Software Requirements

Specification

**for**

**Easy2Ride – Two Wheeler Booking**

**Website**

### Version 1.0 approved

**Prepared by Aditya Prakash Sinha (229310189) Ayush Rakesh Prasad (229310199)**

### Manipal University Jaipur

**19/09/2024**

***Project SRS (Project Based Learning), 2024, MUJ.***

# Table of Contents

[**Table of Contents i**](#_bookmark0)

[Revision History ii](#_bookmark1)

1. [Introduction 1](#_bookmark2)
   1. [Purpose 1](#_bookmark3)
   2. [Project Scope 1](#_bookmark4)
   3. [Overview 1](#_bookmark5)
   4. [System Overview 1](#_bookmark6)
2. [Functional Requirements 2](#_bookmark7)
   1. User Management
      1. Registration and Login
      2. Proﬁle Management
   2. Vehicle Browsing and Selection
      1. Vehicle Listing
      2. Search Functionality
      3. Vehicle Details Page
   3. Booking Process
      1. Date and Time Selection
      2. Pricing Calculation
      3. Booking Conﬁrmation
   4. Payment Gateway Integration
      1. Payment Methods
      2. Transaction Security
   5. Notiﬁcation and Communication
      1. Email and SMS Notiﬁcation
3. **Non - Functional Requirements 3**
   1. Performance 3
   2. Security
   3. Usability
   4. Reliability
   5. Scalability 4
4. **External Interfaces 4**
   1. User Interfaces 4
   2. Hardware Interfaces 4
   3. Software Interfaces 4
   4. Communications Interfaces 4
5. **System Design Constraints**
   1. Frontend Development
   2. Backend Development
   3. Third Party Integrations

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Reg. No.** | **Date** | **Version** |
| Aditya Prakash Sinha | 229310189 | 15/09/2024 | 1.0 |
| Ayush Prasad | 229310199 | 19/09/2024 | 1.1 |

# Introduction

## Purpose

This SRS document outlines the software requirements for an online platform that allows students and residents of Manipal University Jaipur to book two-wheelers. The purpose of this document is to detail the functional and non-functional requirements, constraints, and specifications for the development of the website.

## Project Scope

The platform will enable users to browse and book two-wheelers from their location without needing to visit a rental shop physically. The platform will ensure a seamless, secure experience with clear pricing, booking terms, and availability, thereby streamlining the rental process.

## Overview

This document is divided into sections that describe the system functionalities, non- functional requirements, external interfaces, and design constraints. It is intended for developers, testers, and project stakeholders.

## System Overview

The system will consist of user management, vehicle browsing, booking, payment, and notification modules. The focus will be on ease of use, security, and transparency.

# Functional Requirements

This section provides detailed descriptions of the functional requirements for each module of the two-wheeler booking platform. These requirements specify what the system must do to meet the user needs and business objectives.

## Module 1: User Management

### Registration & Login

* + - * **Registration**:
        + Users must be able to register on the platform by providing their **email address**, **password**, and **personal details** such as name, phone number, and student ID (if applicable).
        + The system should include **client-side validation** for input fields (e.g., valid email format, minimum password length, mandatory fields).
        + A **verification email** will be sent upon registration to confirm the email address before granting full access to the platform.
        + Registration should allow social media login integration (e.g., **Google**, **Facebook**) for convenience.

### Login:

* Users must log in using their registered email and password, or through the social media login if it’s linked.
* The system will implement **JWT-based authentication**, ensuring secure login sessions. JWT tokens will be generated and stored securely on the client side (in cookies or local storage) to maintain session states.
* If a user forgets their password, they should be able to reset it via a **password recovery** mechanism that sends a reset link to their registered email.

## Profile Management

### Profile Viewing and Editing:

* + Users must be able to view and edit their profile information, such as name, contact number, and saved addresses.
  + Profile editing should include validation rules (e.g., valid phone number format).

### Payment Method Management:

* Users can save their preferred payment methods (e.g., credit/debit cards, UPI details) for faster checkout in future bookings.
* The system should store payment details securely, complying with **PCI-DSS**

standards to protect sensitive information.

### Booking History:

* + Users will be able to access their booking history, which includes past, current, and upcoming bookings. This section should display key details such as vehicle type, booking dates, total cost, and status (e.g., confirmed, cancelled).
  + Users should be able to rebook a previously rented vehicle with the same or modified details.

## Module 2: Vehicle Browsing & Selection

* + 1. **Vehicle Listing**

### Available Vehicles Display:

* + The platform will display a catalog of available two-wheelers, which users can browse. Each listing should show key information, such as vehicle type (scooter, bike), brand, price per day, and current availability.

## Filtering Options:

* + Users can filter the displayed vehicles based on various criteria such as:
    - Type: Scooter, bike, electric, etc.
    - Brand: Honda, Yamaha, Bajaj, etc.
    - Price Range: Low to high and vice versa.
    - Availability: Show only currently available vehicles.

### Sorting:

* + Users can sort the list by price, popularity, or other custom criteria such as the newest models.

## Search Functionality

### Search Bar:

* + Users can search for vehicles using keywords (e.g., "Honda", "scooter") to find relevant listings.

### Advanced Filters:

* + The search functionality should also include advanced filters where users can set multiple parameters like vehicle type, price, and brand simultaneously to narrow down their search results.

## Vehicle Details Page

### Detailed Information:

* + Each vehicle should have its own dedicated page showing in-depth details such as vehicle specifications (e.g., engine size, mileage, fuel type), daily rental price, photos, availability calendar, and additional features (e.g., helmets included, luggage racks).

### Reviews & Ratings:

* + The system will allow users to see previous customer reviews and ratings for the specific vehicle to help inform their choice.

## Module 3: Booking Process

* + 1. **Date & Time Selection**

### Booking Calendar:

* + Users will be able to select the start and end date/time for their booking from a calendar interface. The system will display vehicle availability in real time, preventing double bookings.

### Minimum and Maximum Rental Duration:

* + There should be restrictions on minimum and maximum booking durations (e.g., a minimum of 1 day and a maximum of 30 days) based on business rules.

## Pricing Calculation

### Automatic Pricing:

* + Once users select the booking duration, the system will automatically calculate the total rental cost based on:
    - Vehicle type.
    - Rental duration.
    - Any discounts, surcharges, or taxes.

### Dynamic Pricing:

* The system may implement dynamic pricing during high-demand periods (e.g., weekends, festivals) to adjust rental costs accordingly.

## Booking Confirmation

### Booking Summary:

* + Before confirming the booking, the system will provide a detailed summary, including vehicle information, rental period, total cost, and applicable policies.

### Booking Confirmation:

* + Upon user confirmation, the system will send a booking confirmation via both email and SMS. The confirmation should include all booking details, including pickup instructions, and any necessary terms and conditions.

## Module 4: Payment Gateway Integration

* + 1. **Payment Methods**

### Supported Payment Options:

* + The platform will integrate with popular payment gateways like PayPal, Stripe, and UPI to offer users flexibility in how they pay for their bookings.

### Saved Payment Methods:

* + Users can opt to save their payment details for faster checkout during future bookings. Payment details will be securely stored using tokenization methods provided by the respective payment gateway.

## Transaction Security

* SSL Encryption:
  + All payment transactions will be protected by SSL encryption to ensure that sensitive data such as card numbers and personal details are securely transmitted over the internet.
* Secure Payment Processing:
  + The system will adhere to PCI-DSS standards for secure payment processing, ensuring that users’ payment data is handled safely and in compliance with industry standards.

## Module 5: Notifications & Communication

* + 1. **Email & SMS Notifications**

### Booking Confirmation:

* + Users will receive a confirmation notification via email and SMS immediately after successfully booking a vehicle.

### Reminders:

* + The system will send reminder notifications (via email and SMS) to users before their booking starts (e.g., 24 hours prior) and when the booking is about to end.

### Cancellation & Refund Notices:

* + If a user cancels a booking or receives a refund, the system will send an email and SMS notification to confirm the cancellation and provide details of any refund amount or policy.

### System Alerts:

* + In case of system outages, maintenance, or other issues that may affect booking availability, users will be notified in advance through these channels.

# Non - Functional Requirements

Non-functional requirements (NFRs) define the system’s performance, security, usability, and scalability. These are essential to ensure that the system operates efficiently, securely, and consistently, meeting user and business expectations. Here is a detailed elaboration of each of these non-functional requirements:

## Performance

**Simultaneous users:** The system should support up to maximum users at a time without any issues.

**Search speed:** Vehicle search results must be returned within 2 seconds for a smooth user experience.

## Security

**Encryption:** All user data, including personal and payment information, must be encrypted, both at rest and during transmission.

**JWT-based authentication:** Secure login using JSON Web Tokens (JWT) to ensure authenticated sessions.

**Compliance:** The platform must comply with privacy laws like GDPR to protect user data.

## Usability

Usability focuses on the ease with which users can interact with the system. The platform must be designed for intuitive, efficient navigation, regardless of the device being used.

## Responsive design:

* + The website must be responsive, ensuring optimal usability across various screen sizes and devices, including mobile phones, tablets, and desktops.
  + The platform should adopt **responsive web design principles** (e.g., using flexible grids, media queries) to automatically adjust content layout to fit different screen sizes.
  + **Cross-browser compatibility** is necessary, ensuring smooth performance on all modern browsers like Chrome, Firefox, Safari, and Edge.

## Intuitive user interface:

The platform should have an easy-to-use, visually appealing interface that minimizes the learning curve for new users.

* + **Minimalist design** principles should be applied, using clear navigation paths, well-labeled buttons, and simple workflows to guide users through actions like searching for vehicles or making a booking.

## Reliability

* 99.9% uptime: The platform should be available with minimal downtime, translating to high availability for users.
* Daily backups: Automated backups should occur every 24 hours to ensure data is protected and recoverable in case of failure.

## Scalability

* Future growth: The system should be able to scale to handle more users and vehicles as the platform expands.
* Modular architecture: New features should be easily integrated without disrupting existing functionalities.

# External Interfaces

External interfaces define how the system interacts with other systems, users, and external hardware. Here is a detailed overview:

## User Interfaces

### Web-based Interface:

The platform will be accessible through a web interface compatible with modern web browsers. This ensures that users can interact with the system using popular browsers such as:

### Google Chrome

* + **Mozilla Firefox**

### Safari

* + **Microsoft Edge**
* The design should ensure cross-browser compatibility to provide a consistent user experience.

### Hardware Interfaces

* **Standard Computing Devices**:

The system should be operable on standard computing devices with internet access, including:

### Desktops

* + **Laptops**

### Tablets

* + **Smartphones**
* No special hardware is required beyond what is typically available to users with internet connectivity.

### Software Interfaces

* **Backend API**:

The backend of the system will be built using **Express.js**, a web application framework for Node.js, providing the API endpoints necessary for the system's operations.

### Frontend:

The frontend will be developed using **React.js**, a JavaScript library for building user interfaces, ensuring a dynamic and responsive user experience.

### Database Integration:

The system will integrate with a database (e.g., **MongoDB**, **PostgreSQL**) to store and retrieve data related to users, vehicles, and bookings.

### Communication Interfaces

* **Internet Protocols**:

Secure communication will be ensured through:

* + **HTTPS**: For secure data transmission between the client and server.
  + **SMTP**: For sending emails related to notifications and confirmations.
* These protocols will help in maintaining data integrity and security during transmission.

# System Design Constraints

Design constraints outline the limitations and requirements for the system’s architecture and development:

## Frontend Development:

The frontend must be developed using React.js, which provides a component-based approach for building user interfaces, ensuring high performance and maintainability.

## Backend Development:

The backend must use Express.js, which facilitates the development of RESTful APIs, handling requests from the frontend and managing interactions with the database.

## Third-Party Integrations:

The system must integrate with:

Payment Gateways:

Such as PayPal, Stripe, or UPI for processing transactions.

SMS Services: For sending notifications related to bookings and confirmations. Compliance: The system must adhere to local data security and privacy regulations, ensuring that user data is handled appropriately and securely, in line with applicable laws.