Vehicle Doorstep Service

Team Number: 6

Software Requirements Specification Document

Shikhar Pandey - 19BCE10036
Vedant Agrawal - 19BCE10057
Abhishek Srivastava - 19BCE10071
Amit Singh Boran - 19BCE10137
Ashutosh Kishan - 19BCE10200

Version: 1.0.0 Date: 08/27/2021

Table of Contents

1. Introduction
1.1 Purpose
1.2 Scope
1.3 Definitions, Acronyms, and Abbreviations
1.4 References
1.5 Overview

2. The Overall Description
2.1 Product Perspective
2.1.1 System Interfaces
2.1.2 Interfaces
2.1.3 Hardware Interfaces
2.1.4 Software Interfaces
2.1.5 Communications Interfaces
2.1.6 Memory Constraints
2.1.7 Operations
2.1.8 Site Adaptation Requirements
2.2 Product Functions
2.3 User Characteristics
2.4 Constraints
2.5 Assumptions and Dependencies
2.6 Apportioning of Requirements

3. Specific Requirements
3.1 External interfaces

3.2 Functions
3.3 Performance Requirements
3.4 Logical Database Requirements
3.5 Design Constraints
3.5.1 Standards Compliance
3.6 Software System Attributes
3.6.1 Reliability
3.6.2 Availability
3.6.3 Security
3.6.4 Maintainability
3.6.5 Portability
3.7 Organizing the Specific Requirements
3.7.1 System Mode
3.7.2 User Class
3.7.3 Objects
3.7.4 Feature
3.7.5 Stimulus
3.7.6 Response
3.7.7 Functional Hierarchy
3.8 Additional Comments

4. Change Management Process

5. Document Approvals

6. Supporting Information

1. Introduction

This document describes the structural properties and software requirements of the Doorstep Vehicle Repair project.

1.1 Purpose

The purpose of this document is to make the functional and non-functional requirements of the Doorstep Vehicle Repair project easy to comprehend. It also serves the purpose of making the functionality clear to end-users.

1.2 Scope

This SRS document applies to the initial version (release 1.0) of the “Doorstep Vehicle Repair System” software package. This document describes the modelling and requirement analysis of the system. The main aim of the system is to provide a set of protocols that allows users to create, update and delete a service/order from a nearby vehicle repair store. Users can order equipment that they require or can book service to get their vehicle repaired.

1.3 Definitions, Acronyms, and Abbreviations.

ABREVETIONS	DEFINITIONS
DVRS	Doorstep Vehicle Repair System
SM	Store Manager
SI	Store ID
SS	Service Supervisor
VIN	Vehicle Identity Number
DB	Database
UR	User's Request

1.4 References

- D. P. Gilliam, T. L. Wolfe, J. S. Sherif, and M. Bishop, "Software Security Checklist for the Software Life Cycle," in Proc. WETICE 03, 2003, pp. 243-248.

- A. D. Rubin, "Security Considerations for Remote Devices," CACM, vol. 45, pp. 39-44, Dec. 2002.

- IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specification

- J. Peters, and W. Pedrycz, Software

Engineering – An Engineering Approach. New York, NY: Wiley, 2000.

1.5 Overview

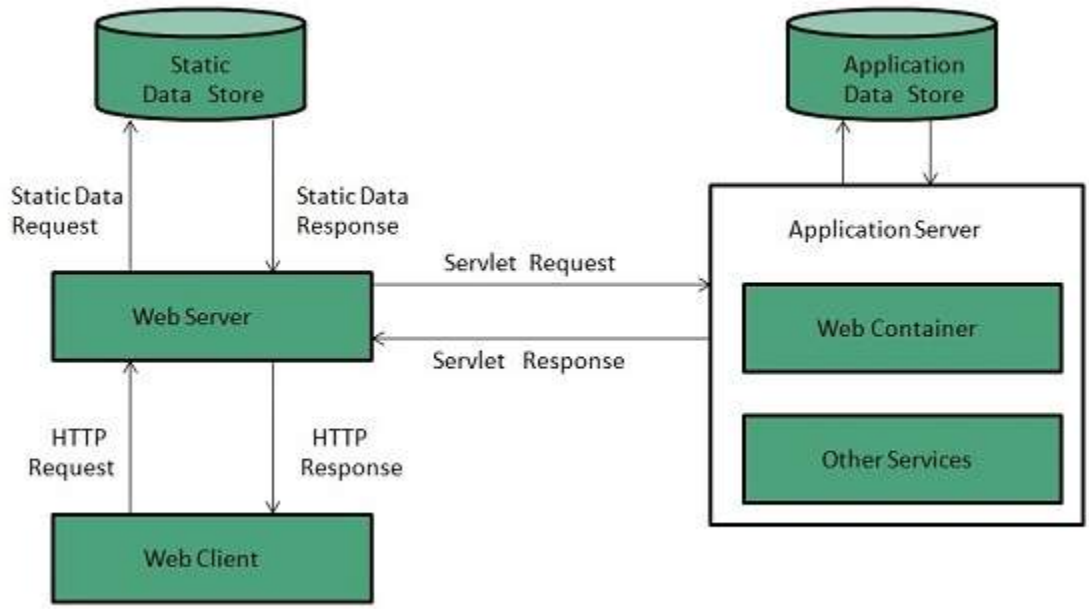
The remainder of this document identifies the actors, use-cases, use case scenarios, activity diagrams, assumptions and dependencies needed for the analysis and design of the Online Doorstep Vehicle Repair software package. The rest of the document contains the overall description of the system, requirements, data model and behavioural description of the system and project planning.

2. The Overall Description

The DVRS is primarily a web-based system so fundamental features related to web-based technologies such as client-server and database properties determine the software requirements of that project. A Mobile application would also be developed which will rely on the API's used in Web Applications.

2.1 Product Perspective

The software product is a standalone system and not a part of a larger system. The system will be made up of two parts. The one part is the complete Backend part (server) which will handle all logical functionality and the API framework. The second part is the frontend (client) part and the Mobile application. This is responsible for the application's user interface. The ECA configures the whole system according to its needs on the server where the system is running.



2.1.1 System Interfaces

- For Server: Django - 3.2.5 (LTS)
- For API: Django Rest Framework- 17.0.2
- For Client: React - 3.12.4

- For Mobile Application: React Native - 0.65
- AWS and Heroku
- Several Third party packages.

2.1.2 Interfaces Specify:

- For Server: Django - 3.2.5 (LTS)
- For API: Django Rest Framework- 17.0.2
- For Client: React - 3.12.4
- For Mobile Application: React Native - 0.65
- AWS and Heroku

The System can be optimised from both frontend and backend. The steps involved precise techniques that will be discussed later in the document. I will provide UI in the form of Web Application and Mobile application.

2.1.3 Hardware Interfaces

There are no hardware interfaces to this software system. The only interfaces are through a computer system.

2.1.4 Software Interfaces

The server runs on http server that is enabled to handle

server pages. It uses a relational database, client application, server application and mobile application.

- For Server: Django - 3.2.5 (LTS)
- For API: Django Rest Framework- 17.0.2
- For Client: React - 3.12.4
- For Mobile Application: React Native - 0.65

2.1.5 Communications Interfaces

The communication between server, client and database will happen through API calls. The API's that are developed using the Django rest framework will act as a link between the apps.

2.1.6 Memory Constraints

Focus groups have determined that our target market has between 128K-512M of RAM,
therefore the design footprint should not exceed 256M. There are no memory
constraints, so state.

2.1.7 Operations

Specify the normal and special operations required by the user such as:

(1) The modes of operation are:

- Customer
- Store Admin
- Store Manager
- Delivery Person
- Anonymous User

(2) The periods when the system has to be down will be decided by the load managers.
(3) All the data processing will take place in the backend. (concept of multithreading will be implemented here)
(4) The backup of the entire data will be done through AWS at 3 am.

2.1.8 Site Adaptation Requirements

Adaptation includes having a stable internet connection, a laptop or a mobile phone.

2.2 Product Functions

- Authentication and Authorization
    - It will be specific to the type of user (Customer, Admin, Manager, Delivery Person)

- Assigning Roles

- For Admin/Manager:

    - Can register their store
    - Able to list their Services
    - Perform CRUD operations
    - Provide Advertisement tools
    - Manage their Customers
    - Contact delivery person
    - Track their services

- For User’s
    - Can register and maintain (CRUD) their profile.
    - Can view all stores
    - Able to access all services provided by the store
    - Place an order/service
    - Contact delivery person
    - Track their orders
    - Modify and cancel their orders
    - Contact Store Admin/Manager
    - Give ratings

- For Delivery Person
    - Can register and maintain (CRUD) their profile.
    - Can view all stores
    - Can reach out to stores
    - Can contact customers
    - Have access to the services provided by DVRS
    - View their ratings

2.3 User Characteristics

The potential user base will include the ones who are mature and own a vehicle. The target age group involves early adults in the age group of 20-40.

2.4 Constraints, Assumptions and Dependencies

The system enables customers to create and book their services from home. This means that the system will work on orders at a large scale.

For the proper working of the system, we can list our assumptions and dependencies as follows.

- Working internet connection.
- A webserver should have basic software installed on the machine, along with third-party packages.
- The server runs on an HTTP server that is ”JSP” enabled.
- A web browser.

2.5 Apportioning of Requirements.

- Interface Requirements
- Functional Requirements
- Non-Functional Requirements
- Performance Requirements
- Security Requirements
- Safety Requirements

3. Requirements

3.1 Other Nonfunctional Requirements

3.1.1 Performance Requirements

1. System should be able to respond to any request within two seconds.

2. In case of report generation, depending on the volume of data maximum time tolerable 15 would be 30 seconds.

3. The system should be able to save and retrieve huge data sets (>200GB).

4. The system must efficiently communicate with ACCOUNTING software.

3.1.2 Security Requirements

1. The system shall use SSL which would act as a security layer for any transactions which include sensitive customer information like bank details or personal details.

2. The system data loss should use encryption and decryption algorithms as a precaution against it.

3. The system would avoid storing cookies for the privacy of customer information.

4. The system shall lock the user after entering the wrong password 3 times.

5. The system's back-end servers will only be accessed by the assigned administrator.

3.2 Software Quality Attributes

3.2.1 Reliability

The entire system shall be reliable if each component of the system works efficiently. If the database is updated time-to-time then the system does not contain ambiguous data and it is reliable.

3.2.2 Availability

The system shall be available 24x7 except for the downtime of the server on which the system is dependent. If the system faces hardware failure then an aberrative page would be available. In a case where the system faces database corruption then data could be retrieved from the server and the service would be available to use.

3.2.3 Portability

It is a web-based application hence built on scripting huge and HTML so that it is fully portable as you can access the application from any device like laptop, PC or mobile.

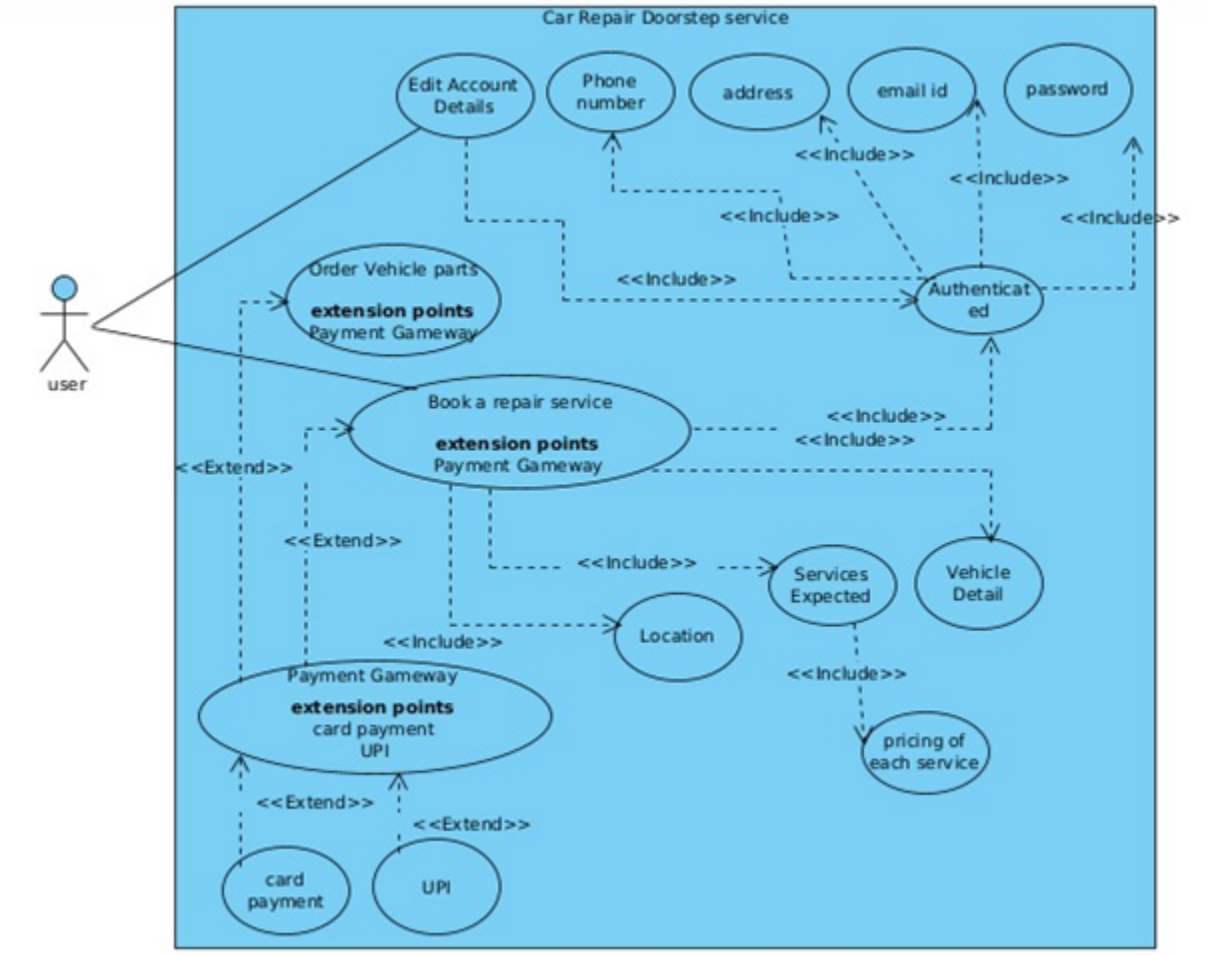
3.3 Design and Implementation Constraints

1. The code with which the system is built may become outdated.

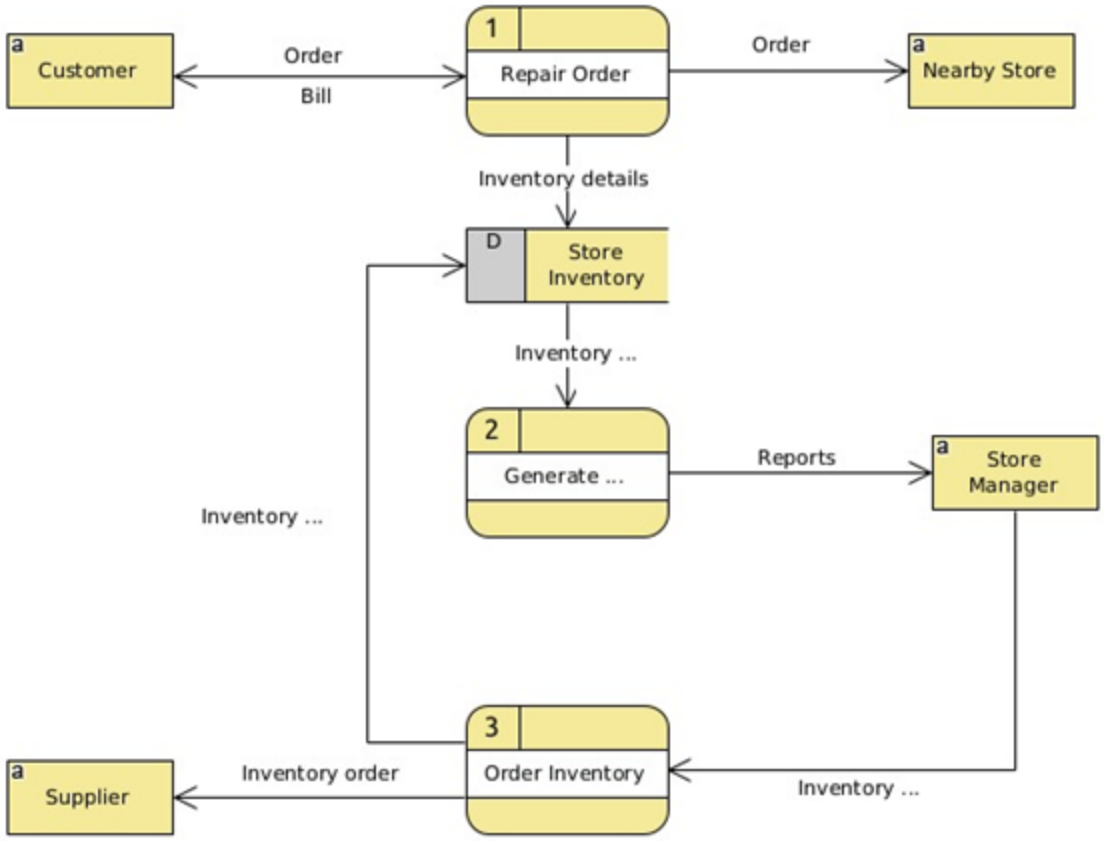
2. User roles will aid in configuring user privileges, which provide access control to the employees or staff members.

3. The finance-related reports generated may not capture all the details as required by the ACCOUNTING software. The system will give the admin a provision to add or modify the fields as per requirement.

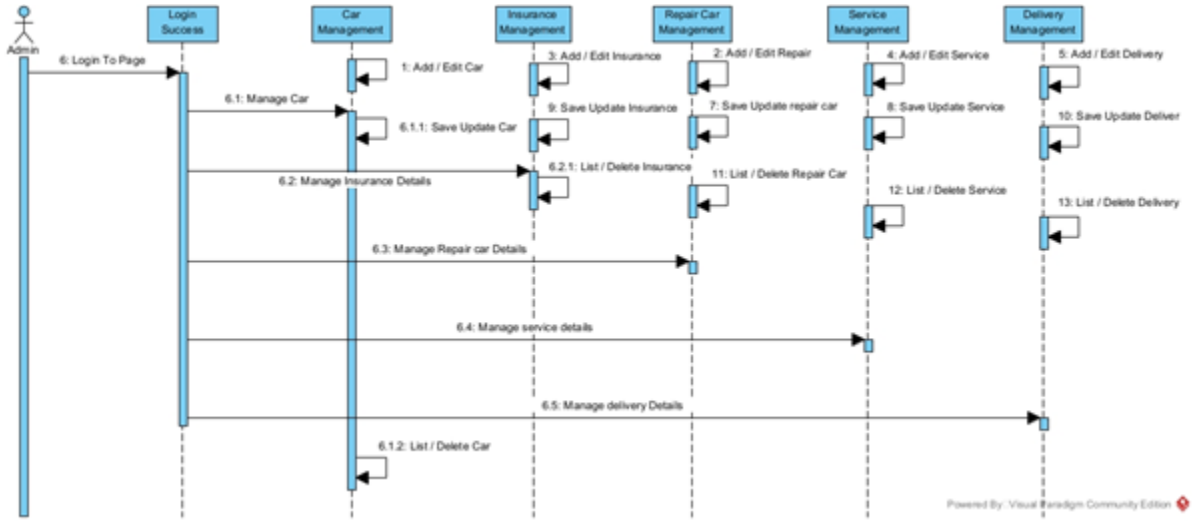
4. Use Case Diagram



5. Data Flow Diagram



6. Sequence Diagram



7. Class Diagram