Akdeniz University

**Software Engineering Project**

Hallet: The Online Platform for Simple Task Assistance and Quick Earnings.

Software Requirements Specification & Analysis

Arda Atakan Uçan, Aslı Yıldırım, Furkan Şenoğlu, Sefer Öztürk

**Team Leader:** Sefer Öztürk

**Product Owner:** Arda Atakan Uçan, Aslı Yıldırım, Furkan Şenoğlu

**Instructor:** Assoc. Prof. Ümit Deniz ULUŞAR

17.06.2023

This report is submitted to the Department of Computer Engineering of Akdeniz University for the Software Engineering course CSE332.

|  |  |
| --- | --- |
| **Abbreviations** | |
| API | Application Programming Interface |
| GHz | Gigahertz |
| GPS | Global Positioning System |
| HTTP | Hyper-Text Transfer Protocol |
| HTTPS | Hyper-Text Transfer Protocol Secure |
| IP | Internet Protocol |
| JSON | JavaScript Object Notation |
| RAM | Random Access Memory |
| REST | Representational State Transfer |
| SMTP | Simple Mail Transfer Protocol |
| SQL | Structured Query Language |
| SSL | Secure Sockets Layer |
| TLS | Transport Layer Security |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Contents**

[**Chapter 1: Design 1**](#_1fob9te)

[**1**](#_3znysh7) **Introduction 1**

[**1.1**](#_2et92p0) **Purpose 1**

[**1.2**](#_tyjcwt) **Product Scope 1**

[**1.3**](#_3dy6vkm) **References 1**

[**2**](#_1t3h5sf) **Overall Description 2**

[**2.1**](#_4d34og8) **Product Perspective 2**

[**2.2**](#_2s8eyo1) **Product Functions 2**

[**2.3**](#_17dp8vu) **User Types and Characteristics 2**

[**2.4**](#_3rdcrjn) **Operating Environment 2**

[**2.5**](#_26in1rg) **Design and Implementation Constraints 3**

[**2.6**](#_lnxbz9) **User Documentation 3**

[**2.7**](#_35nkun2) **Assumptions and Dependencies 3**

[**3**](#_1ksv4uv) **External Interface Requirements 4**

[**3.1**](#_44sinio) **User Interfaces 4**

[**3.2**](#_2jxsxqh) **Hardware Interfaces 8**

[**3.3**](#_z337ya) **Software Interfaces 8**

[**3.4**](#_3j2qqm3) **Communications Interfaces 9**

[**4**](#_1y810tw) **System Features 9**

[**4.1**](#_4i7ojhp) **User 9**

[4.1.1](#_2xcytpi) Creating Job List 9

[4.1.1.1.](#_1ci93xb) Description and Priority 9

[4.1.1.2.](#_3whwml4) Stimulus/Response Sequences 9

[4.1.1.3.](#_2bn6wsx) Functional Requirements 9

[4.1.2](#_2xcytpi) Taking a Job 10

[4.1.2.1.](#_1ci93xb) Description and Priority 10

[4.1.2.2.](#_3whwml4) Stimulus/Response Sequences 10

[4.1.2.3.](#_2bn6wsx) Functional Requirements 11

[**5**](#_qsh70q) **Nonfunctional System Requirements 11**

[**5.1**](#_3as4poj) **Performance Requirements** 11

[**5.2**](#_1pxezwc) **Safety Requirements** 12

[**5.3**](#_49x2ik5) **Security Requirements** 12

[**5.4**](#_2p2csry) **Software Quality Attributes** 12

[**5.5**](#_147n2zr) **Business Rules** 12

**6****System Models 12**

**6**[**.1**](#_3as4poj) **Interaction Models 12**

6[.1.1](#_2xcytpi) Use-Case Models12

6[.1.1.1 Hallet: Create a new Job Posting](#_2xcytpi) 12

6[.1.1.2 Hallet: Apply for a job](#_2xcytpi) 13

6[.1.1.3 Hallet: Reply to job application](#_2xcytpi) 14

6[.1.1.4 Hallet: Edit a created job](#_2xcytpi) 14

6[.1.2](#_2xcytpi) Sequence Diagrams 15

**6**[**.2**](#_3as4poj) **Structural Models 16**

6[.2.1](#_2xcytpi) Object and Class Model 16

6[.2.1.1 Job Class](#_2xcytpi) 17

6[.2.1.2 User Class](#_2xcytpi) 17

**6**[**.3**](#_3as4poj) **Behavioral Models 18**

6[.3.1](#_2xcytpi) Data Driven (Activity Diagrams / Sequence Diagrams) 18

6[.3.2 Data Driven(Sequence Diagrams)](#_2xcytpi) 18

6[.3.2.1 Sequence diagram for creating a new job](#_2xcytpi) 18

6[.3.3 Event Driven Models (State Diagrams)](#_2xcytpi) 19

**6**[**.4**](#_3as4poj) **Test 19**

6[.4.1](#_2xcytpi) Functional Test Cases 19

6[.4.1.1 Creating an Account](#_2xcytpi) 19

6[.4.1.2 Not Allowing Login Without an Account](#_2xcytpi) 19

6[.4.1.3 Filtering Jobs Listed on Home Screen](#_2xcytpi) 20

6[.4.1.4 Rate the Job Doer of Job](#_2xcytpi) 20

6[.4.2](#_2xcytpi) API Test Cases 21

6[.4.2.1](#_2xcytpi) Confirm Retrieval of Correct Job Details for a Specific Job 21

6[.4.3 Non-Functional Test Cases](#_2xcytpi) 22

6[.4.3.1](#_2xcytpi) Browser Compatibility Testing 22

6[.4.3.2 Security Testing](#_2xcytpi) 22

[**Chapter 2: Implementation 23**](#_3l18frh)

**7****List of Change Requests and Assessments 23**

**8****Tasks Performed By Each Team Member 25**

**9****Details of the Final Product (Database, Screens etc.) 26**

9[.1](#_2xcytpi) Mobile Screens 26

9[.1.1 Signup and Login Screens](#_2xcytpi) 26

9[.1.2 Home, Filter Drawer, Job Details Dialog Screens](#_2xcytpi) 26

9[.1.3 Create Job Screen](#_2xcytpi) 26

9[.1.4 My Jobs Screen](#_2xcytpi) 27

9[.1.5](#_2xcytpi) Accept or Reject Request Screen in Requests Tab 27

9[.1.6](#_2xcytpi) Complete Job and Rate Job Doer Screens in Accepted Jobs Tabs 27

9[.1.7 Profile and Add/Remove Skill Screens](#_2xcytpi) 28

[9.1.8](#_2xcytpi) User Ranking and Other Users Profile Screens 28

9.[2 Endpoints Images](#_2xcytpi) 29

9.3 [Database Images](#_2xcytpi) 30

9.3.1 [Job Table](#_2xcytpi) 30

9.3.2 [User Table](#_2xcytpi) 30

9.3.3 [UserCategory Table](#_2xcytpi) 30

9.3.4JobRejectedRequest Table 30

9.3.5JobImages Table 30

9.3.6 [Category Table](#_2xcytpi) 30

9.3.7JobDoerRequests Table 30

9[.4 Web Screens](#_2xcytpi) 31

9[.4.1 Login and Signup Screens](#_2xcytpi) 31

9[.4.2 Home Page](#_2xcytpi) 31

9[.4.3 Job Detail Page](#_2xcytpi) 32

9[.4.4 Create new job page](#_2xcytpi) 33

9[.4.5](#_2xcytpi) User Profile Page 33

[**10**](#_1egqt2p) **Acknowledgment 34**

[**11**](#_3ygebqi) **References 34**

**12** [**Appendix A: Glossary**](#_sqyw64)**34**

**13 Team Member’s CV****35**

# **Chapter 1: Design**

# Introduction

## Purpose

The purpose of Software Requirements Specification (SRS) provides descriptions of requirements of the Hallet Application Web and Mobile Application. It will also explain how the system works, interactions with other parts of the system, interface, system constraints. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team.

## Product Scope

The “Hallet” is a user-friendly online platform that connects people who need help with simple tasks and those who are willing to complete them.

Our aim is to create a reliable environment in our application and to ensure that the work shared by users is done in a shorter time.

The application will be free to download from either a mobile phone application store or similar services. Users must provide user information and location information on their first login, because nearby jobs will be added to the users’ job list. However, users will be able to select the region they want in the filter section according to their situation. Plus, when we receive user information, we optionally request the user's interests. According to the user’s interests’ part, the work list to be shown will be more compatible with the user. However, if the user wants, he can choose a different job from the filter section. Depending on the user's satisfaction with the work done, they can get points from the business providers. A list will be made according to the average of these scores, and in this way, it will be ensured that users with high scores will get more jobs.

Depending on our goal, we will have a complaint and appraisal part so that the user can share their complaints and satisfaction ideas.

## References

- GetStream:

Source: <https://github.com/GetStream>

- Geolocator:

Source: <https://pub.dev/packages/geolocator>

- Flutter Documents

Source: <https://docs.flutter.dev/>

- Information about interfaces:

Source: <https://www.indeed.com/career-advice/career-development/what-are-interfaces>

- Information about REST API

Source: <https://www.ibm.com/topics/rest-apis>

- Recommended minimum Android SDK:

Source:<https://stackoverflow.com/questions/24441178/recommended-minimum-android-app-sdk>

- Tutorial about Use Case Diagrams

Source: <https://www.youtube.com/watch?v=zid-MVo7M-E&t=689s>

- Information about Software Quality Attributes

Source: <https://www.softwaretestingmaterial.com/quality-attributes-in-software-architecture/>

- Information about Business Rules

Source:<https://qracorp.com/guides_checklist/business-rules-and-requirements-are-not-the-same/>

# Overall Description

## Product Perspective

HALLET is an online platform designed to connect users who need help with simple tasks with those who are willing to complete them. The system consists of a web portal and a mobile application that have the same functionality. Both the web portal and mobile application will be used by users to post job offers or accept them. The web portal will also be used to manage the information about the system, such as user profiles and job offers.

The mobile application and web portal are available for users to list, create, and search for jobs. The main page is a page where jobs are listed and can be sorted with different filters. The job creation page allows users to create new job listings. Users can also access a private chat section to communicate with each other users' interests and aptitudes can be viewed and edited on the profile page. Users can also access the "My Jobs" page to keep track of their jobs and work history.

The system relies on a database for data storage and management. Both the web portal and mobile application will communicate with the database over the Internet. Payment transactions will not be handled through the system. Instead, a reporting system will be in place with user reviews to flag non-payment by job posters. In case of non-payment, job doers will be able to report the job poster to the system.

There are no specific resource allocation limitations for the mobile application or web portal. However, the system will be designed to ensure efficient performance and responsiveness on devices with different capabilities and internet connectivity.

## Product Functions

The HALLET application must perform the following major functions:

Allow users to create and manage job postings.

Allow users to search and apply for job postings based on different search criteria.

Enable communication between job posters and job doers through a chat function.

Provide a user profile page where users can view and edit their interests and skills.

Allow users to view their job history and completed jobs on their "Jobs" page.

Enable administrators to manage job postings, user profiles, and payments through the web portal.

Provide a reporting system for users to report issues, such as non-payment or inappropriate behavior, with other users or job posters.

All these functions will be facilitated through a data-centric system that utilizes a database to manage and store information.

## User Types and Characteristics

There are two types of users who will interact with the HALLET application: job-doers/job-posters and administrators.

The job-doer/job-poster user type is the primary user of the system, and they can both post job offers and accept job offers. These users may have varying levels of technical expertise, educational backgrounds, and work experience. They may also differ in terms of their location and language proficiency.

The administrator user type is responsible for managing the system, including adding, modifying, and deleting job offers, user profiles, and payments. Administrators will require higher levels of technical expertise and must be familiar with the HALLET system's operations to effectively carry out their duties. They will also need to have appropriate privileges and access levels to perform their tasks.

It's important to note that while job-doers/job-posters and administrators have different roles and responsibilities within the system, they may also have overlapping needs and requirements.

Therefore, the system must be designed to be user-friendly and easy to navigate for both types of users, with appropriate security measures in place to protect the privacy and security of user data.

## Operating Environment

The operating environment for the software will include both a mobile application and a website. The mobile application will be compatible with Android devices running version 7.0 or later and IOS version 12 or later. The website will be accessible through popular web browsers such as Google Chrome and Apple Safari.

For the mobile application, the software will require a minimum screen resolution of 720x1280 pixels and an internet connection to access HALLET platform and its services. Additionally, the mobile device should have a minimum of 2 GB of RAM and 500 MB of free storage space for the application to operate optimally.

For the website, the software will be designed to be compatible with the latest versions of popular web browsers, including Google Chrome, Apple Safari, Mozilla Firefox, and Microsoft Edge. It will not require any additional software components or applications to function. However, users should have a stable internet connection to access the HALLET platform and its services.

Overall, the software will be designed to operate seamlessly within the specified hardware and software environment. It will be thoroughly tested to ensure that it operates correctly on different platforms and coexists peacefully with other software components or applications.

## Design and Implementation Constraints

The HALLET app will be designed to operate on mobile devices running Android 7.0 or higher and IOS 12 or higher and will be optimized to run smoothly on devices with limited processing power and storage capacity. The app must also be designed to operate within the constraints of popular web browsers, such as Google Chrome and Safari.

Additionally, the application's functionality will be limited by the hardware capabilities of the mobile devices that it will run on. The application may not function optimally on older or low-end mobile devices with limited memory or processing power.

The mobile app will be developed using Flutter framework for both the mobile and web versions to ensure cross-platform compatibility. The developers must be proficient in Flutter development and have experience working with Flutter plugins and packages to integrate third-party services into the app.

The backend application will be designed and implemented in accordance with the .NET platform. Therefore, the software and hardware requirements of the application will be determined to meet the minimum requirements recommended for .NET technologies.

Clean Architecture will be utilized to create a layered structure in the application. This structure will consist of different components such as presentation layer, application layer, domain layer, and data layer.

The app will be designed to work with a variety of communications protocols and data formats to ensure compatibility with different systems. It must be designed to handle multiple tasks or operations in parallel, including multiple users accessing the app simultaneously.

Application must be designed to be secure and protect user data. This could include requirements for specific security protocols, such as SSL/TLS encryption for data in transit, and measures to prevent unauthorized access to user data.

The backend application will be designed and implemented in accordance with the .NET platform. Therefore, the software and hardware requirements of the application will be determined to meet the minimum requirements recommended for .NET technologies. In addition, .NET technologies and tools will be used during application development to ensure that the application is highly performant, scalable, and secure.

Clean Architecture will be utilized to create a layered structure in the application. This structure will consist of different components such as presentation layer, application layer, domain layer, and data layer.

## User Documentation

The Hallet application, although it currently does not have a user manual, will have user documentation created as part of the development process. Additionally, when users first enter the application, an Onboarding screen will be available to help users better understand and use the application's features, providing them with a better understanding of how to use the application.

Furthermore, the Hallet application is designed with a user-friendly interface and familiar design patterns, making it easy for users to navigate without the need for a complex user manual.

## Assumptions and Dependencies

Assumptions:

* Users have access to a stable internet connection to use the application.
* Users have a basic understanding of how to use mobile applications.
* Users have valid email addresses to register for the application.
* Users have Android 7.0 or higher installed on their mobile devices.
* The application will be deployed on the Google Play Store.
* The application will be hosted on a cloud server for scalability.
* Users will accurately provide job descriptions and requirements when posting jobs.
* Users will complete jobs in a timely and professional manner.

Dependencies:

* The application will depend on third-party APIs such as Google Maps and Google Places for location-based services.
* The application's development and deployment will depend on the availability of development tools and libraries such as the Android SDK, Flutter framework,
* The application's performance will depend on the mobile device's hardware specifications, such as RAM and processing power.

# External Interface Requirements

## User Interfaces

* When the user first opens the application, they see the login screen. If they are not a registered user, they can go to the registration screen by clicking on the "Sign up" button.

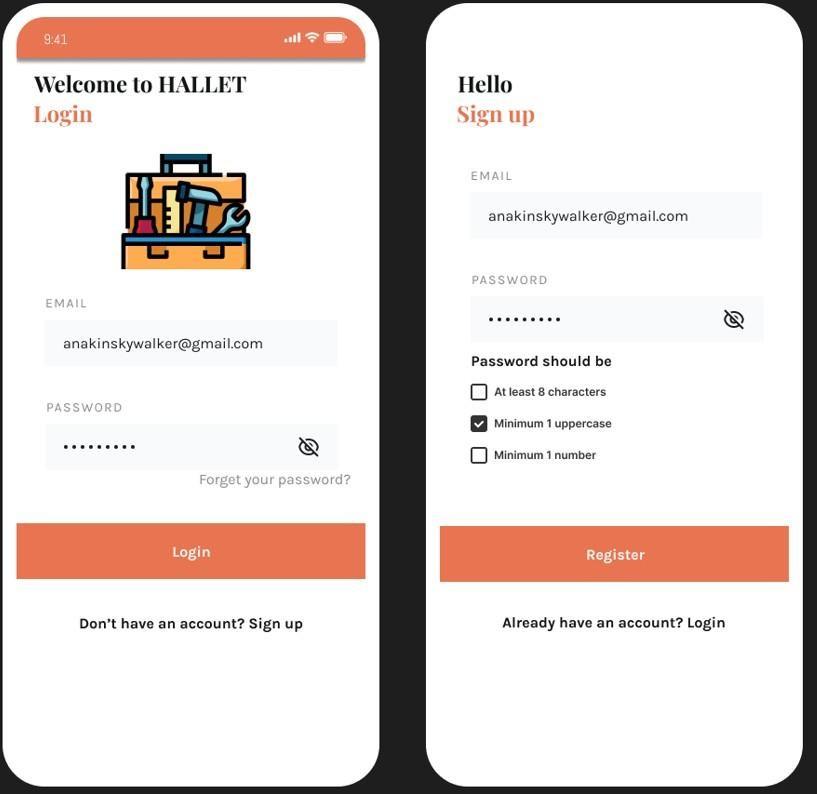


Figure 3.1.2: System Screens

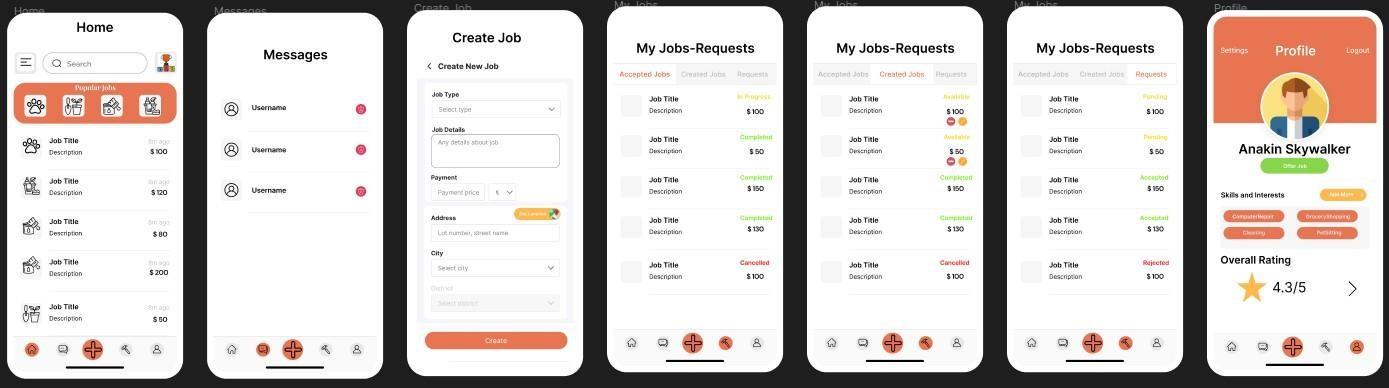
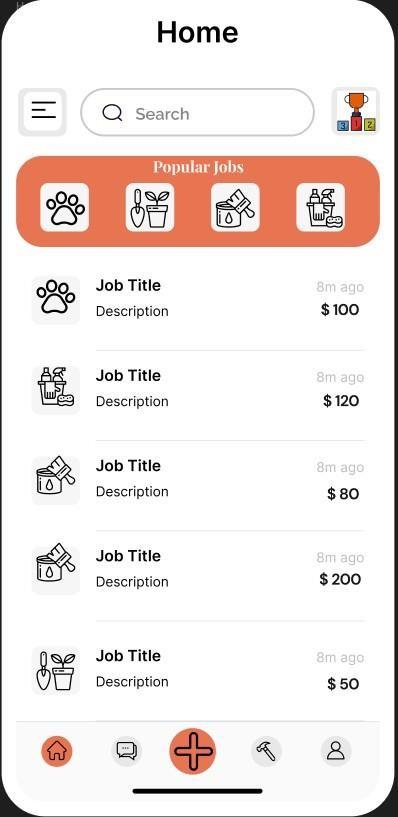
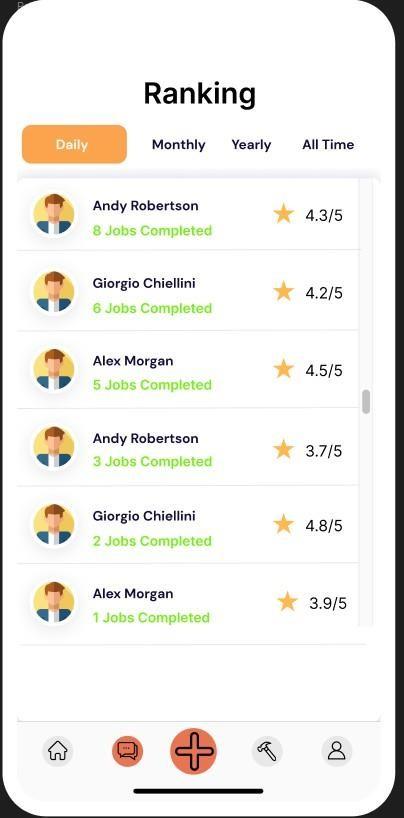
* During registration, the user must choose a password that is at least 8 characters long and includes one uppercase letter and one number.
* After entering their email address and password correctly, they can click the register button to complete the registration process.
* Once the registration process is complete, the user is automatically redirected to the login screen, where they can log in with their registered user information.
* When the user logs in, the Home screen opens, where job offers and popular job areas in the system are listed.
* The Search and Filter sections at the top of the screen make it easy to find a desired job. In the Filter section, selections can be made based on job area or location.
* At the bottom of the screen, the bottom navbar allows easy navigation between screens, and the button indicating the screen being viewed is displayed in a different colour.
* In addition to the Home screen, the Messages, Create Jobs, My Jobs, and Profile screens are available. These screens are shown in detail in Figure 3.1.2.
* You can access the Ranking screen by clicking on the ranking button located to the right of the Search Bar on the Home Screen. On this screen, registered users are listed and ranked according to the number of jobs they have completed weekly, monthly, yearly, or in all-time. By clicking on their usernames, you can easily access their profiles, and if they are ranked higher on the list, you can offer them a job.

Figure 3.1.3: Home Screen and Ranking Screens

* When a job offer on the Home screen is clicked, the Job Offer Detail screen opens, where information such as the job title, description, location, and payment are displayed. The handshake button at the bottom can be used to accept the job offer.

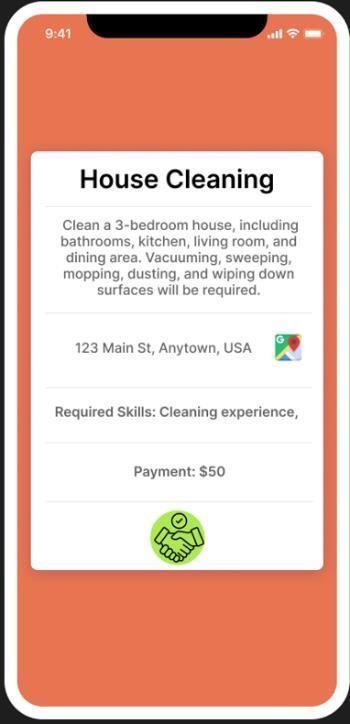


Figure 3.1.4: Job Offer Details

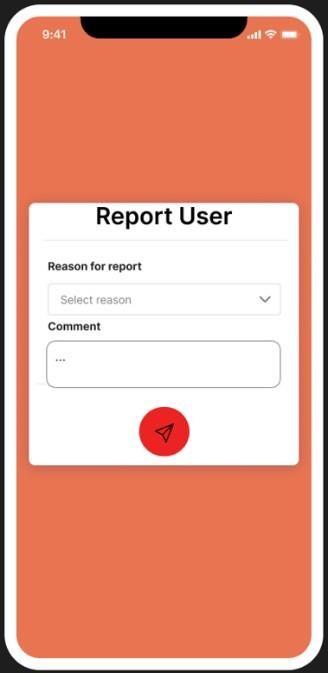
* The Messages section displays the JobDoers/JobPosters you are communicating with. By clicking on one of these conversations, you can open the Chat section, which is shown in Figure 3.1.5.

Figure 3.1.5: Chat Screen

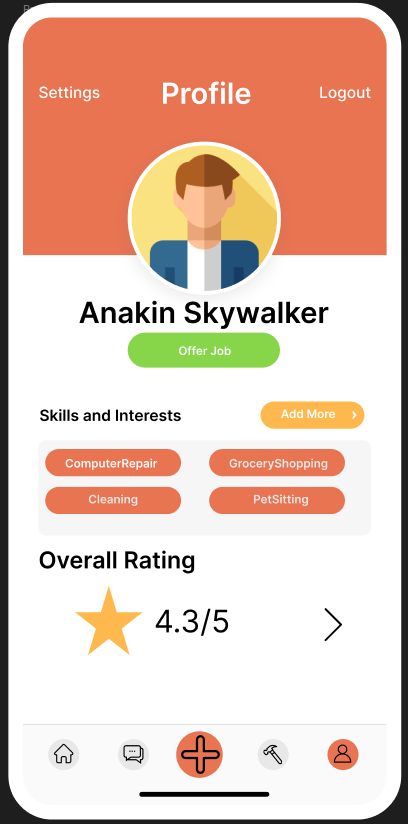
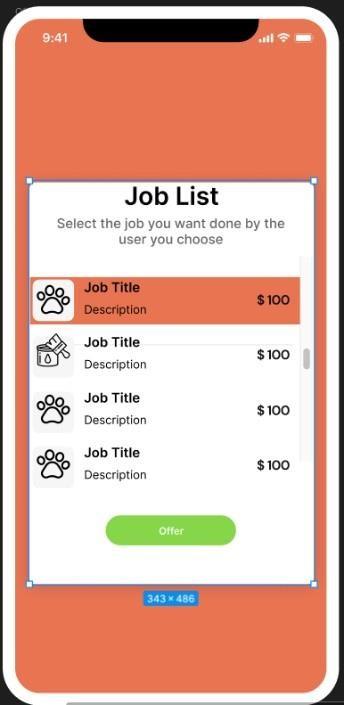
* The My Jobs screen shows the status of the jobs you have accepted or created. Clicking on one of these jobs opens the Job Details screen, which is shown in Figure 3.1.6.
* The chat button next to the JobDoer/JobPoster's profile information can be used to communicate with them.

Figure 3.1.6: Job Details Screen

* If you experience any dispute with a JobDoer/JobPoster during the job process that violates the system rules (such as an employer not paying the agreed-upon amount after the job is completed or the worker not fulfilling the requirements stated in the job description), you can
* Report the user by clicking the report button in the top right corner of the Job Details screen. This allows necessary actions to be taken, such as removing the user from the system.

Figure 3.1.7: Report Screen

* In the profile section, you can see the user's name, skills, interests, and overall score, which is obtained based on evaluations from other users on the system. On this page, you can offer a job to the user by using the "Offer Job" button if you have previously created a job for them to do.

Figure 3.1.8: Profile and Offer Job Screens

* When you click the "Offer Job" button, a list of job postings previously created by you will appear as shown in Figure 3.1.8. From there, you can select a job and send a job request to the user by clicking the "Offer" button.

## Hardware Interfaces

The hardware interfaces of the Hallet project interact with mobile devices (Android and iOS) and web browsers (Google Chrome, Safari, etc.). The application determines the user's location using location services and sends this data to the server. This process is carried out using the HTTP/HTTPS protocols. The application sends the data entered by the users (e.g., account information) to the server using secure protocols such as SSL/TLS. The application will not use hardware such as cameras.

Screen Resolution: The minimum screen resolution for mobile devices to display the user interface of the application should be 720p (1280 x 720 pixels). For the website, a minimum resolution of 1080p (1920 x 1080 pixels) is recommended.

Mobile device processors and memory: The performance of the application will depend on the speed of the processor and memory capacity of the user's device. It is recommended that devices have at least 2 GB of RAM and a quad-core processor with a speed of 1.5 GHz. However, devices with higher specifications are recommended for better performance and user experience.

Internet connection: An internet connection is required for the full functionality of the application. The internet connection is used to interact with the application's servers and data sources.

Data storage: The Hallet application may require data to be stored locally on the user's device. This data can affect the functionality and performance of the application.

## Software Interfaces

The Frontend interface will be developed on the Flutter platform. This interface will enable users to interact with the application and perform visual interface functions. It will interact with the other interfaces of the application to ensure that data is displayed correctly. For example, it will interact with the Google Maps API interface to obtain the user's location and display it on the interface. Similarly, it will interact with the StreamAPI interface to allow users to chat with each other and support the communication features of the application.

The Backend and Database interface will communicate with the Microsoft SQL Server database. This interface will provide access to the data in the database and retrieve the necessary information for the application. It will be developed on the .NET platform.

The Google Maps API interface will provide location determination feature using the Google Maps API. This interface will obtain the user's location by sending requests to the API and display location information on the application interface.

The StreamAPI interface will provide an in-app chat feature using the StreamAPI. This interface will allow users to chat with each other and support the communication features of the application.

## Communications Interfaces

For Mail Communication, SMTP will be used. The format of email messages is usually in HTML and/or plain text format and is used for account authentication, notifications, and other communication functions. SSL/TLS protocols will be used for the security of email messages.

For communication protocols on the network server, HTTP protocol is used to interact with the application's servers. HTTP protocol is commonly used for downloading web pages and communicating with servers. The application must be compatible with the HTTP protocol.

REST API will be used for communication within the application. REST is a commonly used communication and data synchronization protocol for web-based applications and supports data transmission in JSON or XML format. Microsoft SQL Server will be used as the database. REST API will enable the application to retrieve and update data from the database.

Google Maps API will be used to provide the location determination feature of the application. The API will communicate via HTTP protocol and use GPS feature to obtain the user's location. Data will be retrieved in HTTP format and the user's location data will be saved to the interface as well as the database.

# System Features

### User

### Creating Job List

#### Description and Priority

This feature determines the job list that the user will see. This list has high priority because it will be selected according to the location options selected by the user or the characteristics of the user. This filtering has risks because sometimes the job list can be shown differently to what the user wants.

#### Stimulus/Response Sequences

The user is in the “Profile” page.

The user can add skill and interests with the “Add More” button (Optimal) User navigates to the “Home” page.

If the user fills the skill and interest part, the system will give related jobs for the user. The user can choose popular jobs in the filter part or can choose related jobs. If the user did not fill the skill and interest part, the user clicks the “Filter” button.

The system opens a menu showing the different filtering options available to the user.

The system refreshes the search results according to the selected filtering option and displays the filtered results.

#### Functional Requirements

ID: FR1

TITLE: Entering Skill and Interest Input

DESC: A user should be able to write their interests with “Add More” button with that system can recommend related jobs to the user.

RAT: For the system to recommend relevant job postings to the user based on their skills and interests.

DEP: None

ID: FR2

TITLE: Filter Menu

DESC: The system should show filter menu with filtering options.

RAT: For the user to easily select and apply the desired filtering option.

DEP: None

ID: FR3

TITLE: Filtering Jobs

DESC: The system should allow filter the jobs postings like location, popular jobs, related jobs.

RAT: For the system to recommend relevant job postings to the user based on their skills and interests.

DEP: FR1, FR2

ID: FR4

TITLE: Display Jobs

DESC: The system should display job list according to filtered results and filtering options. RAT: For the user easily see and apply relevant jobs

DEP: FR3

### 

### Taking a Job

#### Description and Priority

This feature allows the user to apply for a job and at that moment the job owner can accept or not accept the user with communication of job owner and user. It is a high priority because this feature contains main functions of the platform.

#### Stimulus/Response Sequence

The user is in the “Home” page. The user clicks a job.

The system opens “Job Details” page.

If a user wants to ask questions about the job, the user can ask the job owner via chat. The user applies to the job.

The system adds a job application to the request list in the Jobs page’s request part. The job owner receives the job application. Job owners can accept or reject.

The job owner accepts or rejects the job application then the system adds to the accepted jobs list that user’s application rejected or accepted request.

#### Functional Requirements

### ID: FR5

### TITLE: Chat System

### DESC: A user should be able to ask questions to the job owner through a chat system. The chat system can have chat history.

### RAT: For a user to ask questions to the job-owner.

### DEP: FR4

### 

### ID: FR6

### TITLE: Applying a job

### DESC: Users should be able to apply for a job but then if in the applying process, the user wants to give up the job, the process can be canceled.

### RAT: For a user to applying a job

### DEP: FR4

### 

### ID: FR7

### TITLE: Adding Job Applications to Request List

### DESC: Given that job application, the job owner can reject or accept the job application.

### RAT: For a job owner to look at the user applying for the job.

### DEP: FR4, FR6

### 

### ID: FR8

### TITLE: Adding Reply to Accepted Jobs

### DESC: Given that the Job owner accepts the job application then the user can start a job completing process.

### RAT: For a user begin a job or not

### DEP: FR4, FR6, FR7

### 

# Nonfunctional System Requirements

## Performance Requirements

ID: NF1

TAG: Response Time < ID: FR3 TITLE: Filtering Jobs >

GIST: The fastness of the filtering jobs.

SCALE: The response time of filtering jobs.

METER: Measurements obtained from 1000 times filtering jobs during testing.

MUST: No more than 2 seconds 100% of the time.

WISH: No more than 1 second 100% of the time.

ID: NF2

TAG: Response Time < ID: FR4 TITLE: Display Jobs >

GIST: The fastness of the displaying jobs.

SCALE: The response time of displaying jobs.

METER: Measurements obtained from 1000 times displaying jobs during testing.

MUST: No more than 1 second 100% of the time.

WISH: No more than 0.5 seconds 100% of the time.

ID: NF3

TAG: Response Time < ID: FR7 TITLE: Applying a job>

GIST: The fastness of applying for a job.

SCALE: The response time of applying for a job.

METER: Measurements obtained from 1000 times applying for a job during testing.

MUST: No more than 5 seconds 100% of the time.

WISH: No more than 2 seconds 100% of the time.

ID: NF4

TAG: Response Time < ID: FR8 TITLE: Sending job applications to the job owner >

GIST: The fastness of the sending job applications to the job owner.

SCALE: The response time of sending job applications to the job owner.

METER: Measurements obtained from 1000 times job applications to the job owner during testing.

MUST: No more than 3 seconds 100% of the time.

WISH: No more than 1 second 100% of the time.

ID: NF5

TAG: Response Time < ID: FR8 TITLE: Notifying the user >

GIST: The fastness of notifying the user.

SCALE: The response time of notifying the user.

METER: Measurements obtained from 1000 times notifying the user during testing.

MUST: No more than 3 seconds 100% of the time.

WISH: No more than 1 second 100% of the time.

## Safety Requirements

There may be data losses that occur with unexpecting reasons so that to prevent this issue, we should replicate the database and increase the safety level. Since this action will block some server cores, it may lead to slowness on server-side. Therefore, we need to execute this project in hours where there are the least active users.

## Security Requirements

The application takes some personal information from the user like password, location. Since some of them are very important information, we need to encrypt them on server-side with the most up-to-date encryption model, which is SHA-512. We need to use salt SHA-512 to increase the decrypt resistance.

## Software Quality Attributes

Reliability is the ability of a system to continue to keep operating over time. A software needs to be reliable, and a software will be reliable if the system is available 99.999% of a year, the system does not fail more than one in half a year and the system’s mean time to failure is not more than 5 minutes.

Testability shows how well the system or component facilitates to perform tests to determine whether the predefined test criteria have been met. If a software wants to be testability, then it must be designed with testability in mind, written modular and reusable code, used automated testing tools, provided test data and test cases, implemented error handling and logging.

Usability is described as how the user is utilizing the system effectively and the ease of which users can learn to operate or control the system. Software applications should be user-friendly. Software can be made more usable with understanding user needs, designing for simplicity and clarity, providing feedback, using familiar and consistent interfaces, testing with real users.

## Business Rules

Business rules are different from business requirements. A business rule is a constraint of the business itself that may guide system development.

Business rules of this project is: All users need a valid email address, User’s password must contain at least 8 characters, User must not be blocked from the system to enter the system.

# System Models

## 

## Interaction Models

### Use-Case Models

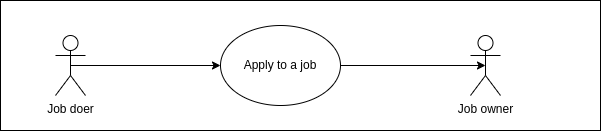
### 6.1.1.1 Hallet: Create a new Job Posting

|  |  |
| --- | --- |
|  | **Hallet: Create a new Job Posting** |
| **Actors** | Job owners |
| **Description** | Any customer (job owner) can create a job posting to hire another customer. This posting will be placed on the home page of the other customers. |
| **Data** | The details of the job. Job title, description, price, job category, etc. |
| **Stimulus** | User command issued by the customer. |
| **Response** | Confirmation that job posting has been created. |
| **Comments** | The customer can receive an invalid input alert when trying to create a job. |

### C:\Users\girnet\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\5E3A6941.tmp

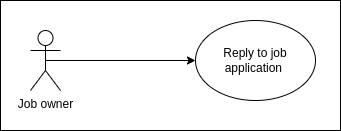
### 6.1.1.2 Hallet: Apply for a job

|  |  |
| --- | --- |
|  | **Hallet: Apply for a job** |
| **Actors** | Job workers and job owners |
| **Description** | Any customer can apply to a job posting to work on it. This application will send a request to the job owner to accept or reject this request. |
| **Data** | Id of the customer that applies for the job and id of the related job. Type of request. |
| **Stimulus** | User command issued by the customer. |
| **Response** | Confirmation that applying for a job has been created. |
| **Comments** |  |



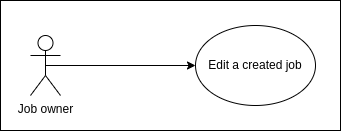
### 6.1.1.3 Hallet: Reply to job application

|  |  |
| --- | --- |
|  | **Hallet: Reply to job application** |
| **Actors** | Job owners |
| **Description** | Job owner users can reply to job application requests to hire the user that is sent that request or reject that. |
| **Data** | Id of the job, id of the user that applied for the job and the response type data. |
| **Stimulus** | Requesting a job application from another user. |
| **Response** | Accepting or rejecting the job application request. |
| **Comments** |  |

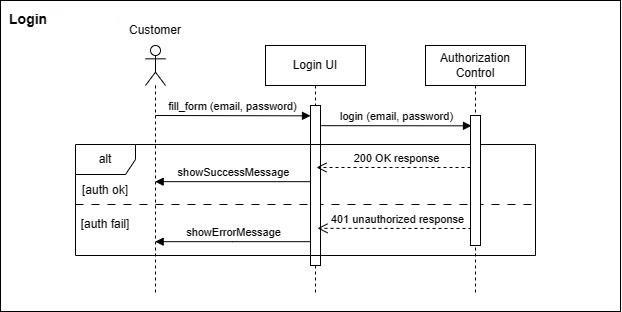


6.1.1.4 Hallet: Edit a created job

|  |  |
| --- | --- |
|  | **Hallet:** Edit a created job |
| **Actors** | Job owners |
| **Description** | A job owner can update the information of their created jobs. To edit a job, the state of that job must be still available for the apply. |
| **Data** | Id of the related job and edited attributes of that job. |
| **Stimulus** | User command issued by the customer. |
| **Response** | Job is updated successfully or an error message. |
| **Comments** | The customer can receive an invalid input alert when trying to edit the attributes of a job. |

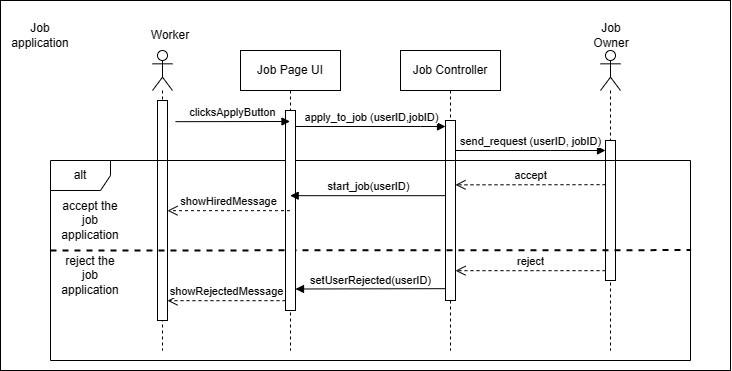


### Sequence Diagrams



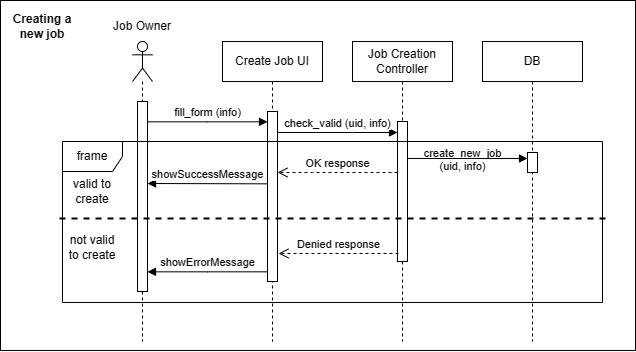
You can read this diagram as follows:

* + - 1. The customer triggers the fill\_form method in an instance of the Login object class, supplying the customer’s email and password information.
      2. That instance sends these email and password to authorization control by login method.
      3. Authorization Control checks that is email registered in the system and is password correct if authorization is successful then it returns 200 OK response and calls showSuccessMessage. However, authorization is failed then, it returns 401 unauthorized response and calls showErrorMessage.



You can read this diagram as follows:

1. Worker triggers clickApplyButton method and in an instance of JobPage object class.
2. That instance calls the apply\_to\_job method with the user and job identifiers in the Job Controller.
3. JobController calls send\_request method with those identifiers to send job request to job owner.
4. The box denoted by “alt” in the top-left corner is a choice box indicating that one of the contained instructions will be executed.
5. If the job owner accept that request JobController instance calls start\_job method with user identifier and JobPage instance calls showHiredMessage method. If the job owner does not accept that offer JobController instance calls setUserRejected method with user identifier and JobPage instance calls showRejectedMessage method.

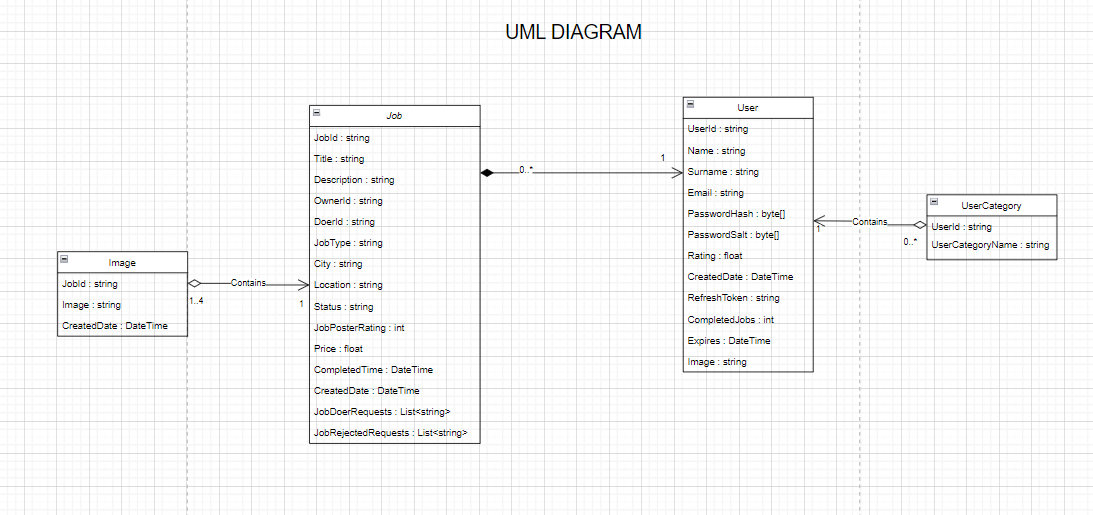


You can read this diagram as follows:

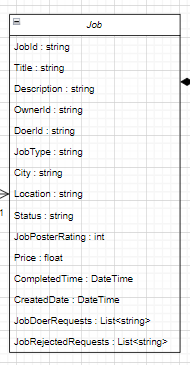
1. Job owner triggers fill\_form method with information in an instance of CreateJob.
2. That instance calls check\_valid method with user identifier and information takes in fill\_form method in an instance of JobCreationController.
3. If this instance returns “OK response” then, this instance calls the database to create a new job with that user identifier and information and CreateJob instance calls showSuccessMessage method. If this instance returns “Denied response” then, CreateJob instance calls showErrorMessage method.

## Structural Models

### Object and Class Model

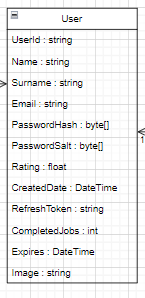


#### Job Class



Job class allows keeping the properties of the jobs created by the user. Moreover, it keeps information such as who got the job, and state.

#### User Class

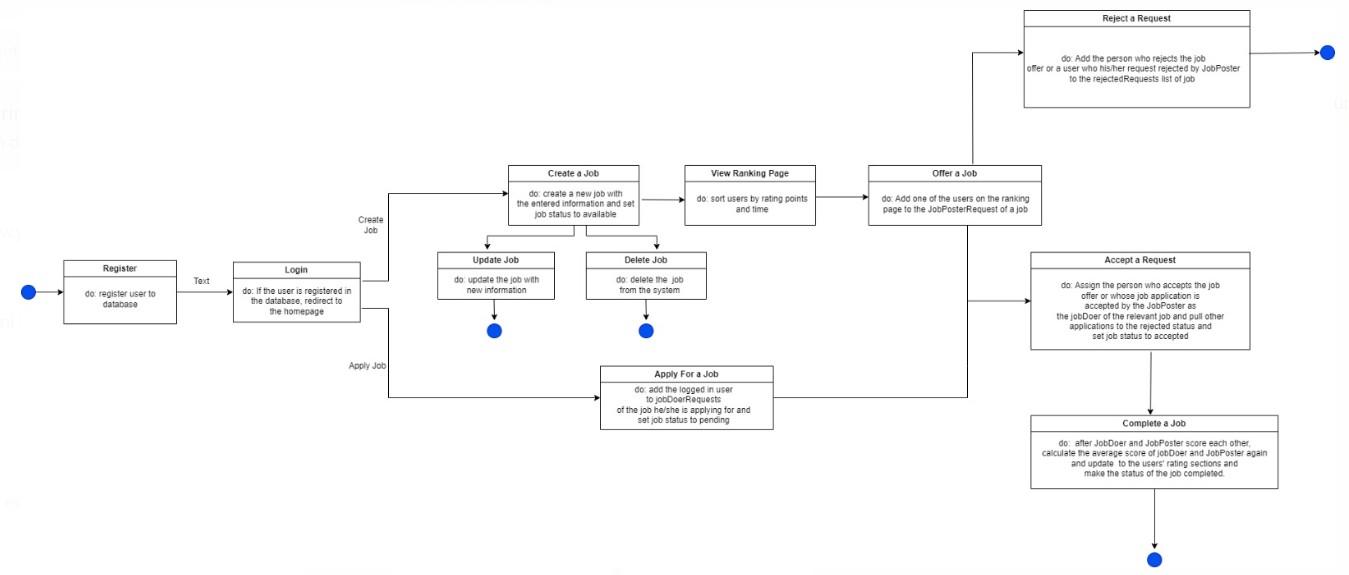


The User class holds the user's information. While each job has an owner, a user can have more than one job

#### 

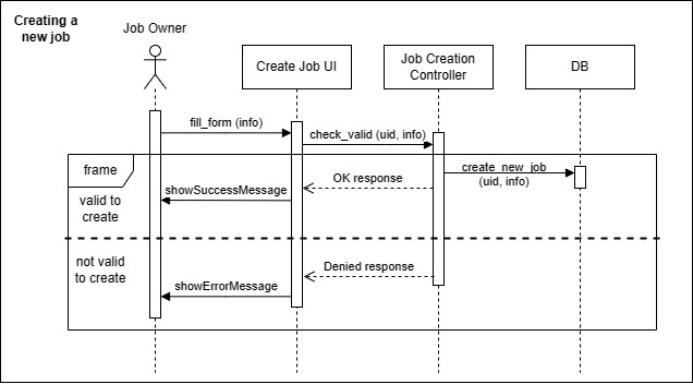
## Behavioral Models

### Data Driven (Activity Diagrams / Sequence Diagrams)

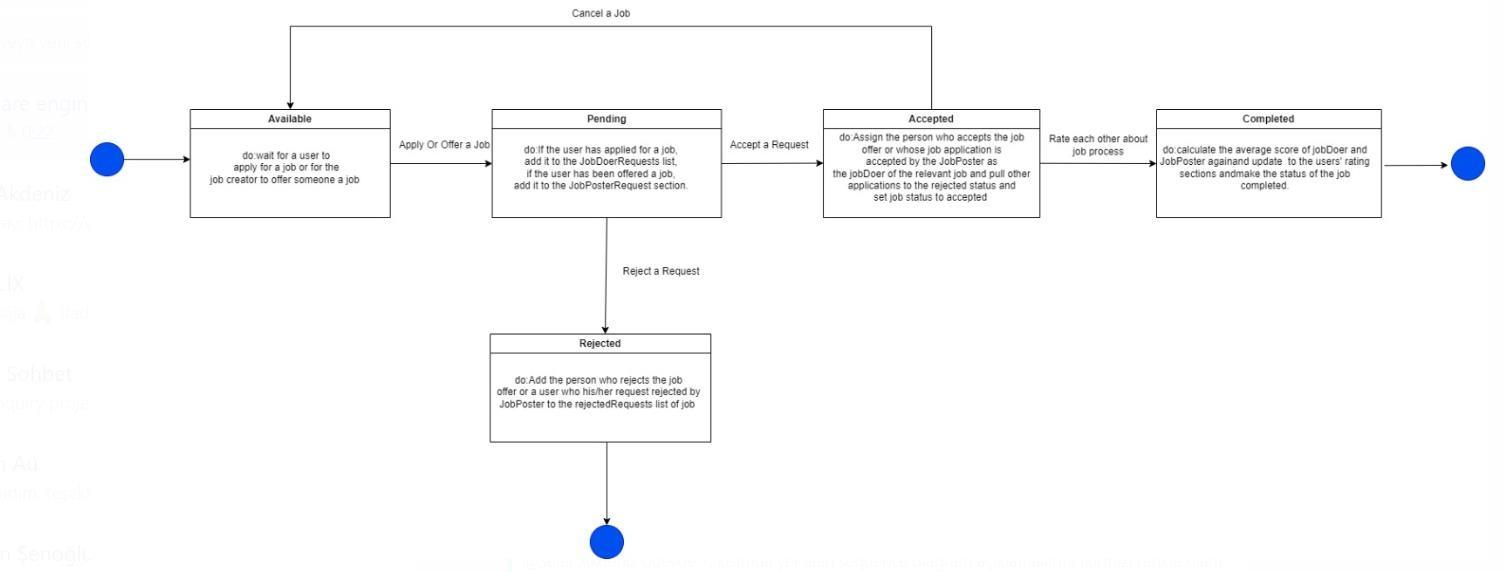


### Data Driven (Sequence Diagrams)

#### Sequence diagram for creating a new job



### Event Driven Models (State Diagrams)



## Test

### 6.4.1 Functional Test Cases

### 6.4.1.1 Creating an Account

Test ID: 1

Test Category : Functional

Objective : Make sure that a user can successfully create a new account.

Preconditions : None

Steps

1. Open the HALLET application in your phone or website
2. Click the Sign Up button
3. Fill in the email,name,surname and password fields with valid user info
4. Press the Register button

Verify that you are creating a new user account using valid user information

Expected Result : The user can successfully create a new account.

Severity : Critical

### 6.4.1.2 Not Allowing Login Without an Account

Test ID: 2

Test Category : Functional

Objective : Testing that a user cannot access the system and use features without logging in

Preconditions : None

Steps

1. Open the HALLET application in your phone or website
2. Verify that the login screen is displayed
3. Enter an email and password that is not registered in the system
4. Press the Login button

Verify that you are unable to login and get the “email or password incorrect” exception

Expected Result : The user cannot access the system and use features without creating an account

Severity : Critical

### 6.4.1.3 Filtering Jobs Listed on Home Screen

Test ID: 3

Test Category : Functional

Objective : Filtering the jobs listed on the homescreen by job type, min and max payment and cities

Preconditions : Must register the system, log in

Steps

1. Open the HALLET application in your phone or website
2. Login the system
3. Press the Filter button on the top left of the home screen
4. In the drawer, to filter jobs, select the job types, minimum/maximum payment, or the cities you want to filter by.
5. Press the Apply Filter button

Verify that you can filter jobs by job types, min/max payment, or cities

Expected Result : Users can filter jobs by job types, min/max payment, or cities

Severity : Major

### 6.4.1.4 Rate the Job Doer of Your Job

Test ID: 4

Test Category : Functional

Objective : Testing the job creator's can rate the job doer after complete a job

Preconditions : Must register the system, log in, create a job and accept the application of one of the job applicants

Steps

1. Open the HALLET application in your phone or website
2. Login the system
3. Open the accepted tab on the MyJobs screen
4. Select the job whose status you want to set to completed
5. Set the status of the job as completed by clicking the Complete button in the Job Details dialog screen.
6. Then rate the user who did the job on the rating screen that opens

Verify that as the owner of the job, you can rate the person doing the job after you set the status of a job to completed

Expected Result : Job creator's can rate the job doer after complete a job

Severity : Minor

### 6.4.2 API Test Cases

### 

### 6.4.2.1 Confirm Retrieval of Correct Job Details for a Specific Job

# 

# Test ID: 5

# 

# Test Category: Job Managing

# 

# Objective: Ensuring the API returns the correct details for a specific job.

# 

# Pre-conditions:

# The /api/Jobs/{Id} endpoint is available.

# A job with a known ID exists in the system.

# Steps:

# Category: API Endpoint Availability

# - Objective: Confirm that the job details endpoint is accessible.

# - Pre-conditions: None

# - Steps:

# Send a GET request to the job details endpoint.

# 

# Verify that the API returns a successful status code (e.g., 200) displaying and showing the endpoint is available.

# 

# Expected Result: The API should return a success status code (e.g., 200) showing that the job details endpoint is accessible.

# 

# Severity: Major

# 

# 2. Category: Get Job Details

# - Objective: Verify that the API returns exact details for a specific job.

# - Pre-conditions:

# 1. The job details endpoint is available.

# 2. A job with a known ID exists in the system.

# - Steps:

# 1. Create a test job in the database with known details and obtain its unique

# ID.

# 2. Send a GET request to the job details endpoint, providing the job ID.

# 3. Check that the API response contains the expected job information, such as

# title, description, title, and any other information about the job's details.

# 

# Validate that the API returns a success status code (e.g., 200) telling that the job details came successfully.

# 

# Expected Result: The API should return the accurate details for the specific job.

# Severity: Major

# 

# 3. Category: Invalid Job ID

# - Objective: Verify the API's response when an invalid job ID is provided.

# - Pre-conditions:

# 1. The /api/Jobs/{Id} endpoint is available.

# 2. No job exists with the provided ID.

# - Steps:

# Send a GET request to the job details endpoint, providing an invalid job ID (e.g., a non-existent ID or an invalid format).

# Check that the API returns an error status code (e.g., 404 or 400) indicating that the job details could not be found.

# Validate that the error response contains a meaningful error message or code to indicate the reason for the failure.

# 

# Expected Result: The API should return an error status code (e.g., 404 or 400) indicating that the job details could not be found.

# 

# Severity: Major

# 

# 

# 

# 

# 

### 6.4.3 Non-Functional Test Cases

### 6.4.3.1 Browser Compatibility Testing

Test ID: 6

Test Category : Non-Functional - Compatibility

Objective : Test if the Hallet web application works properly on different web browsers (Chrome, Firefox, Safari, Internet Explorer, etc.).

Preconditions : The latest stable working version of our web application must be available. The browsers to be tested must be in their most up-to-date versions.

Steps

1. Determine the browsers to be tested.
2. Identify browser versions, operating systems, screen resolutions, and other technical specifications that your application needs to be compatible with.
3. Prepare the test scenarios. These scenarios should cover fundamental functions and features (like login to Hallet, create new job posting and then applying to a job).
4. Apply the defined scenarios in each browser and obtain the results.

Expected Result : Hallet web application is to run smoothly and consistently in every browser.

Severity : Major

### 6.4.3.2 Security Testing

Test ID: 7

Test Category : Non-Functional - Authentication

Objective : Evaluating the authentication mechanisms of the Hallet and testing its security.

Preconditions : There must be already registered users in the system.

Steps

1. Open the Hallet login page by the mobile or web app.
2. Try to login with existing and non-existing credential informations.
3. Try to access home, user profiles or job detail pages without login to the system.

Expected Result : Other users or potential attackers must not gain an access to a user account, important accounts and data will be protected.

Severity : Critical

# 

# 

# 

# 

# 

# 

# **Chapter 2: Implementation**

# List of Change Requests and Assessments

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1.) SUBMITTER - GENERAL INFORMATION** | | | | | |
| **CR#** | #28 MyJobs Sayfası Eklendi | | | | |
| **Type of CR** | ☒ Enhancement | ☐ Defect |  | | |
| **Project/Program/Initiative** | Hallet Mobile App | | | | |
| **Submitter Name** | Sefer Öztürk | | | | |
| **Brief Description of Request** | I request the addition of a "JobHistory" section to the Hallet mobile app. This section should display the history of jobs that the user has logged in, including jobs completed as the owner and jobs completed as the performer, as well as jobs applied for but rejected. | | | | |
| **Date Submitted** | 05/08/2023 | | | | |
| **Date Required** | 05/15/2023 | | | | |
| **Priority** | ☐ Low | ☐ Medium | ☒ High | | ☐ Mandatory |
| **Reason for Change** | To provide users with a better experience and enable them to track their past jobs, I request the addition of this section to the Hallet mobile app. | | | | |
| **Other Artifacts Impacted** | Mockup,Hallet Website | | | | |
| **Assumptions and Notes** | The requested change should align with the design standards of the Hallet mobile app. Additionally, the section should accurately display job and user data and ensure the implementation of appropriate security measures. | | | | |
| **Comments** | - | | | | |
| **Attachments or References** | ☒ Yes | ☐ No |  | | |
| **Link:** | | | | |
| **Approval Signature** | [Approval Signature] | | **Date Signed** | 05/9/2023 | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **2.) PROJECT MANAGER - INITIAL ANALYSIS** | | | | | |
| **Hour Impact** | [#hrs] | | 9 hours | | |
| **Duration Impact** | [#dys] | | 3 days | | |
| **Schedule Impact** | [WBS] | | The requested change will require significant modifications to the user interface of the Hallet mobile app. Additionally, it involves the storage of job history data and the creation of a new database table named RejectedRequests to handle the transactions. | | |
| **Cost Impact** | [Cost] | | This change will result in increased project costs due to additional hours worked and the creation of a new database table. | | |
| **Comments** | Other development tasks not affected | | | | |
| **Recommendations** | It is recommended to share the details of the requested change with the project team | | | | |
| **Approval Signature** | | [Approval Signature] | | **Date Signed** | 05/09/2023 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **3.) CHANGE CONTROL BOARD – DECISION** | | | | | | | |
| **Decision** | | ☒ Approved | ☐ Approved with Conditions | | ☐ Rejected | | ☐ More Info |
| **Decision Date** | | [05/15/2023] | | | | | |
| **Decision Explanation** | | [Document the CCB’s decision] | | | | | |
| **Conditions** | | The Change Control Board has approved the requested change. The impact of the change on the project's schedule and budget has been assessed and accepted. | | | | | |
| **Approval Signature** | [Approval Signature] | | | **Date Signed** | | [05/10/2023] | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1.) SUBMITTER - GENERAL INFORMATION** | | | | | |
| **CR#** | #68 Create new job sayfası eklenecek | | | | |
| **Type of CR** | ☒ Enhancement | ☐ Defect |  | | |
| **Project/Program/Initiative** | Hallet - Web App | | | | |
| **Submitter Name** | Sefer Öztürk | | | | |
| **Brief Description of Request** | A page should be prepared that allows the creation of a job | | | | |
| **Date Submitted** | 23/05/2023 | | | | |
| **Date Required** | 06/06/2023 | | | | |
| **Priority** | ☐ Low | ☐ Medium | ☒ High | | ☐ Mandatory |
| **Reason for Change** | There should be a page for the users to create their new job postings and publish them to the other users. After publishing is completed, this job posting will be listed on the home pages of the users. | | | | |
| **Other Artifacts Impacted** | Home Pages and User Profile Pages | | | | |
| **Assumptions and Notes** | If the required information is not entered while creating the job posting, the user should be warned. Empty information should not be set to null. | | | | |
| **Comments** | *-* | | | | |
| **Attachments or References** | ☐ Yes | ☒ No |  | | |
| **Link:** | | | | |
| **Approval Signature** | *[Approval Signature]* | | **Date Signed** | 23/05/2023 | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **2.) PROJECT MANAGER - INITIAL ANALYSIS** | | | | | |
| **Hour Impact** | *[#hrs]* | | 3 hours | | |
| **Duration Impact** | *[#dys]* | | 2 days | | |
| **Schedule Impact** | *[WBS]* | | The requested change will require the development of the POST method, which allows adding job posting entities in the backend. | | |
| **Cost Impact** | *[Cost]* | | This change will result in increased project costs due to additional hours worked. | | |
| **Comments** | *-* | | | | |
| **Recommendations** | If photos cannot be added while creating a job posting, they can be added after creation. | | | | |
| **Approval Signature** | | *[Approval Signature]* | | **Date Signed** | 23/05/2023 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **3.) CHANGE CONTROL BOARD – DECISION** | | | | | | | |
| **Decision** | | ☒ Approved | ☐ Approved with Conditions | | ☐ Rejected | | ☐ More Info |
| **Decision Date** | | *16/06/2023* | | | | | |
| **Decision Explanation** | | The change made meets expectations as required. | | | | | |
| **Conditions** | | The Change Control Board has approved the requested change. The impact of the change on the project's schedule and budget has been assessed and accepted. | | | | | |
| **Approval Signature** | *[Approval Signature]* | | | **Date Signed** | | 16/06/2023 | |

# Tasks Performed By Each Team Member

Sefer Öztürk (Team Leader) :

* Overall project coordination and management
* Task assignment and tracking for the development team
* Collecting and analyzing mobile application requirements
* User Interface (UI) implementation for Login, Sign Up Pages for Mobile
* User Interface (UI) implementation for Profile Page for Frontend

Furkan Şenoğlu (Mobile Developer):

* Collecting and analyzing mobile application requirements
* Integration with backend APIs to send authorized requests for retrieving and updating data
* User Interface (UI) implementation for Profile, Home, MyJobs,Create New Job, Ranking, Settings Pages
* Sending filtering criteria and displaying filtered job results received from the backend in the mobile app's user interface
* Integration of displaying other users' pages
* Adding Google Maps location feature
* Sending Image properties and displaying job’s images and user’s image received from the backend in the mobile app's user interface
* Displaying the progress of job requests (e.g., accepted, rejected, completed) in the user interface based on the information stored in the backend database
* Allowing users to submit ratings for their completed jobs and Displaying the rating option in the user interface

Arda Atakan Uçan (Frontend Developer):

* Collecting and analyzing frontend application requirements
* Integration with backend APIs to send authorized requests for retrieving and updating data
* User Interface (UI) implementation for Login, Sign Up, Home, MyJobs, Create New Job
* Communicating with the backend to retrieve filtered job results and displaying them in the user interface
* Integration of displaying other users' pages
* Sending Image properties and displaying job’s images and user’s image received from the backend in the mobile app's user interface
* Displaying the progress of job requests (e.g., accepted, rejected, completed) in the user interface based on the information stored in the backend database
* Allowing users to submit ratings for their completed jobs and Displaying the rating option in the user interface

Aslı Yıldırım (Backend Developer):

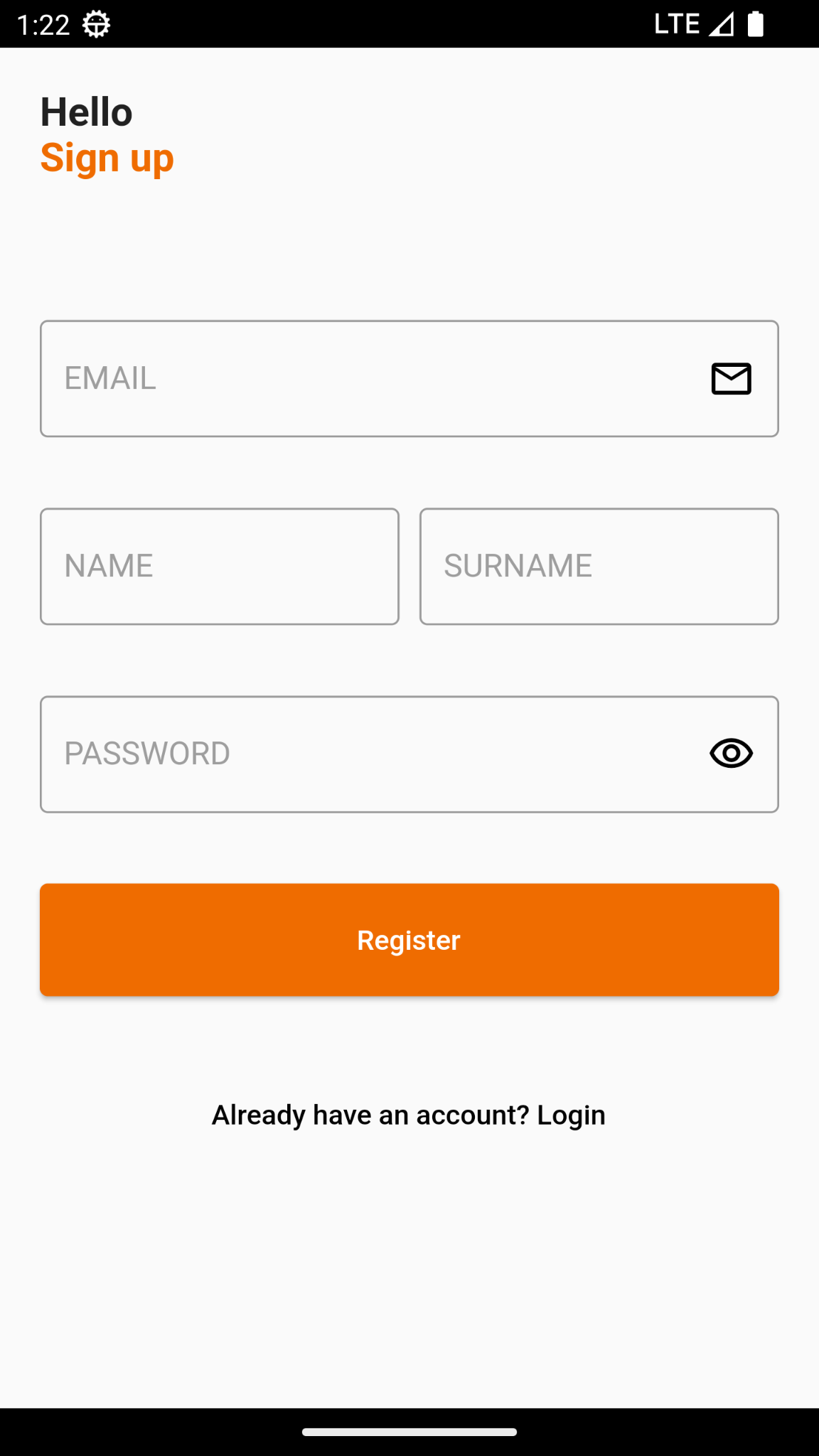
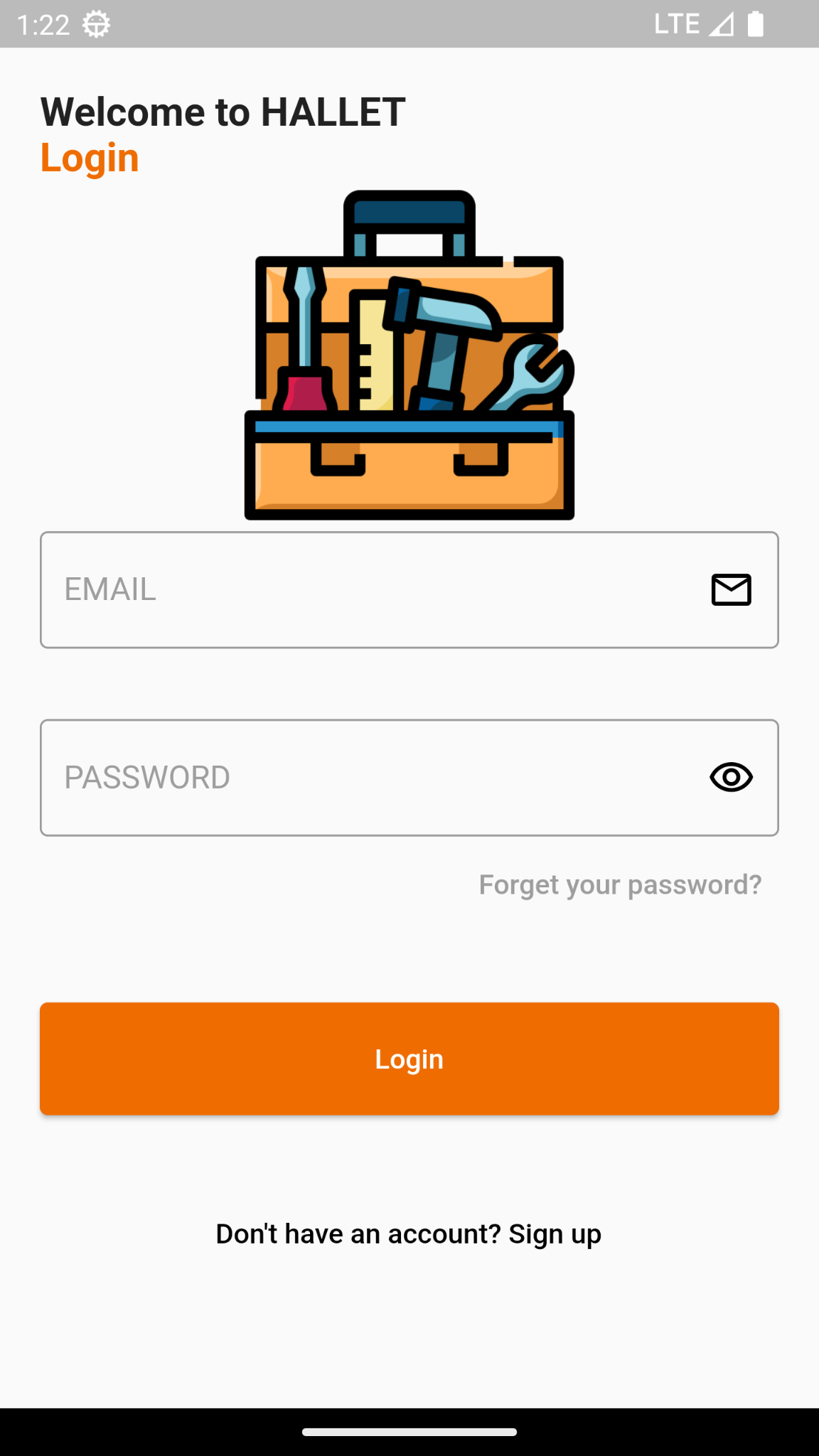
* Collecting and analyzing backend design requirements
* Development of user management functionality (user creation, update, login, retrieval)
* Development of job management functionality (job’s image creation, update, deletion, retrieval)
* Development of job’s images management functionality (job creation, update, deletion, retrieval)
* Creation of endpoints for managing job categories (creation, update, deletion, retrieval)
* Implementation of filtering operations for jobs based on criteria (e.g., category, location) in the backend
* Implementation of authorization and authentication mechanisms for backend APIs, ensuring that only authenticated and authorized users can perform certain actions
* Updating the backend database to reflect the progress and status of job requests (e.g., accepted, rejected, completed)
* Implementation of backend logic for calculating and recording job ratings

# 

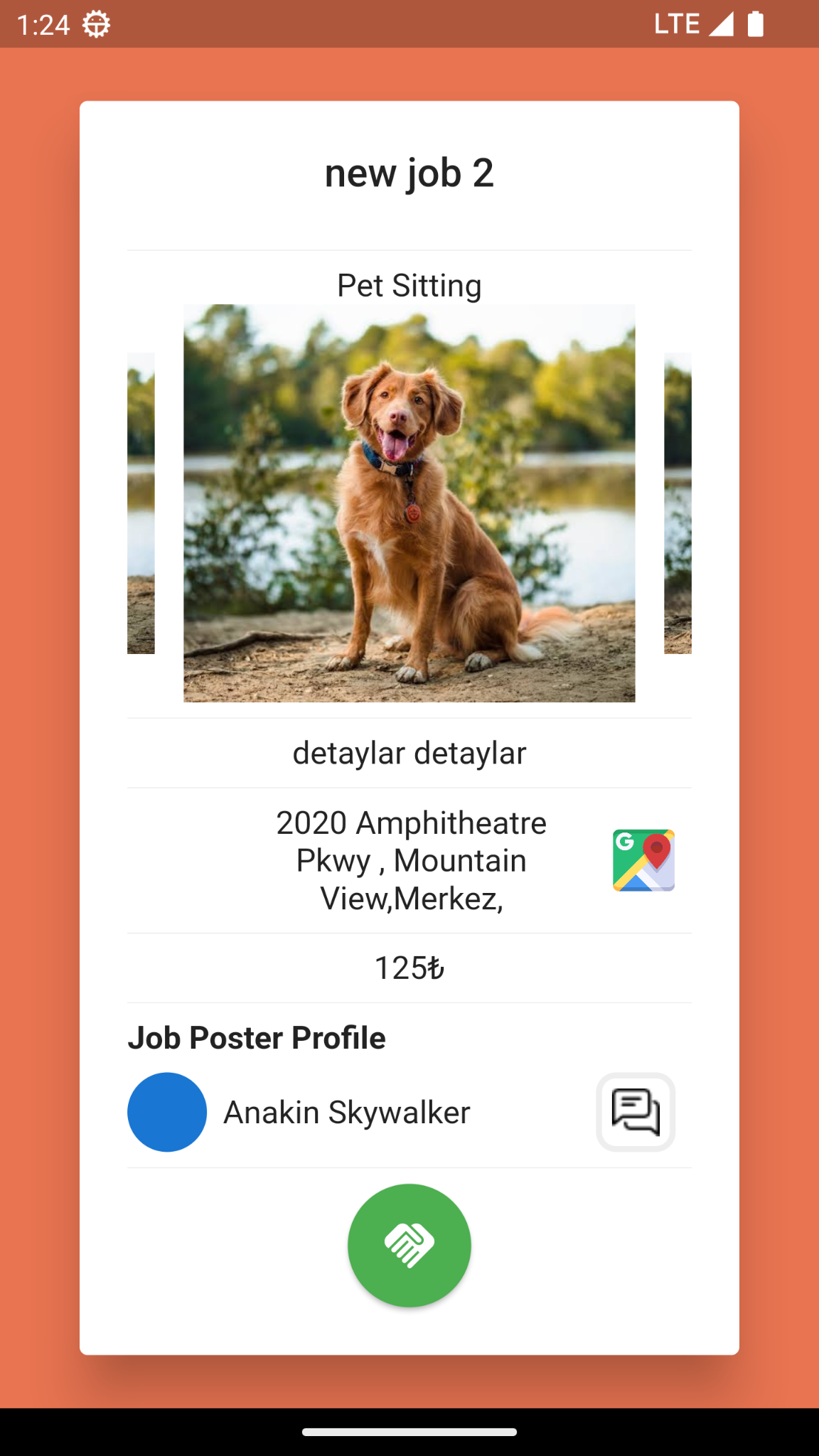
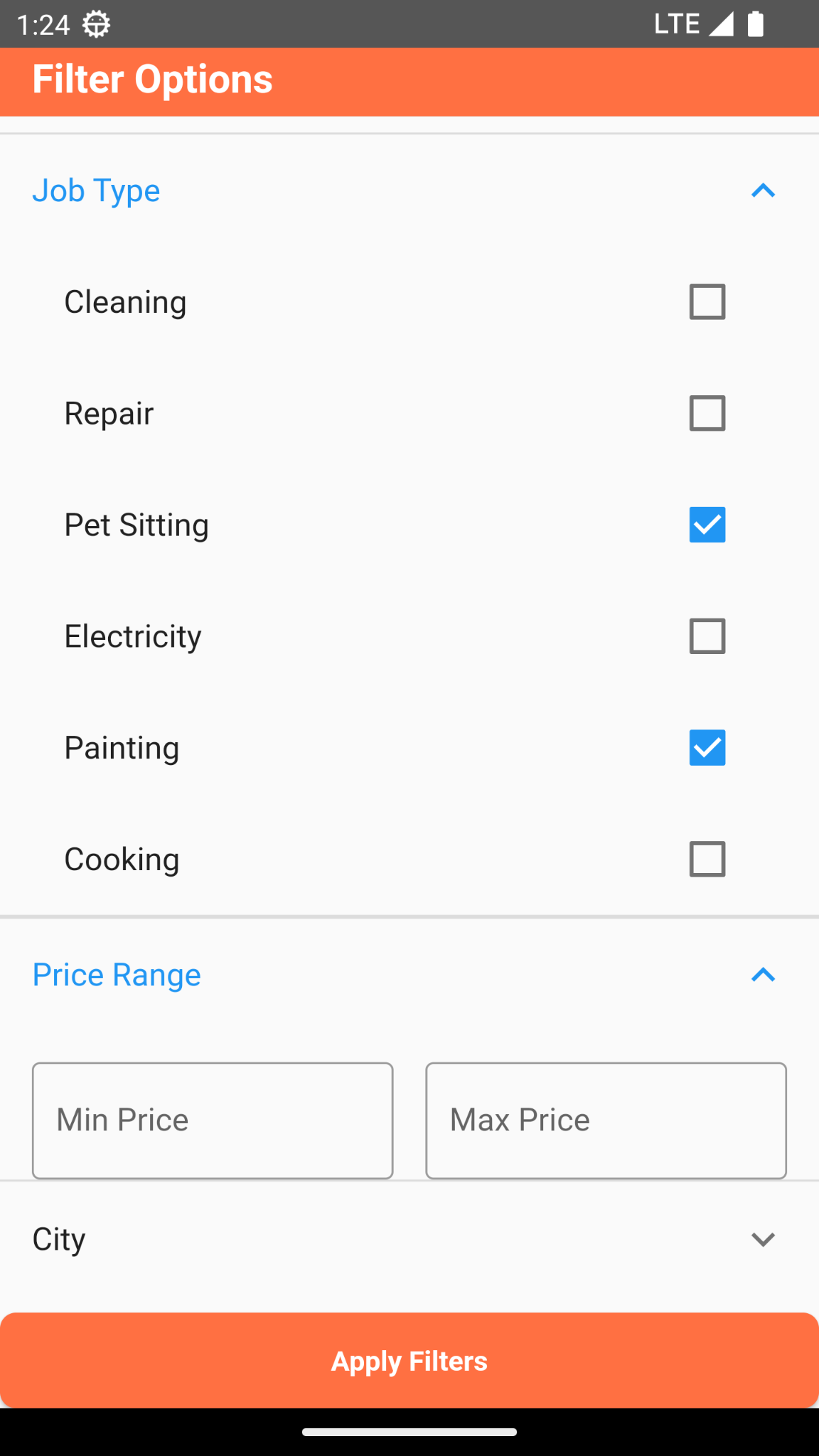
# Details of the Final Product (Database, Screens etc.)

### 9.1 Mobile Screens

9.1.1 Signup and Login Screens

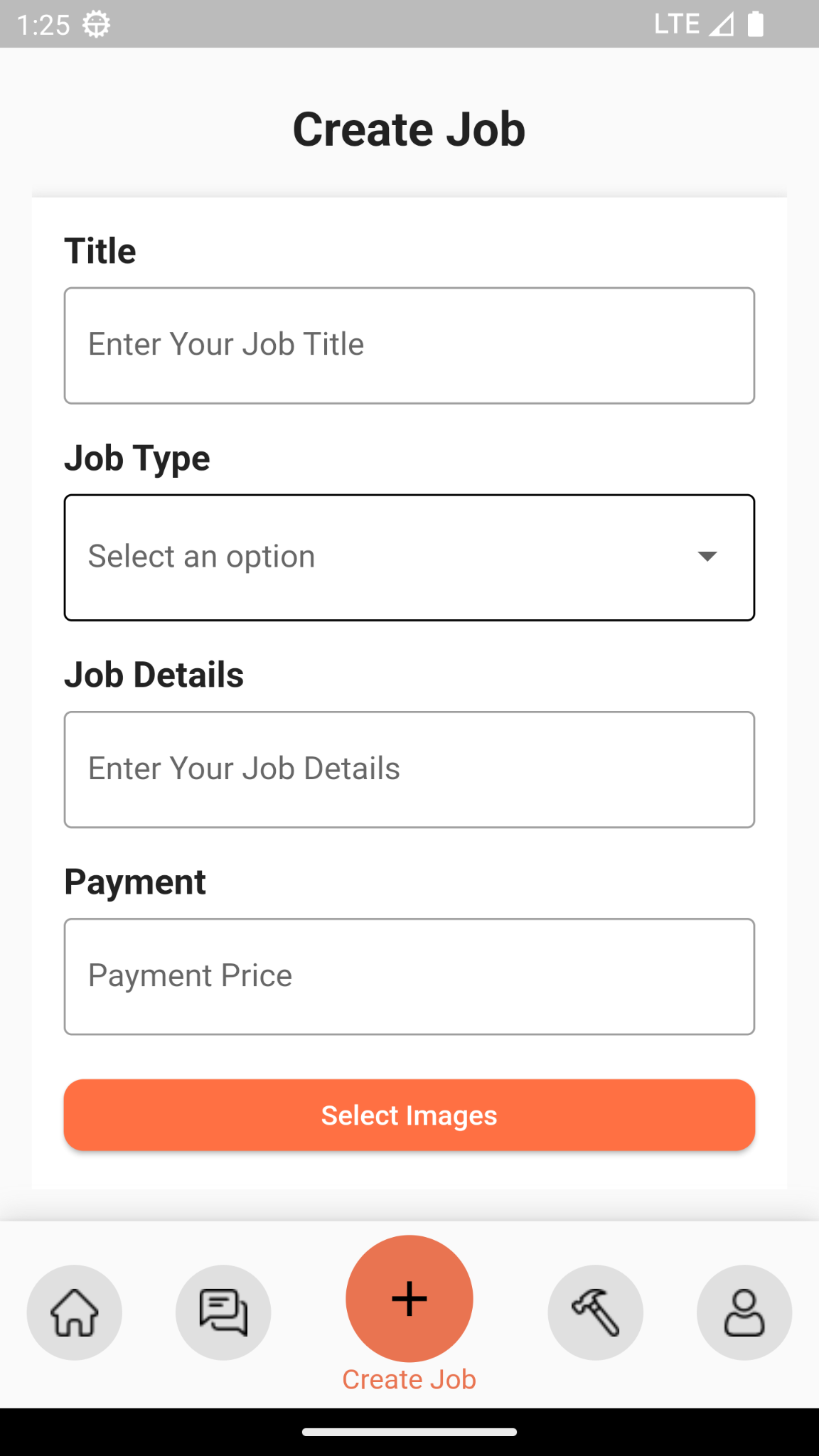
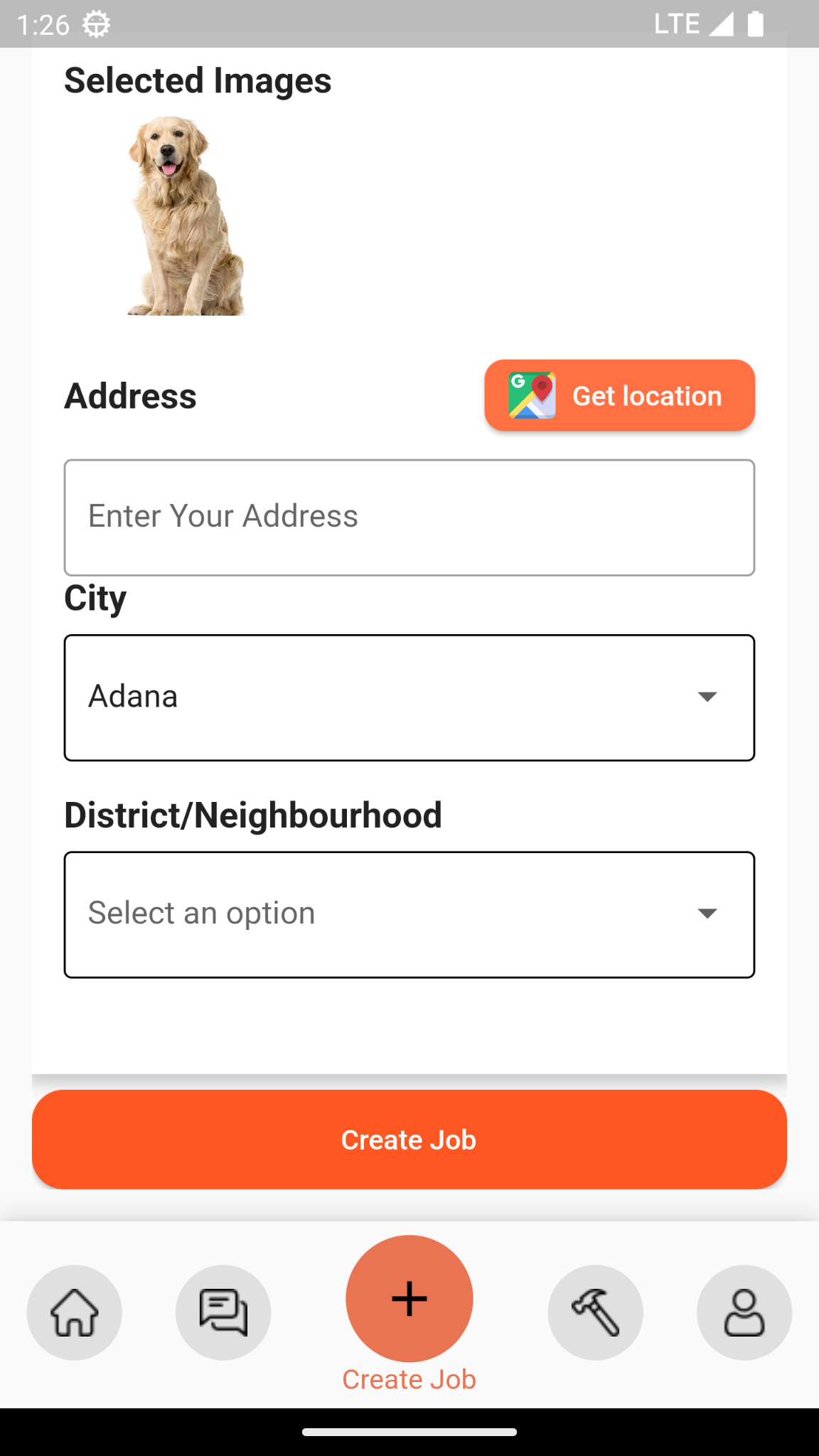


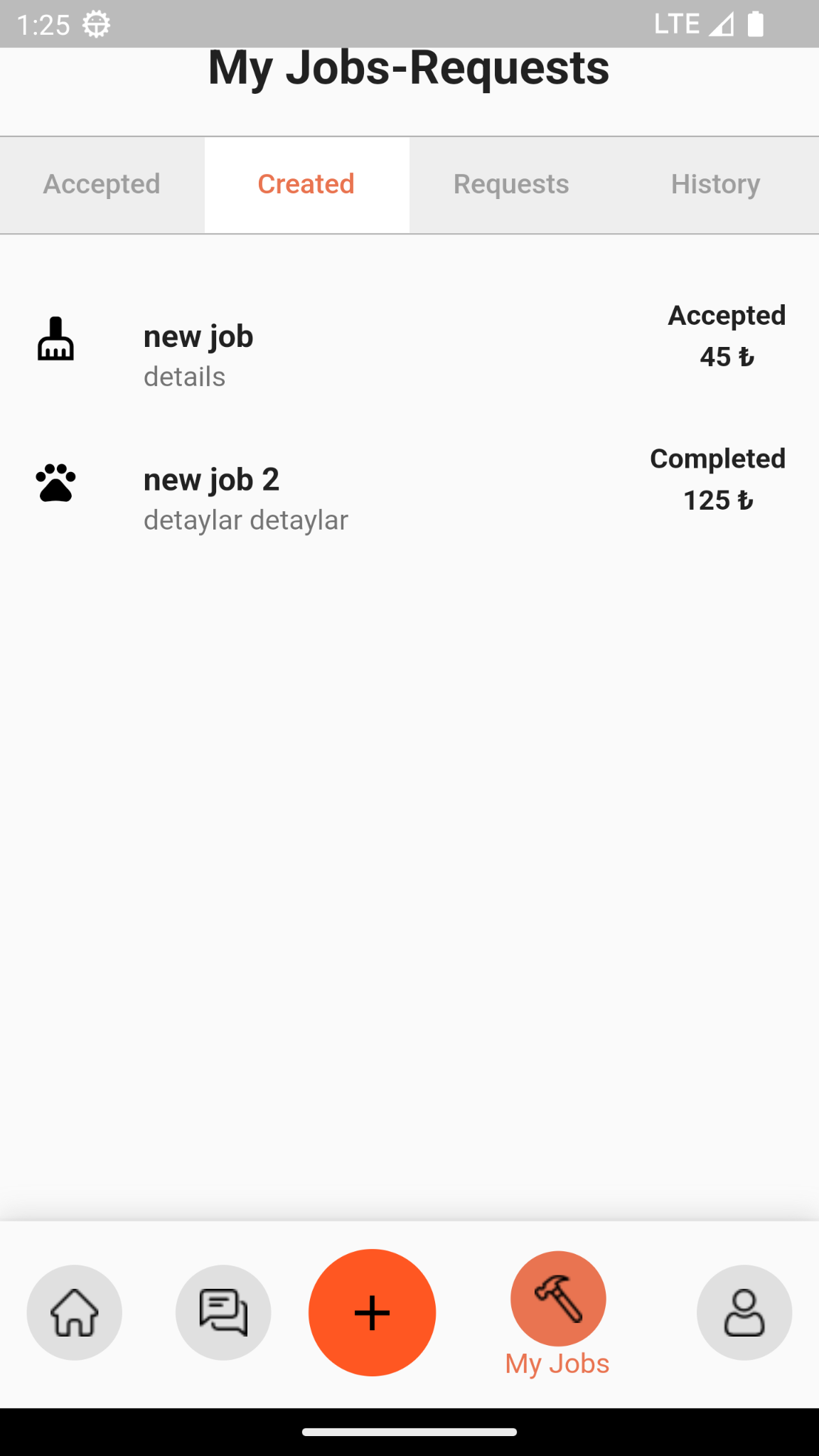
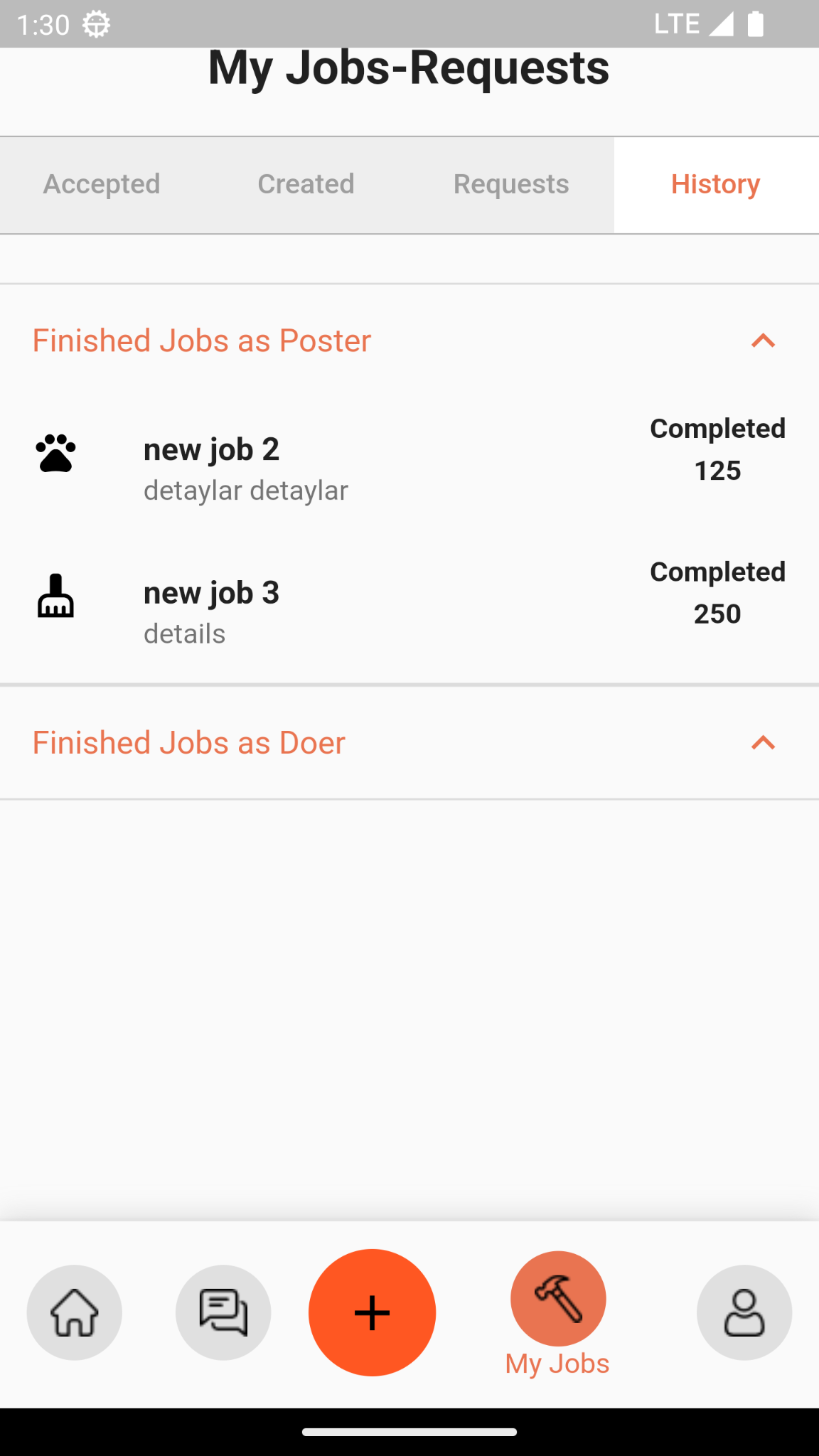
9.1.2 Home,Filter Drawer,Job Details Dialog Screens

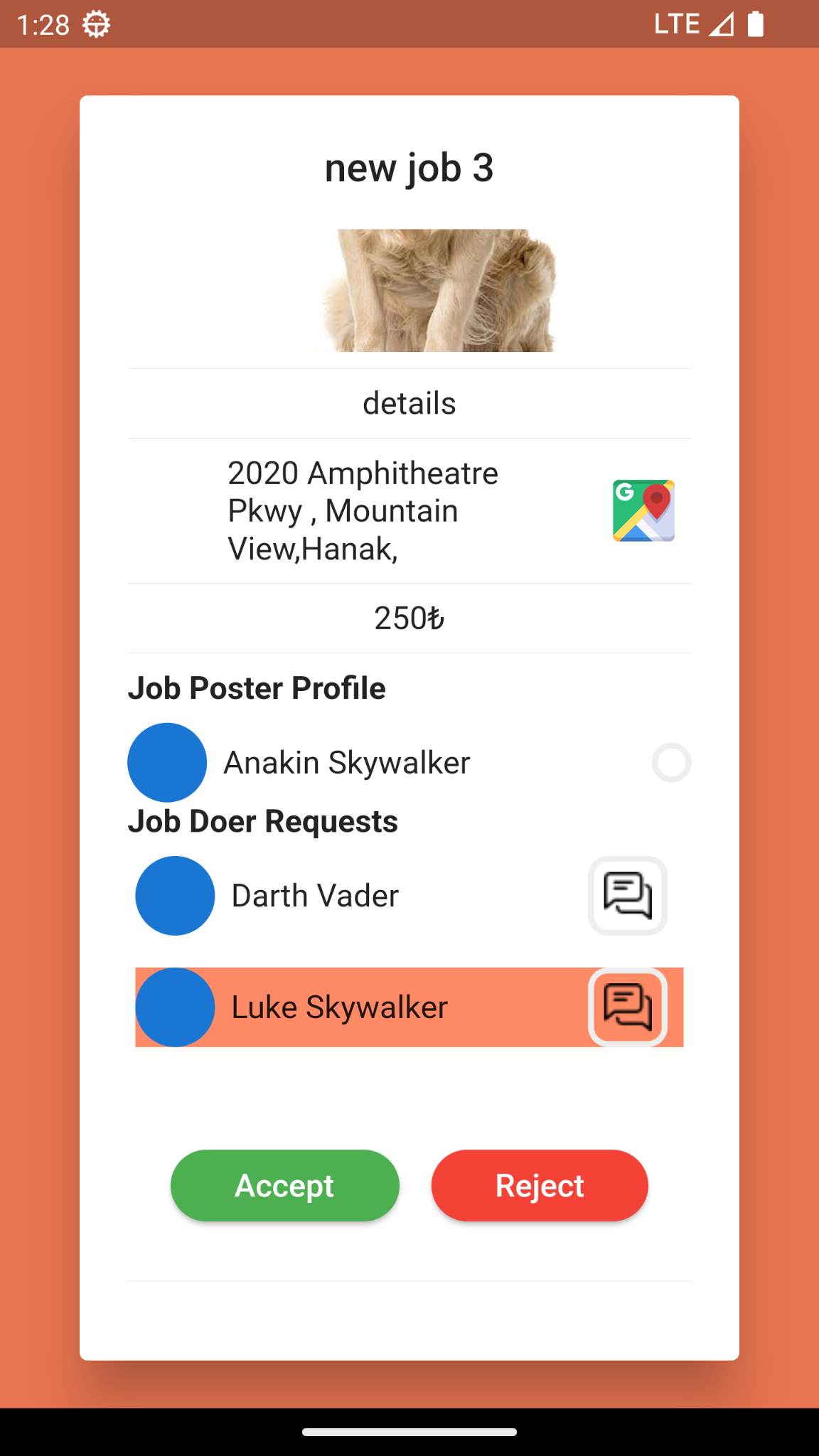


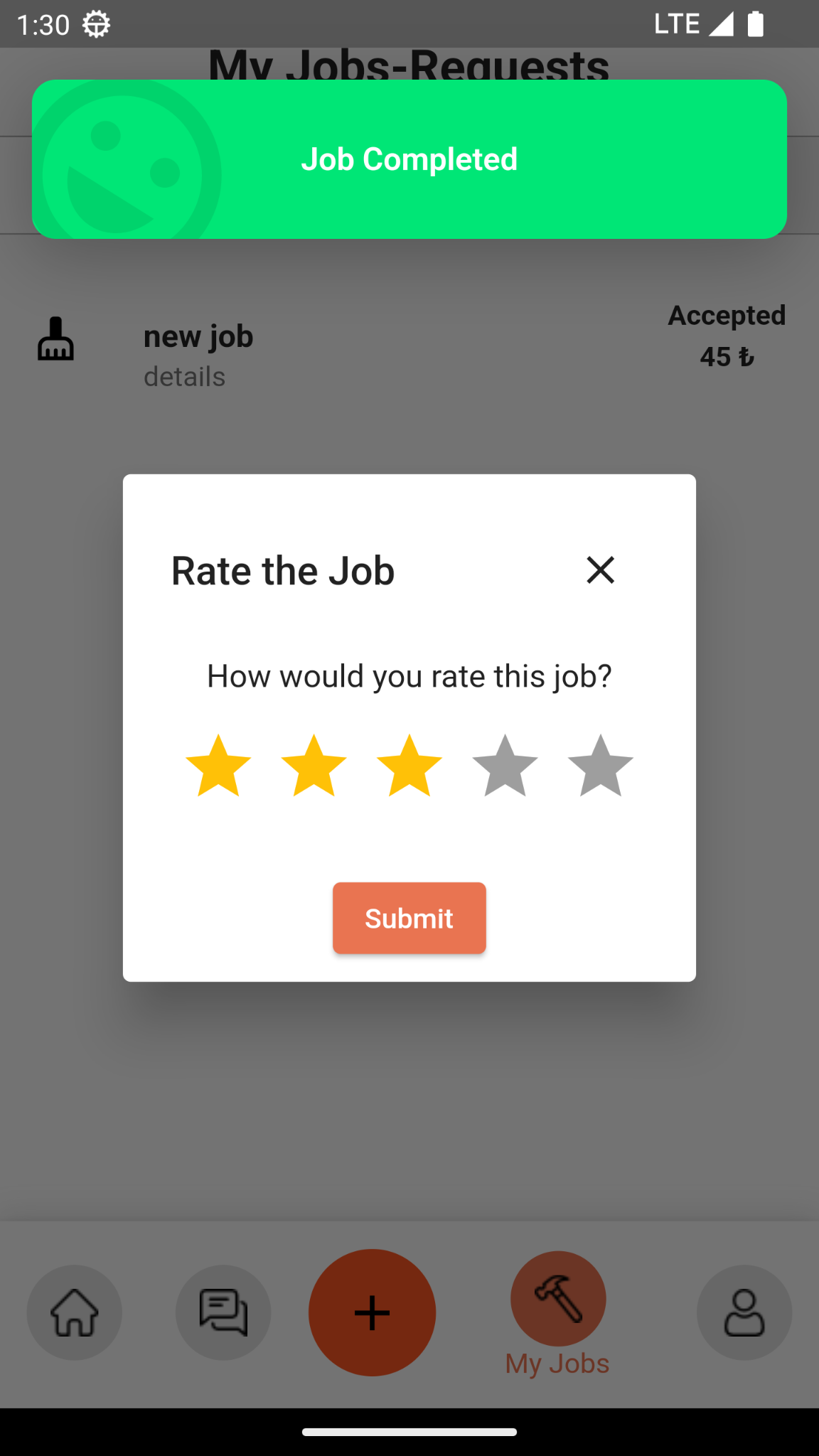
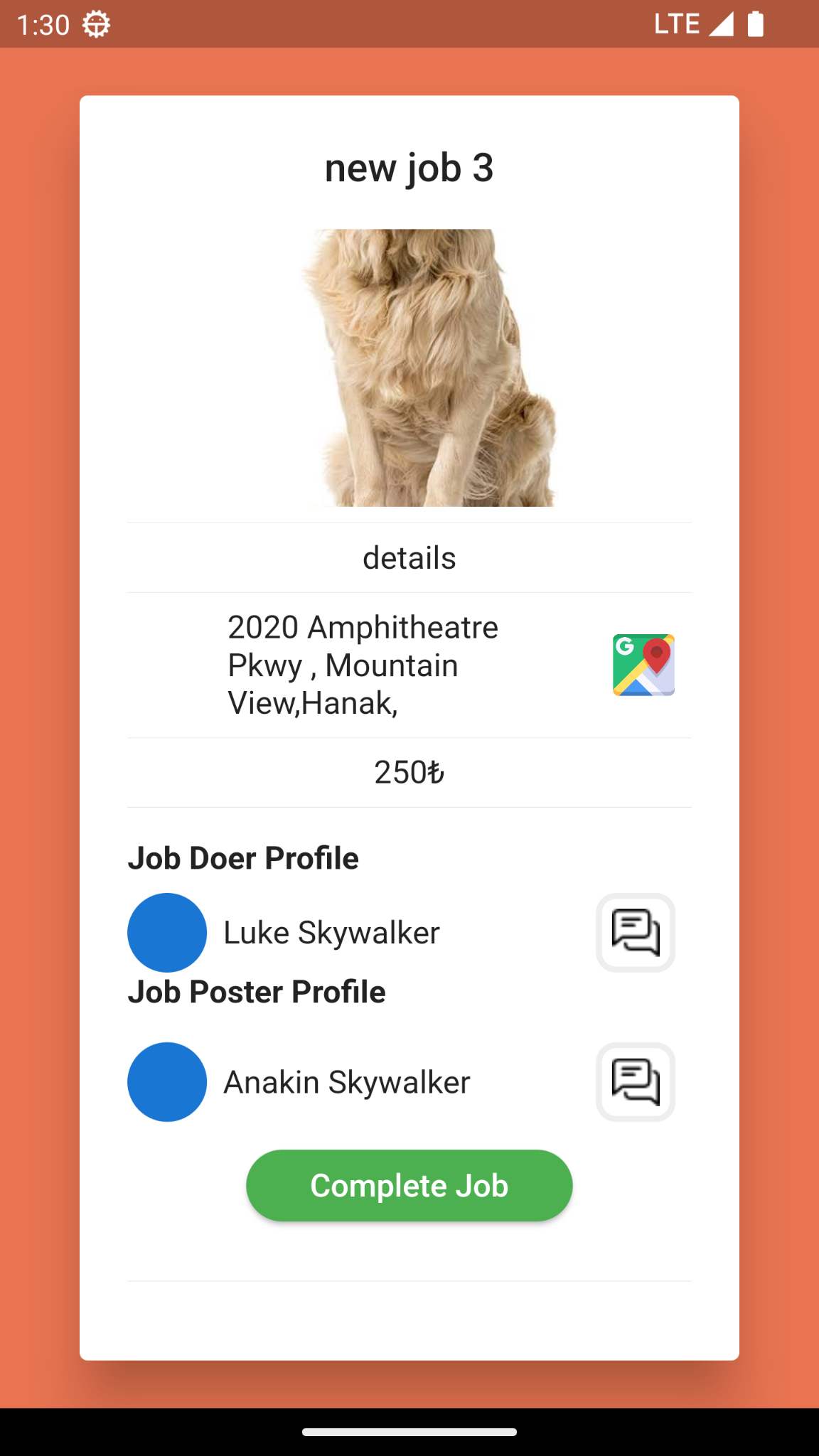
# 

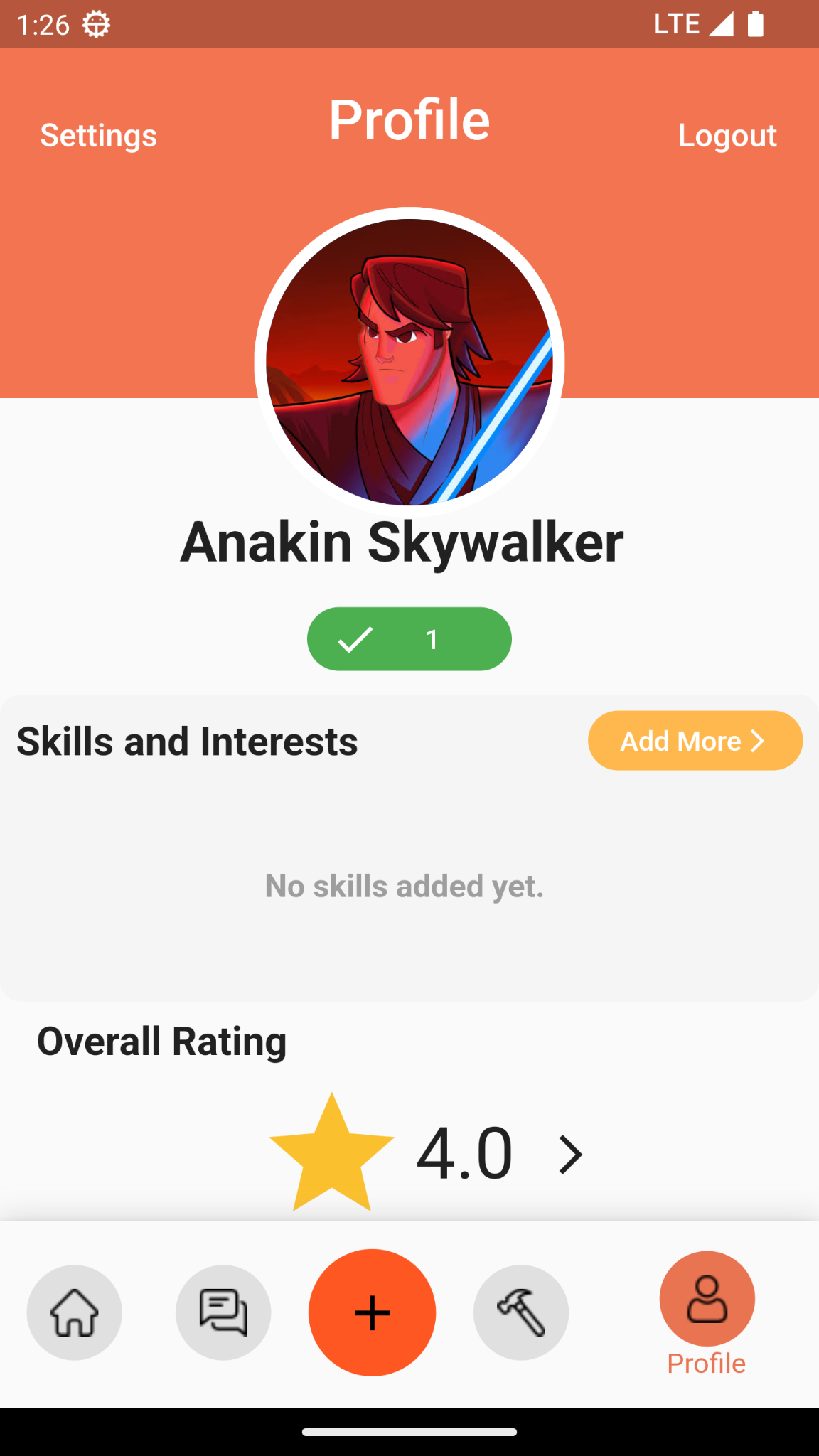
9.1.3 Create Job Screen

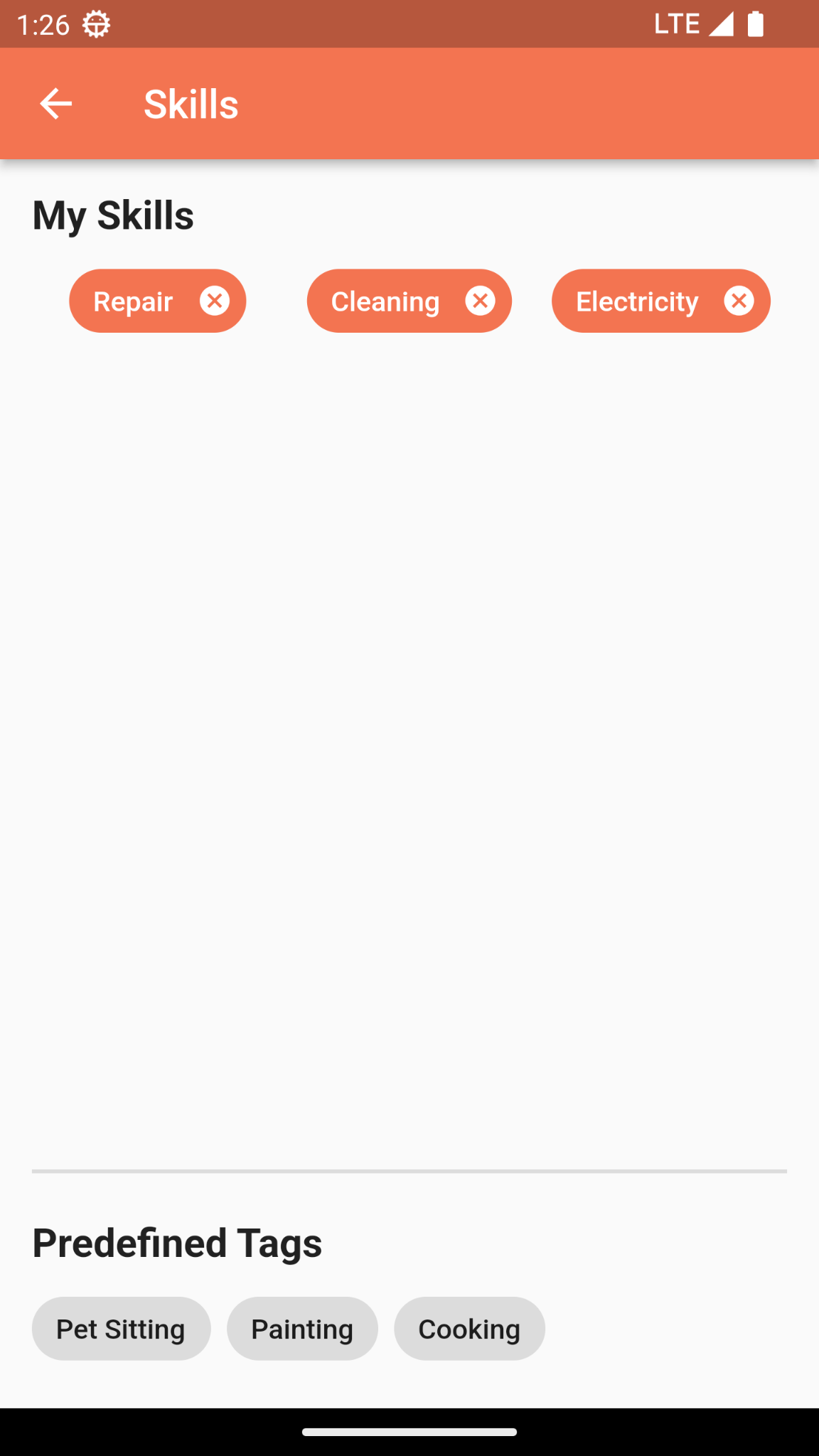


9.1.4 My Jobs Screen

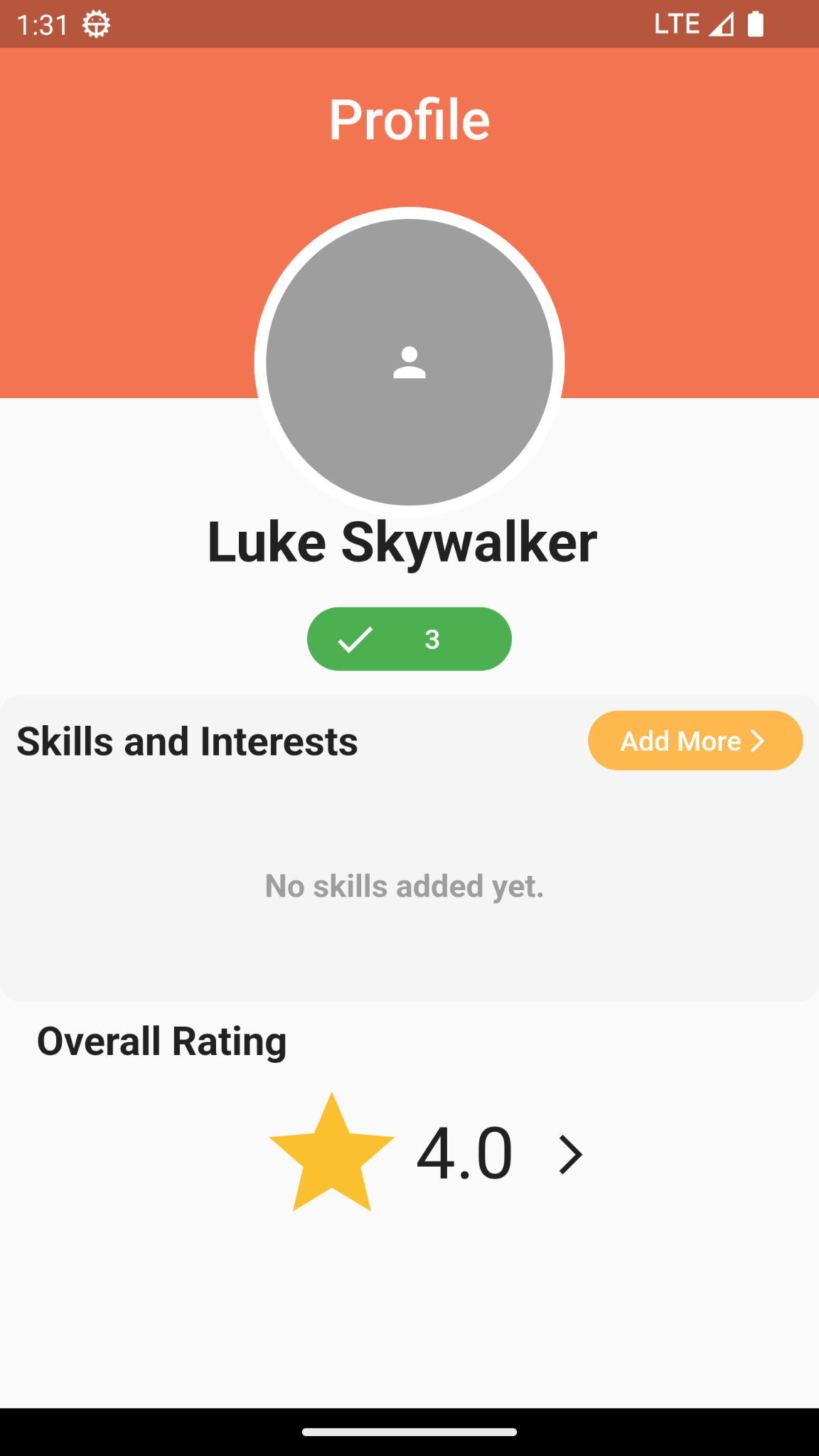
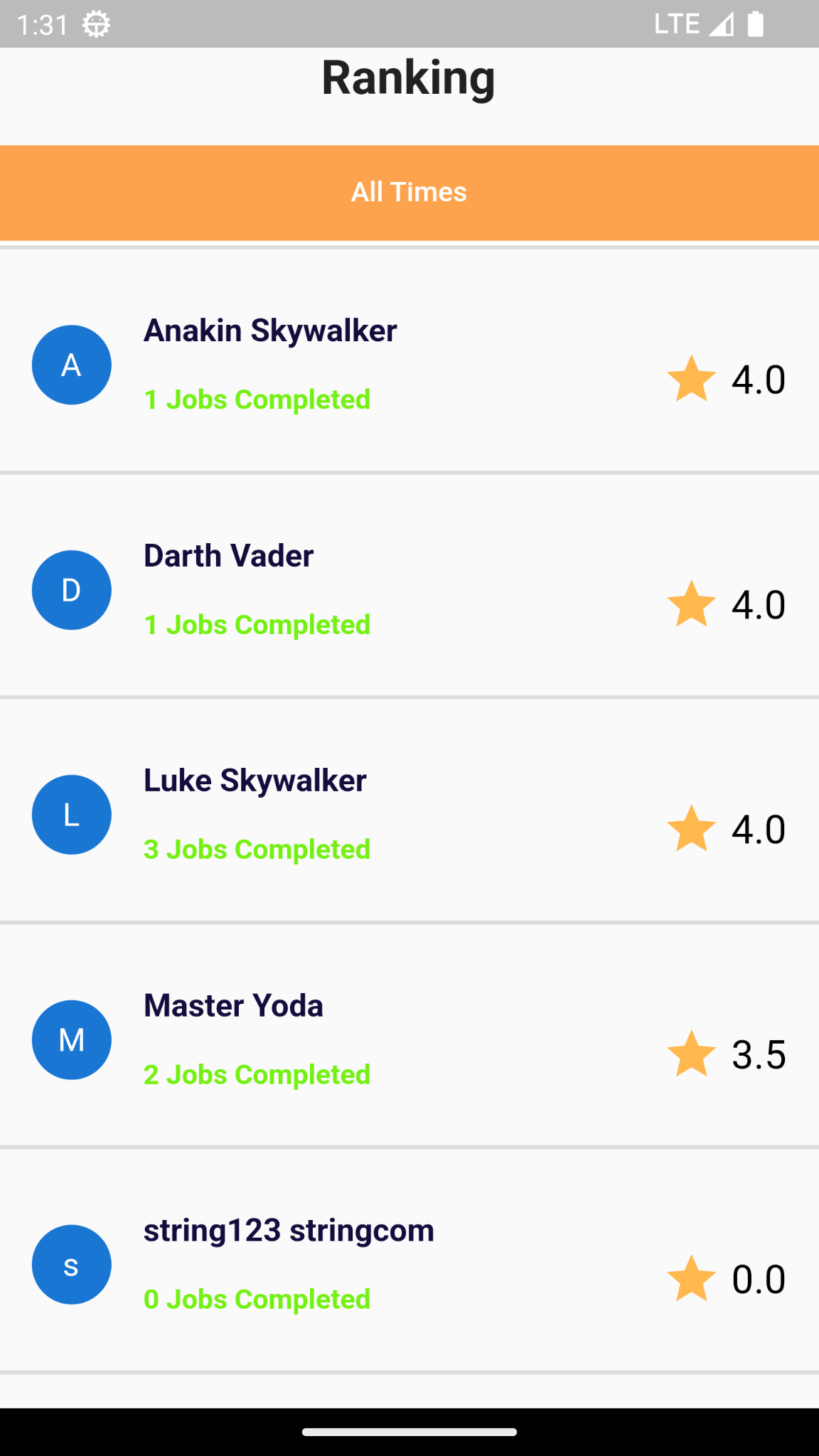
9.1.5 Accept or Reject Request Screen in Requests Tab

9.1.6 Complete Job and Rate Job Doer Screens in Accepted Jobs Tab

9.1.7 Profile and Add/Remove Skill Screens



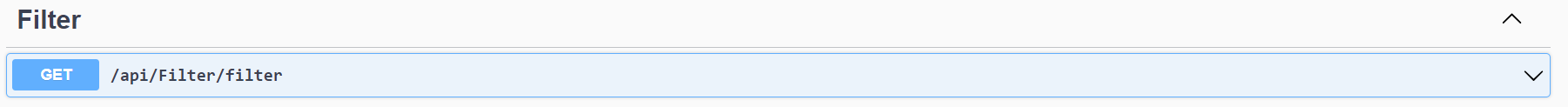
9.1.8 User Ranking and Other Users Profile Screens

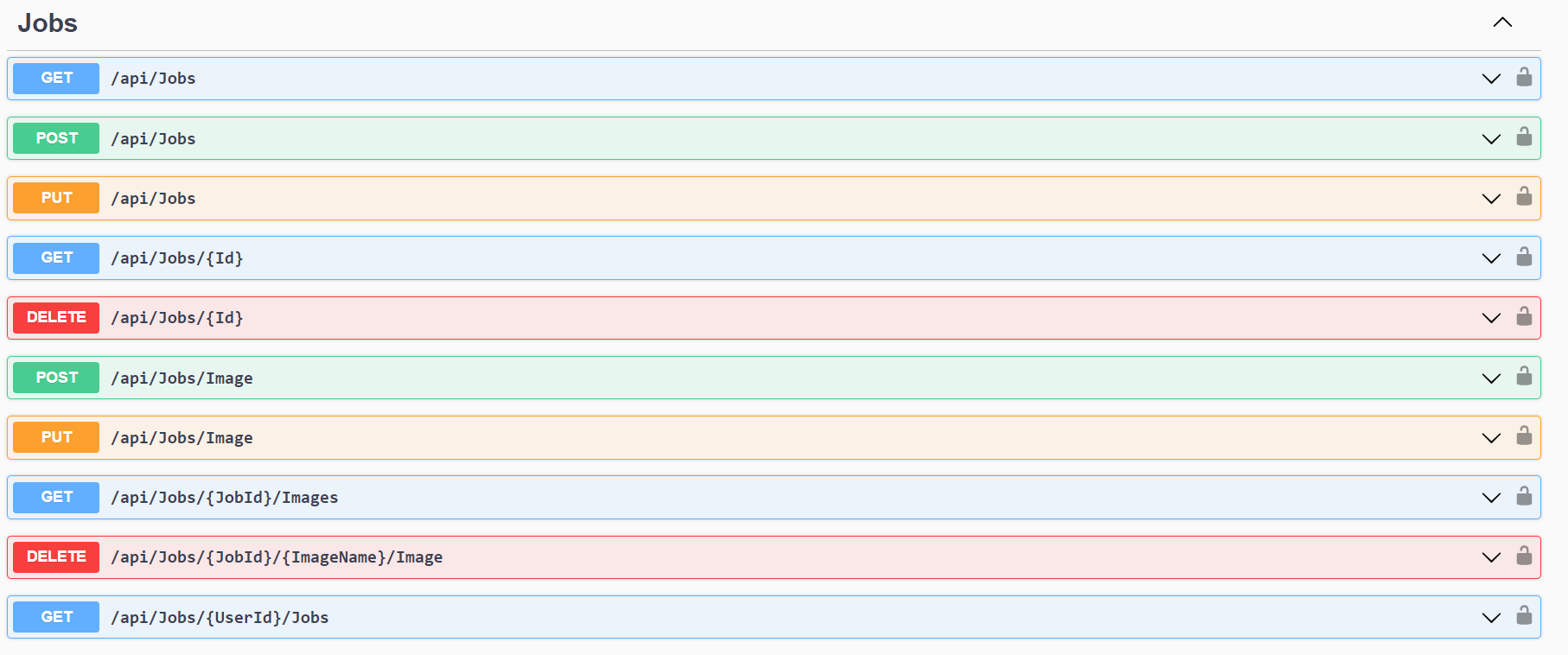


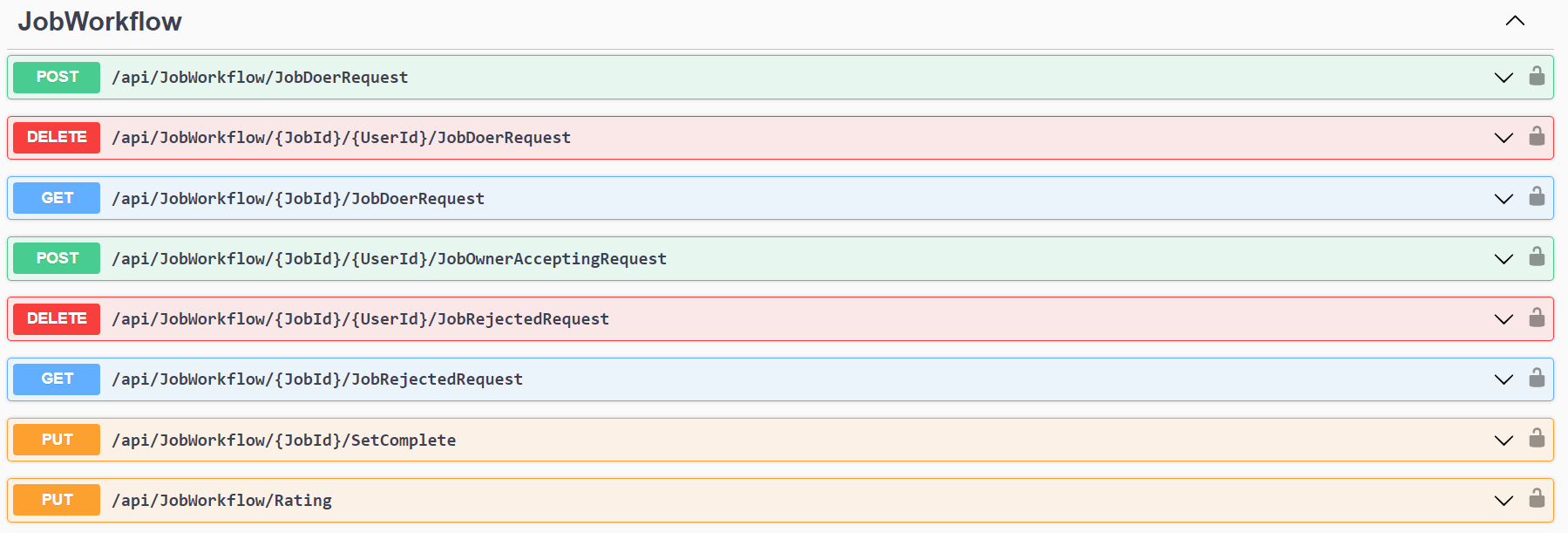
### 9.2 Endpoints Images





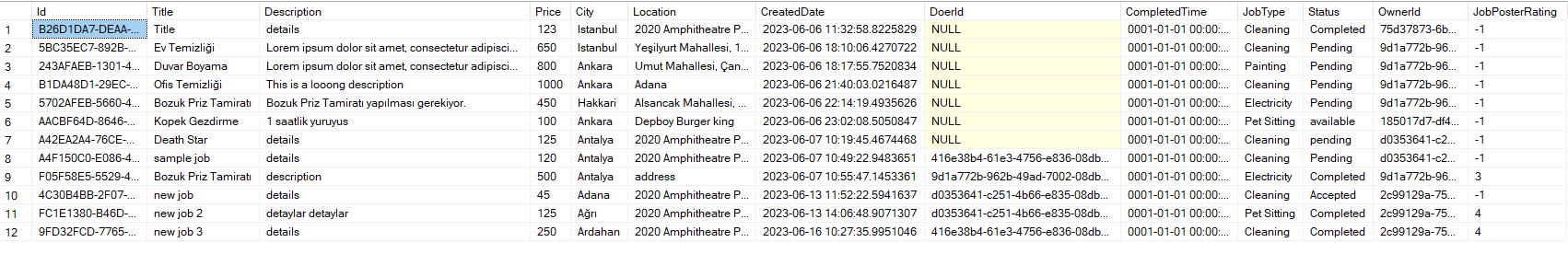




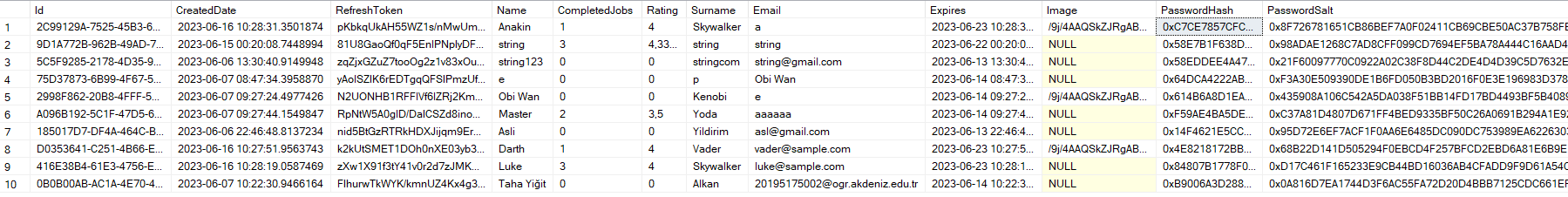


### 9.3 Database Images

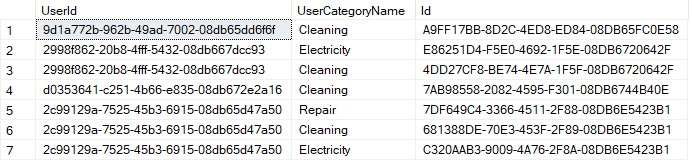
9.3.1 Job Table



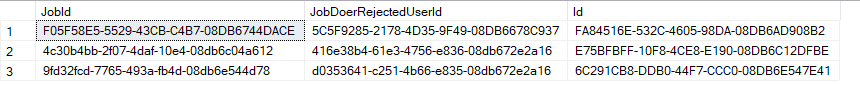
9.3.2 User Table



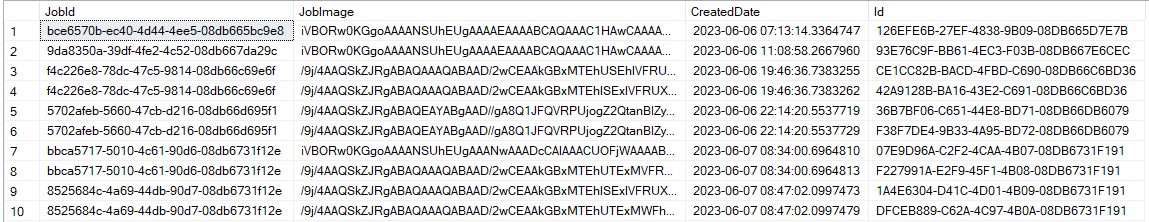
9.3.3 UserCategory Table



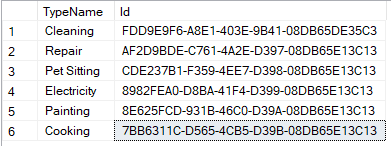
9.3.4 JobRejectedRequests Table



9.3.5 JobImages Table



9.3.6 Category Table

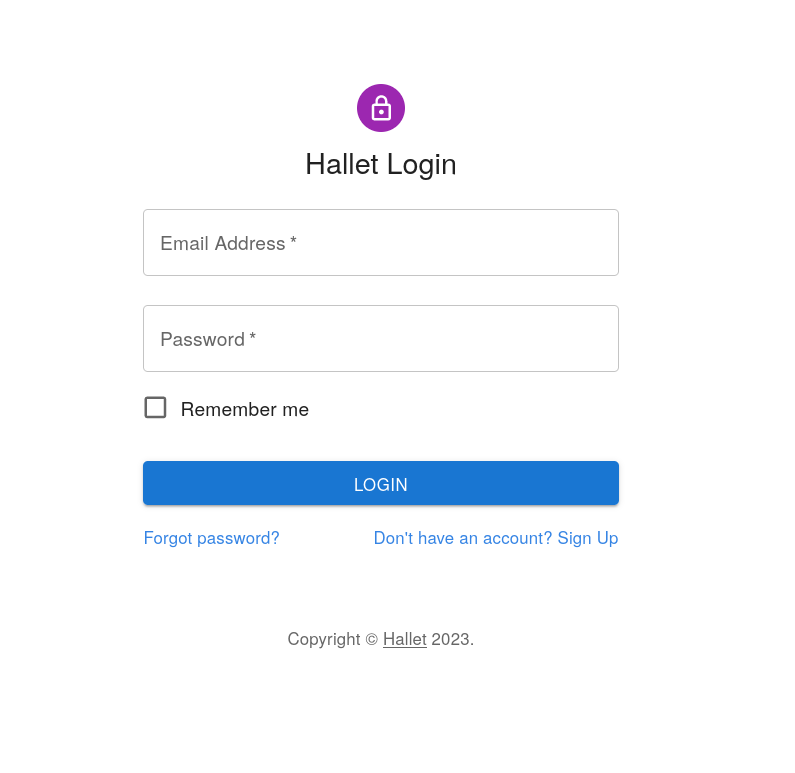
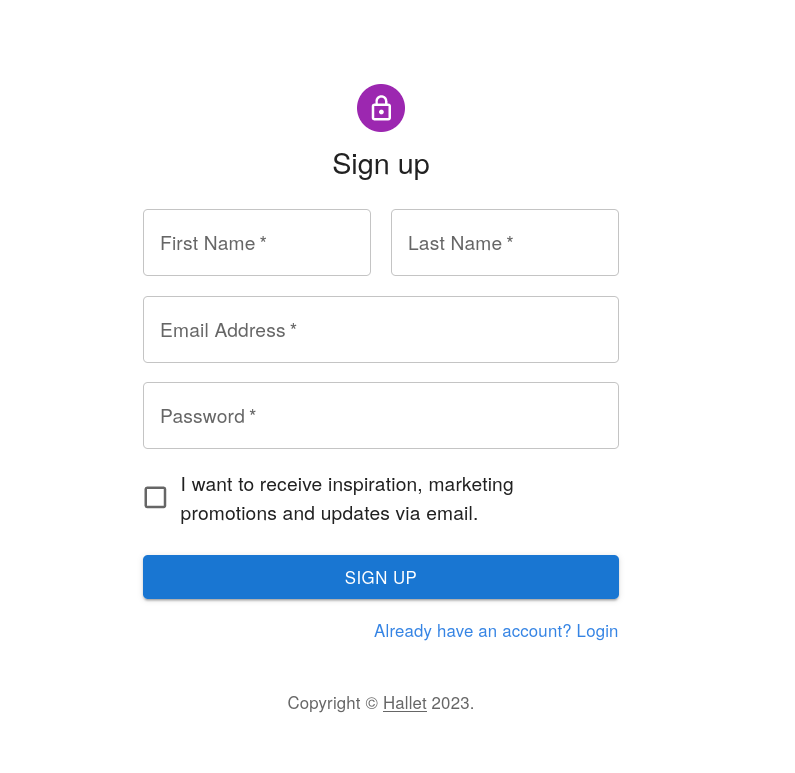


9.3.7 JobDoerRequests Table

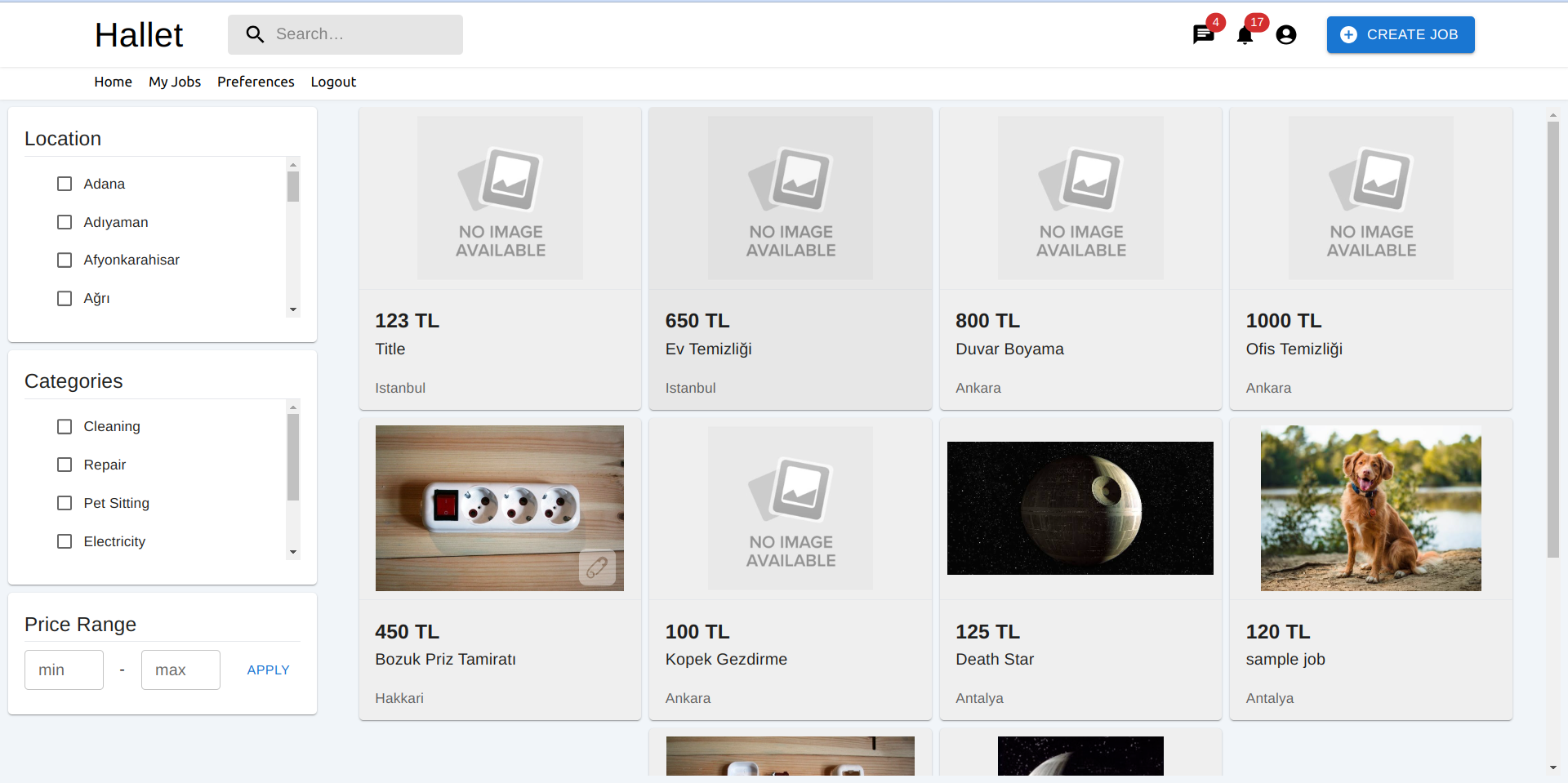


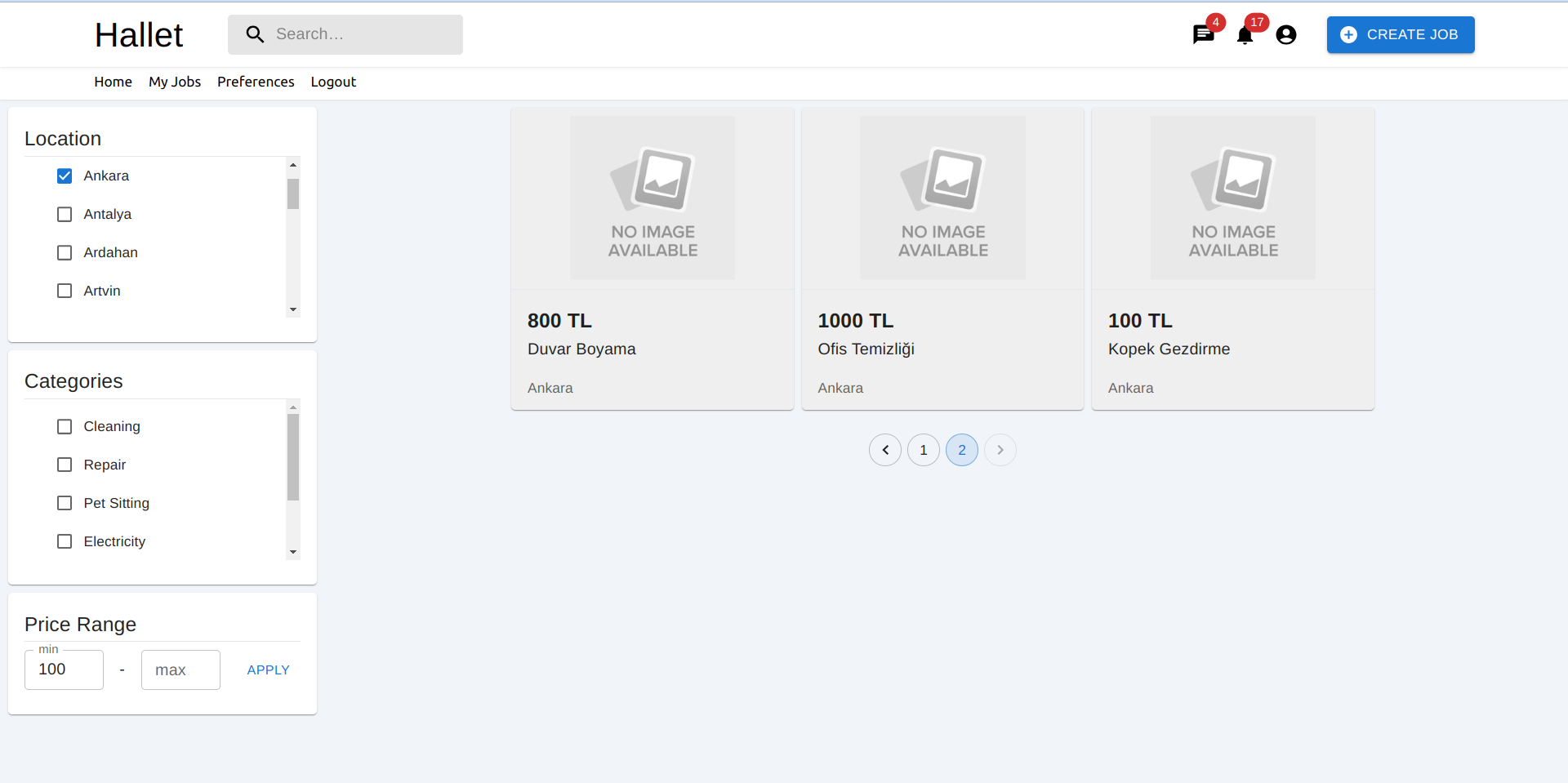
### 9.4 Web Screens

9.4.1 Login and Signup Screens

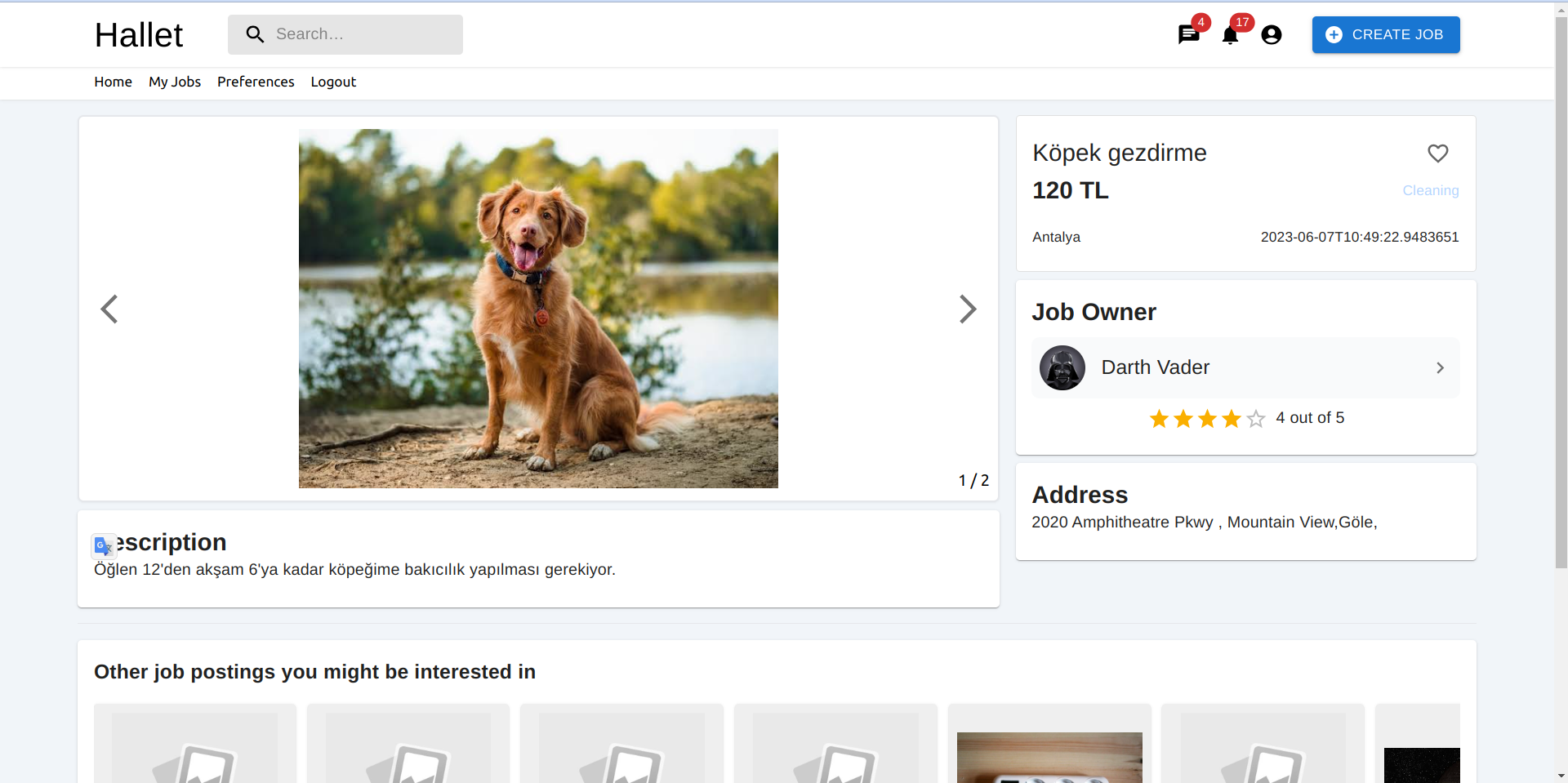


9.4.2 Home Page

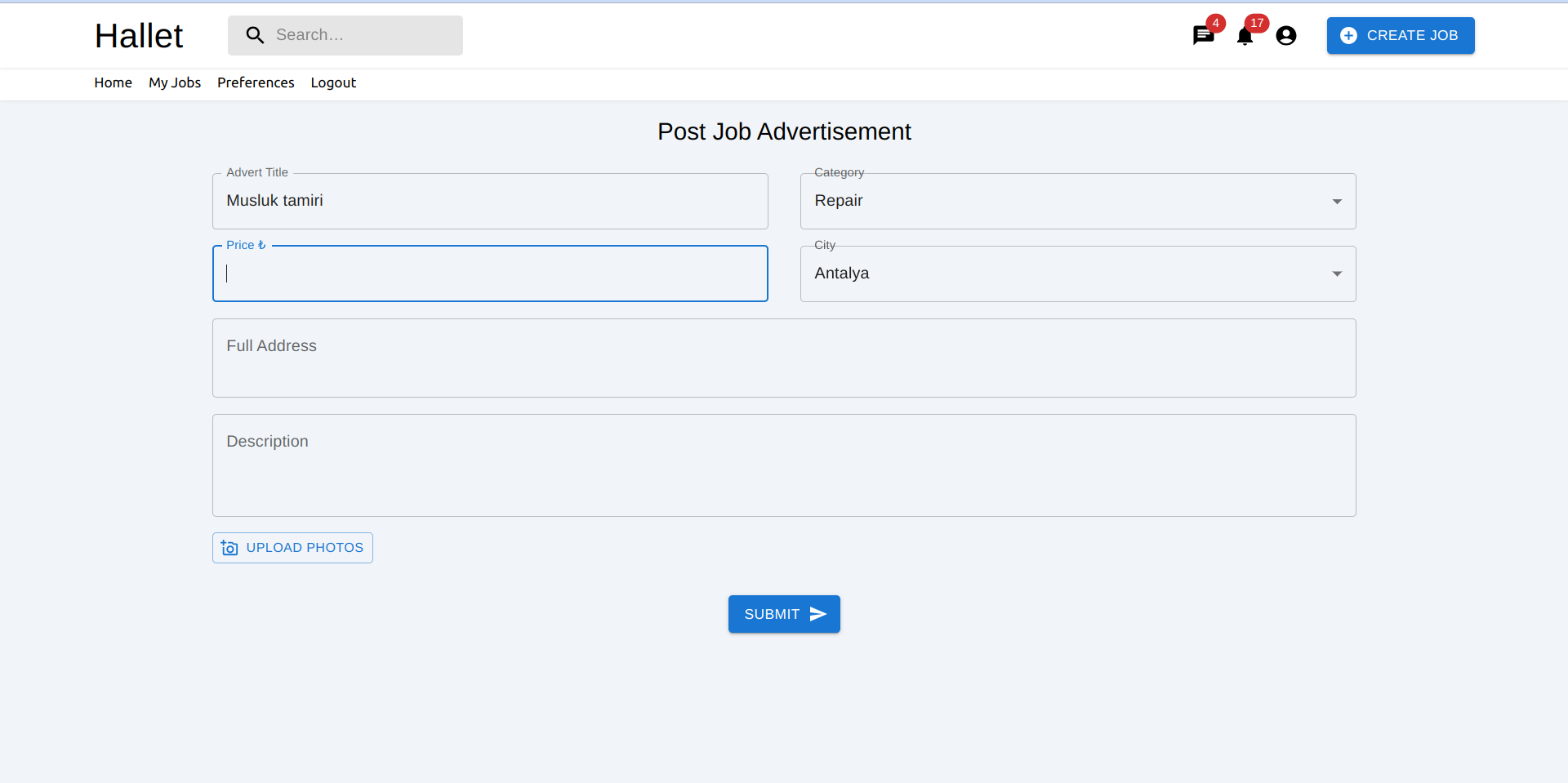




9.4.3 Job Detail Page



9.4.4 Create new Job Page



9.4.5 User Profile Page



# Acknowledgment

We would like to express our appreciation to the following individuals for their valuable assistance in the development of this software:

* Assoc. Prof. Ümit Deniz ULUŞAR : For their expertise and guidance throughout the project.
* Taha Yiğit Alkan : For their extensive testing and helpful feedback.

# References

Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.

12 Appendix A: Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Administrators | The individuals or groups of individuals who are responsible for managing and overseeing a particular system, organization, or network. |
| Clean Architecture | A software design philosophy that emphasizes the separation of concerns and the independence of the business logic from the technical details of the implementation. |
| Database | Collection of all the information monitored by this system. |
| Editor | Person who receives articles, sends articles for review and makes final judgments for publications. |
| Field | A cell within a form. |
| Job Owner | The user who creates an advertisement to get his/her job done. |
| Software Requirements Specification | A document that completely describes all the functions of a proposed system and the constraints under which it must operate. For example, this document. |
| Stakeholder | Any person with an interest in the project who is not a developer. |
| StreamAPI | A feature in Java programming language that was introduced in Java 8. It is a powerful tool for processing collections of objects in a functional and declarative way. |
| User | Any person who uses the application. |
| Worker / Job Doer | The user who gets the job done by applying for the job in the advertisement. |