

FixMyHome

Name: Mintas Alex Andrei

Group: 30236

Table of Contents

[Deliverable 1 3](#_Toc64843130)

[Project Specification 3](#_Toc64843131)

[Functional Requirements 3](#_Toc64843132)

[Use Case Model 3](#_Toc64843133)

[Use Cases Identification 3](#_Toc64843134)

[UML Use Case Diagrams 3](#_Toc64843135)

[Supplementary Specification 3](#_Toc64843136)

[Non-functional Requirements 3](#_Toc64843137)

[Design Constraints 3](#_Toc64843138)

[Glossary 3](#_Toc64843139)

[Deliverable 2 3](#_Toc64843140)

[Domain Model 3](#_Toc64843141)

[Architectural Design 4](#_Toc64843142)

[Conceptual Architecture 4](#_Toc64843143)

[Package Design 4](#_Toc64843144)

[Component and Deployment Diagram 4](#_Toc64843145)

[Deliverable 3 4](#_Toc64843146)

[Design Model 4](#_Toc64843147)

[Dynamic Behavior 4](#_Toc64843148)

[Class Diagram 4](#_Toc64843149)

[Data Model 4](#_Toc64843150)

[System Testing 4](#_Toc64843151)

[Future Improvements 4](#_Toc64843152)

[Conclusion 4](#_Toc64843153)

[Bibliography 4](#_Toc64843154)

# Deliverable 1

## Project Specification

The (Website Name) website is intended to be used by customers in need of a specific job to be done at their house/apartment by a worker and also by workers in need of a customer for their work. The fields of work that I thought about are: Tile Setter, Painter/Decorator, Locksmith, Electrician, Plumber and more. The (Website Name) primary objective is to facilitate the process of finding a good worker for your desired job and also finding a good customer for each worker so that this interaction is no more fulfilled by scams and displeasures.

[Present the project specification.]

## Functional Requirements

First of all, the user is given the possibility of registration and login because in order to use all the facilities of the website and to perform actions on it you must have a valid account. However, the administrator already has an account and he has a wider control area in which he can operate. There are 3 types of users:

The Customer can register, login, update customer details, find workers, request workers for different jobs, rate the worker after the job is done (or not).

The Worker can register, login, update worker details, find customers, request to do a job for a customer, rate the customers after the job is done (or not).

The Administrator can perform modifications related to the databases, he is a super user.

Before any data is stored into the database, it must be verified to meet the required standards in order to have a easy and clean understanding of it. This verification is done by multiple validators written with great care such that the database cannot be updated until every field of the data has been validated.

## Use Case Model 1

### Use Cases Identification

Use-Case: Login

Level: Subfunction

Primary Actor: User

Main success scenario: Login successful

Use-Case: Register

Level: User Goal

Primary Actor: Customer, Worker

Main success scenario: Register successful

Use-Case: Add/Create User

Level: Subfunction

Primary Actor: User

Main success scenario: User successfully added/created

Use-Case: Update User

Level: Subfunction

Primary Actor: User

Main success scenario: User successfully updated

Use-Case: Delete User

Level: Subfunction

Primary Actor: User

Main success scenario: User successfully deleted

Use-Case: Find Users

Level: Subfunction

Primary Actor: User

Main success scenario: List of users successfully returned

### UML Use Case DiagramsO imagine care conține diagramă, schematic Descriere generată automat

## Supplementary Specification

### Non-functional Requirements

Reliability is an important non-functional requirement that refers to a system’s ability to perform its intended functions consistently, without errors or failures. In my project, I took this into consideration by rigorously testing each component in multiple scenarios. This approach ensured that the system not only performed its intended functions successfully, but also gracefully handled cases where unexpected data or requests were encountered. To achieve this, I created specific exceptions within the application and implemented control flows to handle them effectively. (Reliability)

To enhance usability, my application will feature an intuitive user interface that addresses users with varying levels of technical expertise. Users can interact with the application through intuitive buttons, structured forms, and informative content. The application provides users with suggestive buttons that are labeled clearly, which enables them to generate requests and view relevant information with a single click. Using this approach, the application guarantees an effortless user experience. (Usability)

The application was designed in such a way that it can be organized in layers, and a higher layer depends on a lower layer through abstraction. This design aims to minimize dependencies between layers, ensuring that changes to one layer will not require significant modifications to the entire system. As a result, the application's maintainability should not be a concern, provided that the system was structured and modularized appropriately. (Maintainability)

Performance is an important non-functional requirement so the system should be designed to handle a large volume of concurrent requests without slowing down or crashing. This includes optimizing database queries, minimizing response times, and ensuring efficient use of system resources. (Performance)

[Choose 4 NF for your system, describe them and explain why these NF are suitable for your implementation. ]

### Design Constraints

The system is built using the Spring framework, which is a widely used and popular framework for developing Java-based web applications. This framework offers a set of tools and libraries that help with the development of web applications, such as the ability to handle HTTP requests and responses, manage database connections, and more. On the data storage side I worked with MySQL.

My system follows the Model-View-Controller (MVC) architectural pattern, which separates the concerns of the application into three distinct components: the model (data and business logic), the view (presentation layer), and the controller (handles requests and manages communication between the model and view). This pattern ensures that the system is well-structured.

Also, within the application I used some external libraries in order to accelerate the development process and provide additional functionality. In order to map the database tables to Java and also to perform database operations in an efficient way, I used Hibernate and JPARepository. Lombok is another library which reduces the time and effort required for development by providing annotations to reduce boilerplate code in the source code.

[This section needs to indicate any design constraints on the system being built. Design constraints represent design decisions that have been mandated and must be adhered to. Examples include software languages, software process requirements, prescribed use of developmental tools, architectural and design constraints, purchased components, class libraries, and so on.]

## Glossary

1. Administrator: A user with special privileges who is responsible for managing the database, including adding, updating, and deleting users.

2. Customer: A user with normal privileges who is searching for a suitable worker to do a desired job/work.

3. Worker: A user with normal privileges who waits to be contacted by a customer or he can as well be searching for a customer to work for.

4. CRUD operations: Create, Read, Update, and Delete operations that can be performed on the database.

5. Model-View-Controller (MVC): A software design pattern that separates an application into three distinct components: the model, the view, and the controller.

6. Subfunction: A use case that is a part of a larger use case and helps to achieve the main goal of that use case.

7. UML Use Case Diagram: A graphical representation of the use cases in a system and their relationships with other system components.

8. User interface: The graphical user interface (GUI) through which the user interacts with the application.

[Present the noteworthy terms and their definition, format and validation rules if appropriate.]

# Deliverable 2

## Domain Model

1. Name: User

Attributes: id | firstName | lastName | email | phoneNumber | age

Associations: abstract parent class

1. Name: Customer

Attributes: workers | [User attributes]

Associations: a customer

1. Name: Worker

Attributes: job | customer | [User attributes]

Associations: a worker

## Architectural Design

### Conceptual Architecture

O imagine care conține diagramă

Descriere generată automat

The architectural style used for the system is REST (Representational State Transfer) and the architectural pattern is MVC (Model-View-Controller).

The REST architectural style is well suited for designing web-based systems that are scalable, flexible, and easy to integrate with other systems and services. REST defines a set f constraints and principles for creating stateless, client-server systems that communicate using a uniform interface.

The MVC pattern is a widely used pattern in web development that separates the application into three main components: the Model, the View and the Controller. The Model represents the data and the business logic of the application, the View represents the user interface, and the Controller handles user input and updates the Model and View accordingly.

In the case of my application, the REST architecture is used to create a set of resources that can be accessed and manipulated using HTTP requests. For example, the system has resources for tracking the progress made by users. Clients can access these resources using HTTP requests, and the server can respond with the appropriate representations of the resources.

### Package Design

O imagine care conține diagramă

Descriere generată automat

### O imagine care conține diagramă Descriere generată automatComponent and Deployment Diagram

# Deliverable 3

## Design Model

### Dynamic Behaviour

The FixMyHome platform offers a dynamic and interactive experience for both homeowners and handymen, facilitating seamless communication and efficient job management. This section outlines the key features and interactions that users can expect while using the platform.

1. User Registration and Authentication:

Homeowners can easily register on the platform by providing their necessary information, including name, contact details, and preferred handyman category.

The platform employs robust authentication mechanisms to ensure secure access to user accounts, safeguarding personal information and maintaining user privacy.

2. Homeowner Actions:

Homeowners can effortlessly post job requests, specifying the type of service required, such as electrical, plumbing, tiling, painting, or locksmith.

They have the option to provide additional details about the job, including the location, preferred time slots, and any specific instructions.

Homeowners can browse through profiles of available handymen, evaluating factors such as skills, experience, ratings, and reviews to make informed decisions.

3. Handyman Actions:

Handymen can view a list of available job requests and express their interest in specific jobs based on their expertise and availability.

They have the ability to update their availability and indicate their preferred service areas, ensuring efficient job allocation and utilization of resources.

Handymen can communicate directly with homeowners through in-app messaging, allowing them to gather more information about the job or negotiate terms before accepting a job.

4. Job Matching and Acceptance:

The platform utilizes sophisticated algorithms to match homeowners' job requests with suitable handymen, taking into account factors such as location, availability, and skillset.

Handymen receive notifications of matched job requests, enabling them to review the job details and decide whether to accept or decline the job.

Homeowners are promptly notified when a handyman accepts their job request, ensuring transparent and timely communication.

5. In-App Communication:

FixMyHome provides a built-in messaging functionality that allows homeowners and handymen to communicate directly within the app.

This feature facilitates effective communication, enabling users to exchange important information, clarify job requirements, and address any concerns or queries.

6. Real-Time Updates and Notifications:

Users receive real-time updates and notifications regarding job status changes, such as job acceptance, job completion, or changes in availability.

Notifications are delivered through various channels, including in-app notifications, email, or SMS, ensuring that users stay informed throughout the job lifecycle.

7. Job Completion and Feedback:

Upon job completion, homeowners confirm the successful completion and may provide feedback and ratings for the handyman's services.

Handymen have the opportunity to respond to feedback, fostering open and constructive communication between homeowners and handymen.

The feedback and ratings contribute to the handyman's profile reputation, helping future homeowners make informed decisions.

By incorporating these dynamic features and interactions, FixMyHome streamlines the process of connecting homeowners with reliable handymen, fostering transparency, and delivering a user-friendly experience for all parties involved.

### Class Diagram

O imagine care conține diagramă

Descriere generată automat

## Data Model

Entities:

1. User – represents superclass for the Worker and Customer classes.

* First Name
* Last Name
* Email
* Phone Number
* Age

1. Worker – extends the User class and represents the part that offers services.

* Job
* Customer - @ManyToOne relation

1. Customer – extends the User class and represents the part that hires services

* Workers - @OneToMany relation

1. Response – represents a helper class

* timestamp
* statusCode
* status
* reason
* message
* developerMessage
* data

# System Testing

Unit Testing is a crucial software testing technique that focuses on examining the behavior of individual units or components within a software system. Its primary objective is to ensure that each unit of code functions correctly according to its expected behavior. Unit Testing is an integral part of the development process, specifically during the coding phase, and is typically carried out by the developers themselves.

The essence of Unit Testing lies in isolating a specific section of code and thoroughly evaluating its accuracy. These units can take various forms, such as individual functions, methods, procedures, modules, or objects. By subjecting each unit to rigorous testing, developers can verify its functionality in isolation, thereby gaining confidence in its reliability and identifying any potential flaws or errors.

Through Unit Testing, developers can catch bugs early in the development cycle, minimizing the risk of more complex issues arising during integration or system testing. By validating the behavior of each unit individually, developers can ensure that the software's building blocks function as intended, fostering stability and robustness throughout the entire system.

In summary, Unit Testing serves as a critical quality assurance measure by thoroughly examining the performance of individual units or components within software. By conducting these tests during the coding phase, developers can detect and rectify issues early, leading to enhanced software reliability and a smoother development process.

# Future Improvements

1. Enhanced User Matching: Implement an intelligent algorithm that utilizes user preferences and requirements to improve the matching process between homeowners and handymen. Consider factors such as location, availability, skillset, and customer ratings to ensure optimal matches and increase customer satisfaction.

2. Real-Time Availability Tracking: Introduce a feature that allows handymen to update their availability in real-time, providing homeowners with accurate and up-to-date information on the handymen's schedule. This feature would help homeowners find handymen who are immediately available and reduce waiting times.

3. In-App Chat and Video Calls: Enable a built-in communication platform within the FixMyHome app, allowing homeowners and handymen to communicate directly through instant messaging and video calls. This feature would facilitate efficient and seamless communication, enabling handymen to understand the problem better and homeowners to convey their requirements effectively.

4. Secure Payment Integration: Integrate a secure payment gateway within the app, allowing homeowners to make hassle-free payments to handymen for their services. Implement features such as digital invoices, multiple payment methods, and transaction history to ensure transparency and convenience.

5. Service Quality Assurance: Implement a rating and review system where homeowners can provide feedback on the quality of service provided by handymen. This feedback would help homeowners make informed decisions while choosing handymen and encourage handymen to consistently deliver excellent service.

6. Expanded Handyman Profiles: Enhance handyman profiles by including detailed information about their skills, certifications, years of experience, and customer reviews. This additional information would enable homeowners to make more informed decisions when selecting a handyman, fostering trust and confidence in the platform.

7. Service Package Options: Introduce customizable service packages that homeowners can choose from based on their specific needs. For example, a homeowner could select a "Bathroom Renovation Package" that includes a plumber, tile setter, and painter, streamlining the process of hiring multiple handymen for a project.

8. Proactive Maintenance Reminders: Implement a feature that sends automated reminders to homeowners for regular maintenance tasks such as HVAC servicing, plumbing check-ups, or electrical inspections. These reminders would help homeowners stay on top of their home maintenance needs and provide handymen with a consistent stream of service requests.

9. Loyalty Program: Introduce a loyalty program to reward frequent users of the FixMyHome app. Offer benefits such as exclusive discounts, priority booking, or additional service guarantees to incentivize homeowners to continue using the platform and build customer loyalty.

10. Integration with Home Automation Systems: Explore partnerships or integrations with home automation platforms to allow homeowners to seamlessly connect with handymen for assistance with smart home installations, troubleshooting, or repairs. This integration would cater to the growing demand for smart home services and enhance the overall value proposition of FixMyHome.

# Conclusion

In conclusion, the integration of Java Spring Boot and Angular technologies has resulted in an innovative and powerful platform that brings workers and customers together. By bridging the gap between service providers in fields such as Locksmithing, Tile Setting, Painting, Electrical, and Plumbing, and potential customers who reside in various types of dwellings, the application has streamlined the process of finding and hiring for specific jobs. The functionality that allows workers to actively search for customers introduces a new dynamic into the traditional service hiring model, further boosting the efficiency of this marketplace. Overall, this application stands as a testament to the potential of contemporary software development technologies in enhancing user experience, fostering connections, and simplifying the process of hiring professionals for home services. It signifies a major step forward in digital marketplaces, transforming how we find, hire, and interact with skilled workers in our communities.

# Bibliography

<https://support.bigcommerce.com/s/question/0D51B00005YmubDSAR/can-i-edit-the-htmlcss-to-make-it-show-when-a-product-is-out-of-stock?language=en_US>

<https://www.google.com/search?q=how+to+disable+color+for+button+in+disabled&oq=how+to+disable+color+for+button+in+disabled&aqs=chrome..69i57j33i160l2.12672j0j7&sourceid=chrome&ie=UTF-8>

<https://www.google.com/search?q=how+to+set+disable+button+color&oq=how+to+set+disable+button+color+&aqs=chrome..69i57j33l2j33i160l3.10614j0j7&sourceid=chrome&ie=UTF-8>

<https://www.databasestar.com/sql-boolean-data-type/>

<https://www.google.com/search?q=how+to+get+value+from+Subject%3Cnumber%3E&oq=how+to+get+value+from+Subject%3Cnumber%3E&aqs=chrome..69i57j33i160l2.9313j0j4&sourceid=chrome&ie=UTF-8>

<https://www.geeksforgeeks.org/how-to-include-one-css-file-in-another/>

<https://stackoverflow.com/questions/9433448/html-if-statement>

<https://www.tutorialsteacher.com/typescript/typescript-if-else>

<https://www.reddit.com/r/TheInsaneApp/comments/12dmiqj/typescript_project_with_source_code_build_a_heart/>

<https://www.reddit.com/r/Angular2/comments/g9nbpz/adding_add_to_favorite_button/>

<https://stackblitz.com/edit/favorite-component-example>

<https://stackoverflow.com/questions/10853763/how-to-make-td-cell-from-html-table-expand-for-the-row-to-fit-in-one-line>

<https://www.google.com/search?q=how+to+make+a+pop+up+windows+for+editing+product+in+html&oq=how+to+make+a+pop+up+windows+for+editing+product+in+html&aqs=chrome..69i57j33i160.11312j0j7&sourceid=chrome&ie=UTF-8>