Turnitin Rapport de Similitude

Traité le: 24-mai-2023 9:51 PM CET
Numéro : 2101101883
Nombre de mots : 11870
Envoyé : 1

pfe Par Islembargaouietyoussefmrabet Bargaoui

Indice de Similitude	Similarité par source
12%	Internet Sources: 2% Publications : 1% Copies de l'étudiant : 10%

3% match (Travaux des étudiants depuis le 24-mai-2023)
Submitted to University of Carthage on 2023-05-24

1% match (Travaux des étudiants depuis le 24-mai-2023)
Submitted to University of Carthage on 2023-05-24

1% match (Travaux des étudiants depuis le 24-mai-2023)
Submitted to University of Carthage on 2023-05-24

1% match (Travaux des étudiants depuis le 13-mai-2016)
Submitted to University of Portsmouth on 2016-05-13

1% match (Travaux des étudiants depuis le 21-déc.-2015)
Submitted to Monash University on 2015-12-21

< 1% match (Travaux des étudiants depuis le 12-mai-2013)
Submitted to CSU, San Jose State University on 2013-05-12

< 1% match (Travaux des étudiants depuis le 21-oct.-2009)
Submitted to CSU, San Jose State University on 2009-10-21

< 1% match (Travaux des étudiants depuis le 07-janv.-2018)
Submitted to Universiti Teknologi MARA on 2018-01-07

< 1% match (Travaux des étudiants depuis le 01-juin-2017)
Submitted to Universiti Teknologi MARA on 2017-06-01

< 1% match (Internet depuis le 19-mai-2023)
https://tudr.thapar.edu:8080/jspui/bitstream/10266/3977/4/3977.pdf

< 1% match (Internet depuis le 24-mai-2013)
http://www.jahis.jp/wp/wp-content/uploads/giho2001-14.pdf

< 1% match (Travaux des étudiants depuis le 19-mai-2018)
Submitted to University of Melbourne on 2018-05-19

< 1% match (Travaux des étudiants depuis le 17-sept.-2019)
Submitted to University of Melbourne on 2019-09-17

< 1% match (Travaux des étudiants depuis le 18-déc.-2009)
Submitted to Oxford Brookes University on 2009-12-18

< 1% match (Travaux des étudiants depuis le 16-juin-2022)
Submitted to University of Monastir on 2022-06-16

< 1% match (Travaux des étudiants depuis le 04-oct.-2010)
Submitted to University of Greenwich on 2010-10-04

< 1% match (Travaux des étudiants depuis le 16-janv.-2017)
Submitted to HELP UNIVERSITY on 2017-01-16

< 1% match (Travaux des étudiants depuis le 21-mai-2018)
Submitted to HELP UNIVERSITY on 2018-05-21

< 1% match (Travaux des étudiants depuis le 16-mai-2023)
Submitted to The University of Wolverhampton on 2023-05-16

< 1% match (Travaux des étudiants depuis le 11-avr.-2023)
Submitted to BPP College of Professional Studies Limited on 2023-04-11

< 1% match (Travaux des étudiants depuis le 10-avr.-2023)
Submitted to University of Birmingham on 2023-04-10

< 1% match (Travaux des étudiants depuis le 28-mars-2023)
Submitted to QA Learning on 2023-03-28

< 1% match (Travaux des étudiants depuis le 07-avr.-2021)
Submitted to Universiti Tunku Abdul Rahman on 2021-04-07

< 1% match (Travaux des étudiants depuis le 28-janv.-2023)

Submitted to University of Southampton on 2023-01-28

< 1% match (Internet depuis le 09-oct.-2022)
https://www.tdx.cat/bitstream/handle/10803/123976/tcmc.pdf?isAllowed=y&sequence=5

< 1% match ()
Fast, Sven. "Development of a software application for statistical analysis of photovoltaic plant performance", Stellenbosch : Stellenbosch University, 2015

< 1% match ()
Alseid, Marwan N.K.. "Multimodal interactive e-learning: An empirical study. An experimental study that investigates the effect of multimodal metaphors on the usability of e-learning interfaces and the production of empirically derived guidelines for the use of these metaphors in the software engineering process.", School of Computing, Informatics and Media, Department of Computing., 2009

< 1% match (Internet depuis le 22-nov.-2022)
https://su-plus.strathmore.edu/bitstream/handle/11071/5996/Stock%20market%20price%20prediction%20using%20sentiment%20analysis%%20a%20case%20study%20of%20Nairobi%20stock%20exchange%20market.pdf?isAllowed=y&sequence=3

< 1% match (Travaux des étudiants depuis le 22-avr.-2023)
Submitted to University of Salford on 2023-04-22

< 1% match (Travaux des étudiants depuis le 23-avr.-2017)
Submitted to MCC Training Institute on 2017-04-23

< 1% match (Internet depuis le 25-oct.-2003)
http://www.formationsystems.com/cmt/dsp_press_releases.asp?display=detail&id=10

< 1% match (Travaux des étudiants depuis le 19-mai-2023)
Submitted to De Montfort University on 2023-05-19

< 1% match (Travaux des étudiants depuis le 18-mars-2023)
Submitted to National University of Singapore on 2023-03-18

< 1% match (Travaux des étudiants depuis le 31-mai-2018)
Submitted to Federation University on 2018-05-31

< 1% match (Travaux des étudiants depuis le 30-août-2006)
Submitted to University of Huddersfield on 2006-08-30

< 1% match (Internet depuis le 10-nov.-2010)
http://www.kaia.ie/scab.html

< 1% match (Internet depuis le 20-août-2022)
https://oa.upm.es/43855/1/GONZALO_EMMANUEL_EJARQUE_RINALDINI.pdf

< 1% match (Travaux des étudiants depuis le 08-mai-2019)
Submitted to Kingston University on 2019-05-08

< 1% match (Travaux des étudiants depuis le 17-avr.-2023)
Submitted to Roehampton University on 2023-04-17

< 1% match (Ernesto Exposito. "Advanced Transport Protocols", Wiley, 2012)
Ernesto Exposito. "Advanced Transport Protocols", Wiley, 2012

< 1% match (Travaux des étudiants depuis le 24-févr.-2022)
Submitted to University of Southern California on 2022-02-24

< 1% match (Internet depuis le 18-déc.-2022)
https://dk.um.si/Dokument.php?dn=&id=125729

< 1% match (Internet depuis le 28-févr.-2023)
https://git.woozle.org/neale/Marlin/commit/17dddda02455da693eae0552f8da8d7d5db4e0c3?style=unified&whitespace=ignore-all

< 1% match (C. Pons, R. Giandini, G. Baum. "Dependency relations between models in the Unified Process", Tenth International Workshop on Software Specification and Design. IWSSD-10 2000, 2000)
C. Pons, R. Giandini, G. Baum. "Dependency relations between models in the Unified Process", Tenth International Workshop on Software Specification and Design. IWSSD-10 2000, 2000

General intro duction 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 dedi- cated 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 pay- ment integration and job posting features, improving the platform's user-friendliness and value. • "Release 3: Enhanced User Experience", The third and final release will in- troduce new search filters and messaging functionalities, designed to enhance user interactions and facilitate communication among users. 1 Chapter 1 Context of the pro ject 1.1 Introduction This chapter provides an overview of the host organization, details the project devel- oped, outlines the technology choices made, and explains the methodology followed. 1.2 Presentation of the host organization We will present the host organization in this section mentioning a brief description and its areas of activity. 1.2.1 Host organization CodeCooperation is a digital innovation lab founded in 2020 with a passion for trans- forming digital opportunities into exceptional software products. Located in Berlin, with a satellite office in Tunisia, CodeCooperation is a leading digital engineering and consult- ing studio with a startup approach. The company is committed to providing innovative solutions through cutting-edge technology, fostering digital transformation and innovation worldwide. CodeCooperation's flat hierarchy promotes collaboration, innovation, and autonomy among employees. Each department operates independently while contributing to the overall company structure. Services offered by CodeCooperation include software devel- opment, UI/UX design, mobile app and website development, as well as quality assurance. Their iterative approach at startup speed helps clients bring products to market faster, at a lower cost, and with higher quality. The company also assists clients in building a startup mindset and developing a culture of innovation. CodeCooperation has an ex- tensive client portfolio, ranging from startups to governments, helping businesses digitize and automate their processes to achieve success. Additionally, CodeCooperation runs the CodeCooperation Academy and Entrepreneurial Engineering school, providing practical training programs for individuals interested in the IT field. The Academy caters to field switchers, university dropouts, and high school graduates. 2 1.3. Project presentation 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 Next, we will endeavor to present an overview of the project. Providing its general scope, the existing solutions that will highlight the problem, and a proposed solution for it. 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 plat- form 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. 1.4. Project management Methodology • Focusing on connecting candidates with employers, and missing the ability for in- dividuals 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 The decision has been made that Scrum is the most appropriate and well-structured methodology for this project. Scrum is a framework that promotes team collaboration in planning, developing, deliv- ering, 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 figure 1.2 . Figure 1.2: Scrum Structure[11] The figure 1.2 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. 1cm 1.5 Technical Choice In this section, we will outline the tools utilized throughout the project's implementa- tion. This encompasses the programming languages, frameworks, and development tools employed, as well as the software and websites that facilitated collaboration. 1.5.1 Development Technologies & Frameworks Next.JS Next.js [8] 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 Nest.js Nest.js [7] 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 Prisma Prisma [10] 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 MySQL MySQL [6] is a popular open-source relational database management system. It was used as the database for the project. Figure 1.6: MySQL Material UI Material UI [5] is a React UI framework that provides pre-built compo- nents 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 OpenAPI OpenAPI [9] 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 Sonar Sonar [13] 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 Swagger Swagger [15] is a toolset for designing, documenting, and testing APIs. It provides a user-friendly interface for defining API endpoints and parameters, and gener- ates interactive documentation for developers. It also includes testing tools for verifying API behavior. Figure 1.10: Swagger DigitalOcean DigitalOcean [1] 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 Stripe Stripe [14] is an online payment processing platform that helps businesses accept and manage payments securely over the internet. It offers developer-friendly tools and APIs for integrating payment functionality into websites and applications. It was used as a payment gateway. Figure 1.12: Stripe 1.5.2 Collaboration and Development Tools Visual Studio Code (VScode) Visual Studio Code (VScode) [16] 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 1.13: Visual Studio Code Github Desktop Github Desktop [4] 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 1.14: Github Desktop Github Github [3] 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 1.15: Github Slack Slack [12] 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 1.16: Slack Figma Figma [2] 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 1.17: Figma 1.6. General Architecture 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 1.18 offering a comprehensive overview of our system's architecture. Figure 1.18: Deployment Diagram 1.6. General Architecture 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 1.19: 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 1.20 represents our CI process. 1.7. Conclusion Figure 1.20: 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 Sp ecification 2.1 Introduction In this chapter, we begin our work by defining the project's requirements and identi- fying 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. 13 2.2. Requirements Specification • 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 signifi- cant 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 unautho- rized access, data breaches, and cyber attacks. This includes using encryption for sensitive data, implementing authentication and authorization mechanisms, and fol- lowing 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. 2.3. Global Use Cases Diagram • 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 func- tionalities; 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 The figure 2.21 shows the project's global use case diagram, it presents all the use cases and features that the platform will present to the user. 2.3. Global Use Cases Diagram Figure 2.21: Global Use Cases Diagram 2.4. Product Backlog 2.4 Product Backlog The table 2.1 shows our product backlog (all the user stories included in the project) : Table 2.1: Product Backlog As a user i can Register User Story Priority 1 Estimation High Planning Sprint 0 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 2 High Sprint 1 Release 1 through the onboarding proccess As a user i can consult my profile 2 Medium Sprint 1 As a user i can edit my profile 3 Medium Sprint 2 As a user i can edit my ac- (profile picture, gallery, informa- tion) 3 Medium Sprint 2 count settings (personal informa- tion, Visibility, email, password, newsletter) As a user i can delete my account 3 Medium Sprint 2 permanently As a user i can post a job offer 4 High Sprint 3 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 5 Medium Sprint 4 Release 2 offers As a user i can consult offers and 6 Medium Sprint 5 requests posts As a user i can search for job of- 7 High Sprint 6 fers As a user i can filter job offers 7 Medium Sprint 6 As a user i can favorite job offers 8 Medium Sprint 7 Release 3 As a user i can consult other users 8 Medium Sprint 7 profiles As a user i can exchange messages 8 High Sprint 7 with another user 2.5. Conclusion 2.5 Conclusion This chapter has allowed us to identify the key Users and their require- ments 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 regis- tration, 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, includ- ing use case diagrams, textual descriptions, class diagrams and sequence diagrams. 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. 19 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. a. 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. Table 3.1: Description of use case "Create account" 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 1- The actor types in their personal information. Main scenario 2- The actor clicks "Register/Jetzt registrieren". 3- The server verifies the information. 3- If the information is valid, the server Exception 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. if the information is not valid, the server alerts the user b. 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. Table 3.2: Description of use case "Login" 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. 1- The actor types in his credentials Main scenario 2- The actor clicks "Login/Anmelden" 3- The server verifies the credentials Exception 4- if the credentials are correct, the server then creates a session for that account. if the credentials are wrong, the server alerts the user. c. 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. Table 3.3: Description of use case "Reset password" Use Case Scenario Reset password Actors User Pre-conditions The login modal is open The actor has to have an account Post-conditions Password is changed 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". Main scenario 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. 1-if the passwords do not match, an error is displayed. Exception 2- if the email doesn't exist, the server alerts the user 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. a. Class Diagram of the User Story "Create account" The figure 3.4 represents the class diagram of the use case "Create ac- count". Figure 3.4: Class diagram of "Create account" b. 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 informa- tion, 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" c. 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" d. 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 cre- dentials 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" e. Class Diagram of the User Story "Reset password" The figure 3.8 presents the class diagram of the use case "Reset pass- word". Figure 3.8: Class diagram of "Reset Password" f. 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" 3.2.3 Realisation In this Realization we will focus on the interfaces of "Login", "Register" and "Reset password". a. Realization of the use Case "Create account" The figure 3.10, 3.11 and 3.12 show where a new user can register and confirm via email address to create his own account. Figure 3.10: interface of "Create account" 1 Figure 3.11: interface of "Create account" 2 Figure 3.12: account confirmation email b. Realization of the use case "Login" In the modal represented by the figure 3.13 a registered user can login using his credentials to access the web app's features. Figure 3.13: interface of "Login" c. Realization of the use case "Reset password" Here in the figures 3.14, 3.15 and 3.16 is where the registered user can use his email address to reset his account password in case he forgot it. Figure 3.14: interface of "Reset password" email input Figure 3.15: email of password reset Figure 3.16: interface of "reset password" new password input 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. a. Refinement of the use case "Create profile" The figure 3.17 shows the use case diagram of the user story "Create profile" which depends on the use case "Login". Figure 3.17: Use case diagram of "Create profile" The table 3.4 describes the use case "Create profile" textually in the form of tables. Table 3.4: Description of use case "Create Profile" Use Case Scenario Create profile Actors User Pre-conditions The user needs to not have any profile info set up The actor has to be logged in Post-conditions Profile info is created 1- the actor uploads his profile picture. Main scenario 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 b. Refinement of The use case "Consult Profile" The figure 3.18 shows the use case diagram of the user story "Consult profile" that depends on the use case "Login" of course. Figure 3.18: Use case diagram of "Consult Profile" The table 3.5 describes the use case "Consult profile" textually in the form of tables. Table 3.5: Description

of use case "Consult Profile" 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 1- the actor clicks on his profile picture in the navbar Main scenario 2- the profile picture menu is displayed 3- the actor clicks on "Profile/Profil" 4- the user's profile page is displayed 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. a. Class Diagram of the User Story "Create profile" The figure 3.19 presents the class diagram of the use case "Create pro- file". Figure 3.19: Class diagram of "Create profile" b. Sequence Diagram of the User Story "Create profile" Figure 3.20 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 3.20: Sequence diagram of "Create profile" c. Class Diagram of the User Story "Consult profile" The figure 3.21 represents the class diagram of the use case "Consult profile" Figure 3.21: Class diagram of "consult profile" d. Sequence Diagram of the User Story "Consult profile" Figure 3.22 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 3.22: Sequence diagram of "Consult profile" 3.3.3 Realisation This subsection is reserved for the Realization of the use cases developed in sprint 1, they are "Create profile" and "Consult profile". a. Realization of the use case "Create profile" The interfaces represented in the figures 3.23 to 3.31 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 upload • Activity info (Industry, occupation, job title) • Gallery upload • Focus (topics the user focuses on) • Bio input • PDF files upload • personal strengths (Profile tags) Figure 3.23: on-boarding welcome message Figure 3.24: Profile picture upload interface Figure 3.25: Activity input interface Figure 3.26: Gallery upload interface Figure 3.27: Focus input interface Figure 3.28: Bio input interface Figure 3.29: PDF files upload interface Figure 3.30: Personal strengths interface Figure 3.31: User welcoming message b. 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 3.32: Profile interface 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. a. Refinement of the use case "edit profile" The figure 3.33 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 3.33: Use case diagram of "edit profile" The table 3.6 describes the use case "Edit profile" textually in the form of tables. Table 3.6: Description of use case "Edit Profile" Use Case Scenario edit profile Actors User Pre-conditions The actor has to be logged in Post-conditions Profile information are updated 1-the user clicks "Edit profile/Profil bearbeiten" Main scenario 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 b. Refinement of the use case "edit account" The figure 3.34 shows the use case diagram of the user story "Edit account" which is dependent on the use cases "Create account" and "Login" Figure 3.34: Use case diagram of "edit account" The table 3.7 describes the use case "Edit account" textually in the form of tables. Table 3.7: Description of use case "Edit account" Use Case Scenario Edit account Actors User Pre-conditions The actor has to be logged in Post-conditions Account information settings are updated 1- the actor selects a setting tab Main scenario 2- the actor updates the information 3- the actor clicks "Save/Speichern" 4- the data is updated in the database c. Refinement of the use case "delete account" The figure 3.35 shows the use case diagram of the user story "Delete account" that is dependent on the use cases "Create account" and "Login" Figure 3.35: Use case diagram of "delete account" The table 3.8 describes the use case "Delete account" textually in the form of tables. Table 3.8: Description of use case "Delete account" 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 1- the actor clicks "next/weiter" in the delete account tab of the settings Main scenario 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 3.4.2 Conception This step establishes the interactions of the use cases and utilizes Class and Sequence Diagrams to formalize our approach visually a. Class Diagram of the User Story "edit profile" The figure 3.36 represents the class diagram of the use case "Edit profile". Figure 3.36: Class diagram of "edit profile" b. Sequence Diagram of the User Story "edit profile" Figure 3.37 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 3.37: Sequence diagram of "edit profile" c. Class Diagram of the User Story "edit account" The figure 3.38 presents the class diagram of the use case "Edit account". Figure 3.38: Class diagram of "edit account" d. Sequence Diagram of the User Story "edit account" Figure 3.39 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 3.39: Sequence diagram of "edit account" e. Class Diagram of the User Story "delete account" The figure 3.40 presents the class diagram of the use case "Delete ac- count". Figure 3.40: Class diagram of "delete account" f. Sequence Diagram of the User Story "delete account" Figure 3.41 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 pro- vided 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 3.41: Sequence diagram of "delete account" 3.4.3 Realization 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". a. Realization of the use case "Edit profile" In the profile page on the right, the user will find a button "Profil bear- beiten" which if clicked will display edit buttons across the entire page as shown in the figure 3.42 . Figure 3.42: edit Profile interface b. 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 fol- lows: • 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 c. Realization of the use case "Delete account" The interfaces shown in the figures 3.50 and 3.51 allows a user to com- pletely delete his profile, once done, this process cannot be reversed. Figure 3.50: delete account interface Figure 3.51: confirm account deletion interface 3.5. Conclusion 3.5 Conclusion This chapter covers the various steps involved in delivering the first release, which includes refining the requirements, designing, and imple- menting the first three sprints. Chapter 4 Release 2: Job p osts and Payment Integration 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. 62 a. 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. Table 4.1: Description of use case "Create Job Posts" 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 1- the user clicks on "neue Anzeige/New Post" in the Anzeigen Main scenario section of his profile. 2- the user inputs the data that will define the post's content 3- the user clicks "Anzeige aufgeben/Submit post" 4- the server creates a new post in the database b. Refinement of the User Story "Edit a job post" The use-case diagram for the user story "Edit a job post" 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. Table 4.2: Description of use case "Edit job posts" Use Case Scenario Edit a Job Post Actors User Pre-conditions at least one job post is created by the user The user has to be logged in Post-conditions The desired job post is updated 1- the user clicks on the edit button under the job post Main scenario 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 c. Refinement of the user Story "Delete a job post" The use-case diagram for the user story "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. Table 4.3: Description of use case "Delete job posts" Use Case Scenario Delete a Job Post Actors User Pre-conditions at least one job post is created by the user The user has to be logged in Post-conditions The desired job post is deleted from the database 1- the user clicks on the delete button under the post Main scenario in his profile 2- the server deletes the desired post from the database 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. a. 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" b. 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 pre- sented 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 then inputs the desired data that will define the post's con- tent and clicks "Anzeige aufgeben/Submit post". Figure 4.5: Sequence diagram of "Create a job post" c. 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" d. 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" e. 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" f. Sequence Diagram of the User Story "Delete a job post" The figure 4.9 presents how can the user delete one of 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" 4.2.3 Realisation In this section we present the interfaces realised for the use cases of the third sprint which are "Create a job post", "Edit a job post" and " Delete a job post" Realization of the use case "Create a job post" These interfaces shown in the figures 4.10 and 4.11 present the visual UI that is displayed to the user in order to create a new job post. Figure 4.10: Realization of use case "Create a job post" 1 Figure 4.11: Realization of use case "Create a job post" 2 Realization of the use case "Edit a job post" The figure 4.12 and 4.13 presents the interface where the user can edit his job posts. Figure 4.12: Realization of use case "Edit a job post" 1 Figure 4.13: Realization of use case "Edit a job post" 2 Realization of the use case "Delete a job post" The figure 4.14 and 4.15 present the interfaces of the use case "Delete a job post". Figure 4.14: Realization of use case "Delete a job post" 1 Figure 4.15: Realization of use case "Delete a job post" 2 4.3 Sprint 4 Here is the user story that outlines the tasks for our fourth sprint: • 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. The figure 4.16 shows the use case diagram of the user story "Pay for a job post" Figure 4.16: 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. Table 4.4: Description of use case "Pay for a Job Post" 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 1- the user submits a job post Main scenario 2- the user is presented with payment plans 3- the user selects a payment plan 4- the user is redirected to stripe page to complete his payment 5- the user is presented with a summery of the post details Exception 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 if the payment isn't successfull the server returns an error 4.3.2 Conception In this stage, we establish the process of how the identified interactions and use cases should take place. This step is crucial in laying the ground- work for successful development. To achieve this, we utilize Class and Sequence Diagrams to visually represent our logic in a standardized and formal manner. a. Class Diagram of the user Story "Pay for a job post" The figure 4.17 presents the class diagram of the use case "Pay for a job post". Figure 4.17: Class diagram of "Pay for a job" b. Sequence Diagram of the User Story "Pay for a job post" The figure 4.18 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 proceed 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. Figure 4.18: Sequence diagram of "Pay for a job post" 4.3.3 Realisation The figures 4.19, 4.20 and 4.21 represent the "Pay for a job post" inter- face and how the user can make his post visible to the public. Figure 4.19: payment plans interface Figure 4.20: payment realisation interface Figure 4.21: payment confirmation interface 4.4 Sprint 5 This user story shows 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. The figure 4.22 shows the use case diagram of the user story "Consult users job posts", this user story is accessible by anyone even without cre- ating account. Figure 4.22: 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. Table 4.5: Description of use case "Consult users job posts" 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 2- the server returns all visible job posts submitted by the users 1- the

user navigates to all posts page 4.4.2 Conception In this phase, we focus on defining the specific flow and sequence of interactions for the identified use cases. This step is essential for ensuring a solid foundation for development success. To accomplish this, we utilize Class and Sequence Diagrams, which provide a structured and standardized representation of our logic. These diagrams help us visually depict the relationships and actions between different components, ensuring a clear understanding of how the system should behave. a. Class Diagram of the User Story "Consult users job posts" The figure 4.23 represents the class diagram of the use case "Consult users job posts". Figure 4.23: Class diagram of "Consult users job posts" b. Sequence Diagram of the User Story "Consult users job posts" The figure 4.24 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 "news" page, a list of all job posts submitted by the users is displayed, he then can click on a job post to view its details. Figure 4.24: Sequence diagram of "Consult users job posts" 4.4.3 Realization The figures 4.25 to 4.30 present the interface where all visible posts submitted by the users are displayed. The interface is divided into 4 cate- gories: • posts that are of type "Looking for a job" or "Offering a service" (figure 4.26 and 4.29). • posts that are of type "Offering a job" (figure 4.25). • posts that are of type "Looking for a job" or "Offering a service" and are URGENT, this page is displayed in dark mode (figure 4.28). • posts that are of type "Offering a job" and are URGENT, this page is displayed in dark mode (figure 4.27 and 4.30). 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 4.5. Conclusion Figure 4.30: Details of an URGENT job post 4.5 Conclusion This chapter covers all the steps involved in delivering the second re- lease, including the refinement, design, and implementation of the 3 sprints included. Chapter 5 Release 3: Enhanced User Exp eri- ence 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. 87 a. 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. Table 5.1: Description of use case "Search for job posts" 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 2- the server returns the filtered posts. 1- the user enters a search text in the searchBar b. 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. Table 5.2: Description of use case "filter job posts" 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 2- the server returns the filtered posts. 1- the user selects filter options 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. a. 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" b. 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" c. 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" d. 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" 5.2.3 Realisation Realization of the use case "Search for job posts" The interfaces shown in the figures 5.7 and 5.8 present the UI that the user uses to search for job posts. Figure 5.7: Realization of use case "Search for job posts" 1 Figure 5.8: Realization of use case "Search for job posts" 2 Realization of the use case "filter job posts" The interface shown in the figure 5.9 represents the filter form that allows the user to filter the displayed posts depending on any data inputted in this form. Figure 5.9: Realization of use case "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. a. Refinement of the User Story "Favorite job posts" The use-case diagram for the user story "Favorite job posts" is shown in the figure 5.10 . Figure 5.10: 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. Table 5.3: Description of the use case "Favorite job posts" Use Case Scenario Favorite 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 1- The user access the details of a post Main scenario 2- the user clicks on the bookmark button 3- the server adds the post to the user's bookmarks b. Refinement of the User Story "Consult users profiles" The use-case diagram for the user story "Consult users profiles" is shown in the figure 5.11 . Figure 5.11: 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. Table 5.4: Description of the use case "Consult users profiles" 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 1- The user access the details of a post Main scenario 2- the user clicks on "View profile/Profil anschauen" 3- the server redirects the user to the designated profile c. Refinement of the User Story "Message users" The use-case diagram for the user story "Message users" is shown in the figure 5.12 . Figure 5.12: 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. Table 5.5: Description of the use case "Message users" 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 1- the user access the details of a post Main scenario 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 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. a. Class Diagram of the User Story "Favorite job posts" The figure 5.13 represents the class diagram of the use case "Favorite job posts". Figure 5.13: Class diagram of "Favorite job posts" b. Sequence Diagram of the User Story "Favorite job posts" The figure 5.14 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 5.14: Sequence diagram of "Favorite

job posts" c. Class Diagram of the User Story "Consult users profiles" The figure 5.15 represents the class diagram of the use case "Consult users profiles". Figure 5.15: Class diagram of "Consult users profiles" d. Sequence Diagram of the User Story "Consult users profiles" The figure 5.16 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 5.16: Sequence diagram of "Consult users profiles" e. Class Diagram of the User Story "Message users" The figure 5.17 represents the class diagram of the use case "Message users". Figure 5.17: Class diagram of "Message users" f. Sequence Diagram of the User Story "Message users" The figure 5.18 presents how the user can exchange messages with an- other 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 5.18: Sequence diagram of "Message users" 5.3.3 Realization Realization of the use case "Favorite job posts" These interfaces shown in the figures 5.19 and 5.20 present the visual UI that is displayed to the user in order to favorite a job post. Figure 5.19: Realization of use case "Favorite job posts" 1 Figure 5.20: Realization of use case "Favorite job posts" 2 Realization of the use case "Consult users profiles" These interfaces shown in the figures 5.21 and 5.22 present the visual UI that is displayed to the user in order to View another user's profile. Figure 5.21: Realization of use case "Consult users profiles" 1 Figure 5.22: Realization of use case "Consult users profiles" 2 Realization of the use case "Message users" These interfaces shown in the figures 5.23 to 5.26 present the visual UI that is displayed to the user in order to exchange messages with another user. Figure 5.23: interface where the user contacts the job author Figure 5.24: interface where the user inputs his message Figure 5.25: message sent Figure 5.26: Interface of the messenger 5.3.4 Global Classes Diagram We arrive now to the global class diagram that is represented in the figure 5.27, outlining the project's database and the relations between the tables. Figure 5.27: Global Classes Diagram 5.4. Conclusion 5.4 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 con- ducted 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 Tip- pJob 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 but are not limited to 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 (still to be specified by the client) 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. 108 Chapter 1 Chapter 1 Chapter 1 1.5. Technical Choice Chapter 1 1.5. Technical Choice Chapter 1 1.5. Technical Choice Chapter 1 1.5. Technical Choice Chapter 1 1.5. Technical Choice Chapter 1 Chapter 1 Chapter 1 Chapter 2 Chapter 2 Chapter 2 Chapter 2 Chapter 2 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.2. Sprint 0 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.3. Sprint 1 Chapter 3 3.4. Sprint 2 Chapter 3 3.4. Sprint 2 Chapter 3 3.4. Sprint 2