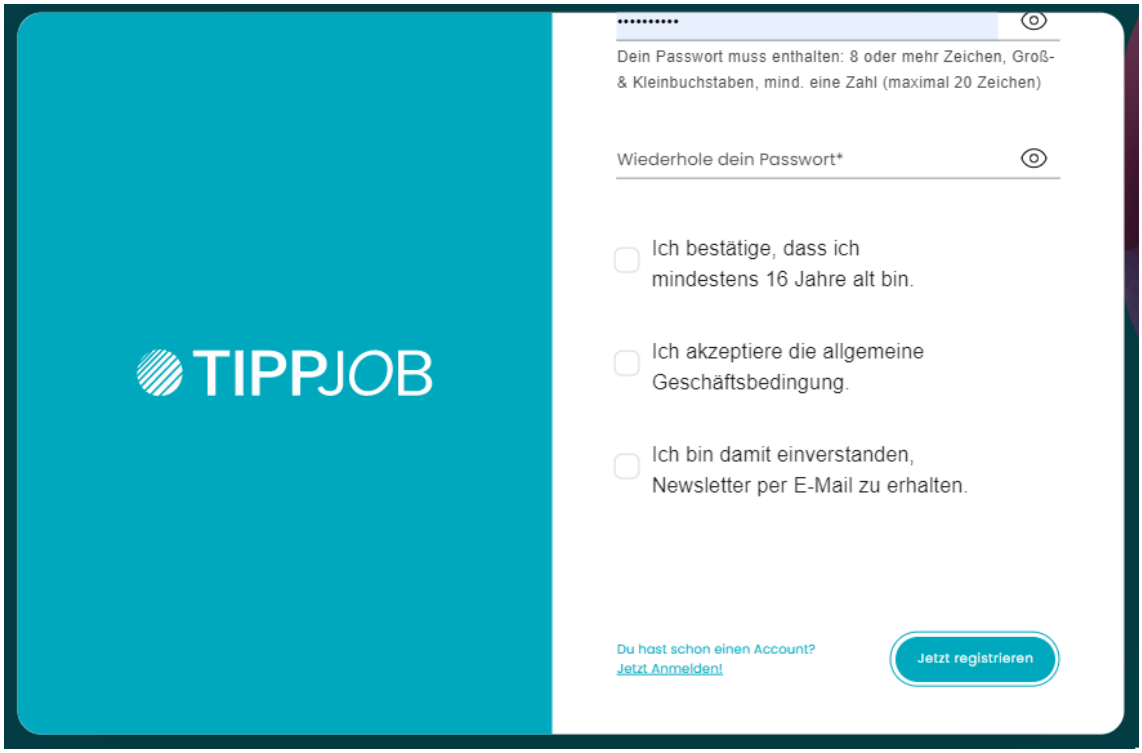

Figure 3.26: interface of "Create account" 2

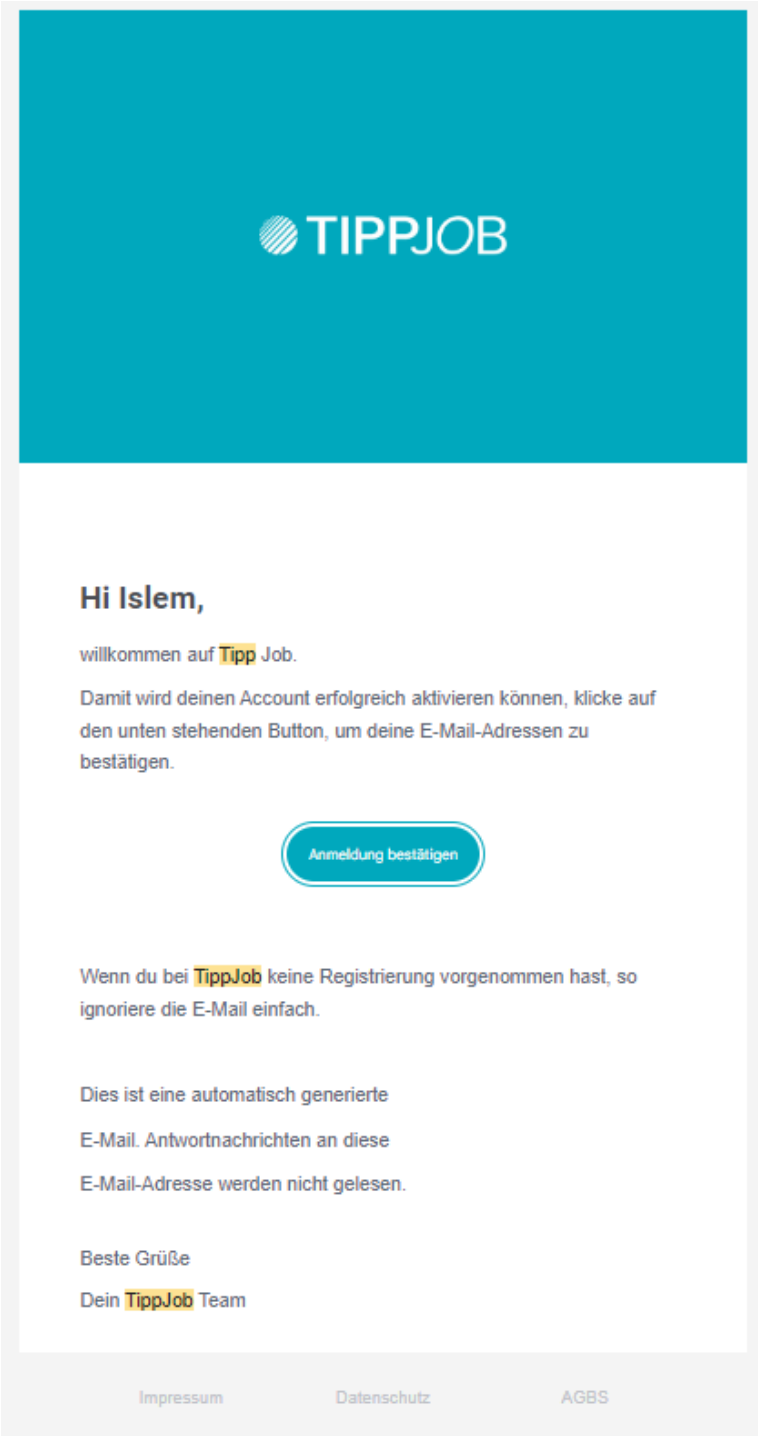

Figure 3.27: account confirmation email

3.5.1.2 Realization of the use case "Login"

In the modal represented by the figure 3.28 a registered user can login using his credentials to access the web app's features.

Figure 3.28: interface of "Login"

3.5.1.3 Realization of the use case "Reset password"

Here in the figure 3.29, 3.30 and 3.31 is where the registered user can use his email address to reset his account password in case he forgot it.

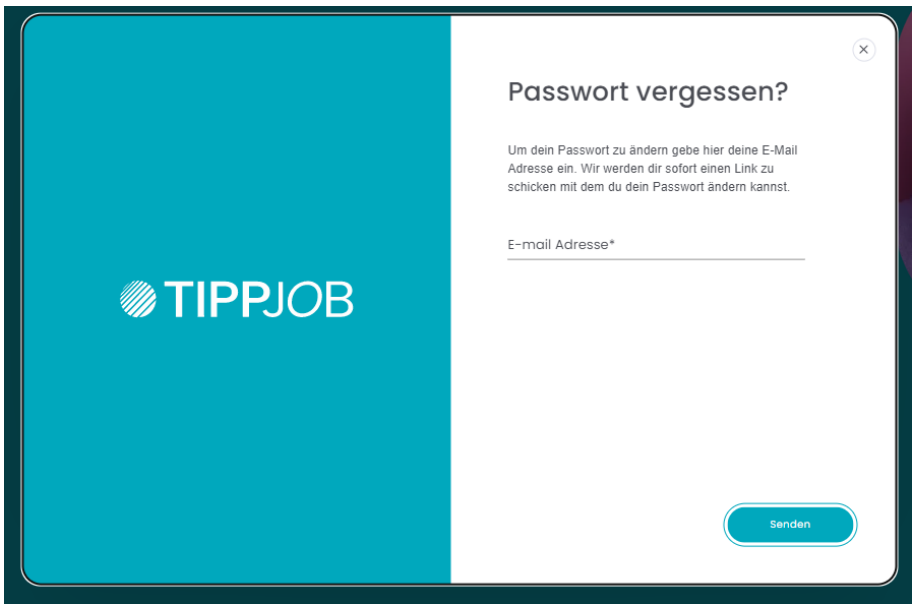

Figure 3.29: interface of "Reset password" email input

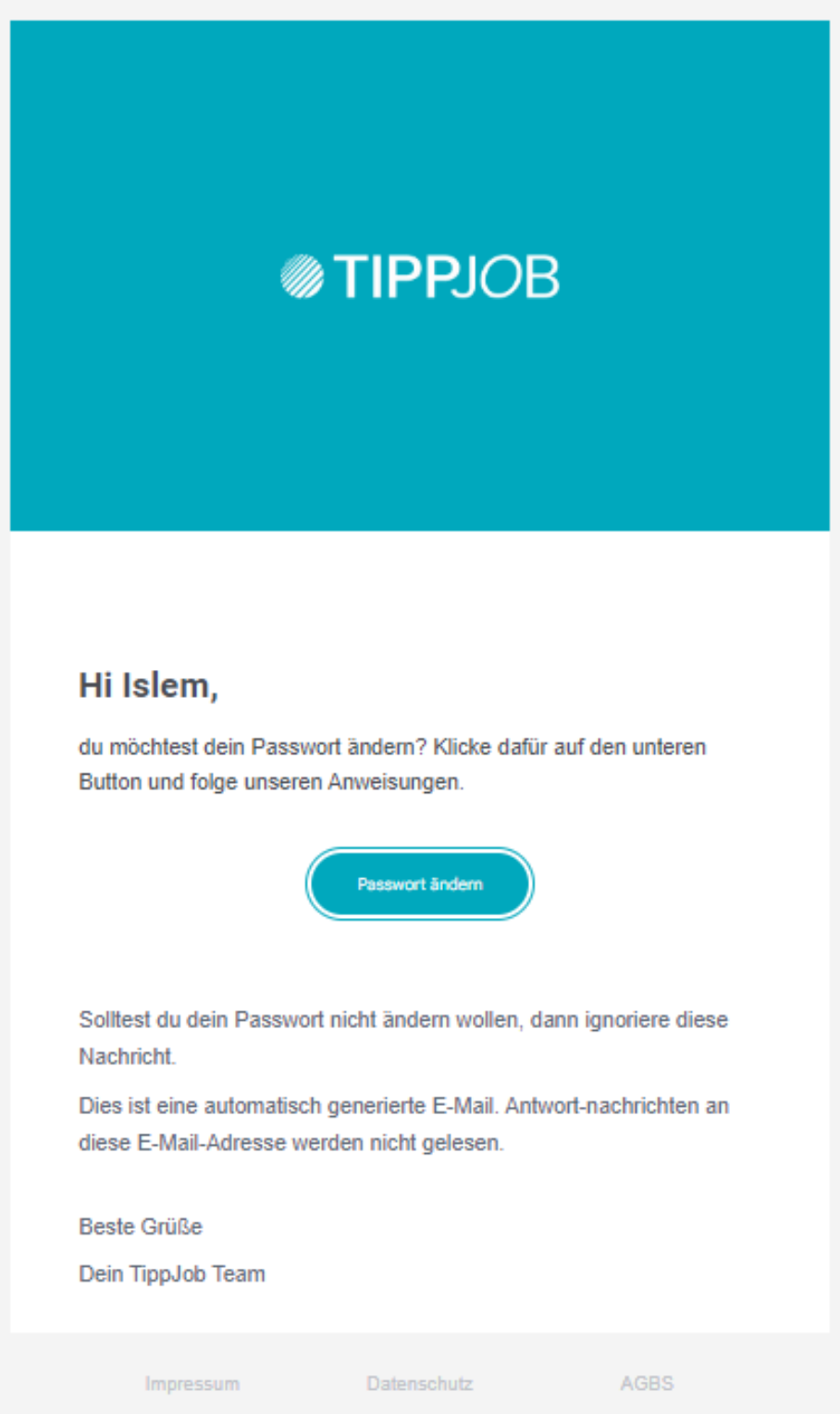

Figure 3.30: email of password reset

Figure 3.31: interface of "reset password" new password input

3.5.2 Realization of sprint 1

This subsection is reserved for the Realization of the use cases developed in sprint 1, they are "Create profile" which is an on-boarding stepper process and "Consult profile" which is simply the user's profile page.

3.5.2.1 Realization of the use case "Create profile"

The interfaces represented in the figures 3.32 to 3.40 represent the steps the user can take to create their profile page. It is important to note that all the following inputs are completely optional, and the user can keep their profile empty of information. In case the user chooses not to provide any information, they can access the on-boarding process again when they log in. The on-boarding process includes the following information inputs:

- Profile picture uploads
- Activity info (Industry, occupation, job title)
- Gallery upload
- Focus (topics the user focuses on)
- Bio input
- PDF files upload
- personal strengths (Profile tags)

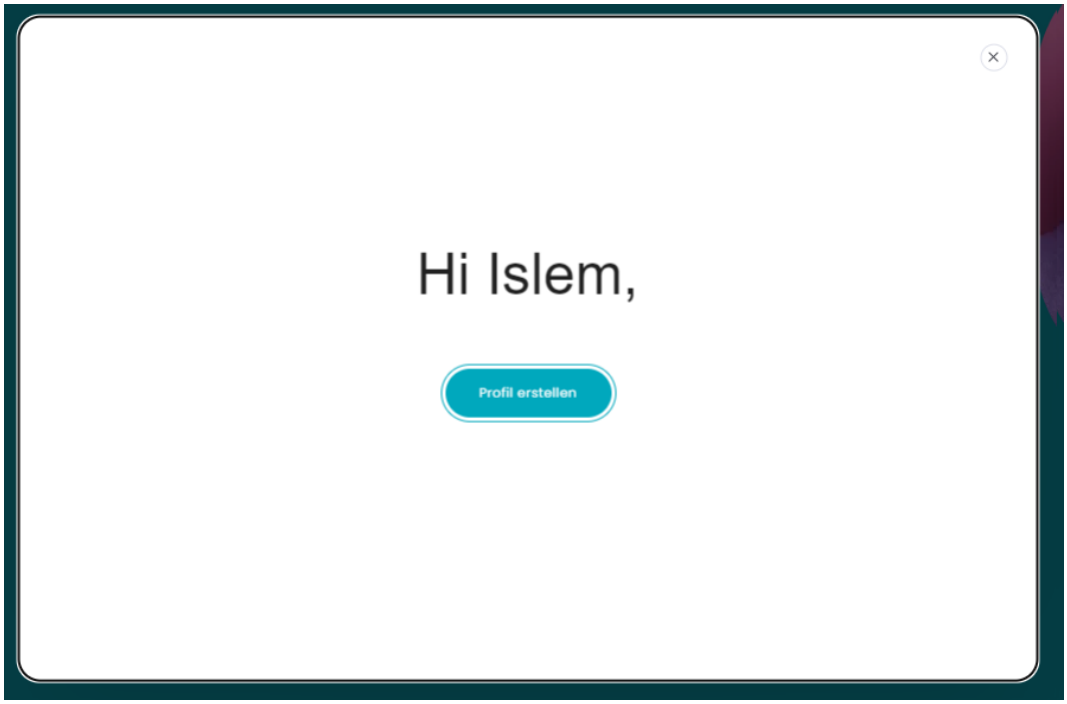

Figure 3.32: on-boarding welcome message

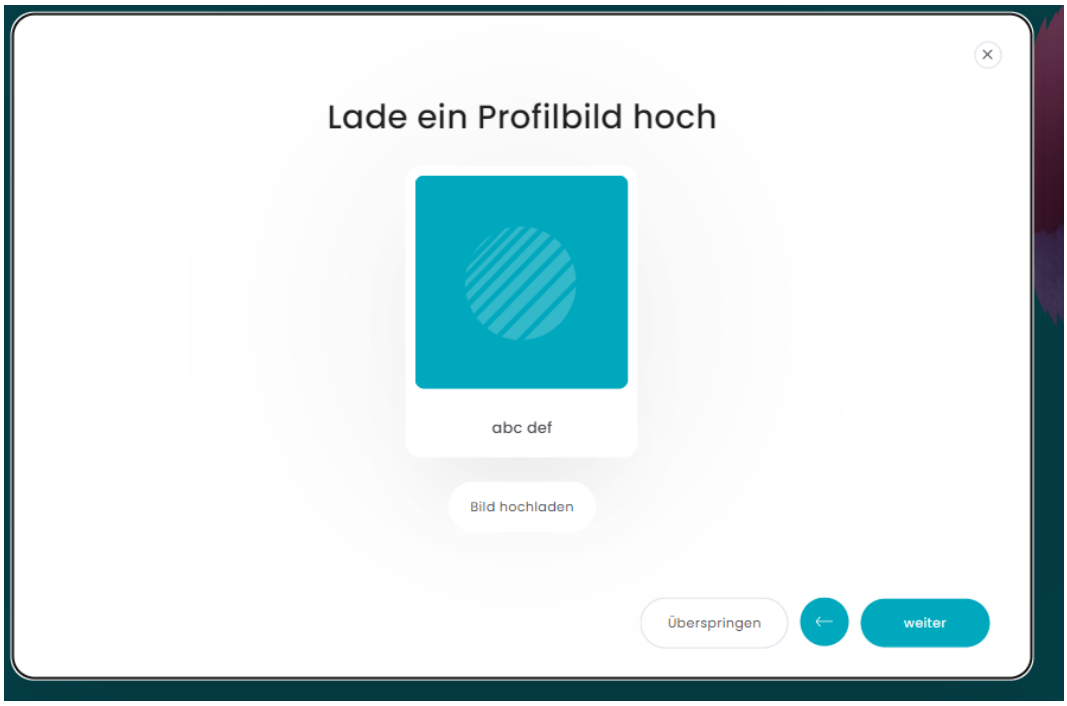

Figure 3.33: Profile picture upload interface

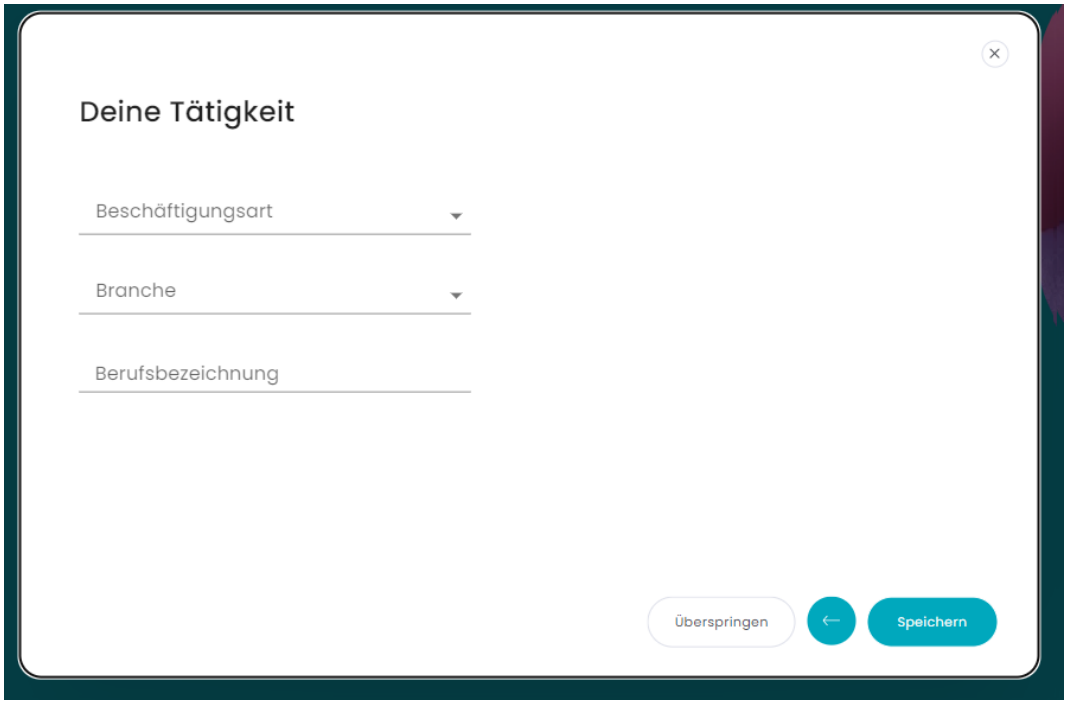

Figure 3.34: Activity input interface

Figure 3.35: Gallery upload interface

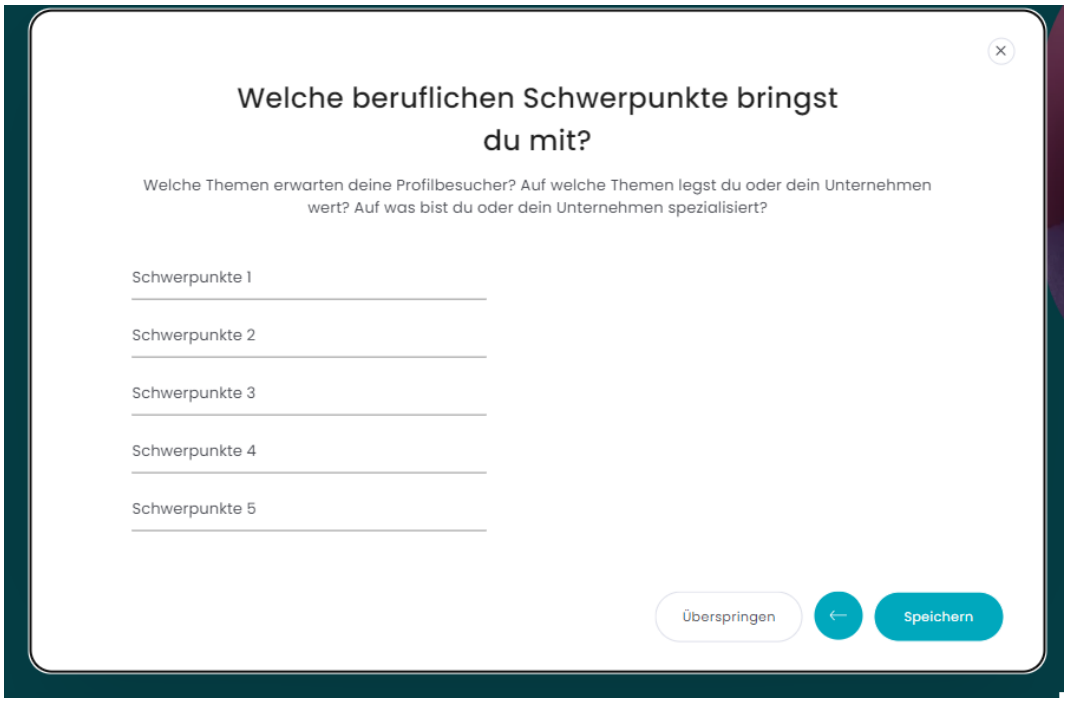

Figure 3.36: Focus input interface

Figure 3.37: Bio input interface

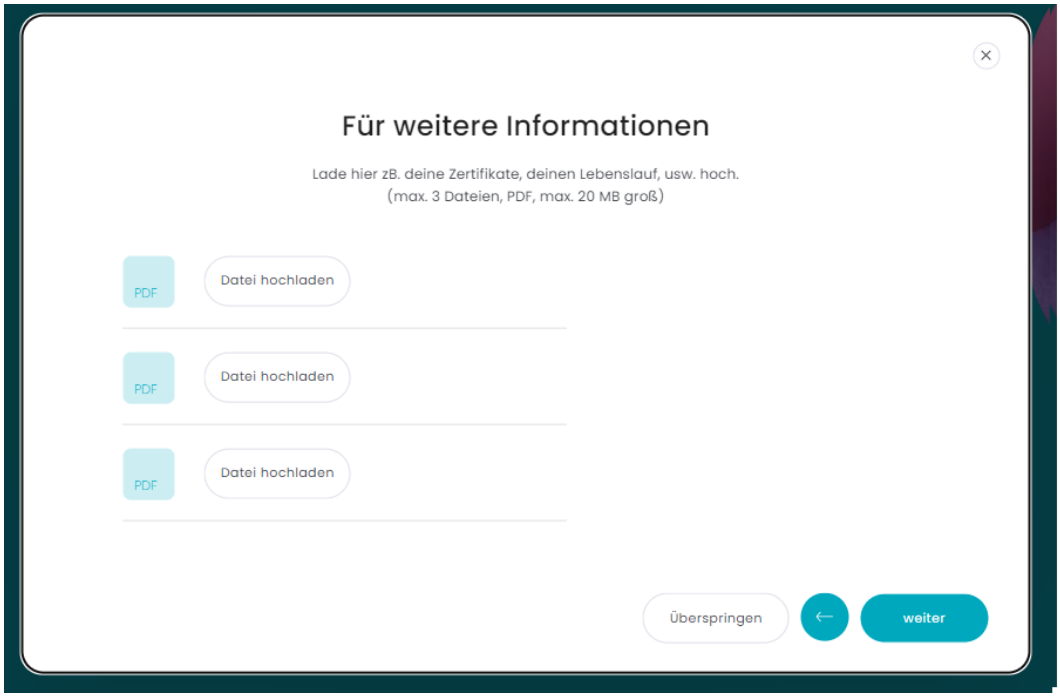

Figure 3.38: PDF files upload interface

Figure 3.39: Personal strengths interface

Figure 3.40: User welcoming message

3.5.2.2 Realization of the use case "Consult profile"

The figure 3.41 shows the interface where the user can consult his own profile page and information, this profile and information page will be public to all users if the owner chooses so.

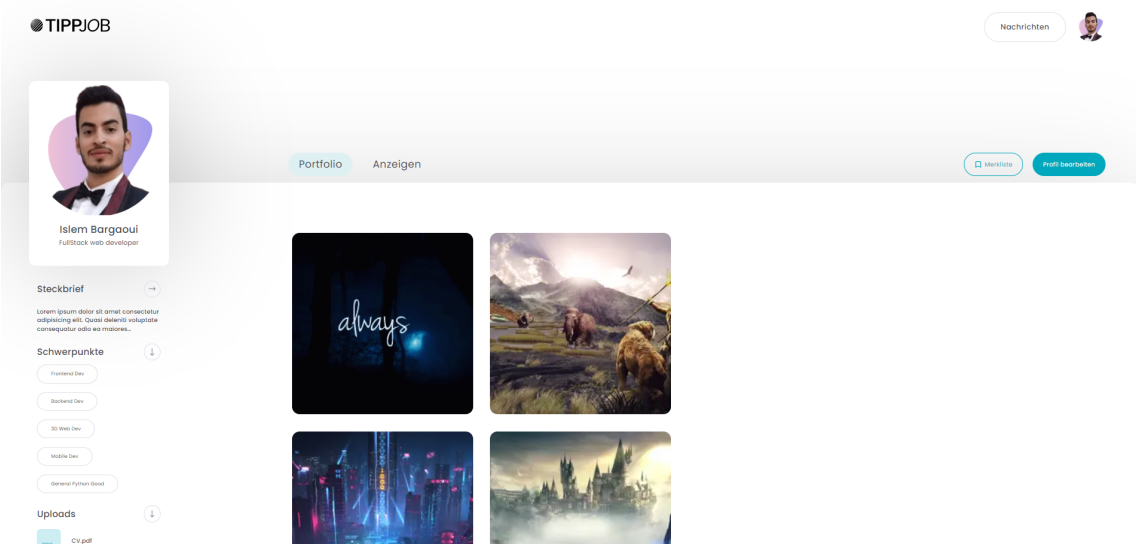

Figure 3.41: Profile interface

3.5.3 Realization of sprint 2

In this subsection we will present the interfaces of the use cases related to sprint 2 which are "Edit profile", "Edit account" and "Delete account".

3.5.3.1 Realization of the use case "Edit profile"

In the profile page on the right, the user will find a button "Profil bearbeiten" which if clicked will display edit buttons across the entire page. as shown in the figure 3.42 .

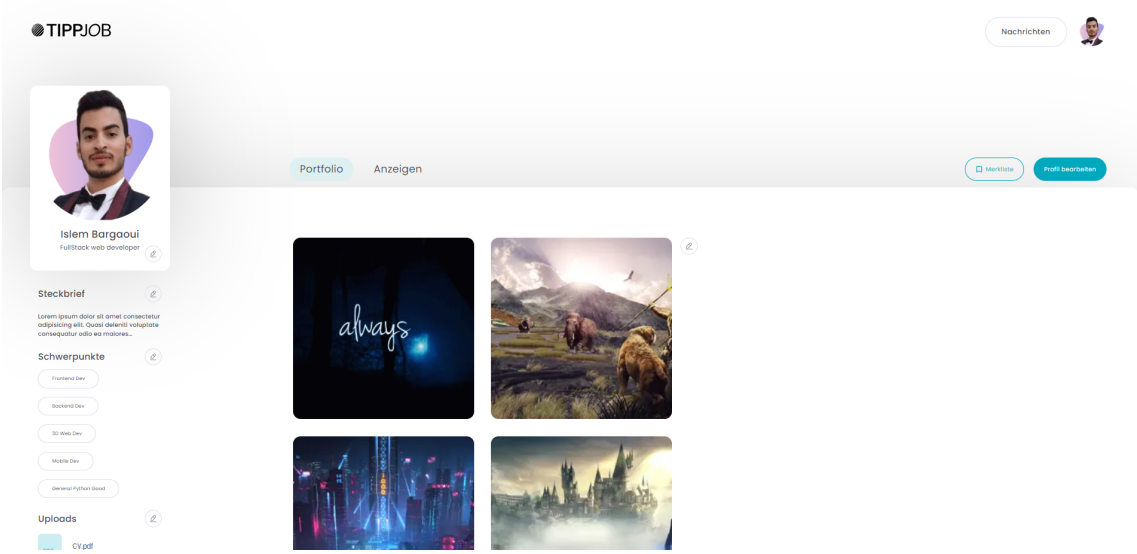

Figure 3.42: edit Profile interface

3.5.3.2 Realization of the use Case "Edit account"

The figures 3.43 to 3.49 show a page called the settings, where the user can edit his personal information such as name, address, email, password, visibility settings or even newsletter setting. The settings tabs are as follows:

- personal Data (first name, last name, gender, address and postal code)
- access data (email, password)
- profile visibility (profile or uploads visibility), here he can choose if his information is visible to the public, companies only or private.
- Newsletter setting

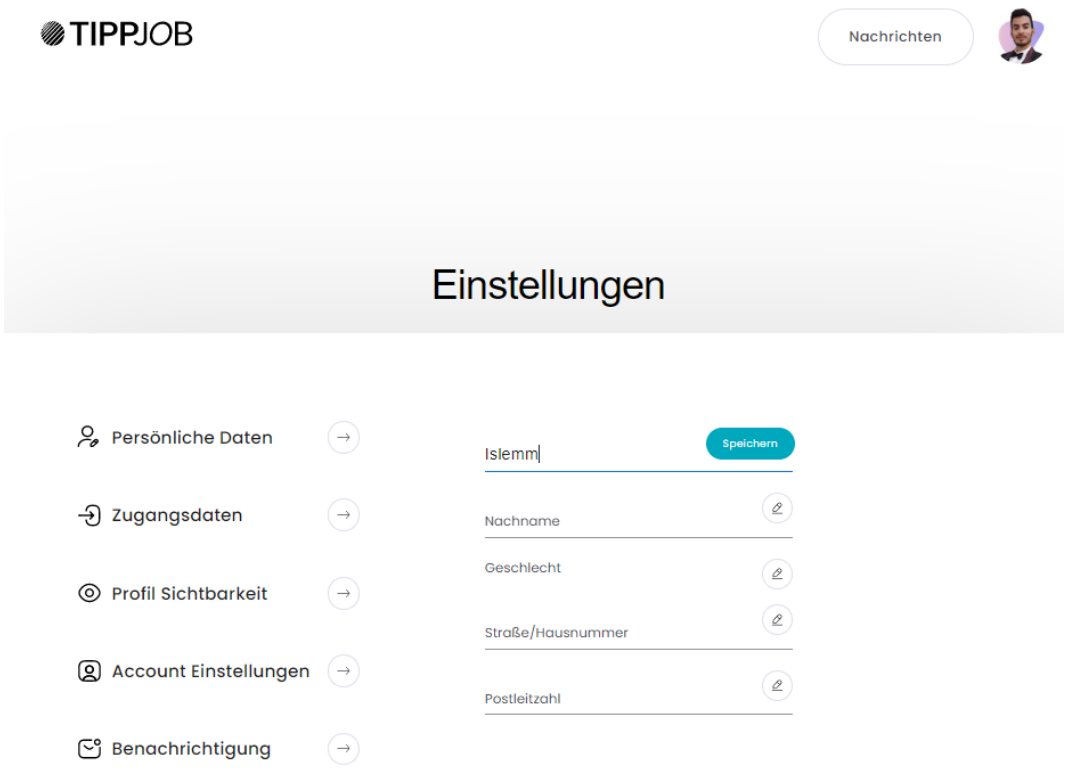

Figure 3.43: edit personal data interface

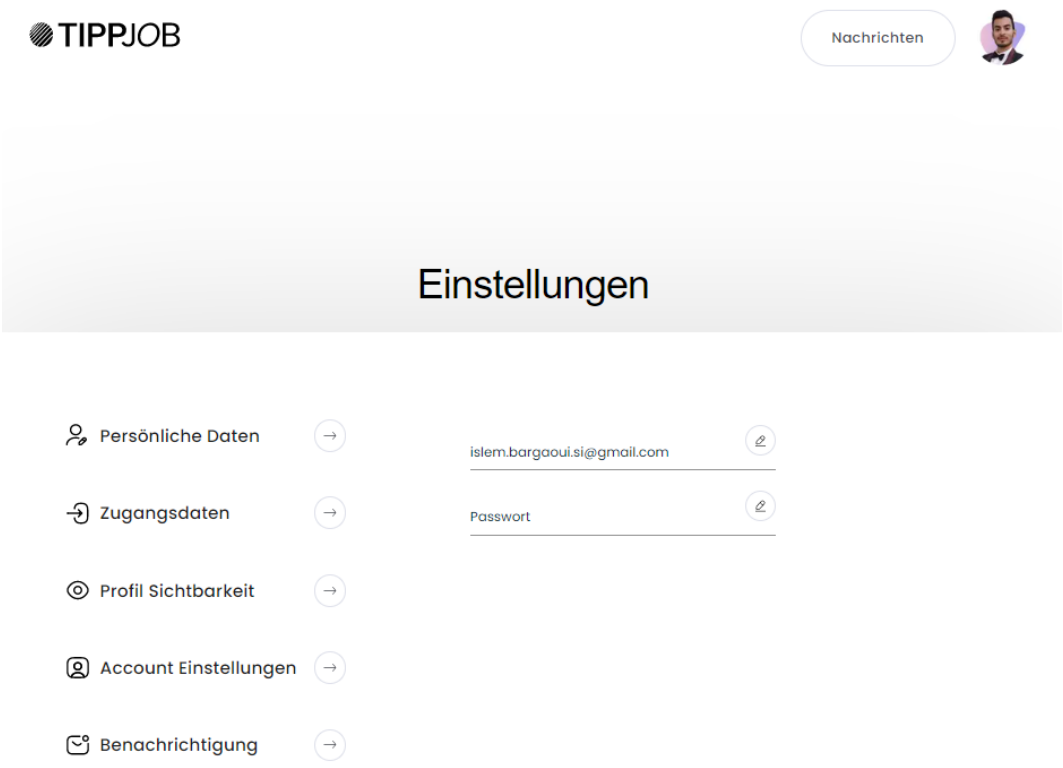

Figure 3.44: edit access data interface

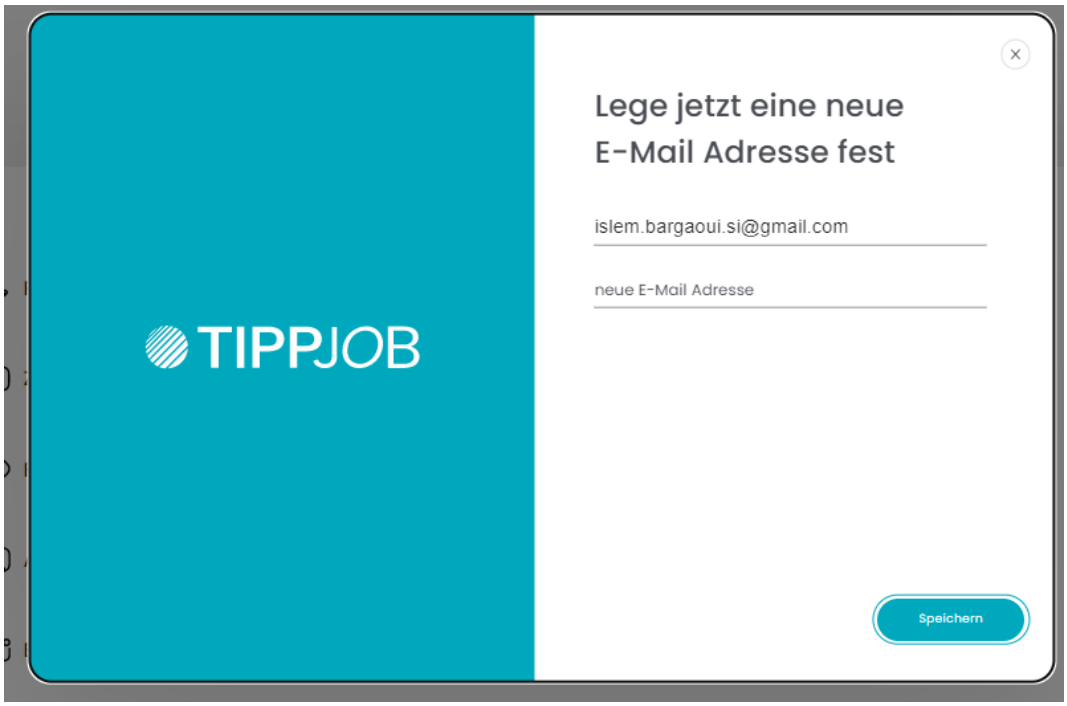

Figure 3.45: edit email interface

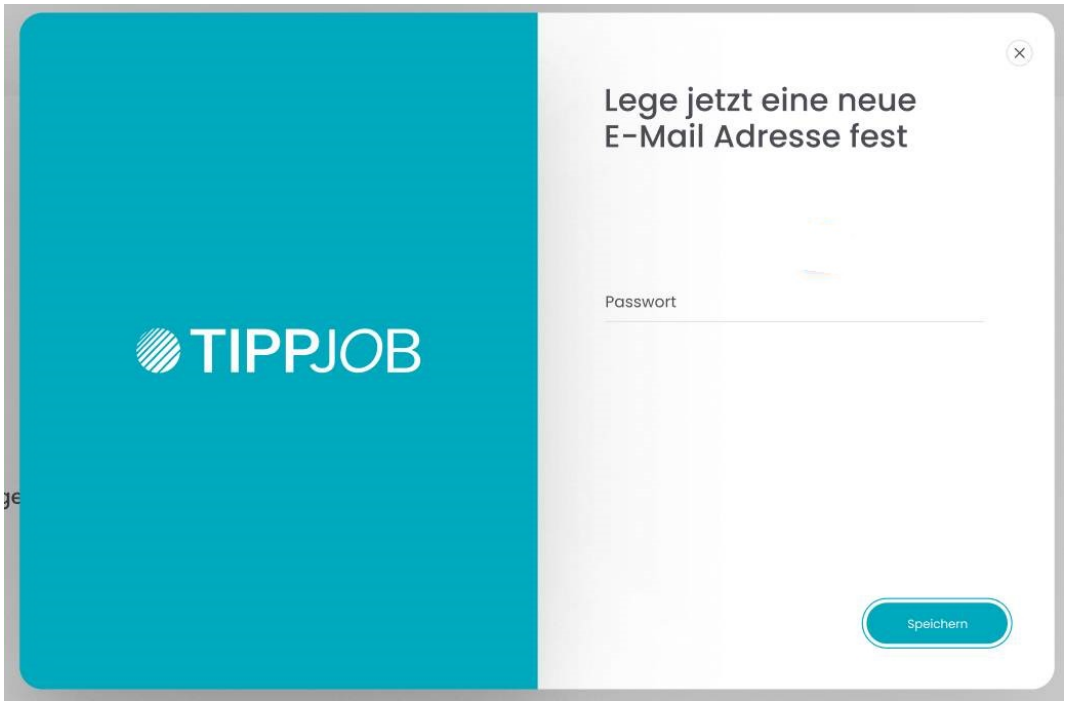

Figure 3.46: password confirmation to update email

Figure 3.47: edit password interface

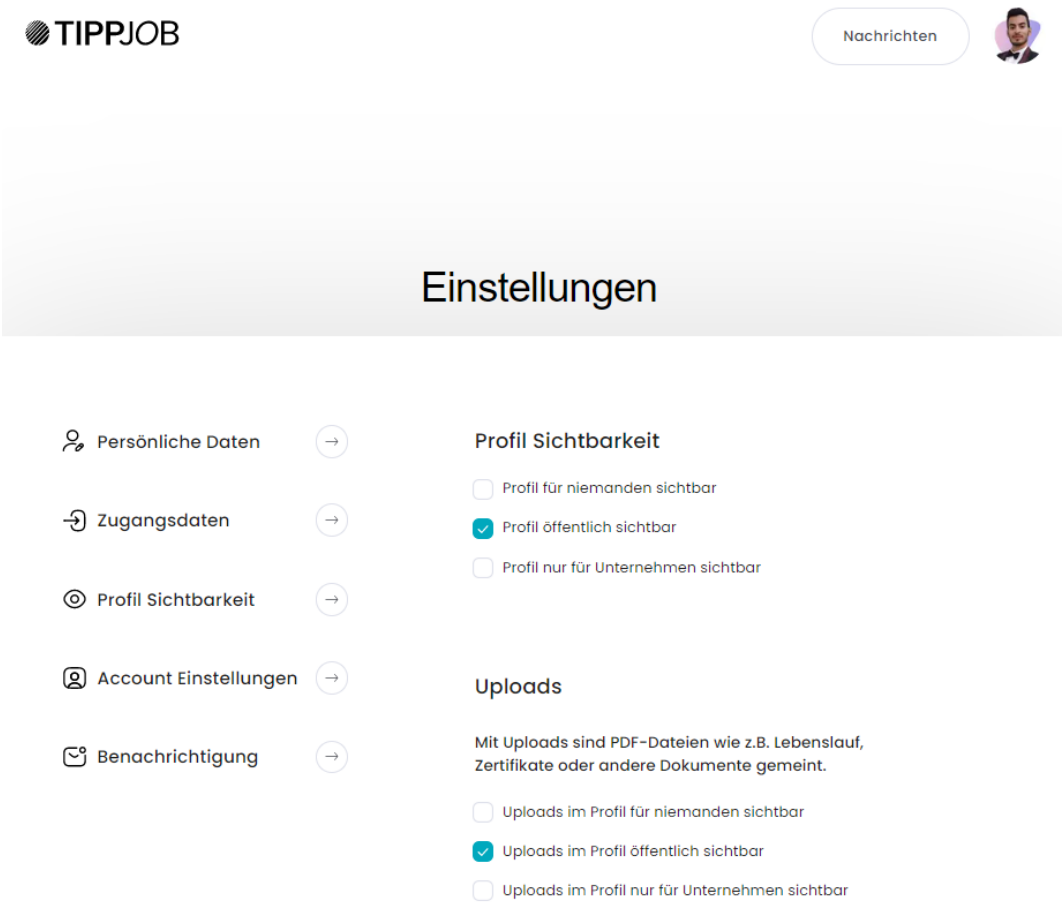

Figure 3.48: edit visibility settings interface

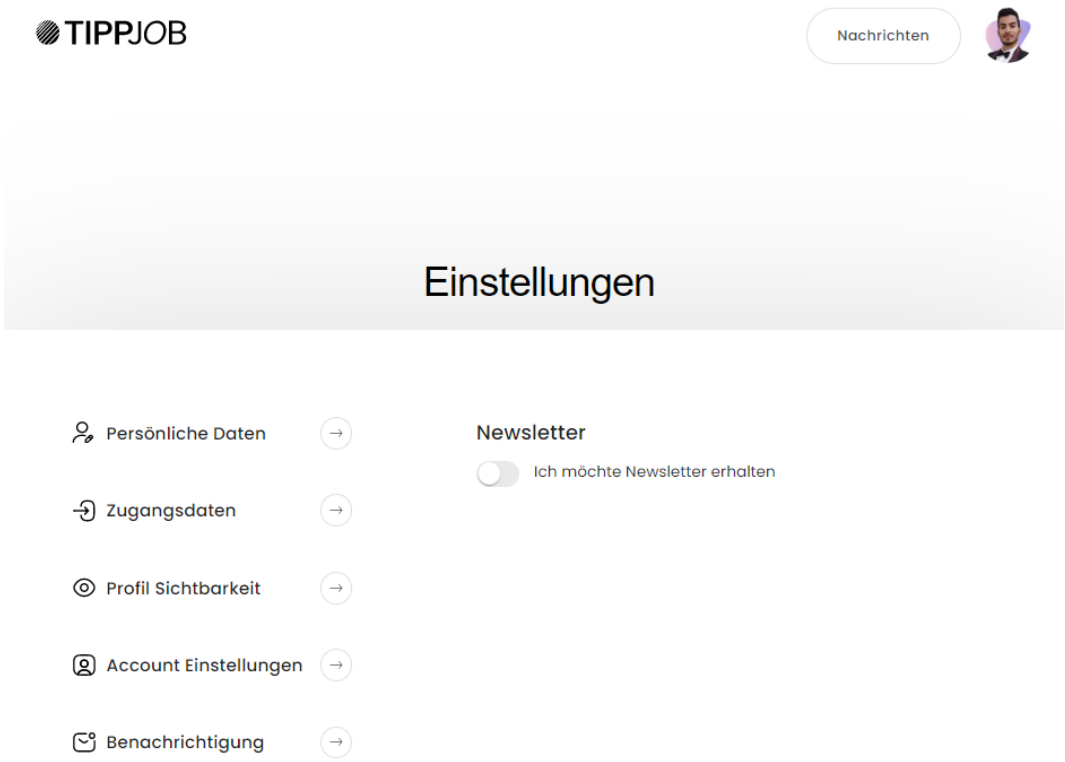

Figure 3.49: toggle newsletter setting interface

3.5.3.3 Realization of the use case "Delete account"

The interfaces shown in the figures 3.50 and 3.51 allows a user to completely delete his profile, once done, this process cannot be reversed.

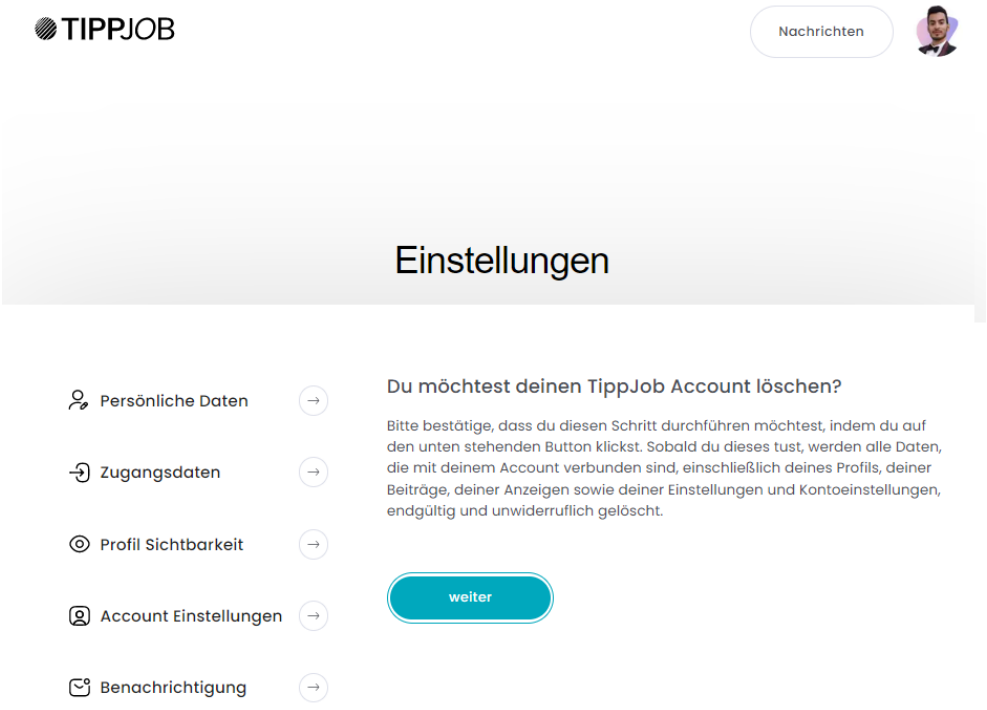

Figure 3.50: delete account interface

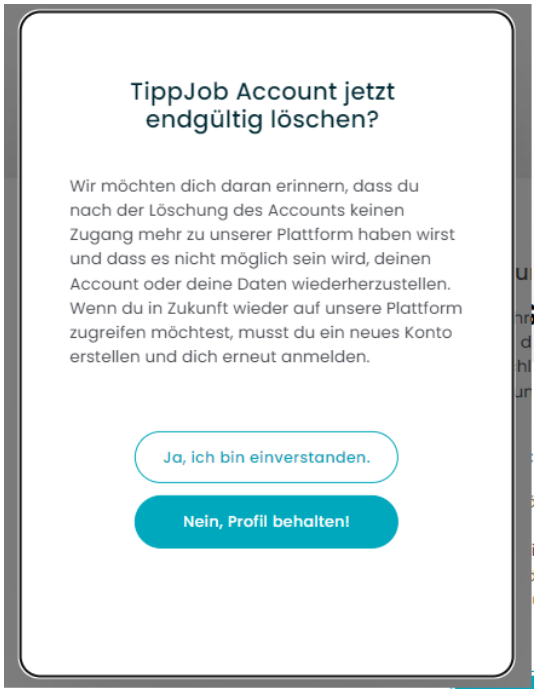

Figure 3.51: confirm account deletion interface

3.6 Conclusion

This chapter covers the various steps involved in delivering the first release, which includes refining the requirements, designing, and implementing the first three sprints.

Chapter 4

Release 2: Integrating job posting system and payment solution

4.1 Introduction

Now we arrive at the fourth chapter of the project which includes Job posting system where the user can manage his job posts as well as view and interact with all available job posts in the platform.
To begin, we define the requirements of our system's actor in the 3 sprints from the system backlog. Following that, we proceed to refine the use cases, which enables us to create class and sequence diagrams for implementation. Ultimately, we showcase the outcome of this release.

4.2 Sprint 3

The following user stories shows what our third sprint consists of:

- As a user, i can create a job offer
- As a user, i can edit my job offers
- As a user, i can delete my job offers

4.2.1 Identification

This section contains a collection of use case diagrams for our use cases, accompanied by a table that provides a textual description of each use case.

4.2.1.1 Refinement of the User Story "Create a Job Post"

The use-case diagram for the user story "create a job post" which is part of the use-case "manage own job posts" is shown in the figure 4.1

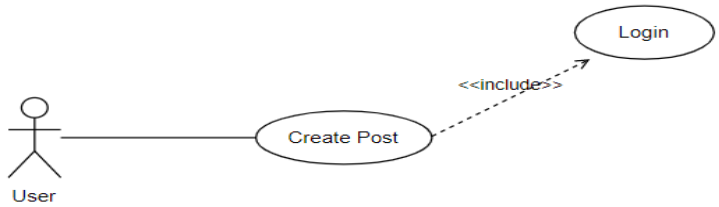

Figure 4.1: Use case diagram of the user story "Create a job post"

The table 4.1 describes the use case "Create a job post" textually in the form of tables.

Use Case Scenario	Create a Job Post
Actors	User
Pre-conditions	The user has to be logged in
Post-conditions	A Job Post related to that user is created
Main scenario	1- the user clicks on "neue Anzeige/New Post" in the Anzeigen section of his profile. 2- the user inputs the data that will defined the post's content 3- the user clicks "Anzeige aufgeben/Submit post" 4- the server creates a new post in the database

Table 4.1: Description of use case "Create Job Posts"

4.2.1.2 Refinement of the User Story "Edit a job post"

The use-case diagram for the user story "Manage Conferences" is shown in the figure 4.2

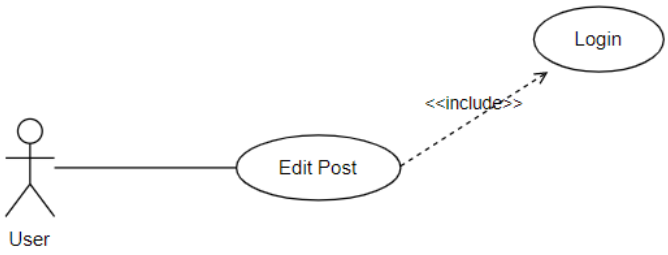

Figure 4.2: Use case diagram of the User Story "Edit a job post"

The table 4.2 describes the use case "Edit a job post" textually in the form of tables.

Use Case Scenario	Edit a Job Post
Actors	User
Pre-conditions	The user has to be logged in at least one job post is created by the user
Post-conditions	The desired job post is updated
Main scenario	1- the user clicks on the edit button under the job post in his profile. 2- the user updates the input with the desired data 3- the user clicks on "Anzeige aufgeben/Submit post" 4- the server updates the post in the database

Table 4.2: Description of use case "Edit job posts"

4.2.1.3 Refinement of the User Story "Delete a job post"

The use-case diagram for the userstory "Delete a job post" is shown in the figure 4.3

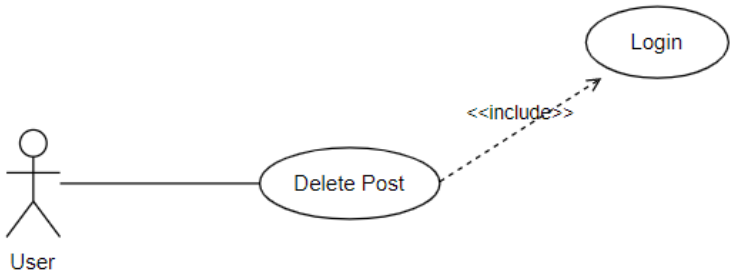

Figure 4.3: Use case diagram of the user Story "Delete a job post"

The table 4.3 describes the use case "Delete a job post" textually in the form of tables.

Use Case Scenario	Delete a Job Post
Actors	User
Pre-conditions	The user has to be logged in at least one job post is created by the user
Post-conditions	The desired job post is deleted from the database
Main scenario	1- the user clicks on the delete button under the post in his profile 2- the server deletes the desired post from the database

Table 4.3: Description of use case "Delete job posts"

4.2.2 Conception

In this step, we define how the interactions and use cases identified in the previous step should occur. This is a critical step that establishes the foundational logic required for successful development. To accomplish this, we employ Class and Sequence Diagrams to represent our logic in a standardized and formal manner.

4.2.2.1 Class Diagram of the User Story "Create a job post"

The figure 4.4 presents the class diagram of the use case "Create a job post".

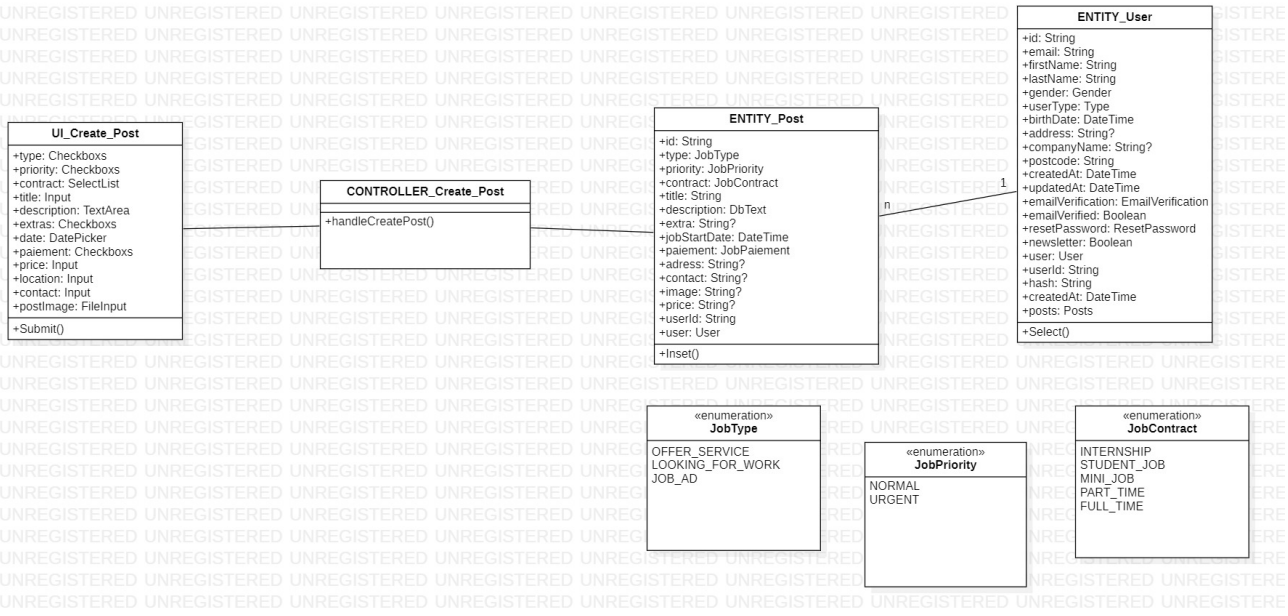

Figure 4.4: Class diagram of "Create a job post"

4.2.2.2 Sequence Diagram of the User Story "Create a job post"

The figure 4.5 presents the procedure that the user does in order to create a new job post, the user has to be logged in to access this user story.

- When the user access the Anzeigen tab of his profile he will be presented with his job posts if he has any as well as a button "neue Anzeige/add post" that opens a modal with a form.
- The user than inputs the desired data that will define the post's content and clicks "Anzeige aufgeben/Submit post".

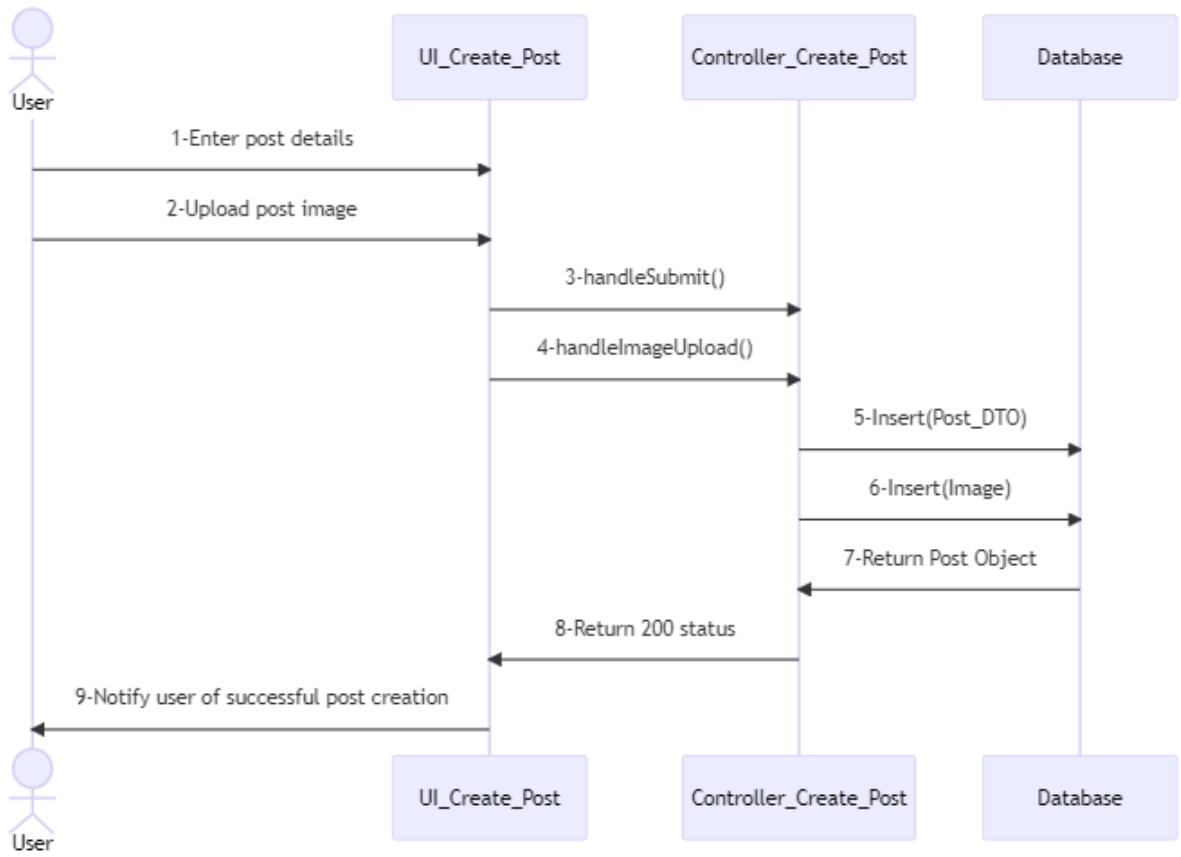

Figure 4.5: Sequence diagram of "Create a job post"

4.2.2.3 Class Diagram of the User Story "Edit a job post"

The figure 4.6 represents the Class diagram of the use case "Edit a job post".

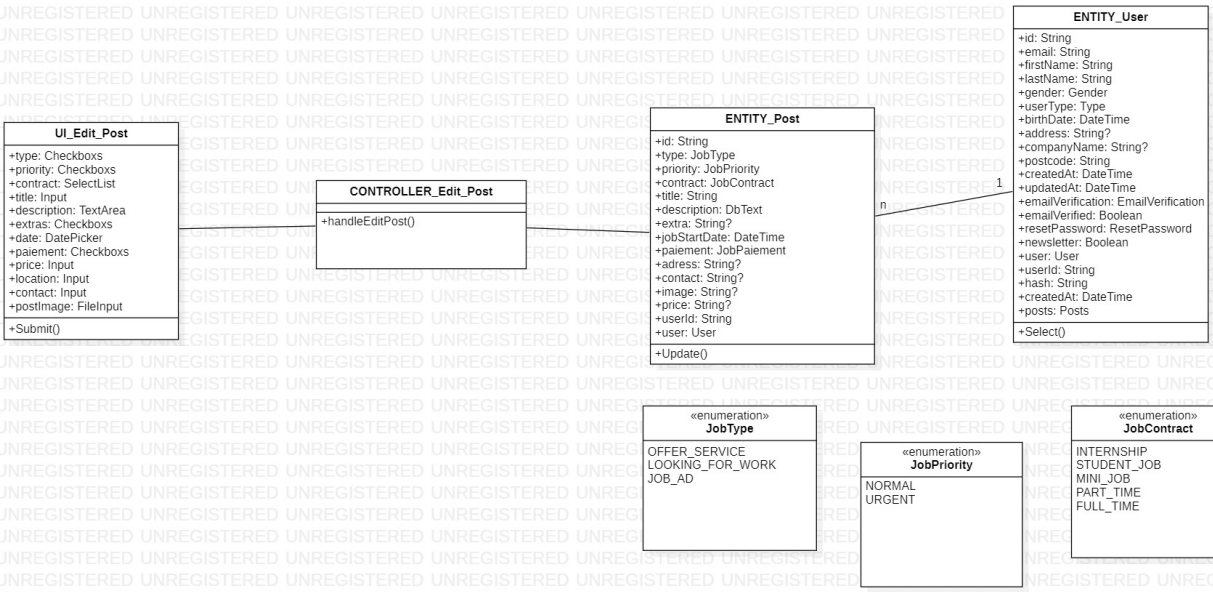

Figure 4.6: Class diagram of "Edit a job post"

4.2.2.4 Sequence Diagram of the User Story "Edit a job post"

The figure 4.7 presents how the user can edit his job posts to update its information, the user needs to be logged in to access this user story.

- when the user navigates to the Anzeigen tab of his profile, he will be presented with his job posts.
- the user can then click on the edit button "pencile icon button" to open a modal where he can edit the inputs to update the job post.
- when he's done he can then click "Anzeige aufgeben/Submit post" to submit the changes.

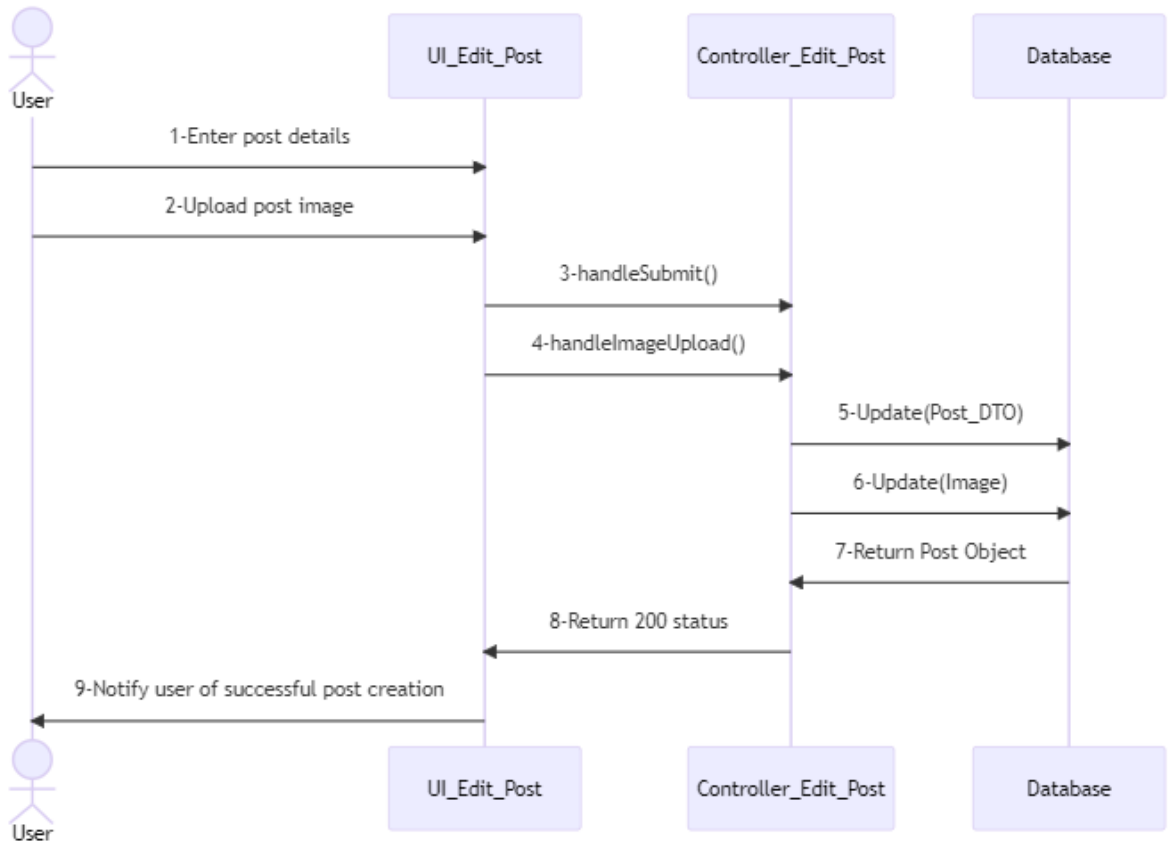

Figure 4.7: Sequence diagram of "Edit a job post"

4.2.2.5 Class Diagram of the User Story "Delete a job post"

The Figure 4.8 represents the class diagram of the use case "Delete a job post".

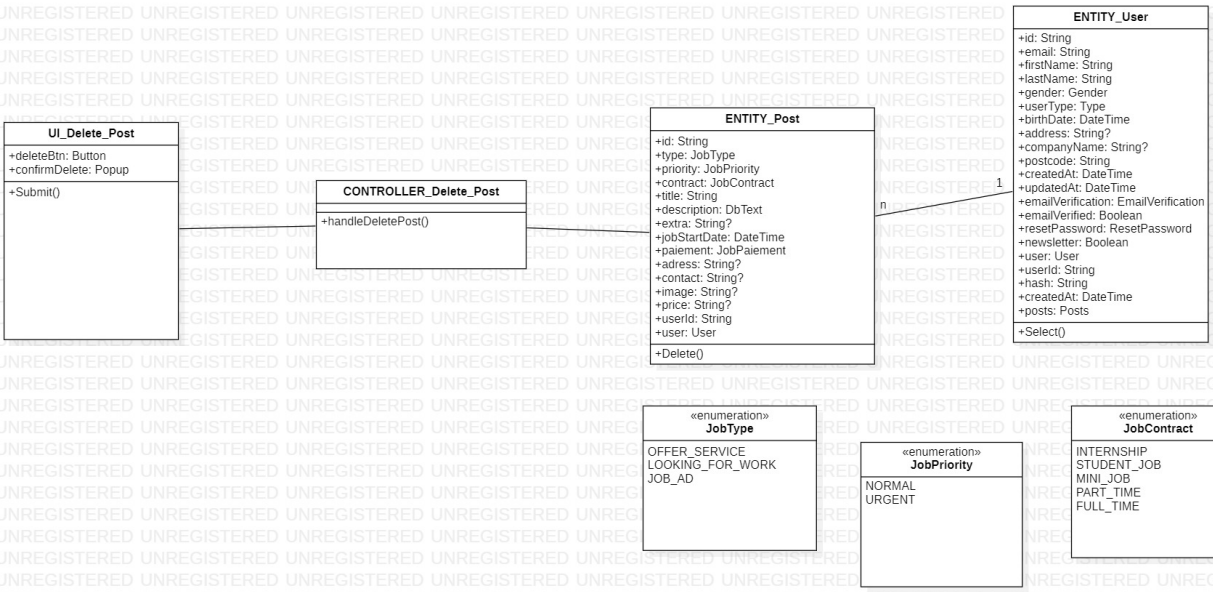

Figure 4.8: Class diagram of "Delete a job post"

4.2.2.6 Sequence Diagram of the User Story "Delete a job post"

The figure 4.9 presents how can the user delete one his job posts, the user has to be logged in to access this user story.

- when the user navigates to the Anzeigen tab of his profile he will be presented with his job posts, he can then click on the delete button under a post and it will be deleted, as simple as that.

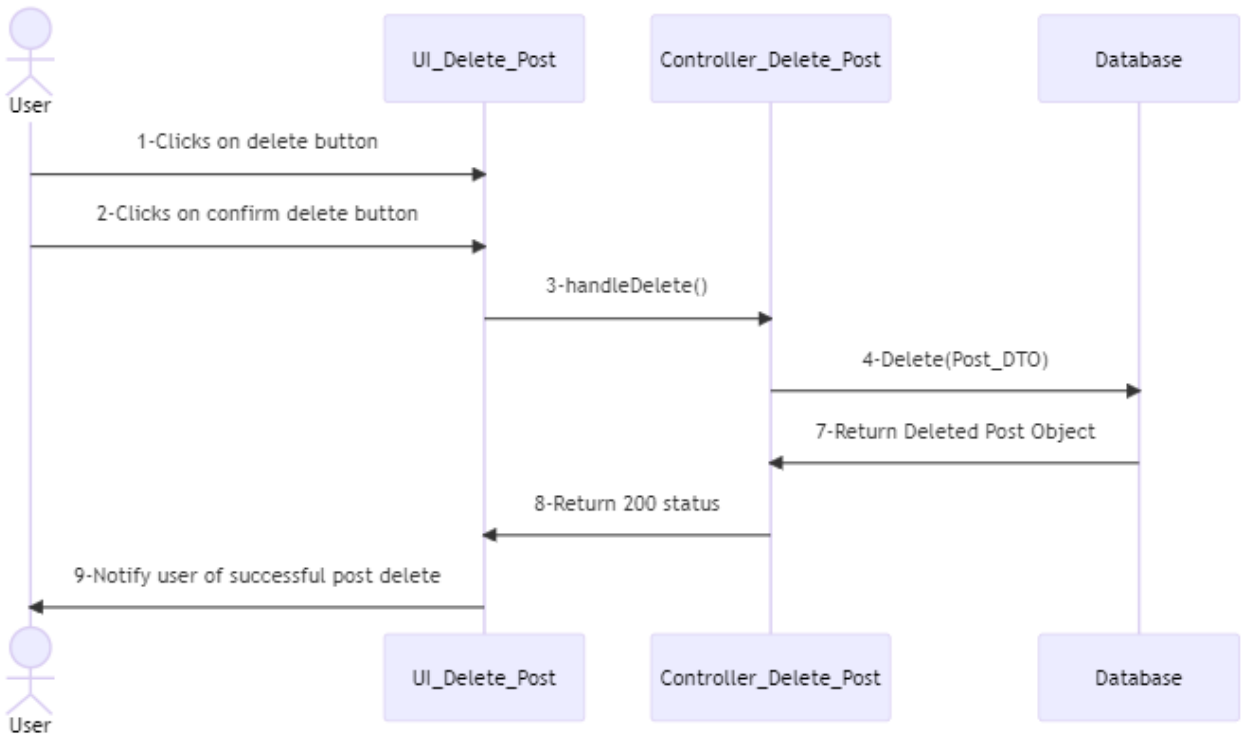

Figure 4.9: Sequence diagram of "Delete a job post"

4.3 Sprint 4

The following user stories show what our forth sprint consists of:

- As a user, i can pay for my job posts .

4.3.1 Identification

this section will present the use case diagram as well as the textual description in the form of a table.

4.3.1.1 Refinement of the Use case "Pay for a job post"

The figure 4.10 shows the use case diagram of the user story "Pay for a job post"

usecase_pay_post.png

Figure 4.10: Use case diagram of the user story 'Pay for a job post'

The table 4.4 describes the use case "Pay for a job post" textually in the form of tables.

Use Case Scenario	Pay for a Job Post
Actors	User
Pre-conditions	The user has to be logged in
Post-conditions	The desired job post is created and visible to the public
Main scenario	1- the user submits a job post 2- the user is presented with payment plans 3- the user selects a payment plan 4- the is redirected to stripe page to complete his payment 5- the user is presented with a summery of the post details 6- the user clicks "Jetzt Anzeige aufgeben" to confirm his payment 7- the server process the request 8- if the payment is successful the post is created and made visible to the public
Exception	if the payment isn't successfull the server returns an error

Table 4.4: Description of use case "Pay for a Job Post"

4.3.2 Conception

In this step, we define how the interactions and use cases identified in the previous step should occur. This is a critical step that establishes the foundational logic required for successful development. To accomplish this, we employ Class and Sequence Diagrams to represent our logic in a standardized and formal manner.

4.3.2.1 Class Diagram of the user Story "Pay for a job post"

The figure 4.11 presents the class diagram of the use case "Pay for a job post".

class_pay_post.png

Figure 4.11: Class diagram of "Pay for a job"

4.3.2.2 Sequence Diagram of the User Story "Pay for a job post"

The figure 4.12 showcase the process of paying for a job post, the user has to be logged in and finished "Create a job post" to access this user story

- when the user successfully finishes creating a post he will be presented with payment plans, he can then choose a plan and procceed with the payment in the stripe gateway interface.
- when he returns back to the web app he will be presented with a summery of the post details and a button that when clicked confirms his payment.

sequence_pay_post.png

Figure 4.12: Sequence diagram of "Pay for a job post"

4.4 Sprint 5

The following user stories show what our fifth sprint consists of:

- As a user, i can consult all job offers in the platform .

4.4.1 Identification

this section will present the use case diagram as well as the textual description in the form of a table.

4.4.1.1 Refinement of the use case "Consult Users job posts"

The figure 4.13 shows the use case diagram of the user story "Consult users job posts", this use story is accessible by anyone even without creating account.

Figure 4.13: Use case diagram of the user story 'Consult users job posts'

The table 4.5 describes the use case "Consult users job posts" textually in the form of tables.

Use Case Scenario	Consult users job posts
Actors	User
Pre-conditions	No pre-conditions
Post-conditions	All job posts of all the users in the platform is displayed
Main scenario	1- the user navigates to all posts page 2- the server returns all visible job posts submitted by the users

Table 4.5: Description of use case "Consult users job posts"

4.4.2 Conception

In this step, we define how the interactions and use cases identified in the previous step should occur. This is a critical step that establishes the foundational logic required for successful development. To accomplish this, we employ Class and Sequence Diagrams to represent our logic in a standardized and formal manner.

4.4.2.1 Class Diagram of the User Story "Consult users job posts"

The figure 4.14 represents the class diagram of the use case "Consult users job posts".

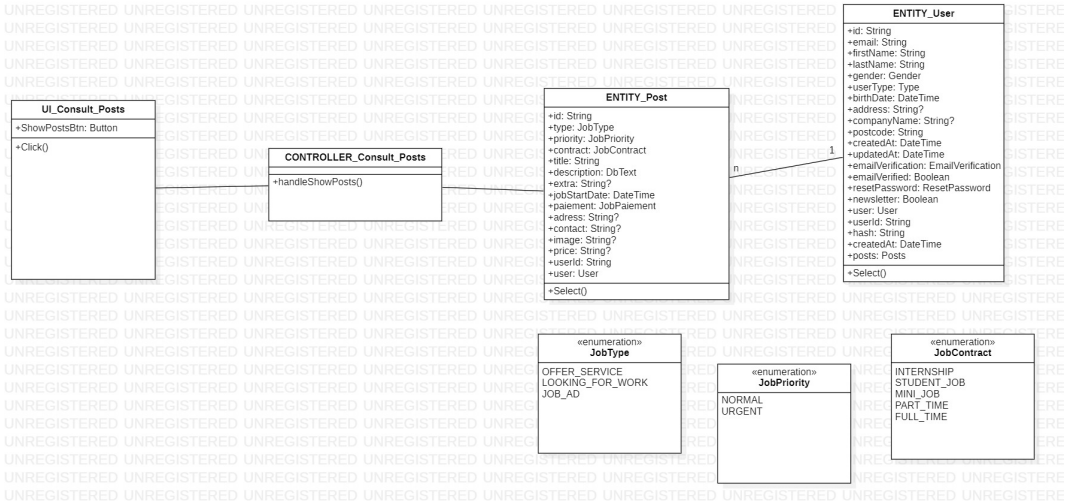

Figure 4.14: Class diagram of "Consult users job posts"

4.4.2.2 Sequence Diagram of the User Story "Consult users job posts"

The figure 4.15 presents how the user can view all visible job posts submitted by the users, the user can access this user story without logging in.

- when the user navigates to all posts page he will be presented with all job posts submitted by the users, he then can click on a job post to view its details.

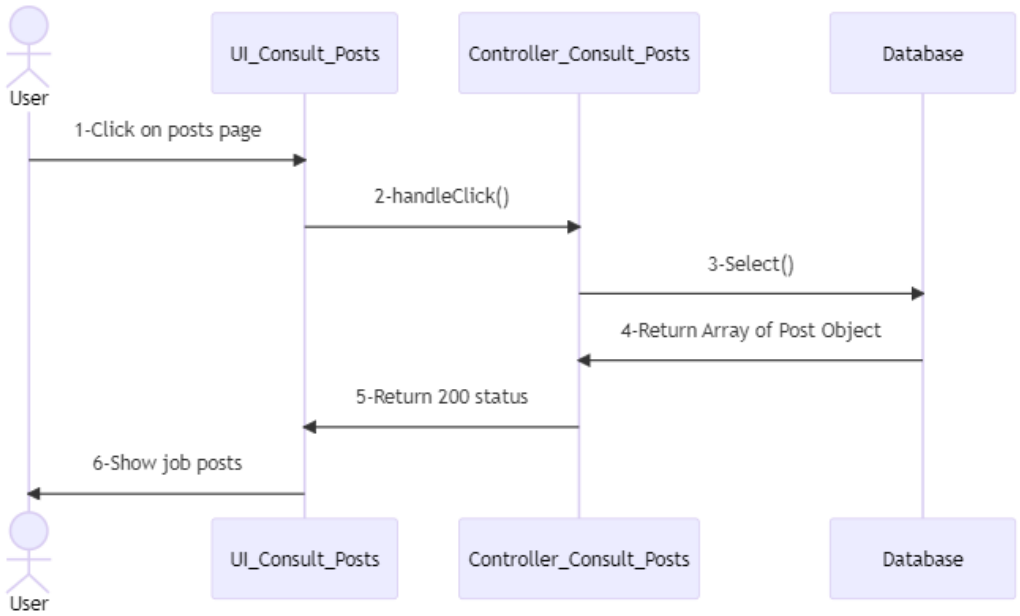

Figure 4.15: Sequence diagram of "Consult users job posts"

4.5 Realization

Realization of the use case "Create a job offer"

these interfaces shown in the figures 4.16 and 4.17 present the visual UI that is displayed to the user in order to create a new job post.

Figure 4.16: Realization of use case "Create a job post" 1

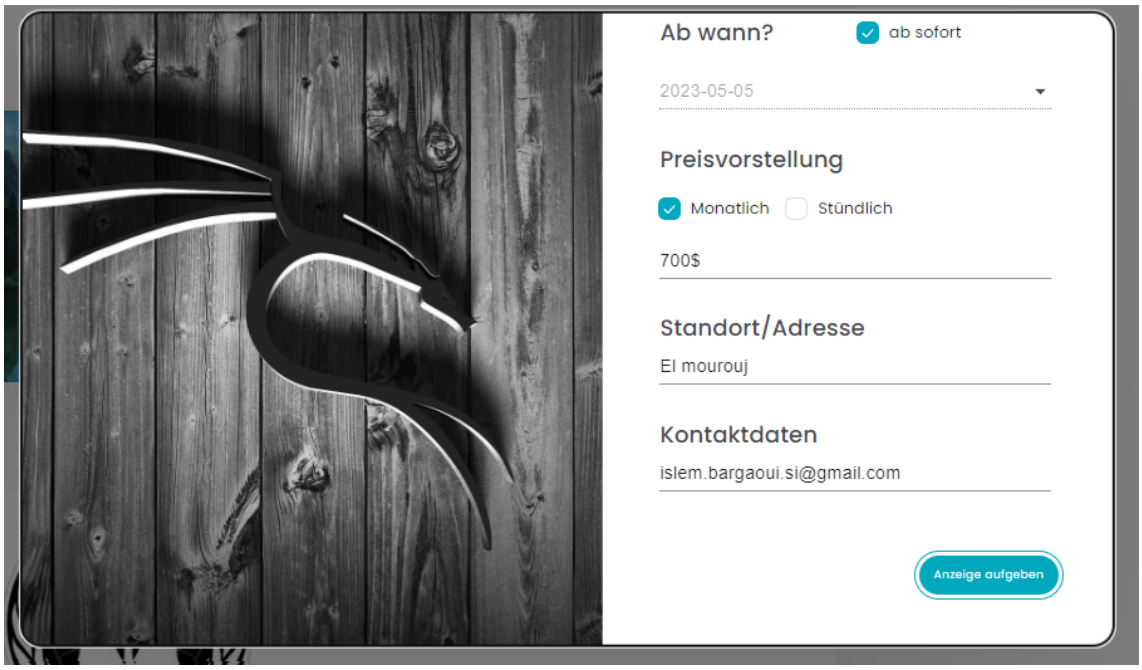

Figure 4.17: Realization of use case "Create a job post" 2

Realization of the use case "Edit a job post"

The figure 4.18 and 4.19 presents the inteface where the user can edit his job posts.

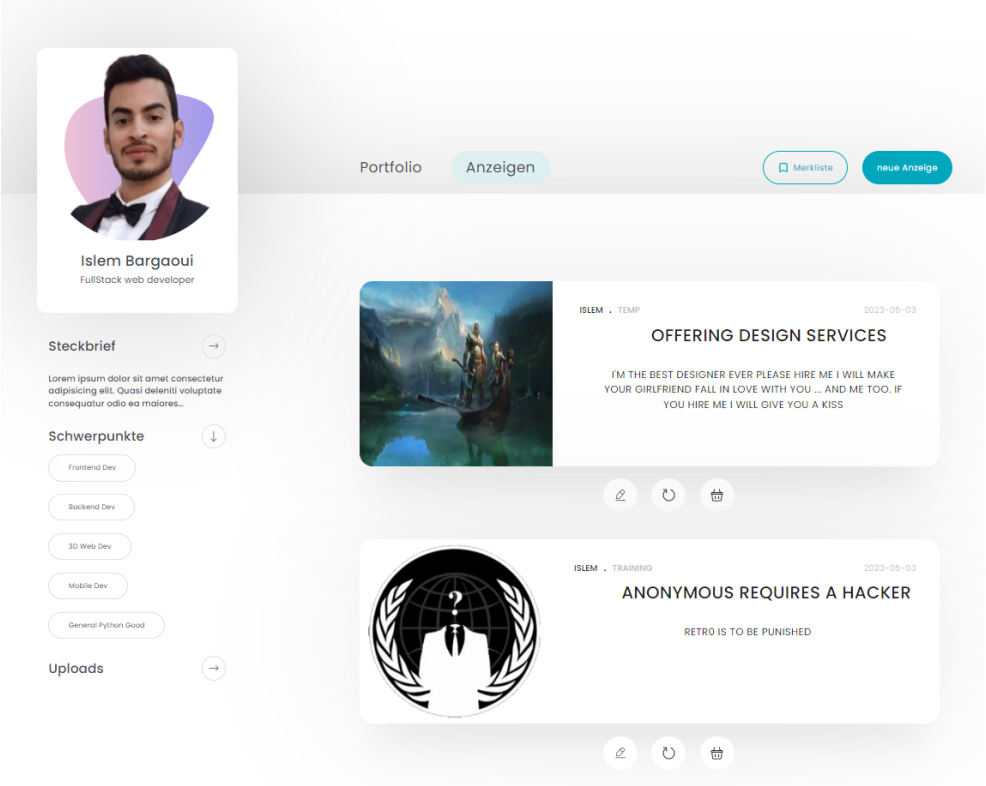

Figure 4.18: Realization of use case "Edit a job post" 1

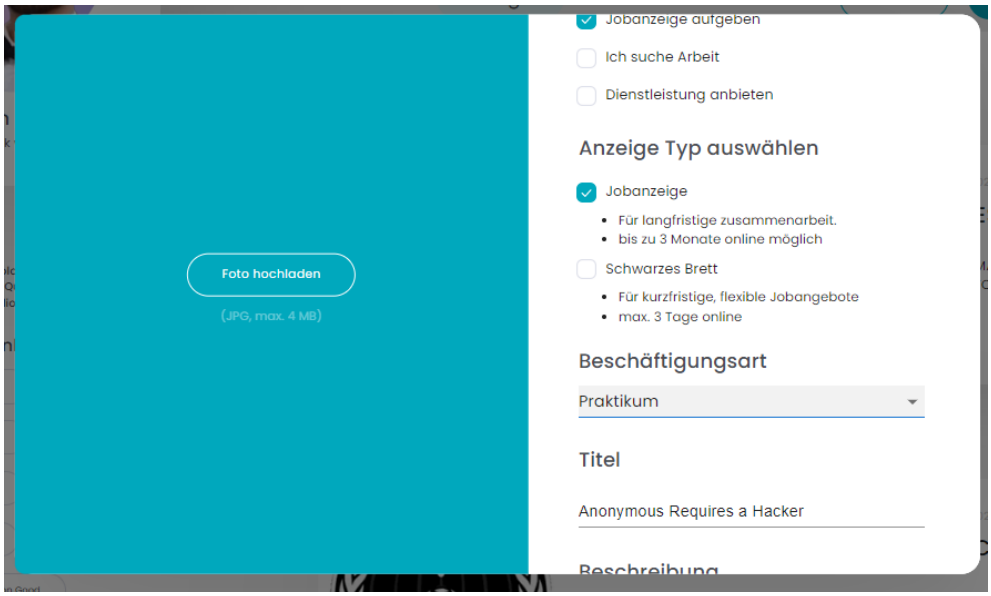

Figure 4.19: Realization of use case "Edit a job post" 2

Realization of the use case "Delete a job post"

The figure 4.20 and 4.21 present the interfaces of the use case "Delete a job post".

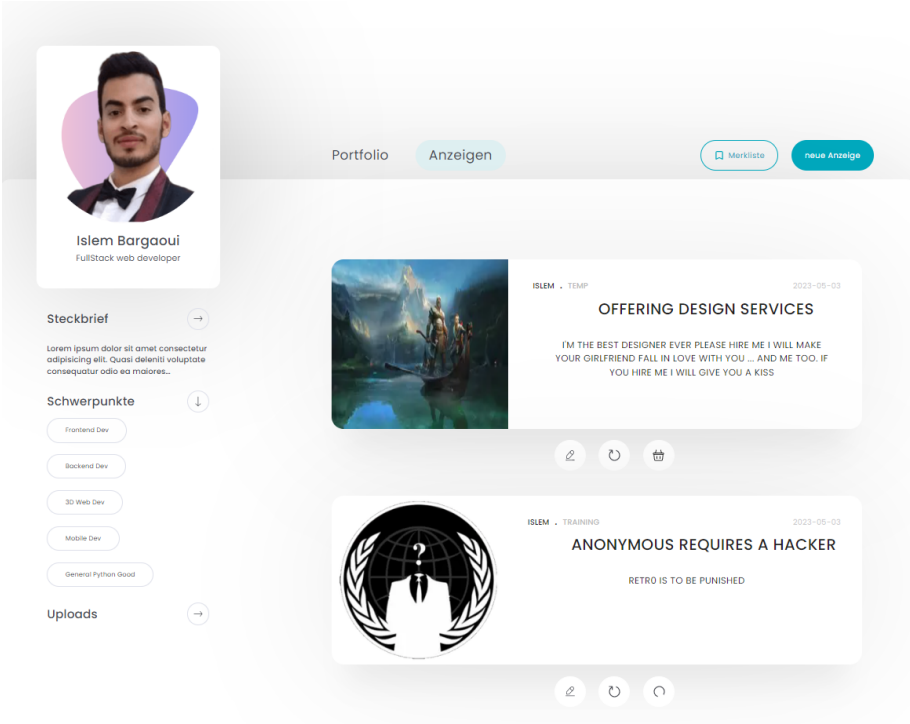

Figure 4.20: Realization of use case "Delete a job post" 1

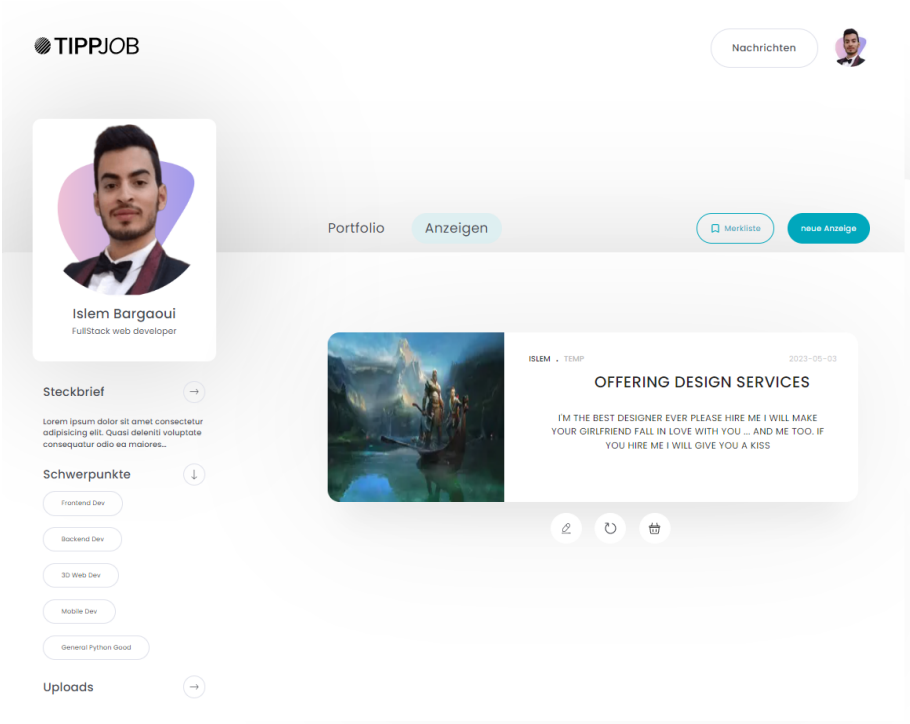

Figure 4.21: Realization of use case "Delete a job post" 2

Realization of the use case "Pay for a job post"

The figure 4.22, 4.23 and 4.24 represent the "Pay for a job post" interface and how the user can make his post visible to the public.

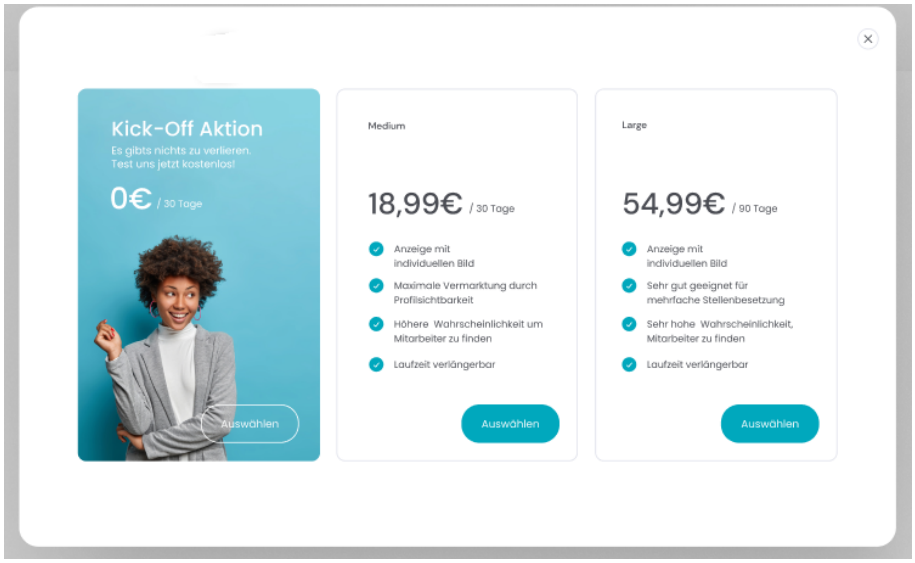

Figure 4.22: payment plans interface

this_interface_is_still_under_development.png

Figure 4.23: payment realisation interface

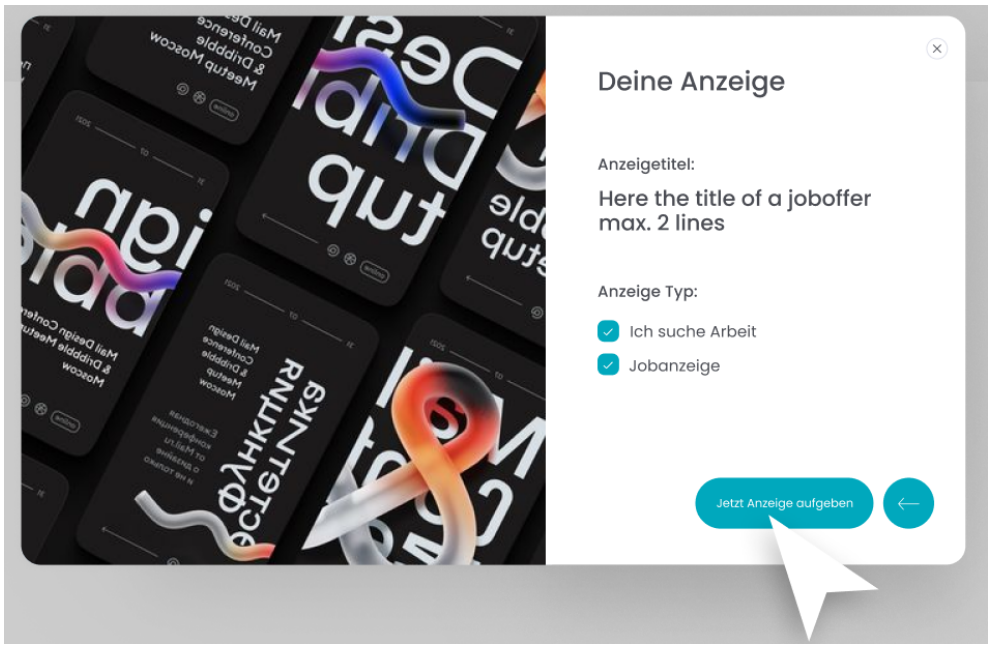

Figure 4.24: payment confirmation interface

Realization of the use case "Consult users job posts"

The figures 4.25, 4.26, 4.27 and 4.28 presents the interface where all visible posts submitted by the users are displayed. the interface is divided into 4 categories:

- posts that are of type "Looking for a job" or "Offering a service".
- posts that are of type "Offering a job".
- posts that are of type "Looking for a job" or "Offering a service" and are URGENT, this page is displayed in dark mode.
- posts that are of type "Offering a job" and are URGENT, this page is displayed in dark mode.

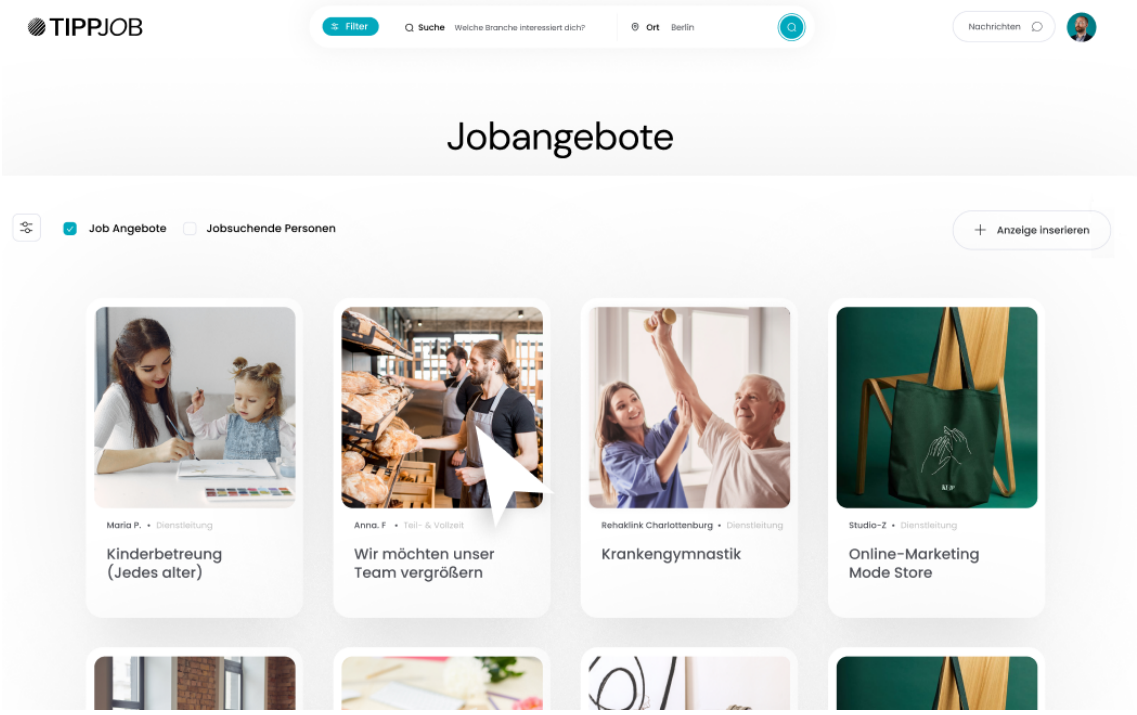

Figure 4.25: interface of all posts that are Offering a job

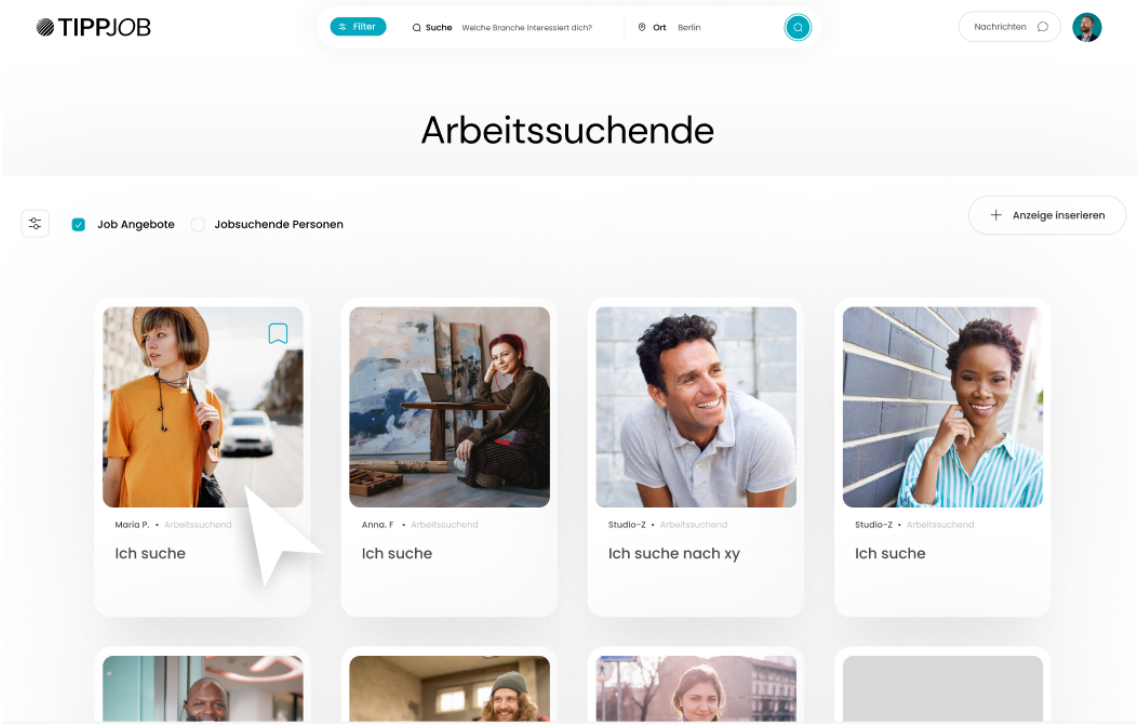

Figure 4.26: interface of all posts that are Looking for a job

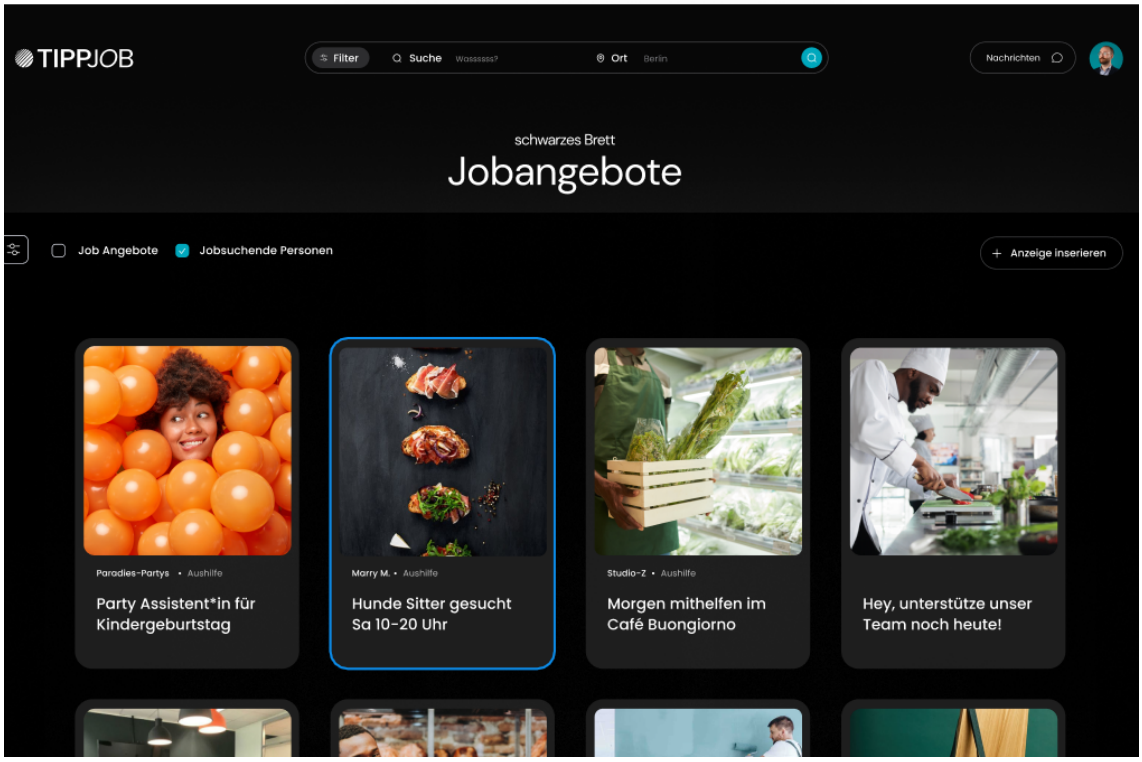

Figure 4.27: interface of all URGENT posts that are Offering a job

Figure 4.28: interface of all URGENT posts that are Looking for a job

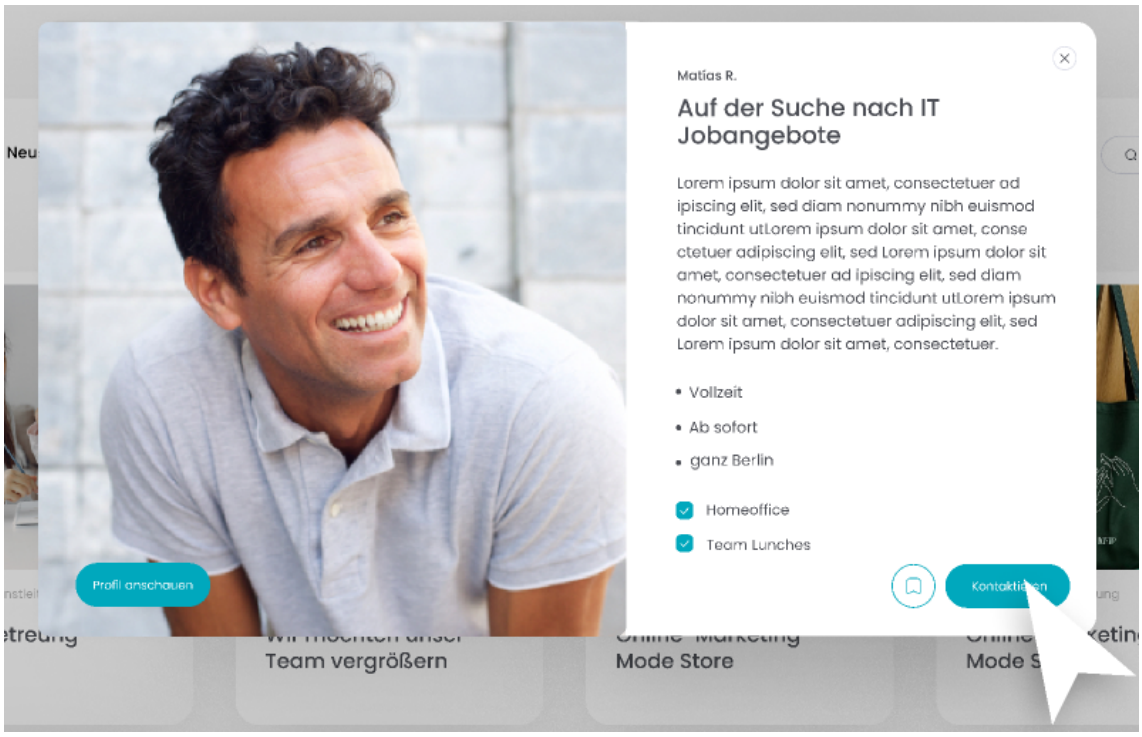

Figure 4.29: Details of a job post

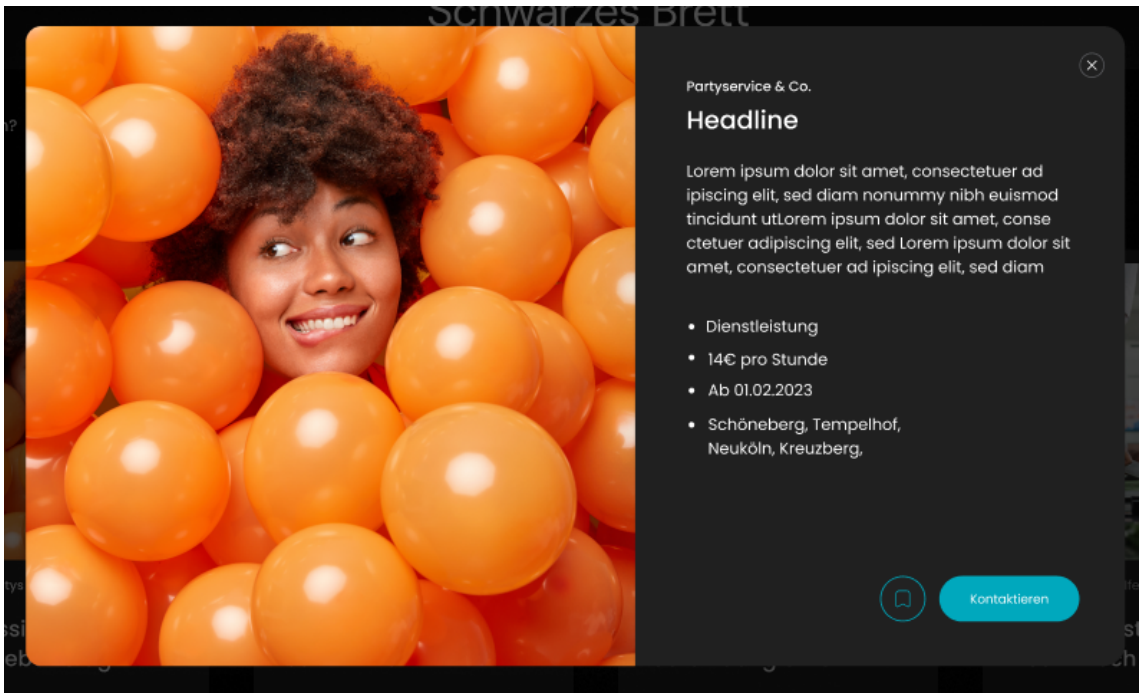

Figure 4.30: Details of an URGENT job post

4.6 Conclusion

This chapter covers all the steps involved in delivering the second release, including the refinement, design, and implementation of the 3 sprints included.

Chapter 5

Release 3: Enhancements to the User Experience

5.1 Introduction

We have now reached the final chapter of the project, which focuses on improving the user experience (UX). This includes features such as the ability to view user profiles, create a bookmark list for job posts, use a search and filter system, and utilize a messenger.

To begin this chapter, we first identify the requirements of our system's users during the two sprints outlined in the system backlog. We then refine the use cases, which allows us to create class and sequence diagrams for implementation. Finally, we present the outcome of this release.

5.2 Sprint 6

The following user stories shows what our sixth sprint consists of:

- As a user, i can search for job posts
- As a user, i can filter job posts

5.2.1 Identification

This section contains a collection of use case diagrams for our use cases, accompanied by a table that provides a textual description of each use case.

5.2.1.1 Refinement of the User Story "Search for job posts"

The use-case diagram for the user story "Search for job posts" is shown in the figure 5.1

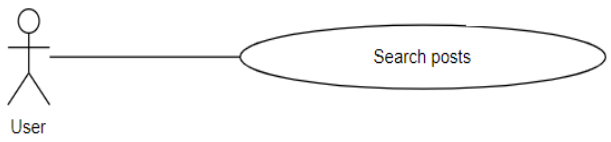

Figure 5.1: Use case diagram of the user story "Search for job posts"

The table 5.1 describes the use case "Search for job posts" textually in the form of tables.

Use Case Scenario	Search for job posts
Actors	User
Pre-conditions	No pre-conditions
Post-conditions	The posts are filtered depending on the search query
Main scenario	1- the user enters a search text in the searchBar 2- the server returns the filtered posts.

Table 5.1: Description of use case "Search for job posts"

5.2.1.2 Refinement of the User Story "filter job posts"

The use-case diagram for the user story "filter job posts" is shown in the figure 5.2

Figure 5.2: Use case diagram of the user story "filter job posts"

The table 5.2 describes the use case "filter job posts" textually in the form of tables.

Use Case Scenario	Filter for job posts
Actors	User
Pre-conditions	No pre-conditions
Post-conditions	The posts are filtered depending on the filter queries
Main scenario	1- the user selects filter options 2- the server returns the filtered posts.

Table 5.2: Description of use case "filter job posts"

5.2.2 Conception

In this step, we define how the interactions and use cases identified in the previous step should occur. This is a critical step that establishes the foundational logic required for successful development. To accomplish this, we employ Class and Sequence Diagrams to represent our logic in a standardized and formal manner.

5.2.2.1 Class Diagram of the User Story "Search for job posts"

The figure 5.3 represents the class diagram of the use case "Search for job posts".

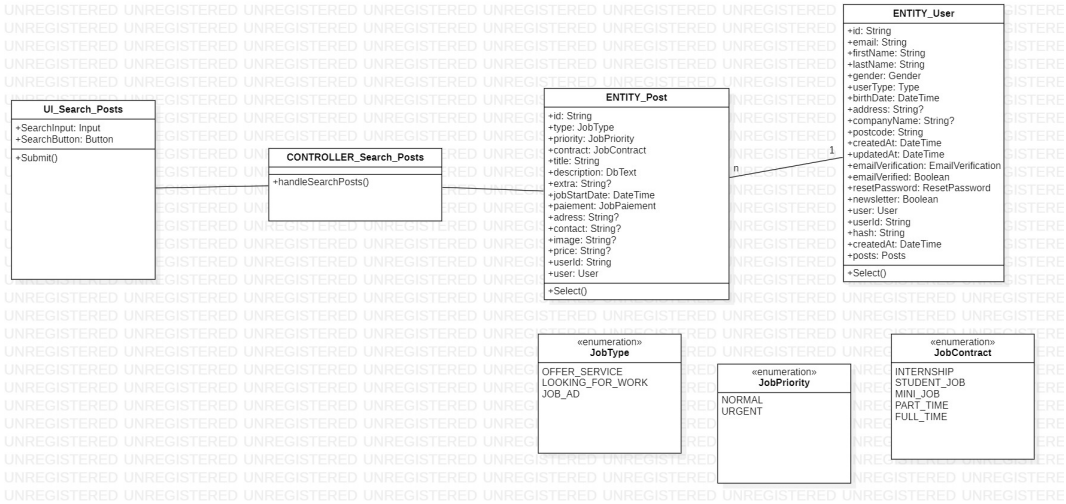

Figure 5.3: Class diagram of "Search for job posts"

5.2.2.2 Sequence Diagram of the User Story "Search for job posts"

The figure 5.4 presents how the user can search for a job posts by a text from the title or the description, the user can access this user story without logging in.

- The user is able to enter a text in the search bar to search by query in the title or the description
- The user can also specify a location in the search bar to filter by it in addition to the text query. (the location is also a text field).

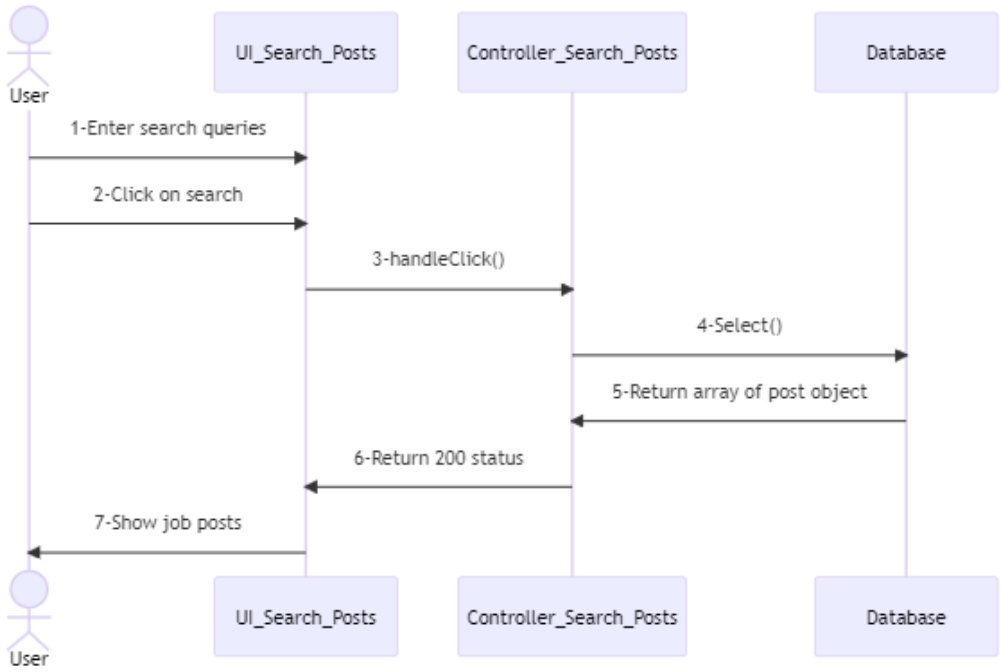

Figure 5.4: Sequence diagram of "Search for job posts"

5.2.2.3 Class Diagram of the User Story "Filter job posts"

The figure 5.5 represents the class diagram of the use case "Filter job posts".

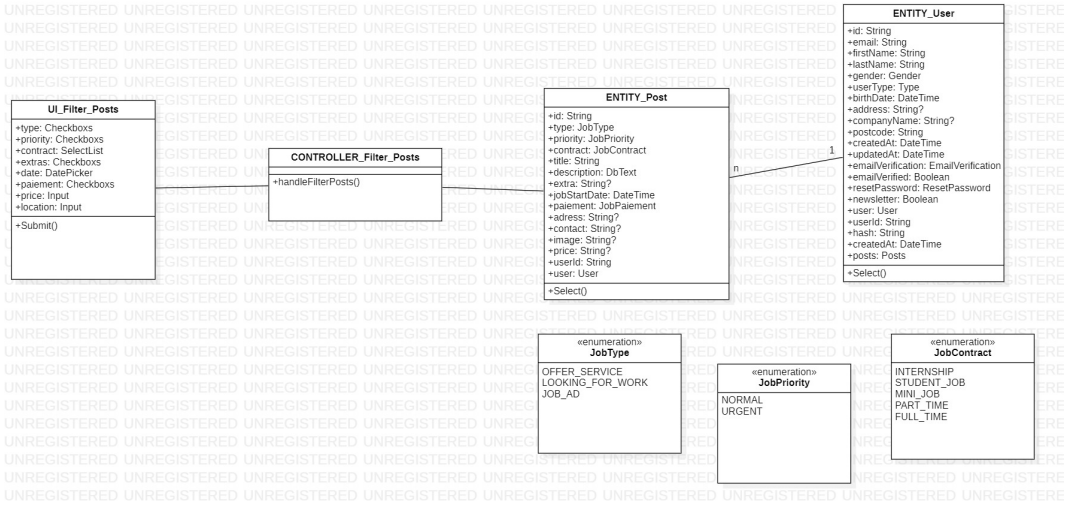

Figure 5.5: Class diagram of "Filter job posts"

5.2.2.4 Sequence Diagram of the User Story "Filter job posts"

The figure 5.6 presents how the user can filter the displayed job posts, the user can access this user story without logging in.

- The user is able to click on "Filter" in the search bar and select the filter options from the opened popup.

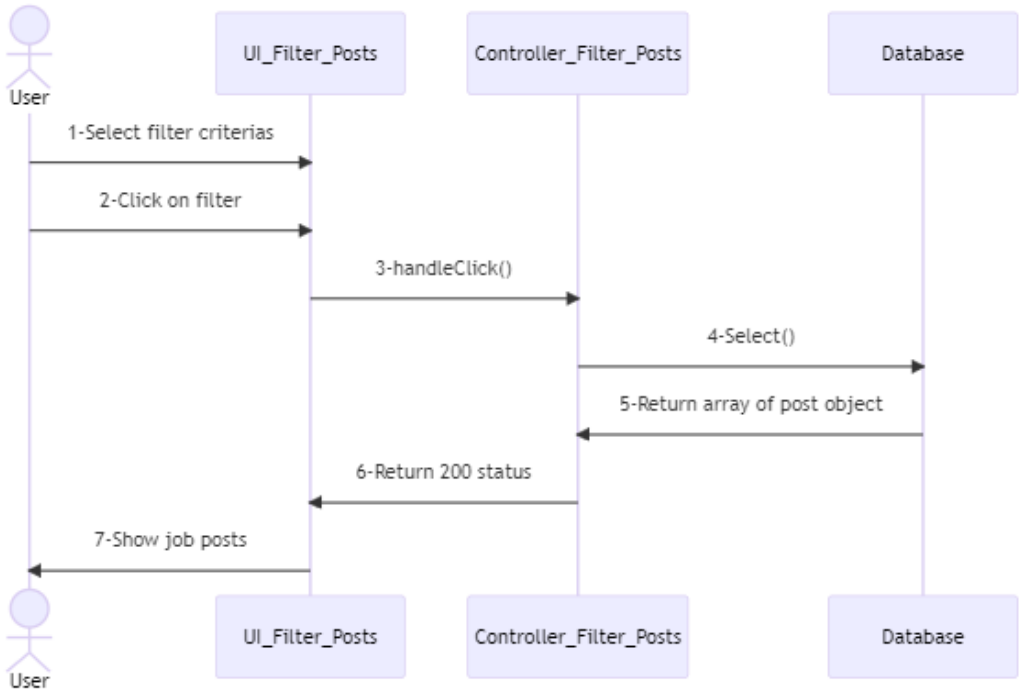

Figure 5.6: Sequence diagram of "Filter job posts"

5.3 Sprint 7

The following user stories shows what our seventh sprint consists of:

- As a user, i can favorite job offers.
- As a user, i can consult other users profiles.
- As a user, i can exchange messages with another user.

5.3.1 Identification

This section contains a collection of use case diagrams for our use cases, accompanied by a table that provides a textual description of each use case.

5.3.1.1 Refinement of the User Story "Favorite job posts"

The use-case diagram for the user story "Favorite job posts" is shown in the figure 5.7

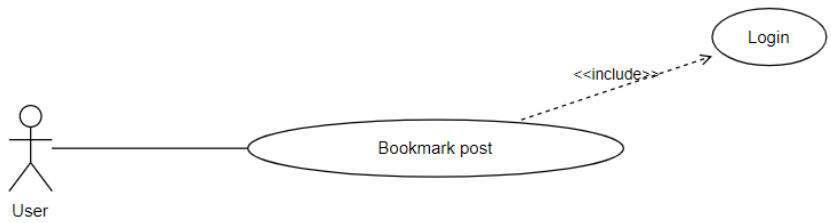

Figure 5.7: Use case diagram of the user story "Favorite job posts"

The table 5.3 describes the use case "Favorite job posts" textually in the form of tables.

Use Case Scenario	Favorite for job posts
Actors	User
Pre-conditions	The user has to be logged in
Post-conditions	The designated post is added to the bookmarks of the user
Main scenario	1- The user access the details of a post 2- the user clicks on the bookmark button 3- the server adds the post to the user's bookmarks

Table 5.3: Description of the use case "Favorite job posts"

5.3.1.2 Refinement of the User Story "Consult users profiles"

The use-case diagram for the user story "Consult users profiles" is shown in the figure 5.8

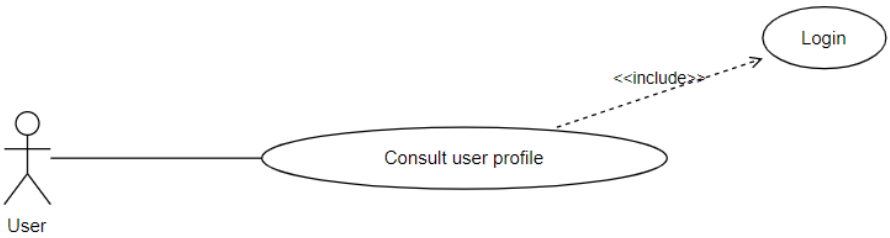

Figure 5.8: Use case diagram of the user story "Consult users profiles"

The table 5.4 describes the use case "Consult users profiles" textually in the form of tables.

Use Case Scenario	Consult users profiles
Actors	User
Pre-conditions	The user has to be logged in
Post-conditions	The designated user's profile is displayed
Main scenario	1- The user access the details of a post 2- the user clicks on "View profile/Profil anschauen" 3- the server redirects the user to the designated profile

Table 5.4: Description of the use case "Favorite job posts"

5.3.1.3 Refinement of the User Story "Message users"

The use-case diagram for the user story "Message users" is shown in the figure 5.9

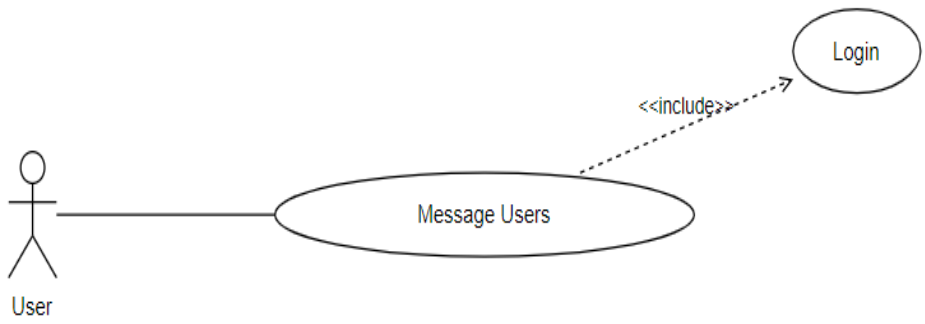

Figure 5.9: Use case diagram of the user story "Message users"

The table 5.4 describes the use case "Message users" textually in the form of tables.

Use Case Scenario	Message users
Actors	User
Pre-conditions	The user has to be logged in
Post-conditions	A conversation is launched between the 2 users
Main scenario	1- the user access the details of a post 2- the user clicks on "Message/Kontaktieren" 3- the user is redirected to the messenger UI 4- the user sends a message 5- a conversation is started between the 2 users

Table 5.5: Description of the use case "Message users"

5.3.2 Conception

In this step, we define how the interactions and use cases identified in the previous step should occur. This is a critical step that establishes the foundational logic required for successful development. To accomplish this, we employ Class and Sequence Diagrams to represent our logic in a standardized and formal manner.

5.3.2.1 Class Diagram of the User Story "Favorite job posts"

The figure 5.10 represents the class diagram of the use case "Favorite job posts".

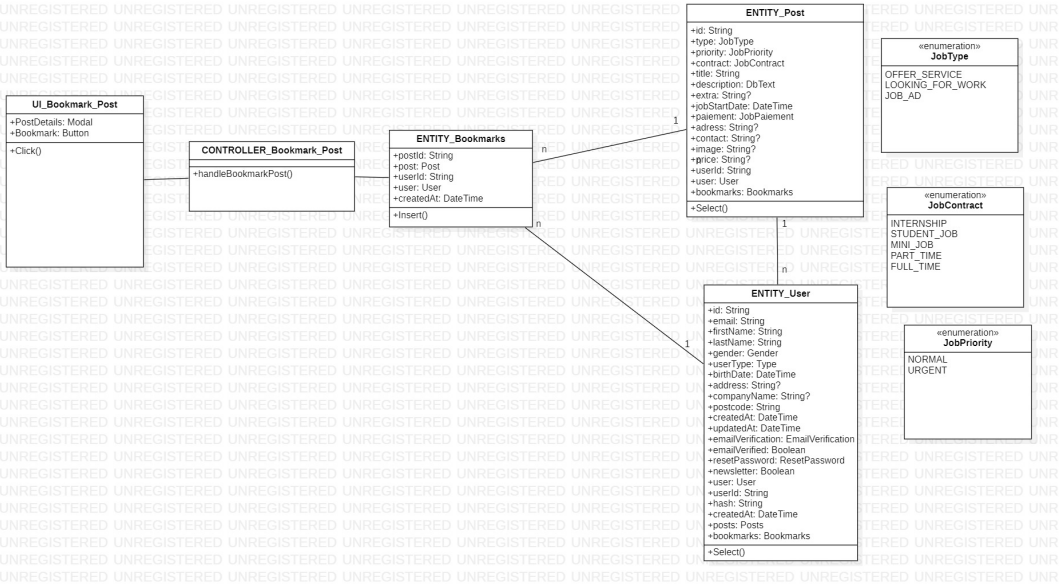

Figure 5.10: Class diagram of "Favorite job posts"

5.3.2.2 Sequence Diagram of the User Story "Favorite job posts"

The figure 5.11 presents how the user can Favorite job posts so he can access them later, the user has to be logged in to access this user story.

- when the user clicks on the favorite button in the details of a post, the post is then added to the user's bookmarks.

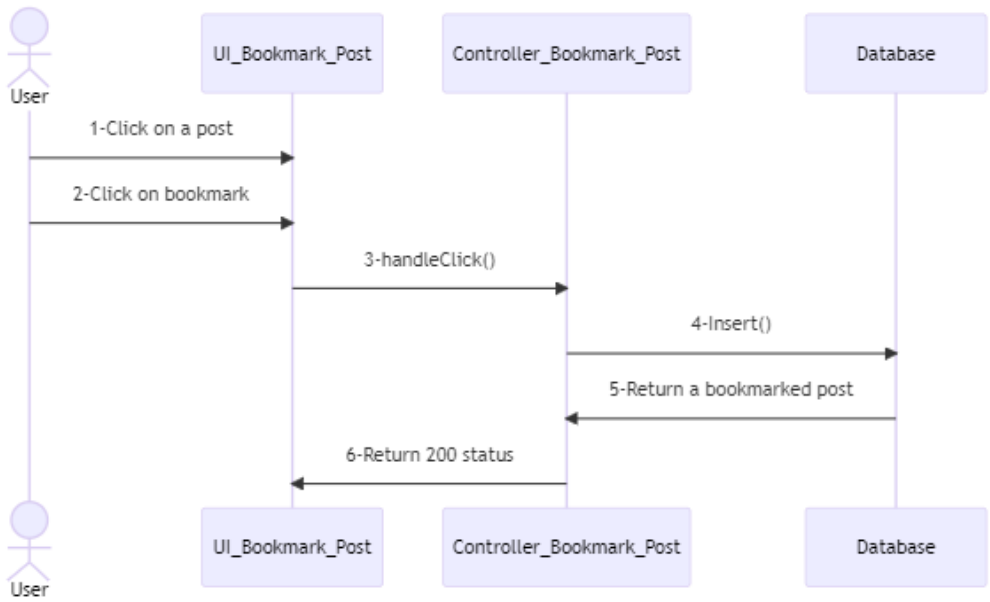

Figure 5.11: Sequence diagram of "Favorite job posts"

5.3.2.3 Class Diagram of the User Story "Consult users profiles"

The figure 5.12 represents the class diagram of the use case "Consult users profiles".

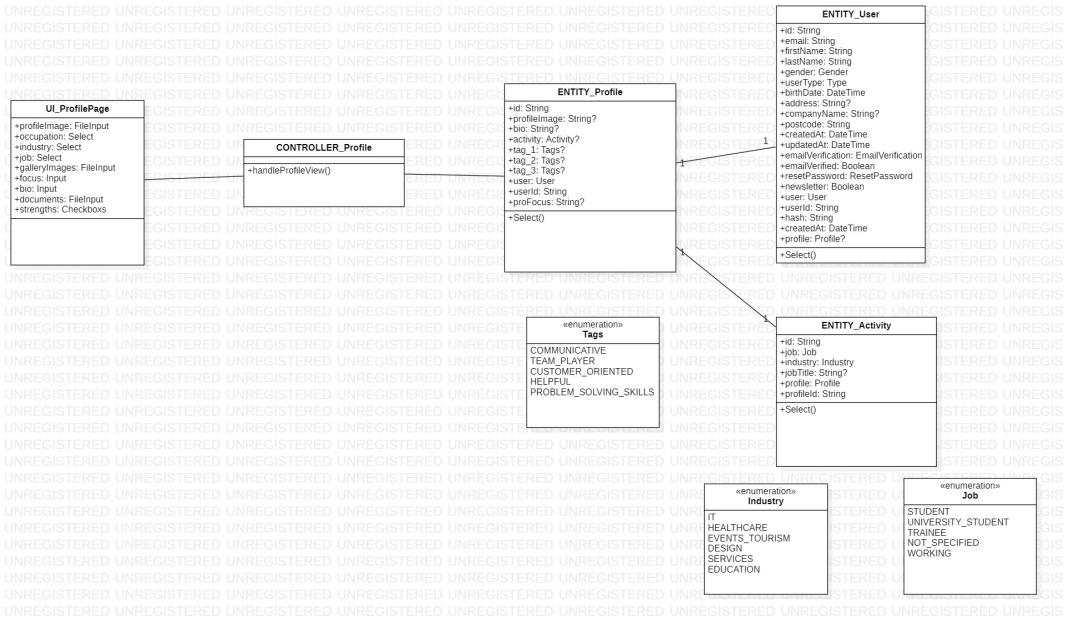

Figure 5.12: Class diagram of "Consult users profiles"

5.3.2.4 Sequence Diagram of the User Story "Consult users profiles"

The figure 5.13 presents how the user can view other users profile page, the user has to be logged in to access this user story.

- when the user clicks on "View profile/Profil anschauen" in the details of a post he will be redirected to the designated user's profile

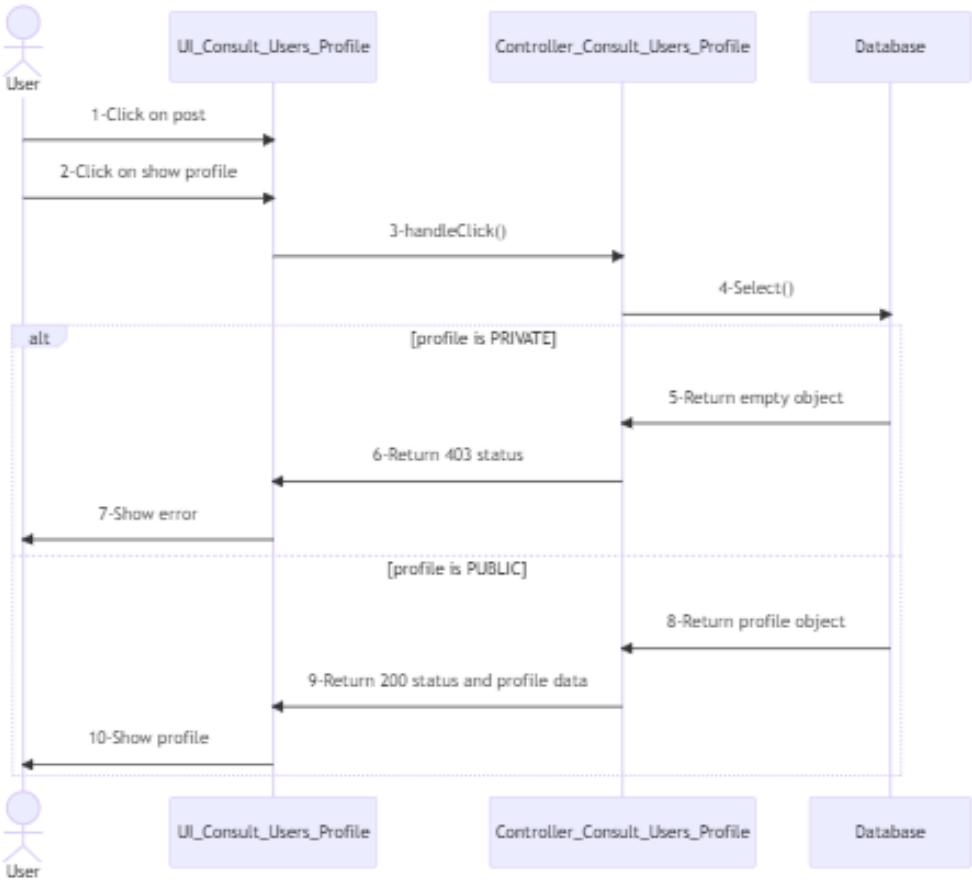

Figure 5.13: Sequence diagram of "Consult users profiles"

5.3.2.5 Class Diagram of the User Story "Message users"

The figure 5.14 represents the class diagram of the use case "Message users".

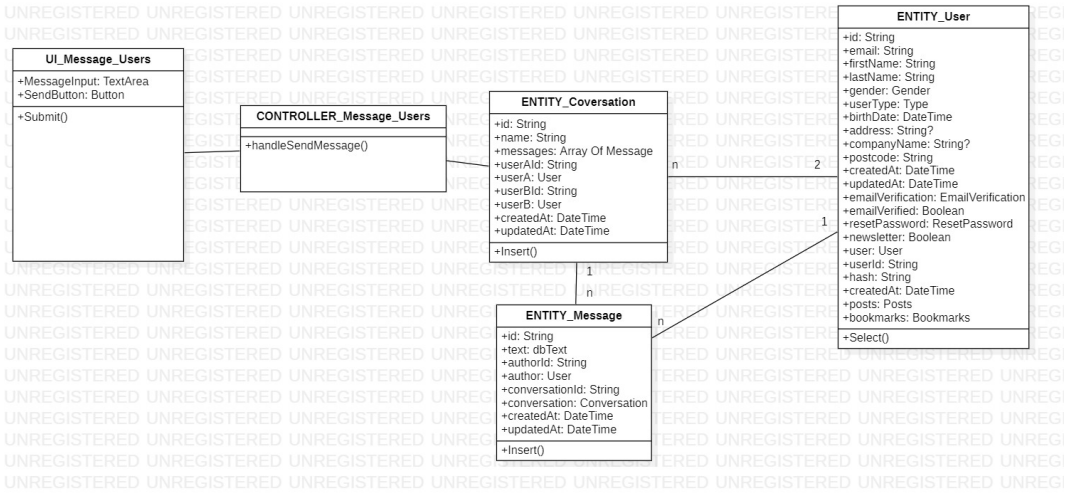

Figure 5.14: Class diagram of "Message users"

5.3.2.6 Sequence Diagram of the User Story "Message users"

The figure 5.15 presents how the user can exchange messages with another user, the user has to be logged in to access this user story.

- when the user clicks on "Message/Kontaktieren" in the details of a post he will be redirected to the messenger interface.
- when the user sends a message a new conversation is created and saved, and the message is sent to the other user.

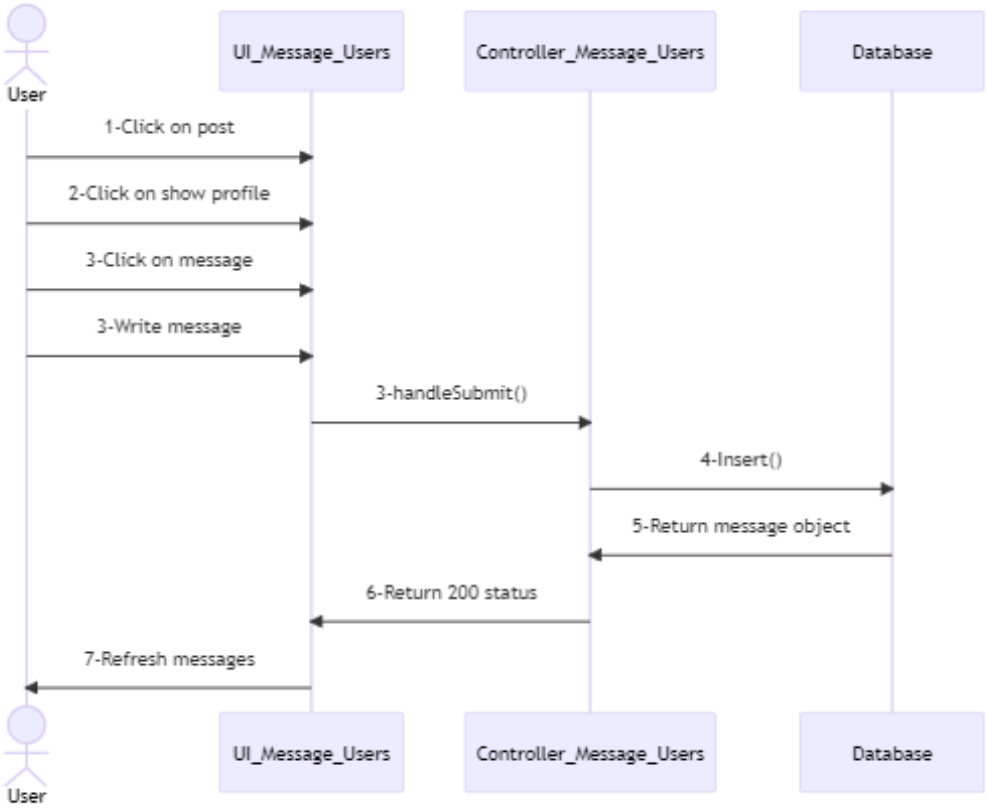

Figure 5.15: Sequence diagram of "Message users"

5.3.3 Global Classes Diagram

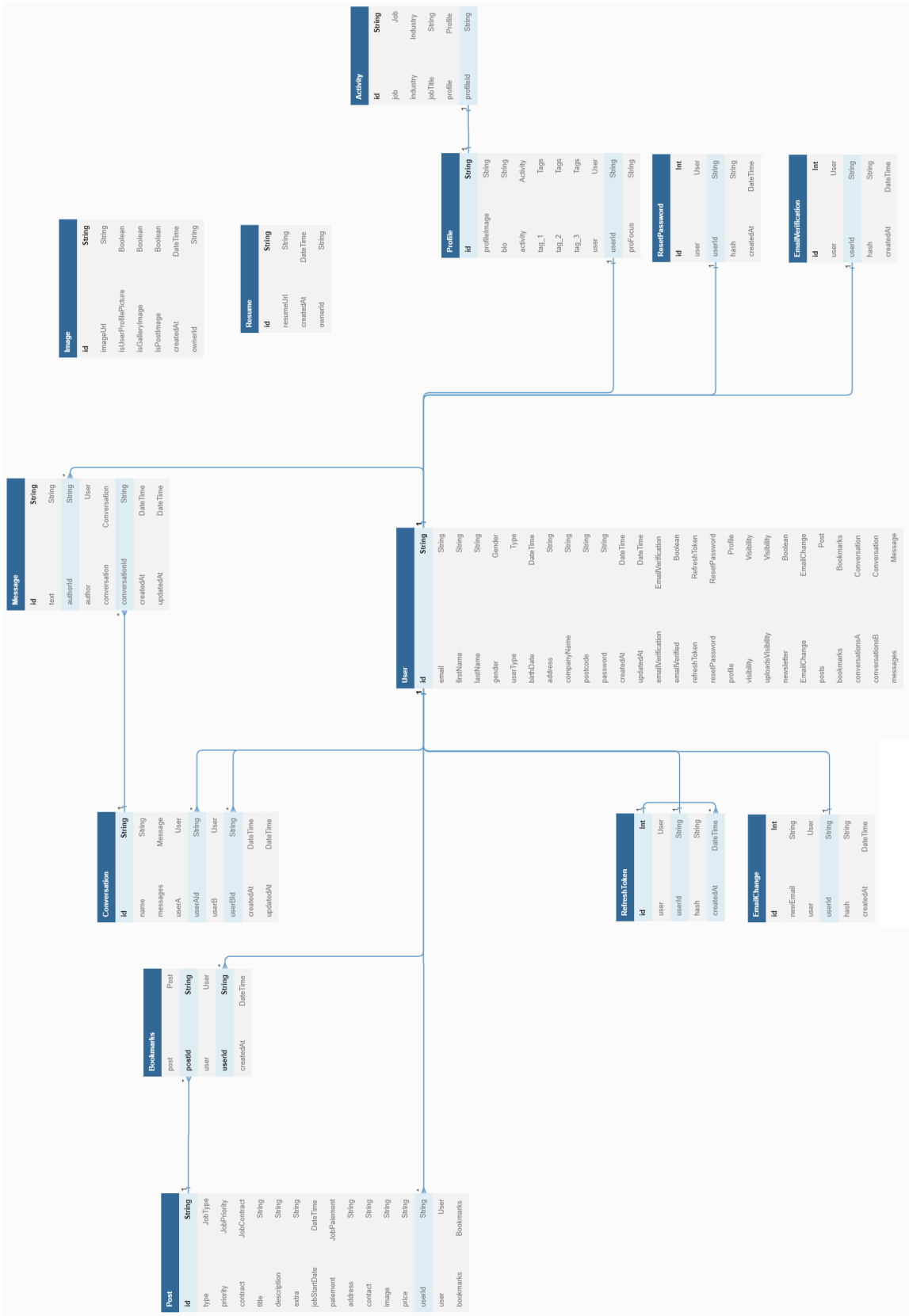

Figure 5.16: Global Classes Diagram

5.4 Realization

Realization of the use case "Favorite job posts"

These interfaces shown in the figures 5.16 and 5.17 present the visual UI that is displayed to the user in order to favorite a job post.

Figure 5.17: Realization of use case "Favorite job posts" 1

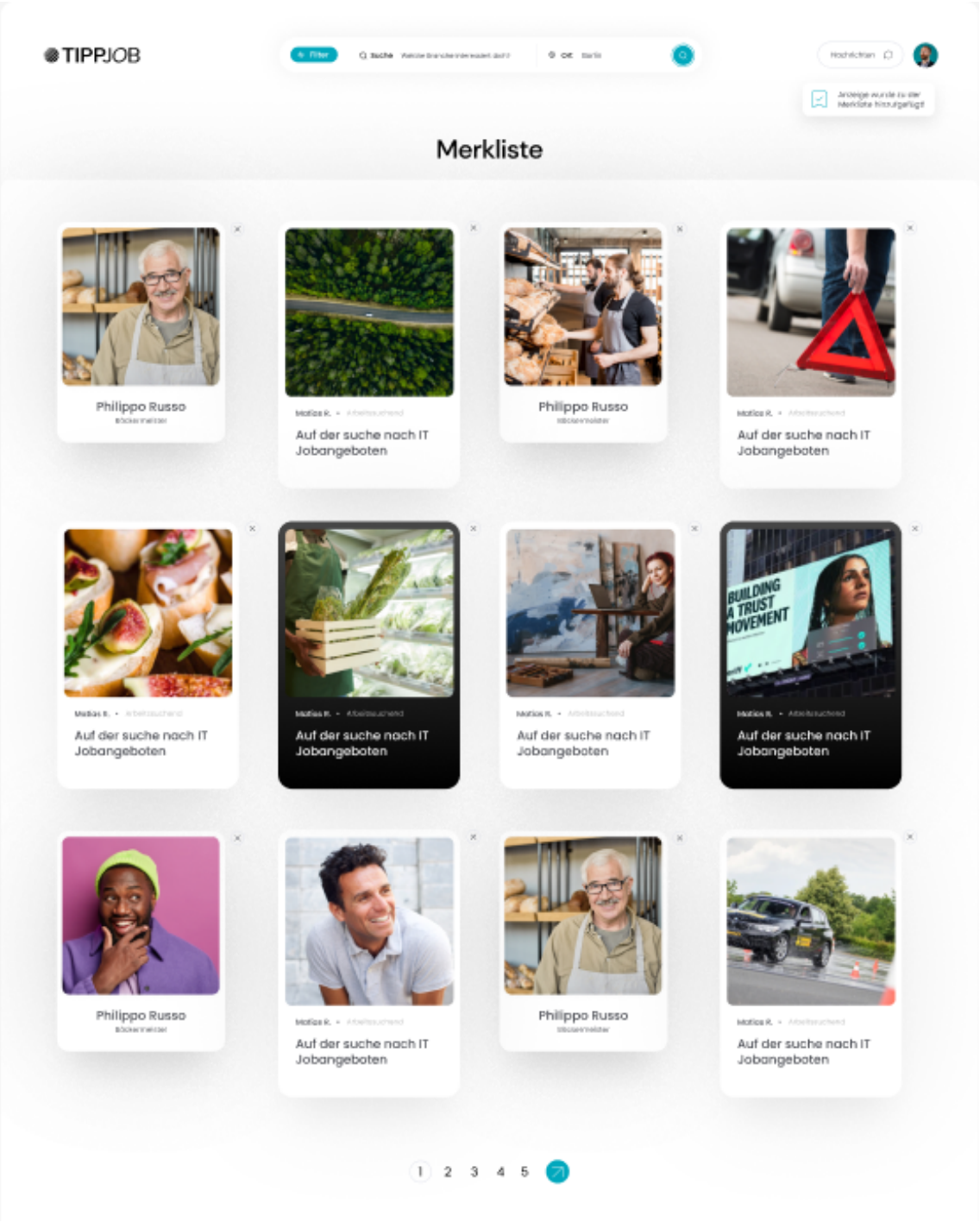

Figure 5.18: Realization of use case "Favorite job posts" 2

Realization of the use case "Consult users profiles"

These interfaces shown in the figures 5.18 and 5.19 present the visual UI that is displayed to the user in order to View another user's profile.

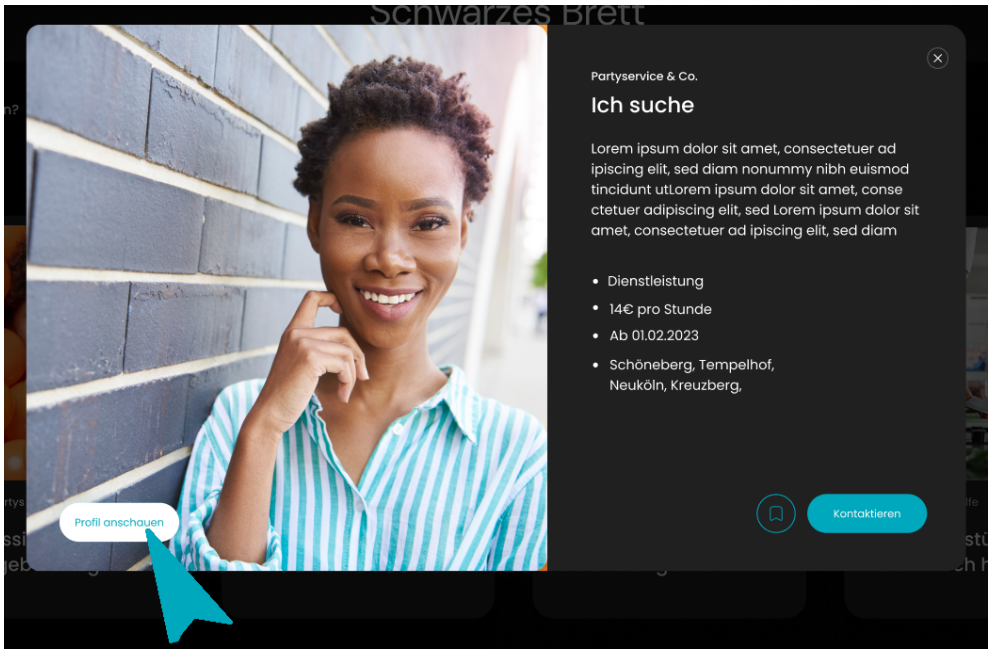

Figure 5.19: Realization of use case "Consult users profiles" 1

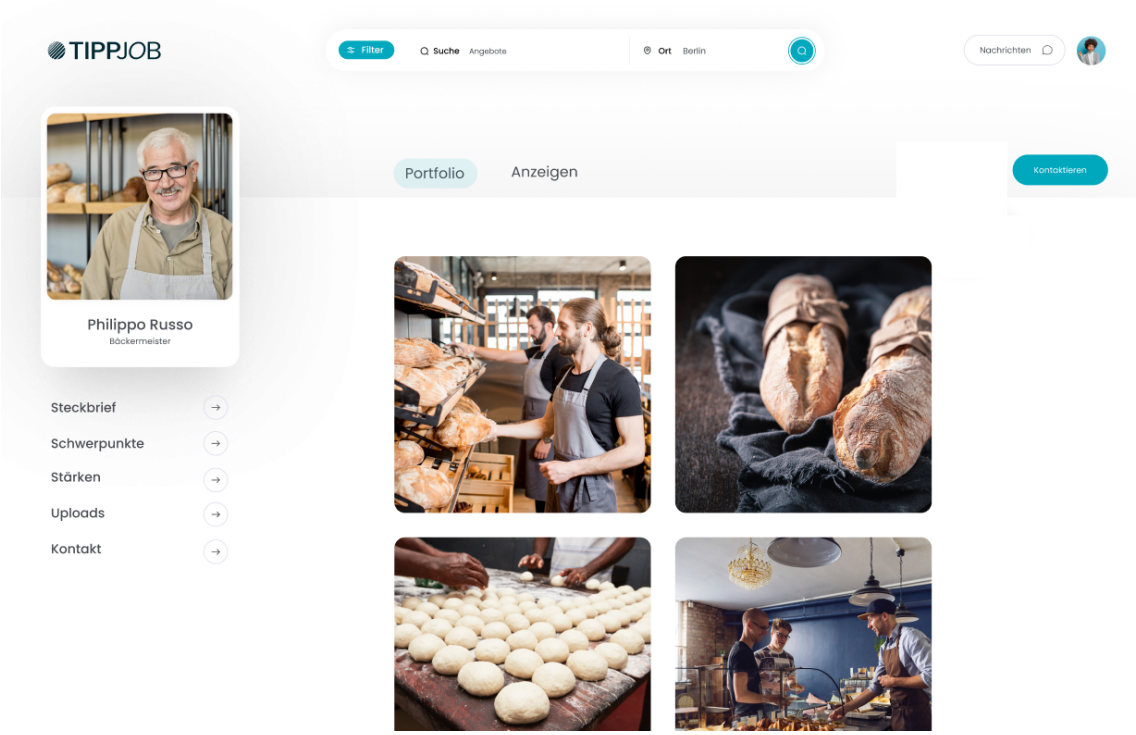

Figure 5.20: Realization of use case "Consult users profiles" 2

Realization of the use case "Message users"

These interfaces shown in the figures 5.20 to 5.23 present the visual UI that is displayed to the user in order to exchange messages with another user.

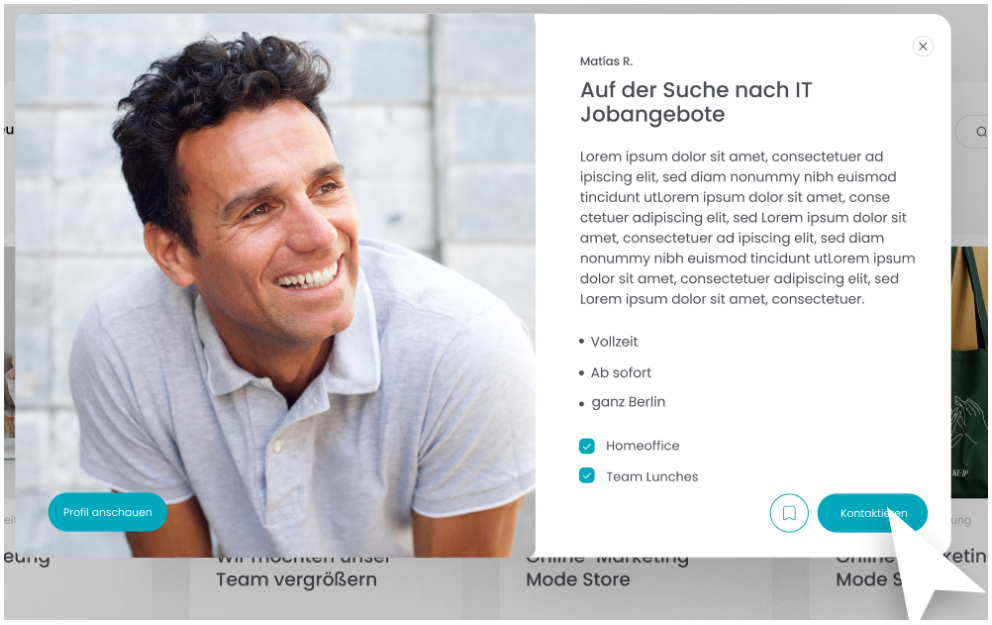

Figure 5.21: interface where the user contacts the job author

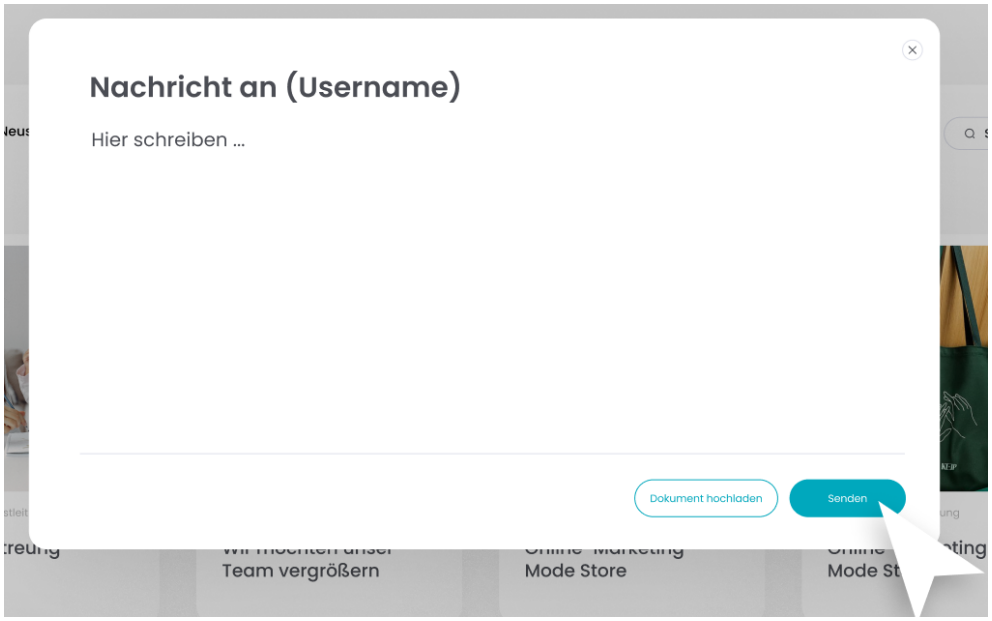

Figure 5.22: interface where the user inputs his message

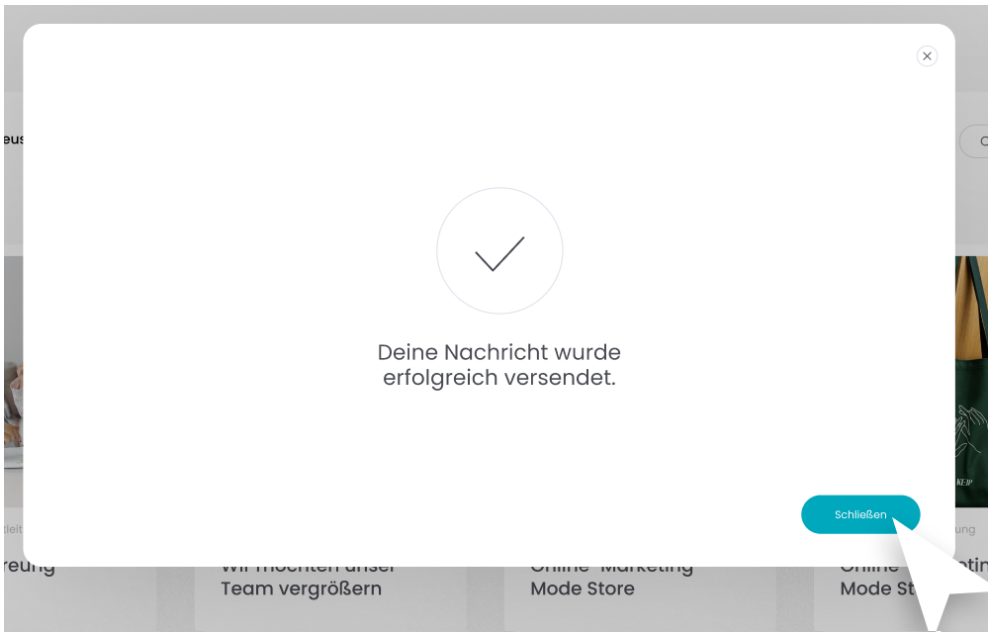

Figure 5.23: message sent

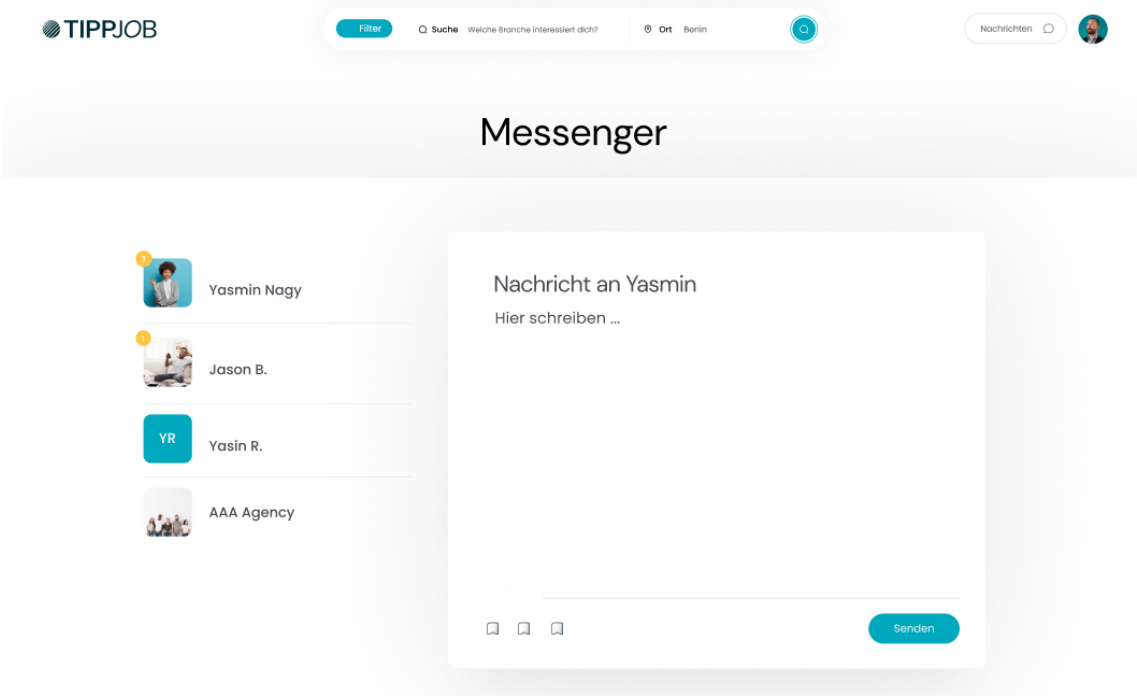

Figure 5.24: Interface of the messenger

5.5 Conclusion

This chapter covers the entire process involved in delivering the third and final release, including the refinement, design, and implementation of the last two sprints.

General conclusion

Our project is the focus of the end-of-study internship that was conducted at CodeCooperation GmbH, with the goal of obtaining a Bachelor's Degree in Computer Science.

Over the course of the four months we spent working on this project, we had the opportunity to learn new methodologies, technologies, and problem-solving approaches that we had not previously encountered. Our work on this project has increased our knowledge and proficiency in the field of web development. We are proud and satisfied with our achievements, having successfully met the objectives, reached milestones, and adhered to the deadlines set by the company.

To summarize, we were able to create a job posting platform called TippJob where users can post job offers of any kind and type, and other users can view them and contact the job owner for application. The platform was built using Next.js and Open-API as a front end, and Nest.js and Prisma ORM as a back-end, with MySQL as the database. The most important features of the platform include job posting system, profile building, messenger, and UX enhancements such as search, filter, and bookmarks.

Our platform has implemented Stripe payment processing to ensure a seamless and secure transaction experience for our users. Stripe was integrated via API and is a leading payment processing service known for its robust security measures, including encryption and fraud detection. By integrating Stripe, we have made it easier for our users to make post payments with just a few clicks while providing them with the confidence that their data is being protected.

As we get closer to the end of this report, we would like to mention that we still aim to add more features into the application and enhance some of the existing ones in order to make it ready for public use, as these releases are only the MVP (Minimal valuable product) of the platform.

Netography

[1] *Digital Ocean*. URL: https://www.digitalocean.com/ (visited on 03/28/2023).

[2] *Figma*. URL: https://www.figma.com// (visited on 03/26/2023).

[3] *Github*. URL: https://github.com/ (visited on 03/21/2023).

[4] *Github Desktop*. URL: https://desktop.github.com/ (visited on 03/21/2023).

[5] *Material UI*. URL: https://mui.com/ (visited on 03/09/2023).

[6] *MySQL*. URL: https://www.mysql.com/ (visited on 03/08/2023).

[7] *NestJS*. URL: https://nestjs.com/ (visited on 03/05/2023).

[8] *NextJS*. URL: https://nextjs.org/ (visited on 03/02/2023).

[9] *OpenAPO*. URL: https://www.openapis.org/ (visited on 03/19/2023).

[10] *Prisma ORM*. URL: https://www.prisma.io/ (visited on 03/07/2023).

[11] *Scrum*. URL: https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale/1443834-scrum-guide-de-la-methode-agile-star/ (visited on 02/26/2023).

[12] *Slack*. URL: https://slack.com/ (visited on 03/25/2023).

[13] *Sonarcloud*. URL: https://www.sonarsource.com/ (visited on 03/20/2023).

[14] *Swagger*. URL: https://swagger.io/ (visited on 03/23/2023).

[15] *Visual Studio Code*. URL: https://code.visualstudio.com/ (visited on 03/21/2023).