**Software Requirements Specification Document**

**Title: Online fuel Delivery System**

**Group - 03**

**Table of Content**

**1 . Introduction**

1.1. Purpose

1.2. Document Conventions

1.3. Intended Audience and reading suggestions

1.4. Product Scope

1.5. Reference

**2. Overall Description**

2.1. Product Perspective

2.2. Product functions

2.3. User Classes and characteristics

2.4. Operating Environment

2.5. Design and Implementation constraints

2.6. User Documentation

2.7. Assumptions and dependencies

**3. External Interface Requirements**

3.1. User Interface

3.2. Hardware Interface

3.3. Software Interface

3.4. Communication Interface

**4. Specific Requirements**

4.1. Requirements

4.a. Functional Requirements

4.b. Non functional Requirements

4.2. Use cases

4.3. Business Rules

**5. Other Non functional Requirements**

5.1. Performance requirements

5.2. Safety Requirements

**6. Conclusion**

**1. Introduction**

Fuel delivery app is a delivery solution that enables fuel delivery start-

ups to make safe and hassle-free deliveries of fuel to the customers. For getting fuel

deliveries, customers first have to register on the fuel delivery app. Then they have to

login, enter their location and fuel requirements. After this, they have to complete

their payment by choosing a payment method.

**1.1 Purpose**

 The purpose of OFDS software requirement specifications document is to

provide a detailed overview of our software product, its parameters and

goals.

 Skyline web development basically assists customers to request their

products online, and also help its administrators to monitor and control the

stock.

 The main purpose of developing an OFDS app is to provide a complete

mobility solution that allows the customers to connect directly with the

suppliers from almost anywhere.

 No need to stand in long queues getting your car tank refilled. Get immediate

access to gas anywhere, anytime on tap of a button. Likewise, you can reserve

your frequent requirements in advance.

**1.2 Document** **Conventions**

|  |  |
| --- | --- |
| **Acro** | **Abbreviation** |
| SRS | Software requirements specification |
| OFDS | Online Fuel delivery System |
| OS | Operating Systems |
| DBMS | Database Management System |

**1.3 Intended Audience and Reading Suggestions**

* This is intended for the operation manager, customers and the coordinator. It

will be implemented under the guidance of coordinators

* The project will play an important role of assisting operations manager and

customers, in terms of managing stock and online equipment hiring respectively.

**1.4 Product Scope**

Timely & Cost-effective

* You can save time and can get the fuel in few minutes after ordering it.
* One of the best advantages of this gas and fuel delivery app is it save lots of

time and money. Also, a sufficient amount of fuel can be supplied to desired

place at the expected time.

* It allows the user to pay according to their comfortability.

**Emergency** **crisis**:

* It works as a savior in emergencies. In the time of problems like shortage of

fuel, this app can provide you with fuel at the place so you can go ahead

hassle freely.

**Location** **Sharing**:

* This feature makes it easy for users to share their location and access

**Request Fuel:**

* You can choose the type, quantity, and category of fuel they have to order.

**Track Delivery:**

* Track of the location of the delivery vehicle in real-time on a map with ETA.

**Easy Payments:**

* Choose the payment option as per your convenience. You can pay for your

order using multiple payment methods including cash on delivery.

**Transaction History:**

* Keep a track of your previous fuel delivery requests and bills.

.

**Enter Vehicle Info:**

* Add a new vehicle with a one-time process to streamline recurring requests.

**Schedule Delivery:**

* Choose a time window for delivery that works best for you.

**Manage Profile:**

* Dedicated section to manage your profile, favorites, vehicles and more.

**1.5 References**

The references are:

Https// krazytech.com/projects

[Https://www.perforce.com](https://www.perforce.com)

**2. Overall Description**

**2.1 Product Perspective**

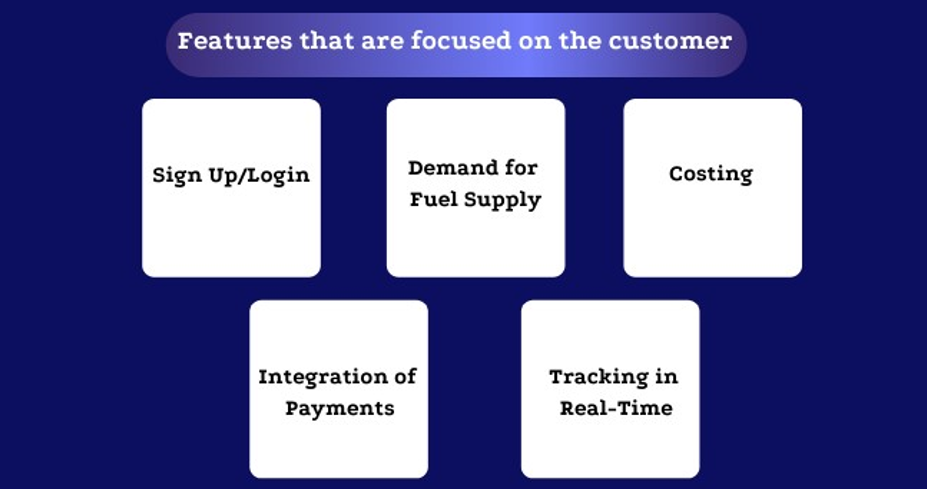
The product/service specified in the SRS document is a replacement for an existing

system which was maintained manually by the organization.

**2.2** **Product** **Functions**

**CUSTOMER**

* Requires customer to register an account or login
* View all products for sale
* User request
* Calculate total sales for requested fuel



**ADMINISTRATIVE**

* Add / delete service
* Update equipment
* Update requested service
* Update deliveries / collections

**AGENT**

**Sign Up/Login:**

The driver can download and update various versions of the program, as well

as register or login for it.

**Activity status:**

The driver may include their service availability state, which enables them to

toggle their working mode’s availability status on and off.

**Table of Contents:**

On the dashboard, the driver will see the ordered, planned, and completed

deliveries, as well as the earnings.

**Features in Real-Time:**

The driver has the option of accepting or rejecting the obtained offer, as well

as tracking the order in real-time.

**Details regarding the consumer:**

The driver may access the customer’s contact details, such as name, phone

number, venue, and fuel, after approving the order and be able to contact

them.

**2.3 User Classes and Characteristics**

* The users of the fuel application should be able to sign up with their emails or

phone numbers.

* On the Push Notification they will be frequent updates relating to their orders and

pricings from time to time as well as price fluctuations

* Should the user experience any inconvenience or errors before/after placing orders

the online support would be available 24/7 at the tip of the fingers will just a simple

* push of button or call the online support team immediately, they will be assisted

accordingly.

The Customer should be able to the following

* Signing up/Logging in
* Sing up with Google
* Sign up with Email/Phone Number
* Access to Push Notification
* Access online support for any problems or error
* Access to their Order History
* Access to current order in progress

The Driver should be able to the following

* Signing up/logging in
* Updating the push notification
* Access to customer records
* Access to location and live tracking

The Admistrator should be able to the following

* Have access to both the customer and driver’s panel dashboard
* Update the pricing of fuel
* Calculate the cost of orders and estimate delivery time
* Access to both the customer and driver’s record history
* Availability for query support and take calls 24/7
* Make a new request to enroll in the website
* Cancel an existing request
* View request details

### **2.4 Operating Environment**

* centralised database
* client/server system
* Operating system: Windows.
* database: SQL + database
* platform:Java/PHP

### **2.5 Design and Implementation Constraints**

* The system shall be developed using open-source tools and shall run on windows operating system
* The global schema, fragmentation schema, and allocation schema.
* SQL commands for above queries/applications
* Explain how various fragments will be combined to do so.
* Implement the database at least using a centralized database management system.
* The system shall be developed using Java/PHP

### **2.6 User Documentation**

* FAQ, Step by step guide, Support teams with hotlines.
* It will provide easy use of the app starting from how to install it to helping the user enter accurate details. There will also be a video tutorial on how to make your fuel order. Users will be guided on how to use different features of the system. There will be screenshots explaining main features of the system.

Explanations of error messages and troubleshooting guides too.

### **2.7 Assumptions and Dependencies**

Let us assume that this is a centralized management system and it is used in the following application:

* Calculation of most frequent requesting customers and calculating appropriate reward points for them.

## **3. External Interface Requirements**

### **3.1 User Interfaces**

* **Front-end software:** Android Studio
* Back-end software: SQL+

### 3.2 Hardware Interfaces

* Windows.
* A browser which supports HTML, Java

### **3.3 Software Interfaces**

|  |  |
| --- | --- |
| **Software used** | **Description** |
| Opertaing System | Windows operating System for its best support And user Friendliness |
| Database | To store customer information and booking and product information |
| Android studio | To implement the project,Java and Kotlin language for interactive support |

### **3.4 Communications Interfaces**

This project supports all types of web browsers. We are using simple electronic forms for the requesting forms. The platform itself, consist of chat for services, and the specified email can be used to communicate with customers.

**4. Specific Requirements**

**4.1. Requirements**

**Functional Requirements:**

* **User Authentication and Authorization:**

Users must be able to register an account securely. The system should support multi-factor authentication. Different user roles (e.g.,customer, delivery driver) with specific permission.

### **Order** **Placement**:

### Users should be able to browse and select fuel types. Specify thequantity of fuel needed. Provide delivery location details, includingaddress and any specific instructions.

### **Delivery** **Assignment** **and** **Tracking**:

### Delivery orders must be automatically assigned by the system todrivers who are available. A real-time map is essential for effective route planning by drivers. Users ought to be able to monitor the whereabouts and status of their deliveries.

### **Payment** **Processing**:

### safe management of credit card data. support for a number of payment options, including credit cards and mobile payments. creation of invoices or receipts automatically

### **User** **Feedback**:

### Users can provide feedback and ratings for each delivery. A mechanism for the system to analyze and respond to feedback.

### **User** **Account** **Management**:

### Users should be able to update their profile information. Allow users to view order history and invoices.

### **Notification** **System**:

### Notifications on order confirmation, delivery status updates, and completion are sent to users. Push alerts for mobile apps.

**Non -Functional Requirements:**

* **Performance**:

During peak hours, the system should be able to process at least X orders per minute. User engagements (such as placing orders and tracking) should have response times of fewer than Y seconds.

* **Reliability**:

With the exception of planned maintenance, the system should be operational 99 percent of the time. Establish data backup procedures with a Z-hour recovery time target (RTO).

* Scalability:

### Plan the system to accommodate an increasing volume of orders and users over time. Make sure the program can grow horizontally by expanding its server or instance count.

### **4.2 Use Cases**

An actor in use case modeling specifies a role played by a user or any other system that interacts with the subject.

An Actor models a type of role played by an entity that interacts with the subject (e.g., by exchanging signals and data), but which is external to the subject.

There the system will have the following actors

* Customer
* Agent
* Admin

**a) Use** **Cases**: The whole use case diagram will have below use cases:

## **For** **customer**

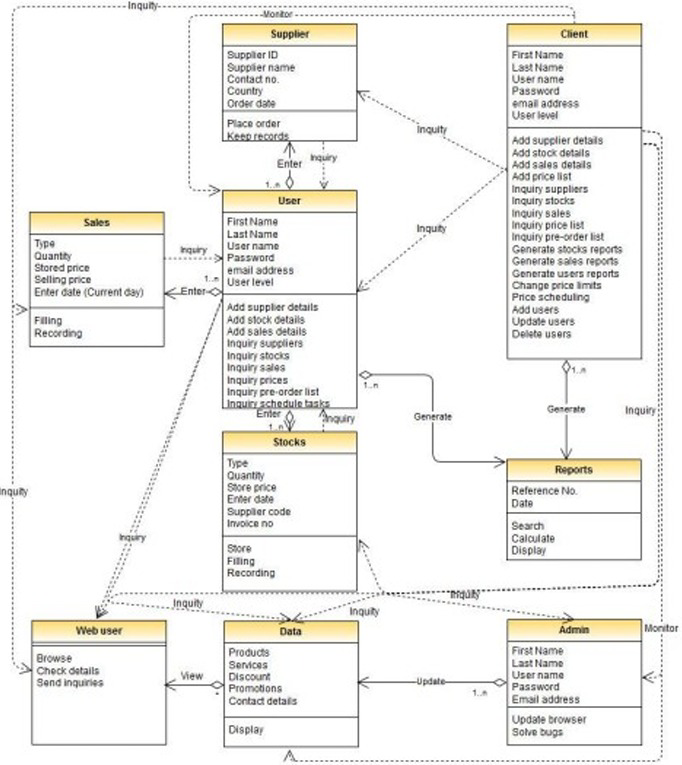
* **Purchase** **Online**: Main Use case, the ability for the customer to successfully make a purchase
* **Make** Registration: Purchase Online will include make registration.
* **Login**: All necessary details to authenticate user.
* **Payment**: Finalizing money transfer can also choose to have the money paid in person.
* **Evaluation** of the driver: Authenticating the driver is indeed on the way.
* **Detect** Location: The use case that shows where the customer is relative to the driver.
* **Log** **out**: After completion, logging out of the app can now be done.

**For** **Agent**

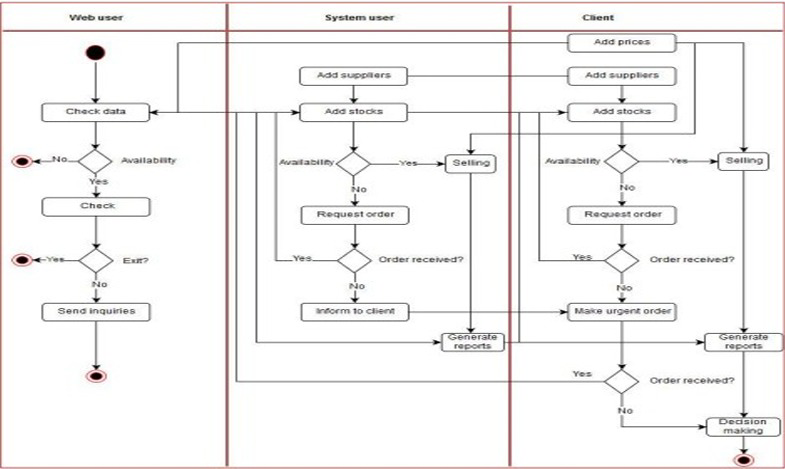
* Register
* Login
* Use the map: See where the customer is
* Evaluation of the customers: Authenticating where customer is and proceed to driving where they are.
* Log out

## **For** **Admin**

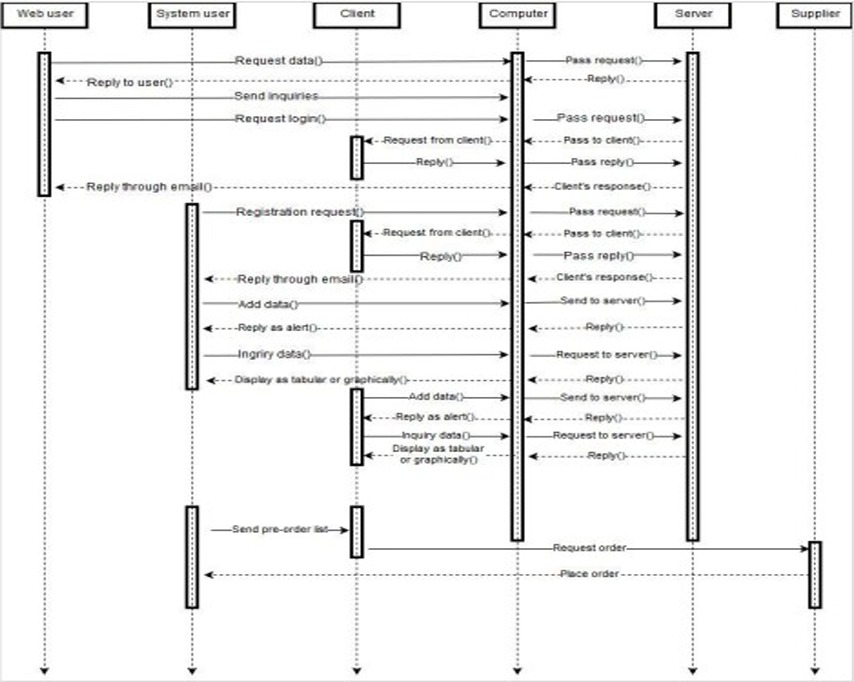
* Follow the petrol driver: Ensure the driver is on the proper route
* Add new petrol driver: Add new driver to the list of drivers
* Determine price: Update the pricing of the fuel
* Add offers: Special discounts that which might be added in it
* **Class** **Diagram**:

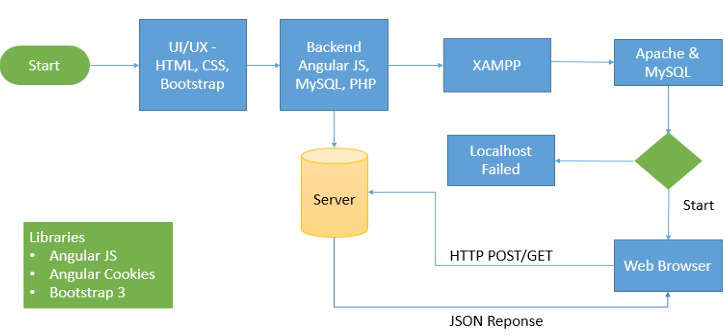


* **Activity** **Diagram**:



* **Sequence** **Diagram**:





* **Data** **Flow** **Diagram**:

### **4.3.** **Business** **Rules**

Admin Portal should be created which should such that the More privileges need to be provided to the admin, proper terms and conditions and FAQS and Support and Contact Us, such type of information must also be available to the users so that they can go to the specific section if any problem

If a selected driver does not respond within a predetermined time window, another driver can be selected.

Employee onboarding is a lengthy process.

Employees must, among other things, be trained, given system access, and assigned assets.

All new employees will be given a work email address.

## **5. Other Nonfunctional Requirements**

### **5.1 Performance Requirements**

* The application accommodates a maximum of one thousand concurrent users.
* System screens exhibit no greater than 2 seconds response time.
* Application takes 0.5 second to reply to users input
* Application uses very small internet data to load a page or for a customer to place an order

### **5.2 Safety** **Requirements**

* In Case of a system crash, there is a database backup for every data in the system.
* If a user loses the password, they can contact the administrator and retrieve their password.
* Minimal strain on battery life.

**Security** **Requirements**

* The application identifies all of the user applications before
* allowing them to use its capabilities.
* Two-factor authentication is enabled with OAuth2, JSON is used for profile encryption.
* The application ensures that the name of the customer placing an order and payment databases exactly matches the name printed on the customer social security card.
* Sensitive data is not distributed among third party mediators
* The delivery address is taken from your current google map location.

### **5.4 Software Quality Attributes**

**AVAILABILITY**:

The application runs 24 hours every day of the week and deliveries are made every day at any time

**CORRECTNESS**:

The application never identifies incorrect user or incorrect address for the delivery

**MAINTAINABILITY**:

The administrators and petrol manager maintain correct schedules for every delivery

**USABILITY**: The delivery schedules satisfy a maximum number of customer’s needs

**Conclusion:**

To sum up, a mobile app for fuel delivery provides a practical and effective replacement for the conventional method of car refueling. Customers can avoid driving to a gas station and save time by using on-demand fuel delivery services.Real-time pricing updates, on-demand fuel delivery services, and integration with smart home systems are just a few ways to improve the app and add even more features and advantages. Predictive analytics, in-app customer service, and gamification elements are examples of further improvements.

The app may grow to offer battery level monitoring, charging station location information, and perhaps res- idential charging station installation services as the num- ber of electric vehicles on the road rises. All things considered, a fuel delivery smartphone app provides a practical and effective substitute for conventional