Software Requirements Specification

for

PostOffice

Version 1.0

Prepared by Ihor Huralskyy,

Taras Kizlo, Mariia Kunyk, Nazarii Tymtsiv

14 March, 2021

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the PostOffice system. It will explain the purpose and features of the system, what the system will do and the constraints under which it must operate. This document is intended for both the stakeholders and the developers of the system.

1.2 Intended Audience and Reading Suggestions

This project is a prototype for the cargo management system and it is useful for companies that want to deliver their products to clients in the most simple way and as well it is useful for the clients, which can track their cargoes.

1.3 Project Scope

This software system will provide a unified service for postal companies where they can manage orders and services for post office clients who can check the order's status. This system will be designed to maximize the post office's managers' productivity by providing tools to assist them in creating orders and changing their statuses. Also, this system is designed to allow a client to track their orders and receive notifications about changes in its status. The system contains a database with Orders, Cargoes, and Events entities. Above all, we hope to provide a comfortable user experience along with the best pricing available.

2. Overall Description

2.1 Product Perspective

The PostOffice system will consist of an online registration form, an administrators' portal, a backend database to store and process user's information and order data. This system will be used to manage the information and status of the cargoes. The functionality provided by the mobile operators will be embedded into the application to notify users about changes in status of their orders by SMS.

2.2 Product Features

The system will have the following major features:
1. Creating order in 3 steps with all needed information by a post office employee.
2. Sending SMS with TTN number to an inputted phone number by post office employee when order is created.
3. Displaying all orders with valid TTN numbers and with their statuses for post office employees. 4. Managing orders by post office employees.
5. Getting TTN number in SMS by the client.
6. Displaying order timeline by TTN number got from SMS for the client.
7. Completing an order if it has already been delivered to the client.

8. Displaying saved by an employee or client changes dynamically.

2.3 User Classes and Characteristics

There are two types of users that interact with the system: administrators and clients. Each of these two types of users has a different use of the system so each of them has its own requirements. Clients can only use the application to see order timeline, order history and receive SMS notifications. Administrators will manage orders, for example creating orders, adding and removing cargoes, editing current order location, deleting orders, etc. They are managing the overall system so there is no incorrect information within it. The administrators can manage the information for each order.

2.4 Operating Environment

Operating environment for the PostOffice system is as listed below:

1.Two databases: Redis and MongoDB;
2.Client/server system;
3.Operating system: Windows;
4.Platform: .Net Core.

2.5 Assumptions and Dependencies

The system will be developed by using the .NET Core platform with C# language. Other dependencies will also include:

- Angular
- SignalR
- MongoDB
- Redis
- Azure Functions
- Docker

Angular framework will be used, so the system will work only as a native app without reloading the page. Another constraint is that the system will be available and usable in Chrome, Opera, Mozilla Firefox, and Safari browsers, but not in Internet Explorer. The system may also have an adaptive design for small screens (smartphones, tablets).

3. Requirement specifications

3.1 Functional Requirements

This section contains all functional requirements of this project. They describe how our system should work and behave, so these requirements are very important and must be implemented in full.

3.1.1 The Employee Functional Requirements

The list of the post office employee functional requirements descriptions:

1. The employee shall be able to see the 'Register Order' and 'View Order List' panels.

2. The employee shall create an order in 3 steps and every step requires the previous one to be filled.
3. The order step shall have “description” and “phone” fields.
4. The order step shall have Sender and Receiver location groups with the required “city” and “street” fields.
5. The cargo step shall have a minimum of one item.
6. The cargo item shall have width, height, and length number fields.
7. The employee shall add and remove the cargo item when creating an order.
8. The third creating order step shall contain a confirmation question with confirm button.
9. The system shall send SMS with TTN number to an inputted phone number in the first creating order step after confirmation in the third step.
10. The system shall display a page with information that the order has been successfully created.
11. The employee shall be able to see the order list via 'View Order List' panel.
12. The employee shall see a valid TTN number for each order.
13. The employee shall see each order status (New, Delivering, Delivered, Completed, Canceled).
14. The order status shall change when the employee changes the order’s current location, cancel the order, or the client completes the order.
15. The employee shall be able to delete an order from the system.
16. The employee shall be able to edit the current order location (city and street) to a new one.
17. The employee shall be able to cancel the order.

3.1.2 The Client Functional Requirements

The list of the post office client functional requirements descriptions:

1. The client shall get the TTN number in SMS.
2. The client shall be able to see 'View Order History' panel.
3. The client shall be able to see an order timeline by TTN number got from SMS.
4. The client shall be able to complete an order if it has already delivered to him.

3.2 Non-functional Requirements

This section contains all non-functional requirements of this project. They describe what our system is supposed to be, different system properties. They also concentrate on the user's expectations, so it is good to have them, although these requirements are non-mandatory.

3.2.1 Performance

The list of the performance requirements descriptions:

1. The application shall load and be usable within 2 seconds.
2. The system shall be able to support 1000 simultaneous users.
3. Response time for queries and updates shall be not over 500 ms.

3.2.2 Reliability

The list of the reliability requirements descriptions:

1. The system shall work even if errors have occurred.
2. The system shall be available 75% time per month.
3. Downtime after a critical failure shall not exceed 5 hours. The average time shall be 2-5 hours.

3.2.3 Usability

The list of the usability requirements descriptions:

1. The user shall see all his or other users' changes dynamically.
2. The system shall work as a native app without reloading the page.
3. The interface shall be easy to learn without a tutorial and allow users to accomplish their goals without errors.

3.2.4 Supportability

The list of the supportability requirements descriptions:

1. The system shall be available and usable in Google Chrome, Opera, Mozilla Firefox, and Safari browsers.

3.2.5 Maintainability

The list of the maintainability requirements descriptions:

1. The application shall use continuous integration so that features and bug fixes can be deployed quickly without downtime.
2. The system shall be easy for testing.

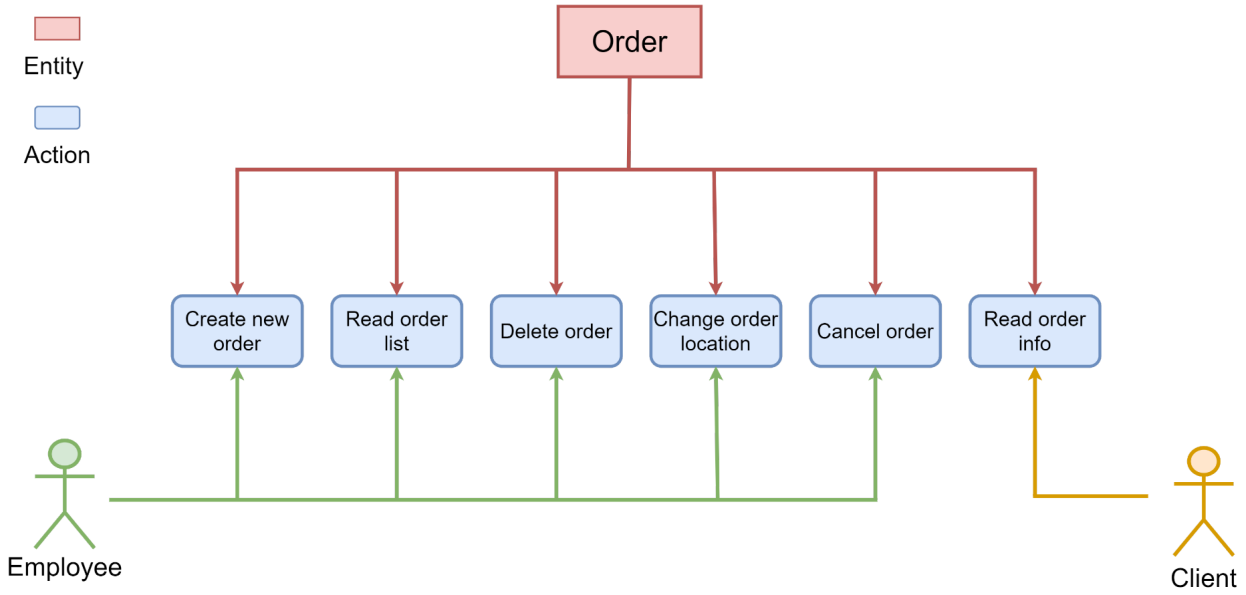
3.2.6 Security

The list of the security requirements descriptions:

1. Access permissions for application may only change by the system’s administrator.
2. Password requirements - length, special, characters, expiry.
3. All external communications between the server and clients must be encrypted.

4. Other Requirements

Use case diagram



This is a use case diagram of our project. As you can see we have two actors: Employee and Client. The domain area for both of them is Order.

User Stories and Acceptance Criteria

User story: As an employee
I want to be able to register a new order
So that order can be delivered

Acceptance criteria:

- An employee can only submit a form by filling in all required fields
- An employee shall create order in 3 steps and every step requires the previous one to be filled
- An employee shall enter sender and receiver location as a first step
- An employee shall add cargo as a second step
- An employee shall enter cargo width, height, and length
- An employee can go from the second step to the third if only they have added a cargo
- An employee shall confirm order creation as a third step
- An employee shall enter a user phone number
- User will receive a notification via SMS after successful registration
- Notification should contain TTN number

User story: As an employee
I want to be able to see all orders
So that I can find all orders

Acceptance criteria:

- An employee can see next order information: TTN, status, description and current location
- An employee can sort on next columns: TTN, status, description and current location

User story: As an employee
I want to be able to change order location
So that a user can track delivery progress

Acceptance criteria:

- An employee can only submit a form by filling in all required fields
- An employee should enter a new city and street
- Order status shall change to delivered when a new location is the same as the receiver location.
- The change of location must be added to the order history

User story: As an employee
I want to be able to delete an order
So that order no longer exist

Acceptance criteria:

- An employee should confirm deletion of an order
- The deletion must be added to the order history

User story: As a user
I want to be able to track order history
So that I can always know what happened with an order

Acceptance criteria:

- User should see order history only after he enters in TTN number
- User should see an empty result if he enters an invalid TTN number
- User shall see changes in order history dynamically