Software Requirements Specification for Moto Bike

Submitted To:

Sir Aatif Hussain

Submitted By:

Shahbaz Rafique	2021-CS-16
Saad Munir	2021-CS-19
Hammad Younas	2021-CS-20

Department of Computer Science
University of Engineering and Technology Lahore, Pakistan

Table Of Contents

1. Introduction

1.1 Purpose

The purpose of this project is to create a website for people who love motorcycles. On this website, you can buy all kinds of motorcycle-related stuff like bikes, parts, and accessories. The website is easy to use, and you can quickly find what you want. It has clear categories, detailed product info, and pictures. You can also read what other customers say about the products. You can compare different products and pay for them easily with different cards. If you have questions, there's a team to help you.
This project is not just for customers; it also has special roles for Admin (who manages everything), Sellers (who can upload products), and Customers (who buy things). Admin can keep an eye on what everyone does and generate reports. In a nutshell, this project aims to make buying motorcycle stuff online easy and enjoyable for everyone involved.

1.2 Scope

The Moto Bike Management System project's scope is to develop a comprehensive e-commerce platform for motorbike enthusiasts. It will serve as an online marketplace where users can access a wide range of high-quality motorbikes, parts, accessories, and components. This platform will cater to three primary user roles: Admin, Seller, and Customer. Customers can shop for products, view detailed information, read reviews, and place orders with multiple payment options, including the integration of the Stripe Payment Gateway. The project facilitates direct communication between users, including a live chat room for customers. It also incorporates a reporting system for tracking product and seller performance. With a focus on user-friendliness, this project aims to create a global community of motorbike enthusiasts, offering a seamless and secure online shopping experience while enhancing user interaction and knowledge sharing.

1.3 Abbreviations and Definitions

- **SRS:** Software Requirement Specification - A paper detailing the functional and non-functional needs for the e-commerce website.
- **CMS:** Content Management System - A platform that makes it simple to create, update, and manage website content.
- **UI:** User Interface - The website's graphical interface and interactive components.
- **UX:** User Experience - The total interaction that users have with the website, including its usability, navigation, and overall happiness.
- **SKU:** Stock Keeping Unit - A special identification or code for each item in the inventory.
- **API:** Application Programming Interface - A collection of guidelines that permit interaction between various software programs.
- **CMS:** Customer Management System - A client account, profile, and preference management system.
- **B2B:** Business-to-Business - Online commercial exchanges involving two companies, such a dealer and a firm.
- **B2C:** Business-to-Consumer - E-commerce exchanges between a company and a single client.
- **Cart:** Shopping Cart - A virtual cart that allows customers to add and arrange their chosen goods before checking out.
- **Wishlist:** A function that lets customers bookmark things for later usage or acquisition.

- **Search Algorithm:** The method by which search results are retrieved and presented in response to user requests.
- **Payment Gateway:** A service that makes it easier for clients and businesses to make payments and transfers online.
- **Inventory Management:** The method of monitoring and controlling product stock levels, which includes notifications for out-of-stock situations and restocking.
- **Responsive Design:** Ensuring that the website adjusts and performs properly across a range of platforms, such as tablets, smartphones, and PCs.
- **Load Time:** The amount of time it takes a user's browser to fully load a page.
- **User Account:** Users can have personalized profiles with their information, past orders, and preferences.
- **Product Category:** A categorization scheme (e.g., cement, plastic pipes, electrical supply) for grouping and arranging things according to their kind or function.
- **Product Description:** Comprehensive details on a product, such as features, specs, and usage guidelines.

1.4 Overview

This project is all about creating a user-friendly website for people who love motorcycles. It offers a wide variety of high-quality motorbikes, parts, and accessories from well-known brands. Customers can easily browse and shop with clear product categories, detailed descriptions, and customer reviews to help them make smart choices. They can also compare products and read what others think. Checkout is quick and simple with different payment options available. Plus, there's a dedicated customer service team to assist with any questions. The project also includes roles like Admin, Seller, and Customer, each with specific tasks and privileges. Admin can oversee everything, generate reports, and manage activities. This project's core goal is to make buying motorcycle gear online easy and enjoyable for everyone involved.

1.5 Contact

- **Project Team:**
  List the key members of the project team who are involved in the creation and maintenance of the SRS.
  - Name: XYZ
  - Phone No: (+1) 234 242467
  - Fax No: 00112244
  - Email Address: email@gmail.com
- **Client or Stakeholder Contacts:**
  Identify the primary client or stakeholder representatives who should be contacted for questions, clarifications, or issues related to the requirements.
  - Name: XYZ
  - Phone No: (+1) 234 242467
  - Fax No: 00112244
  - Email Address: email@gmail.com

1.6 References

- **Payment Card Industry Security Standards Council (PCI SSC):**
  The PCI SSC sets and maintains standards for payment card security, including the Payment Card Industry Data Security Standard (PCI DSS).
  https://www.pcisecuritystandards.org

- **Payment Card Industry Security Standards Council (PCI SSC):**
  W3C develops and maintains web standards and guidelines, including HTML and CSS.
  https://www.w3.org
- **United Nations Economic Commission for Europe (UNECE):**
  UNECE regulations set standards for various aspects of motorcycles, including safety, emissions, and noise levels. UNECE Regulation No. 78, for example, addresses motorcycle helmets.
  https://www.ungeneva.org/en/about/organizations/unece#:~:text=The%20United%20Nations%20Economic%20Commission,promote%20pan-European%20economic%20integration
- **Society of Automotive Engineers (SAE):**
  SAE sets standards for various aspects of automotive and motorcycle engineering, including components, materials, and performance.
  https://www.sae.org/standards

2. Overall Description

2.1 Product Perspective

The intended motorcycle website's product viewpoint, which focuses on the sale of motorbikes, safety gear, gasoline, and spare parts, provides insightful information about the history and context of the system. This website serves as an all-in-one, feature-rich e-commerce platform tailored to the various requirements of motorbike buyers and dealers. It is a stand-alone product that meets the needs of the motorcycle industry; it is not a member of a wider product family.
The motorbike website will depend on several external interfaces to offer a smooth user experience in connection to a bigger system. These interfaces include payment gateways for safe transactions, order fulfillment systems for shipping and logistics, databases external to access real-time information on fuel prices and spare part availability, and systems for user reviews and feedback to improve the overall user experience. These other relationships are essential to building a comprehensive motorbike marketplace that guarantees the usefulness and functioning of the website go beyond its independent existence.

2.2 Product Functions

- Payment Gateway
- Chat App
- Emailing System
- Report Generating

2.3 User Characteristics

Project Actors:
The system has three types of users: Admin, Seller, and Customer.

- Admin
  Admin is the top of the hierarchy user. Admin can add new sellers and can also manage their membership. He can also view the products from all the users. He can also generate the required analytical reports as per required and can also contact the seller and the customer with the help of the email system and the chat room.
- Sellers
  Sellers are the key points of any e-commerce website because they must add and sell the products. He can also add new products and can also give offers to the customers. He can also chat with the

admin and the customers as a seller must satisfy the needs of the customer. He can also generate analytical reports as per required.

- Customers
  Customers are those who are interested in purchasing motorbikes and their parts. He can also give reviews and ratings to the products after purchasing them. Customers can also become a seller by applying for a membership which is then having to accept by the admin

2.4 Operating Environment

Technology Stack:

- Language (C#/Java): Node js, Express js
- Platform (Web/Desktop) Web Frontend Technology (Simple HTML/Bootstrap/any other library): React js, MDB React, Bootstrap
- IDEs: Visual Studio Code

2.5 Implementation Constraints

- Trouble with Data Formatting:
  Sometimes, it's tough to get the data organized in the way we need it. It's like trying to tidy up a messy room but with information.
- Struggling to Build a Secure Chat App:
  Creating a chat app that keeps your messages secret can be quite challenging. It's like building a secret code that only you and your friend can understand.
- Managing Product Quantity Issues:
  When you have a store, it can be hard to make sure you always have enough products in stock. It's like trying to bake cookies, but you're not sure how many chocolate chips you have left.
- Protecting User Data:
  Keeping people's personal information safe can be a real puzzle. It's like making sure no one can peek into your diary unless you want them to.

2.6 General Constraints

- **Limited Scalability:**
  The project may have limitations in terms of scalability, especially if it's designed for a small-scale or specific use case. As the system grows or if there is a need to accommodate a larger number of users or motorbikes, it may require significant modifications and upgrades to handle the increased load.
- **Security Concerns:**
  Depending on the implementation and security measures in place, the project may be vulnerable to security breaches, data theft, or unauthorized access. Protecting sensitive data, such as user information and motorbike details, is crucial and may require additional security features.
- **Lack of Real-time Updates:**
  The project's ability to provide real-time updates on the status and location of motorbikes may be limited, especially if it relies on periodic data synchronization. Real-time tracking and monitoring may be challenging to implement, which could be a limitation for certain use cases.
- **Maintenance and Support:**
  Over time, software projects like this one may require regular maintenance and updates to keep up with changing technologies and requirements. The lack of a dedicated maintenance plan or support team can lead to difficulties in addressing issues, implementing new features, and ensuring the system remains up-to-date and functional.

2.7 Assumptions

- **Data Accuracy:**
  The system assumes that data entered by users or collected from motorbikes, such as location, mileage, and maintenance records, is accurate and truthful. It may not account for deliberate data manipulation or inaccuracies.
- **Stable System Environment:**
  The system assumes a stable and reliable server environment for hosting and running the application. Server outages or technical issues may not be addressed in the assumptions.
- **Internet Connectivity:**
  The system may assume that users have consistent internet connectivity when interacting with it. This may affect the real-time functionality and data synchronization.

3. Requirement Specification

3.1 Functional Requirements

functional requirements constitute a precise and measurable description of the specific functions, features, and behaviors that a software system or application must possess to satisfy the needs and expectations of its users and stakeholders. These requirements serve as the foundation for designing, developing, and testing the software, outlining its intended capabilities and how it should behave when users interact with it. Functional requirements are characterized by their clarity, testability, realism, completeness, consistency, and traceability, ensuring that all essential system functions are identified and can be verified for compliance with the specified criteria. Our Functional requirements are as follows:

3.1.1 User Management:

Customers can create and manage their profiles.
Sellers can create and manage their profiles.
Admin can manage user profiles, including customers and sellers.

3.1.2 Product Catalog:

The website offers a wide range of motorbikes, bike parts, accessories, and components.
Products are categorized and organized for easy browsing.
Detailed product descriptions, high-quality images, and customer reviews are available.
Customers can compare different products.

3.1.3 Shopping and Payment:

Customers can add products to their cart.
A seamless checkout process is provided.
Payment can be made using various payment cards, with Stripe Payment Gateway integration.
Customers can view their previous purchase reports.
Sales and discounts are available to customers.

3.1.4 Customer Support:

Customer support is available, with a dedicated customer service team.
Customers can use a chat feature to communicate with sellers and admin.

3.1.5 Seller Features:

Sellers can manage their profiles and inventory.
They can apply sales and discounts to their products.
Sellers can chat with customers during the purchase process.

3.1.6 Admin Features:

Admin can manage user accounts (both customers and sellers).
They can view and manage all products.
Admin can generate various reports, including seller and product ratings, order details, and payment information.
Email communication with users is facilitated.
Admin can switch between Light and Dark modes for the interface.
A chat room is available for communication between admin, sellers, and customers.

3.1.7 Product Management:

Sellers can upload and manage motorbikes and accessories.
Admin can delete products or listings that do not meet quality standards.

3.1.8 Reviews and Ratings:

Customers can leave reviews and ratings for products and services.
Ratings are categorized, including customer ratings and product ratings.

3.1.9 Role-Based Access:

The system has three types of users: Admin, Seller, and Customer.

3.1.10 Database Management:

The system maintains various database tables to store user, product, order, and other relevant data.

3.1.11 Email Communication:

Email communication is available for users to contact each other and for admin to contact users.

3.1.12 Dark Mode:

Users have the option to switch between Light and Dark modes for the interface.

3.1.13 Chat Functionality:

A chat room is implemented for real-time communication between users and admin.

3.1.14 Payment Gateway:

Integration with the Stripe Payment Gateway for secure payment transactions.

3.1.15 Reports:

The system generates various reports for admin, sellers, and customers, including sales, product ratings, and order details.

3.1.16 Product Categories:

Products are organized into categories for easy navigation.

3.1.17 Membership Management:

Admin can manage seller memberships.

3.1.18 Customer Conversion:

Customers can apply to become sellers, subject to admin approval.

3.2 Non-Functional requirements

Non-functional requirements are essential characteristics that specify how a software system should perform or behave, rather than defining what the system should do functionally. These requirements encompass various qualities, such as **performance**, **reliability**, **security**, **usability**, and **scalability**, among others. They are crucial for ensuring that the software not only meets its functional objectives but also adheres to critical aspects related to speed, robustness, user experience, and more. Non-functional requirements provide a set of criteria that help developers, designers, and stakeholders evaluate and benchmark the overall quality and performance of a software system, thus contributing to its success in real-world use.

3.3 Design and Infrastructure Constraints

3.3.1 Scalability:

The system must be designed to handle potentially high traffic and a growing number of users. The infrastructure should support horizontal scaling to accommodate increased load.

3.3.2 Security:

The website should adhere to industry-standard security practices, including encryption for data transmission and storage, secure user authentication, and protection against common web vulnerabilities (e.g., SQL injection, cross-site scripting).

3.3.3 Performance:

The system should deliver responsive performance, with fast page loading and smooth user interactions. This might require optimizing code, using content delivery networks (CDNs), and implementing caching mechanisms.

3.3.4 Data Storage:

Consider the need for a reliable and scalable database system to store product information, user data, order details, and other critical data. The database should support efficient querying and data retrieval.

3.3.5 Backup and Disaster Recovery:

Implement regular data backups and a disaster recovery plan to ensure that data is not lost in case of system failures or other emergencies.

3.3.6 Redundancy and Failover:

Design the system with redundancy and failover mechanisms to minimize downtime. This includes redundant servers, load balancing, and failover strategies for critical components.

3.3.7 Compliance:

Ensure that the system complies with legal and regulatory requirements, especially in the areas of data privacy, online payments, and e-commerce operations. This may include compliance with GDPR, PCI DSS, and other relevant standards.

3.3.8 Third-Party Integrations:

Plan for integrating third-party services like the Stripe Payment Gateway and email communication systems. Ensure these integrations are well-documented and that potential changes in third-party APIs are accommodated.

3.3.9 Infrastructure Costs:

Consider the budget constraints for maintaining servers, databases, and other infrastructure components. Optimize the infrastructure to minimize operational costs.

3.3.10 Cross-Browser Compatibility:

Design the user interface to be compatible with various web browsers and screen sizes to ensure a consistent user experience.

3.3.11 Mobile Responsiveness:

Ensure that the website is responsive to different mobile devices and screen sizes, as a significant portion of users may access the site from mobile devices.

3.3.12 Technology Stack:

Specify the technology stack and libraries to be used in the development of the website, including the programming languages, web frameworks, and third-party tools.

3.3.13 User Load Testing:

Conduct load testing to understand the system's performance under heavy user loads and ensure that it can handle peak traffic without performance degradation.

3.3.14 Geographical Considerations:

If the website is intended for a global audience, consider infrastructure for content delivery and hosting that provides low latency and high availability in various regions.

3.3.15 Accessibility:

Ensure that the website is designed and developed to be accessible to individuals with disabilities, adhering to WCAG (Web Content Accessibility Guidelines) standards.

3.3.16 Version Control and Deployment:

Implement version control for code management and have a well-defined deployment process to maintain consistency and manage changes.

3.3.17 Data Retention:

Define data retention policies, particularly for customer data, in line with data protection regulations. Ensure that obsolete data is appropriately purged.

3.3.18 Scalable Chat Service:

Implement a chat system that can handle real-time communication between users without performance bottlenecks and consider the scalability of this feature.

4. Respond to Changes

4.1 Level of Change

Functional and non-functional requirements indicate a high level of change in terms of system scalability and security. The need for scalability improvements is critical to support potential high traffic. Enhancing security measures is also a high-level change, given the importance of protecting user data and transactions.

4.2 Priority of the Change

The highest priority is assigned to security improvements. Ensuring data security and user trust is critical in an e-commerce platform. Scalability is the next in priority due to its direct impact on user experience. Other changes, such as performance optimization and data storage enhancements, follow in priority.

4.3 Communication

Communication about these changes should be comprehensive. For high-priority changes, such as security and scalability, the entire development and infrastructure teams should be informed through meetings and documentation. Performance-related changes require communication with the development team, while data storage improvements should involve the database administration team.

4.4 Cost

High-priority changes like security and scalability may require significant budget allocation. Security tools, penetration testing, and encryption measures come with costs. Scalability improvements, including server resources and load balancing, also have budget implications. Performance enhancements might involve costs for performance testing tools. Database improvements, while less costly, may require some budget allocation for maintenance and potential upgrades.

4.5 Approval

Approval for high-priority changes, particularly security measures, should be obtained from the security officer or compliance team. Scalability changes need approval from the project manager or stakeholders. Performance-related changes require approval from the project manager or lead developer. Database changes should be approved by the database administrator or DBA team.

4.6 Updating Documentation (If necessary)

Documentation should be updated as follows (if necessary):
Security and compliance documentation should be revised to include the new security measures.
Scalability improvements should be documented in the system architecture documentation.
Performance guidelines and best practices should be updated to reflect the performance enhancements.
Database schema documentation should be updated to reflect changes and optimizations.

4.7 Change log (For previous changes)

A change log should be maintained to track changes and updates, including:
Dates and descriptions of security measures, patches, and updates.
Scalability changes and their impact on the system.
Performance-related changes, optimizations, and their outcomes.
Database schema changes and updates, including any alterations in data storage.

5. Approval

5.1 Approval Criteria:

5.1.1 Approvers:

For high-priority security changes, the approvers include the Security Officer and Compliance Team.
Scalability improvements require approval from the Project Manager and relevant stakeholders.
Performance-related changes need approval from the Project Manager or Lead Developer.
Database schema changes and optimizations are subject to approval by the Database Administrator or DBA Team.

5.1.2 Review Process:

Each change proposal, whether related to security, scalability, performance, or data storage, should undergo a thorough review process.
Reviewers should assess the proposed changes based on their impact, feasibility, and alignment with project objectives.
The review process should include security assessments, scalability assessments, code reviews, or database schema reviews, depending on the nature of the change.

5.1.3 Revision Process:

If the proposed change does not meet the initial criteria or requirements, it should be subject to a revision process.
The revision process may involve addressing feedback from reviewers, making necessary adjustments, and re-submitting the proposal for review.

5.1.4 Comments and Feedback:

Throughout the review and revision process, comments and feedback should be documented for each change proposal.
Reviewers and approvers should provide clear and constructive feedback to ensure that changes meet the required standards.

5.1.5 Approval Signature

Approvals for each type of change, whether security, scalability, performance, or data storage, should be accompanied by the digital signatures of the respective approvers.

5.1.5.1 For Point 1):-

Developer / Company **Client / Stakeholder**

_____ _____

5.1.5.2 For Point 2):-

Developer / Company	**Client / Stakeholder**
_____	_____

5.1.5.3 For Point 3):-

Developer / Company	**Client / Stakeholder**
_____	_____

5.1.6 Approval Date:

Each approved change proposal should be assigned an approval date, indicating when it was reviewed and approved.

Reviewed Date	**Approval Date**
10/10/2023	10/10/2023

5.2 Changes:

For changes that are approved, a change log should be maintained to track the details of the changes, including dates and descriptions of security measures, scalability improvements, performance optimizations, and database schema changes.

5.3 Rejections Criteria

The rejection criteria should be clearly defined for each type of change. If a change proposal does not meet the criteria for security, scalability, performance, or data storage, it may be rejected. Rejection criteria should be specific and based on the project's requirements and standards.