Find a Master

Requirements Determination and Use Case Analysis / System Proposal / Analysis Phase   
(Homework No.2)

Project team: Team 01

Instructor: Dr. Araz Yusubov

Submitted in partial fulfillment of the requirements of the INFT 2303: Systems Analysis and Design course project

|  |  |
| --- | --- |
| GitHub repository | https://github.com/ADA-SITE-INFT2303-2022-Spring/systems-development-project-team-01 |
| Version date | Version information |
| 06.03.2022 | Initial draft |
| 26.03.2022 | Final version |

|  |  |
| --- | --- |
| Other documents in the package | |
| Find a Master-Requirements (1) | We added all parts of this assignment in one file for more convenience |
|  |  |
|  |  |

|  |  |  |
| --- | --- | --- |
| Team member | Contribution to this homework (NOT the project) | Estimated % |
| Aytaj Najafova | Participate actively, always on time, introduction part, functional and non-functional requirements design constraints and some user story descriptions done by her. Also, she found reliable sources for reading and improving our project which we added in references. | 25% |
| Unal Imanov | Brainstorming in use case analysis, use case 3 and 4 done by his help. Responsible and always on time. | 25% |
| Faraz Baghernezhad | Participate actively, always on time. Brainstorming in use case analysis use case 1 and 2 done by him. Also, he found reliable sources for reading and improving our project which we added in references. | 25% |
| Arzuman Alakbarli> | Brainstorming in use case analysis. External actor descriptions and some user story descriptions done by him. Responsible and always on time. | 25% |

# Table of Contents

Table of Contents

[Table of Contents 3](#_Toc99229671)

[1. Introduction 3](#_Toc99229672)

[Definitions 3](#_Toc99229673)

[2. Requirements Definition 3](#_Toc99229674)

[Functional Requirements 3](#_Toc99229675)

[Nonfunctional Requirements 4](#_Toc99229676)

[Design Constraints 4](#_Toc99229677)

[3. Use Case Analysis 4](#_Toc99229678)

[External Actor Descriptions 4](#_Toc99229679)

[User Story Descriptions 5](#_Toc99229680)

[Use Case Diagram 6](#_Toc99229681)

[Use Case Descriptions 7](#_Toc99229682)

[Use Case 1 7](#_Toc99229683)

[Use Case 2 8](#_Toc99229684)

[Use Case 3 9](#_Toc99229685)

[Use Case 4 9](#_Toc99229686)

[References 10](#_Toc99229687)

# Introduction

This is part of the System Proposal for a hypothetical project “**Find a Master** “ submitted for partial fulfillment of the requirements of the Systems Analysis and Design course in the School of Information Technologies and Engineering at ADA University, Baku, Azerbaijan.

This document is designed for defining requirements and creating use cases by adding specific information like; **Actor, planned release, triggered actions, Main (Success) Flow etc.**

## Definitions

|  |  |
| --- | --- |
| Term | Definition |
| MVP | The simplest version of a product that may still be deployed is known as a minimal viable product (MVP) |
| AWS | |  | | --- | | Amazon Web Services, database for storing data | |

# Requirements Definition

## Functional Requirements

**1.Master-related requirements:**

1.1 The system shall permit repairers to sign up by entering their name, surname, address, card and contact information, and specialization.

1.2 T The system shall permit the repairer to see the description of the problem, location, date, and time of the request by customers.

1.3 The system shall permit the repairers to approve or reject the request.

1.4 The system should save the reserved date and time ranges on the timesheet to detect the available hours of master.

1.5 The system shall permit the repairer to receive the payment after completing the request.

1.6 The system should allow the repairer to rate and write any comment about the customer if he has any.

1.7 The system may give an opportunity of live chat to master for advising how the work should be done in case of simple tasks.

1.8 The system may send a notification to the repairer in case of receiving requests, messages, and feedback from customers

**2. Customer-related requirements**

2.1 The system shall permit customers to sign up by adding their name, surname, contact and card information.

2.2 The system shall permit the customers to create problem description by entering exact time range, location, and send request for master.

2.3 The system should detect and present the available hours of master.

2.4 The system shall permit the customers to make payments over the system.

2.5 The system should give an opportunity to customers for a rating and write feedback about the performance of master.

2.6 The system may allow the customers to use live chat with the repair person to get guidance in an online environment in case of simple tasks.

2.7 The system may send a notification to the customer in case of approval or rejection, or comment and message by repairer.

## Nonfunctional Requirements

**1.Operational**

1.1 The system shall make the application accessible for IOS and Android users.

**2.Performance**

2.1 The system shall be able to perform with data of more than 500,000 people.

2.2 The system shall be active continuously for 24 hours.

**3.Security**

3.1 The system shall request the users to confirm the privacy agreement before the registration.

3.2 The system shall encrypt the card and personal data of users.

3.3The system may have a fingerprint security option for logging in to the application.

## Design Constraints

Design constraints - If we consider that this system is a mobile application, designers will face different constraints.

Firstly, budget is the main constraint because not every project has an unlimited budget. 15k AZN is allocated for the design expenses and this limit should be considered.

Secondly, the mobile app will require a compact and responsive design to easily locate all details on the screen. Additionally, the timeline should be taken into consideration as each page should have its own priority.

# Use Case Analysis

## External Actor Descriptions

|  |  |
| --- | --- |
| Actor | Role Description |
| Customer | Customer will add the information about items and how problem happens and add address and request the repair person for the specified time. |
| Repair Man | Repair man will add his specialization and address. And he will be able to accept or reject the order within given time. |
| Operators of our System | Operators will analyze the request from customer and direct it to the relevant repair man. (In the MVP this role planned but for future we plan to automate this part, too) |
| Database System, AWS cloud | This is the actor where we store all data that operate over our system |
| Notification System | This is the actor for notifying all users according to their processes proceed on the system. There are two options; 1. Notification with triggered action configured beforehand by development team, 2. Manual notification which can be sent by admins anytime in any topic. |

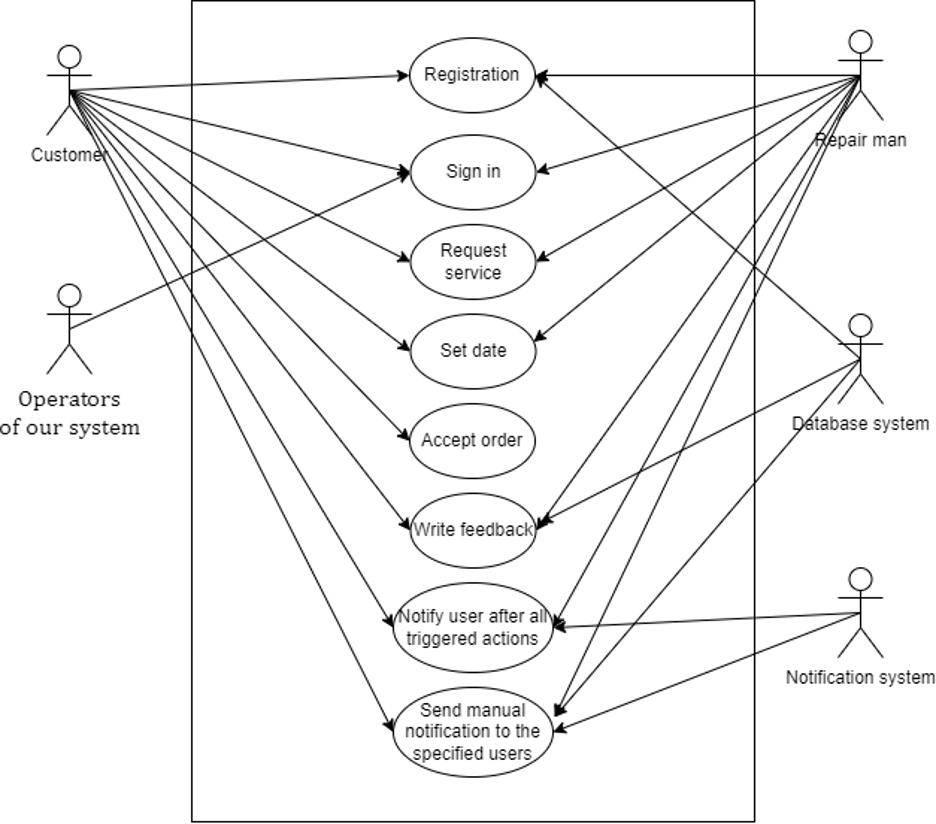
## 

## User Story Descriptions

|  |  |  |
| --- | --- | --- |
| User story name | Description | Release |
|  |  |  |
| Customer Registration | Customers sign up to become a user of “Find Master” application by adding his personal information including name, surname, address, contact information, card information. | R1 |
| Repair man Registration | Repair man sign up as Master to become a user of “Find Master” application for selling his/her services by adding his personal information including name, surname, address, contact information, card information, specialization. | R1 |
| Request for master | Customers add problem description and send the request for calling Master. | R1 |
| Master accepts the request from customer. | Repair man can view the request details; problem description, date, location and the either accept or reject the request. | R1 |
| Operating payment over system | Customers will be able to add bank accounts and make payment over the system. After repair man complete his work, he will be able to get his payment over system. | R1 |
| Set availability for Masters | Master set the time of availability on the system so that system detect she/he is available for request or not. | R2 |
| Feedback and rating for all users | After completing the operation, the customer can write detailed feedback about the result of master’s work and review them over system. In addition, masters can do the same for customers. | R2 |
| Live chat between customer and repair man. | Sometimes repair man does not need to go offline location and can repair problem by advising what to do. For this reason, there will be paid plan for advice to repair man. | R3 |
| Sending manual notification | Admin or operations with the send notification permission can write a notification message, choose the auditory {Customer, repair man} and send time. | R3 |

### 

### Use Case Diagram



## Use Case Descriptions

### Use Case 1

|  |  |
| --- | --- |
| **Use Case Number:** | UC-01 |
| **Use Case Name:** | Customer Registration |
| **Actor(s):** | Customer, Repair, Database which store the registered data. |
| **Description:** | Users sign up to become a user of “Find Master” application by adding his personal information including name, surname, address, contact information, card information. |
| **Priority (Release)** | R1 |
| **Trigger:** | External - Registrant registering as a customer or repair man. |
| **Pre-condition(s):** | * Application has been setup and configured. * Application is running and open for registrations. * User has accessed to the system via application. |
| **Main (Success) Flow:** | 1. Registrants add phone number to register. 2. Registrant request verification code 3. System sends verification code to number. 4. Users verify phone number by adding code.  * If code is not correct, system shows error message Return to step 2  1. System asks to fill required information 2. Users fill all required information. 3. System request to add valid password  * If password is not strong, system shows example for strong password  1. System verifies password and all information.  * If there is problem while verification, system shows appropriate error message Return to step 1  1. System shows success message of registration to user |
| **Alternate Flows:** | Alternate Flow #1: After Step 4 in success scenario System will display the option to Cancel the registration process. The following steps would occur:   1. Users will be able to cancel registration and reject to fill in information. 2. System asks; “Are you sure to cancel” 3. Users approve the reject 4. User directed to the first screen. |
| **Post Condition:** | 1. User did not complete all requirements information, so system does not complete registration and does not store registration data too. 2. The system has errors while registration, so the process is not complete, and the system does not store registration data. |
| **Requirements:** | 1 – The user shell register with personal number  2 – The user shell be able to cancel registration during the process,  3 – The user shell be able to add bank accounts – The system verifies the registration process and shows success message. |

### Use Case 2

|  |  |
| --- | --- |
| **Use Case Number:** | UC-02 |
| **Use Case Name:** | Request for Master |
| **Actor(s):** | Customer |
| **Description:** | Customers add problem description, date and location and request for master |
| **Priority (Release)** | R1 |
| **Trigger:** | External – Customer request a repair man over system |
| **Pre-condition(s):** | - Customer logged in to system.  - Customers add description of problem, location, and meeting time |
| **Main (Success) Flow:** | 1. Customers choose items with problem. 2. Customers describe problem and how it happens. 3. Customers choose location 4. Customers choose suitable time 5. Customer request for master.  * If any field is not filled appropriate message will be shown and customer will back to request page |
| **Alternate Flows:** | Alternate Flow #1: After Step 4 in success scenario System will display the option to Cancel the order option. The following steps would occur:   1. Customer will choose request. 2. Customer click cancel order. 3. System request to write cancel reason. 4. Customers add reason. 5. System delete the order |
| **Post Condition:** | Users choose the past time on the calendar; system did not complete the order and do not store order to database. |
| **Requirements:** | - User shell specify item which has problem  - User shell delete the order  - User shell be able to set time and location. |

### Use Case 3

|  |  |
| --- | --- |
| **Use Case Number:** | UC-03 |
| **Use Case Name:** | |  | | --- | | Master accepts/rejects the request from customer. &Set availability time | |
| **Actor(s):** | Repair man. |
| **Description:** | Repair man read the problem, see the time and location then accept/reject order |
| **Priority (Release)** | R1 |
| **Trigger:** | External – Master accepts order.  External- Master reject order. |
| **Pre-condition(s):** | * Master logged in to system. * Master view order description. |
| **Main (Success) Flow:** | 1. Master gets notification about new order 2. Master read order description 3. Master clicks the accept/reject order.  * If time is not suitable, set new time and request customer to approve. * If master reject order, he/she must write the reason of rejection. |
| **Alternate Flows:** | Alternate Flow #1: After Step 2 System will display the option to ask for another time from customer. The following steps would occur:   1. Master clicks request another time. 2. Master chooses new time. 3. Master sends new time to customer approve. 4. Success message of request displays on screen. |
| **Post Condition:** | 1. Customers choose the past time on the calendar; system did not complete the order and do not store order to database. |
| **Requirements:** | 1. Repair man can view order 2. Repair man can request time change 3. Repair man can either accept or reject order 4. Repair man should write reject reason |

### Use Case 4

|  |  |
| --- | --- |
| **Use Case Number:** | UC-04 |
| **Use Case Name:** | Operating payment over system |
| **Actor(s):** | System, Customer. |
| **Description:** | Customer will be able to add bank accounts and make payment over system. After repair man complete his work, he will be able to get his payment over system. |
| **Priority (Release)** | R2 |
| **Trigger:** | External – User send payment.  External – System keep payment on own account until operation complete status, then transfer to Master account |
| **Pre-condition(s):** | * Customer set order status as completed |
| **Main (Success) Flow:** | 1. Customer request order. 2. System transfer money from customer account to system account. 3. Customer set order status as completed.  * If order is not completed, user add reason of not completing and request payment back  1. System transfer money to Master account. |
| **Alternate Flows:** | Alternate Flow #1: After Step 3 System will display the option to customer to change status of work as incomplete and request money back within 30days.  Users click to change status of order to incomplete.  System asks the reason.  Users add reason and request back.  System displays the message of, “Your order will be analyzed, and we will back to you”.  Alternate Flow #2: After Step 4 System will put limit master account to take money from account within 30 days.  Repair man request to take money.  System shows message of, “Without completing 30 days of your service, you cannot take money from your account”. |
| **Post Condition:** |  |
| **Requirements:** | 1. Order status should be complete 2. Customer and master accounts on system should be active. |

These are use cases for our MVP and Release 1. Remining will be developed further.

# References

1. <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/how-to-identify-actors/>

2. <https://www.techopedia.com/definition/27809/minimum-viable-product-mvp>

3. <https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/>