Software Requirements Specification

for

VAMS

**Version 1.0**

**Prepared by**

**Group Name: VAMS**

| **Varun Nair** | **60004210197** | **varunrnair2002@gmail.com** |
| --- | --- | --- |
| **Mohit Dhatrak** | **60004210214** | **msdhatrak25@gmail.com** |
| **Smit Dama** | **60004210244** | **smitdama09@gmail.com** |
| **Aradhya Sakalley** | **60004210253** | **sakalleyaradhya@gmail.com** |
|  |  |  |

| **Instructor:** | **Dr Kiran Bhowmick** |
| --- | --- |
| **Course:** | **Software Engineering** |
| **Lab Section:** | **C32** |
| **Teaching Assistant:** | **NA** |
| **Date:** | **23/02/2024** |

**Contents**

**Revisions iii**

**1** **Introduction 1**

1.1 Document Purpose 1

1.2 Product Scope 1

1.3 Intended Audience and Document Overview 1

1.4 Definitions, Acronyms and Abbreviations 1

1.5 Document Conventions 1

1.6 References and Acknowledgments 2

**2** **Overall Description 3**

2.1 Product Perspective 3

2.2 Product Functionality 3

2.3 Users and Characteristics 3

2.4 Operating Environment 3

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

**3** **Specific Requirements 5**

3.1 External Interface Requirements 5

3.2 Functional Requirements 6

3.3 Behaviour Requirements 6

**4** **Other Non-functional Requirements 7**

4.1 Performance Requirements 7

4.2 Safety and Security Requirements 7

4.3 Software Quality Attributes 7

**5** **Other Requirements 8**

**Appendix A – Data Dictionary 9**

**Appendix B - Group Log 10**

**Revisions**

| **Version** | **Primary Author(s)** | **Description of Version** | **Date Completed** |
| --- | --- | --- | --- |
| 1.0 | Varun Nair,  Mohit Dhatrak,  Smit Dama,  Aradhya Sakalley | Initial version of SRS. | 23/02/24 |

# Introduction

VAMS is a mobile application designed to revolutionize urban mobility and delivery services. It allows restaurants to send delivery requests, which are then displayed to users who can choose to fulfill these tasks. Users can rent e-scooters from nearby stations to complete the delivery tasks and are paid for their services. Expanding its services further, VAMS incorporated an innovative e-scooter rental system, allowing users to easily rent e-scooters from nearby stations to complete their delivery tasks. This integration of e-scooter rentals with delivery services not only provides users with a convenient and eco-friendly mode of transportation but also opens up new opportunities for individuals looking to earn by completing delivery tasks. Through its strategic evolution, VAMS has transformed into a comprehensive urban mobility and delivery platform, seamlessly integrating food, transportation, and courier services. Its commitment to innovation and user-centric offerings positions VAMS as a dynamic player in the rapidly evolving landscape of urban mobility and delivery services.

## Document Purpose

This document serves as the Software Requirements Specification (SRS) for the VAMS mobile application, version 1.0. Its purpose is to clearly outline the technical aspects of the product we aim to develop. It will detail how the initial version of the application will interact with users, hardware, software, and other interconnected systems. By specifying the requirements, technical details, and limitations of the project, this document will guide us in utilizing all factors correctly, meeting all product requirements, and delivering the best possible product to our users.

## Product Scope

The VAMS mobile application is designed to offer a convenient and efficient urban mobility and delivery service. It includes features such as user registration, delivery request management, e-scooter rental, payment processing, and user feedback. Users can register and log in to the app to view available delivery tasks, rent e-scooters from nearby stations, complete delivery tasks, and receive payment for their services. The application aims to provide a seamless experience for users looking to earn by completing delivery tasks while also promoting eco-friendly transportation options through e-scooter rentals.

## Intended Audience and Document Overview

This document is structured to provide a comprehensive overview of the VAMS mobile application project. It begins with an introduction to the app's purpose and features, followed by a description of its intended audience and the document's organization. The document then outlines the technical aspects of the project, including its requirements, interfaces, and constraints. The intended audience for this document includes developers, testers, technical personnel, and project stakeholders involved in the development, testing, and deployment of the VAMS mobile application. It aims to provide clear and concise information about the application's functionality, requirements, and technical specifications to facilitate the development process and ensure the successful delivery of the product.

## Definitions, Acronyms and Abbreviations

* AES: Advanced Encryption Standard
* API: Application Programming Interface
* FTP: File Transfer Protocol
* GPU: Graphics Processing Unit
* GPS: Global Positioning System
* GUI: Graphical User Interface
* HTTP: Hypertext Transfer Protocol
* JSON: JavaScript Object Notation
* RAM: Random Access Memory
* SQL: Structured Query Language
* SSD: Solid State Drive
* SRS: Software Requirements Specification
* SSL: Secure Sockets Layer
* TLS: Transport Layer Security
* UI: User Interface
* URL: Uniform Resource Locator
* XML: Extensible Markup Language

## Document Conventions

The following conventions were followed while creating the document:

We have used the IEEE standards for document formatting.

Overall Description:

* The font used is Arial, font size for title is 14 and font size for text is 12.
* Italics have been used for comments.
* 1” margin has been maintained throughout the document.
* The text is single spaced.

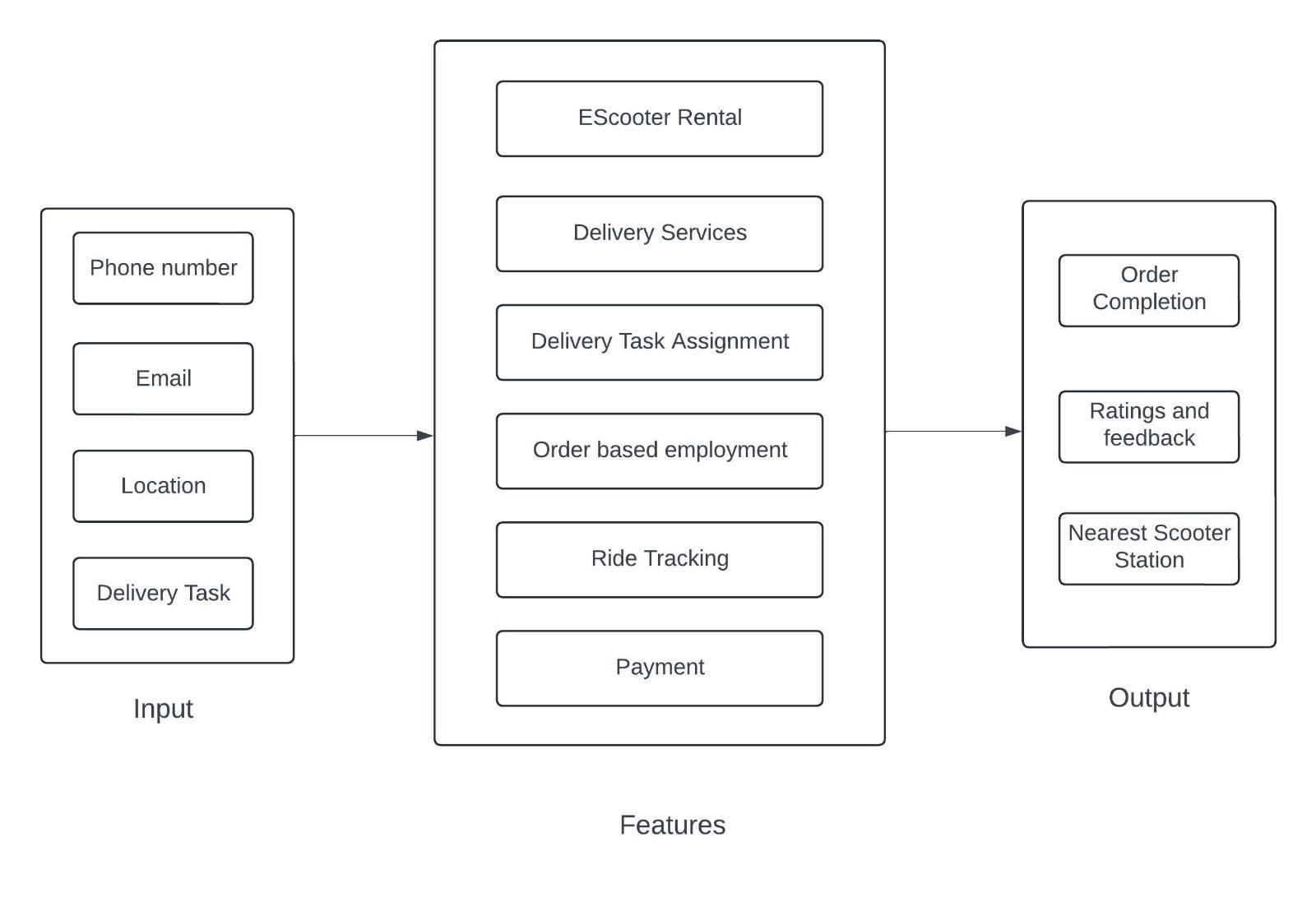
## References and Acknowledgments

1. Swiggy’s official website: [Swiggy](https://www.swiggy.com/)
2. Zomato’s official website: [Zomato](https://www.zomato.com/)
3. Google Material Design for e-commerce: [Material Design](https://material.io/design)
4. IEEE Xplore: [IEEE Xplore](https://ieeexplore.ieee.org/)

# Overall Description

## Product Perspective

VAMS is envisioned as an innovative platform that integrates urban mobility and delivery services, providing users with a convenient and efficient way to fulfill delivery tasks while promoting eco-friendly transportation options. It represents a new, self-contained product designed to enhance urban mobility and delivery experiences. The application serves as a central hub for users looking to earn by completing delivery tasks and for those seeking a sustainable mode of transportation through e-scooter rentals.



The VAMS application interacts with various external interfaces, including user devices (smartphones, tablets, etc.), e-scooter stations, payment gateways, and delivery task management systems. User interaction points encompass features such as user registration, delivery request management, e-scooter rental, payment processing, and user feedback. This diagram illustrates the interconnectedness of VAMS with its environment, emphasizing its role as a comprehensive platform for users to engage with urban mobility and delivery services seamlessly.

## Product Functionality

* **Delivery Task Management**: Restaurants can send delivery requests to the app, which are then displayed to users.
* **E-Scooter Rental**: Users can view nearby e-scooter stations on a map and rent e-scooters for a specified duration.
* **Payment Processing**: Users can pay for e-scooter rentals and delivery tasks through the app, with payment information stored securely.
* **User Feedback**: Users can provide feedback on completed delivery tasks and e-scooter rentals, helping to improve the app's services.

## Users and Characteristics

1. Delivery Task Users:

* **Characteristics:** These users are individuals looking to earn by completing delivery tasks. They use the VAMS app to view and select available delivery tasks, rent e-scooters, and complete delivery tasks.
* **Usage Frequency:** High, as they rely on the app for earning opportunities.
* **Technical Expertise:** Moderate to low technical expertise required, as the app is designed to be user-friendly and intuitive.
* **Security/Privilege Levels:** Standard access levels, with no special privileges beyond basic user functionalities.
* **Importance:** Crucial for the app’s core functionality, as they drive the completion of delivery tasks and contribute to the platform’s success.

1. E-Scooter Rental Users:

* **Characteristics:** These users are individuals looking to rent e-scooters for personal use or for completing delivery tasks. They use the VAMS app to locate nearby e-scooter stations, check e-scooter availability, and rent e-scooters.
* **Usage Frequency:** Moderate to high, depending on their need for e-scooter rentals.
* **Technical Expertise:** Similar to delivery task users, with moderate to low technical expertise required for navigation and usage.
* **Security/Privilege Levels:** Standard access levels, similar to delivery task users.
* **Importance:** Important for promoting eco-friendly transportation options and supporting users in completing delivery tasks efficiently.

1. Administrators/Managers:

* **Characteristics:** These users have administrative privileges and are responsible for managing aspects of the VAMS platform, such as e-scooter station management, delivery task management, and user support.
* **Usage Frequency:** Moderate, as they perform administrative tasks periodically to maintain and optimize the platform.
* **Technical Expertise:** Higher technical expertise required, as they may need to manage backend systems and databases.
* **Security/Privilege Levels:** Elevated access levels with administrative privileges to manage and oversee various aspects of the platform.
* **Importance:** Crucial for the smooth operation and management of the VAMS platform, ensuring its functionality and efficiency.

**Importance of Users:**Regular users are the most important users for VAMS, as they form the core user base and drive the app’s daily usage and engagement. Their satisfaction and continued usage are essential for the success and sustainability of the platform. Occasional users and administrators/managers are also significant but hold comparatively less importance in terms of daily usage and overall user engagement. However, their satisfaction and effective management are still crucial for the overall functionality and success of VAMS.

## Operating Environment

VAMS operates within a versatile environment, ensuring compatibility with various hardware platforms, operating systems, and web browsers to provide widespread accessibility and seamless performance across different devices.

* **Hardware Platform:** VAMS is designed to operate on standard computing devices such as smartphones, tablets, laptops, and desktop computers, ensuring accessibility across a wide range of hardware platforms.
* **Operating Systems:** Compatible with popular operating systems including but not limited to: Android (versions X.X and above), iOS (versions X.X and above), Windows (versions X and above), macOS (versions X and above), Linux (various distributions), Others (e.g., iPadOS, Wear OS, watchOS).
* **Web Browsers:** Optimized for use on major web browsers such as: Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge, Brave, and Others with modern standards-compliant rendering engines.

**Minimum Platform Requirements:**

* **Android:** Version 7.0 (Nougat) and above
* **iOS:** Version 12.0 and above
* **Windows:** Windows 10 and above
* **macOS:** macOS 10.12 (Sierra) and above
* **Linux:** Various distributions (specific requirements may vary)
* **Web Browsers:** Google Chrome (latest version), Mozilla Firefox (latest version), Apple Safari (latest version), Microsoft Edge (latest version), Brave (latest version), Others with modern standards-compliant rendering engines.

## Design and Implementation Constraints

Constraints for VAMS:

1. **Hardware Limitations:** The application must be optimized to run efficiently on a variety of devices, including smartphones, tablets, and computers, to ensure a seamless user experience. It should take into account the varying hardware specifications of these devices.
2. **Integration with External Services:** VAMS’s functionality relies on integration with external services such as e-scooter stations, payment gateways, and delivery task management systems. Compatibility and reliability of these services are essential for the app’s functionality.
3. **Security Considerations:** Ensuring the security of user data, payment transactions, and communication channels is paramount. Compliance with industry standards and regulations, as well as implementing robust encryption and authentication mechanisms, are essential constraints that must be addressed.
4. **Performance Optimization:** The application must be optimized for performance to handle high volumes of traffic during peak usage times. This requires considerations for server scalability, database optimization, and efficient code architecture to ensure responsiveness and minimal downtime.
5. **User Accessibility:** VAMS should be designed to be accessible to users with disabilities, adhering to accessibility standards and guidelines. This includes features such as screen reader compatibility, keyboard navigation, and alternative text for visual elements, ensuring inclusivity for all users.
6. **Data Handling and Storage:** Efficient handling and storage of data, including user profiles, delivery task information, and e-scooter rental history, impose constraints on database design and management. Scalability and data integrity must be ensured to accommodate growth and maintain system performance.

These constraints influence decisions regarding technology selection, architecture design, and implementation strategies for VAMS. Addressing these constraints effectively is crucial for delivering a robust, secure, and user-friendly application that meets the needs of its users and stakeholders.

## User Documentation

For VAMS, the user documentation will include a comprehensive user manual outlining the app's features and functionalities. It will provide guidelines for navigating and utilizing the software effectively. Additionally, an online help system will be available within the app, offering contextual assistance and tutorials to guide users through various tasks and workflows. A detailed FAQ section will also be provided for quick reference and troubleshooting common issues. Users will have access to a dedicated "Contact Us" feature, allowing them to easily reach customer support representatives for assistance with any inquiries, concerns, or technical issues they may encounter while using the VAMS platform.

## Assumptions and Dependencies

1. E-Scooter Availability: The assumption that there will be a sufficient number of e-scooters available at each station for users to rent is crucial. Any limitations in e-scooter availability could impact user satisfaction and the ability to fulfill delivery tasks efficiently.

2. Integration with Mapping Services: Assumptions regarding seamless integration with mapping services for locating e-scooter stations and planning delivery routes are significant. Any issues or limitations in these integrations could disrupt the functionality of VAMS, affecting user experience and task completion efficiency.

3. User Adoption and Task Completion: Assumptions about user adoption rates, engagement levels, and completion of delivery tasks are essential. If these assumptions are inaccurate, it could impact the availability of delivery services and the overall success of VAMS.

4. E-Scooter Maintenance and Service: The assumption that e-scooters will be regularly maintained and serviced to ensure their proper functioning is critical. Any disruptions in e-scooter availability due to maintenance issues could affect the efficiency of delivery task completion.

These assumptions shape the development and operation of VAMS and should be carefully considered and monitored to ensure the platform's success.

# Specific Requirements

## External Interface Requirements

### User Interfaces

1. Home Screen: The home screen serves as the central hub for users, providing quick access to key functionalities such as viewing available delivery tasks, renting e-scooters, and managing tasks. It may also display personalized recommendations based on user preferences and task history.

2. Delivery Task Listings: Users can browse available delivery tasks, including task details such as task name, pickup/delivery locations, task rewards, and task deadlines. They can select a task to view more details and choose to accept or reject it.

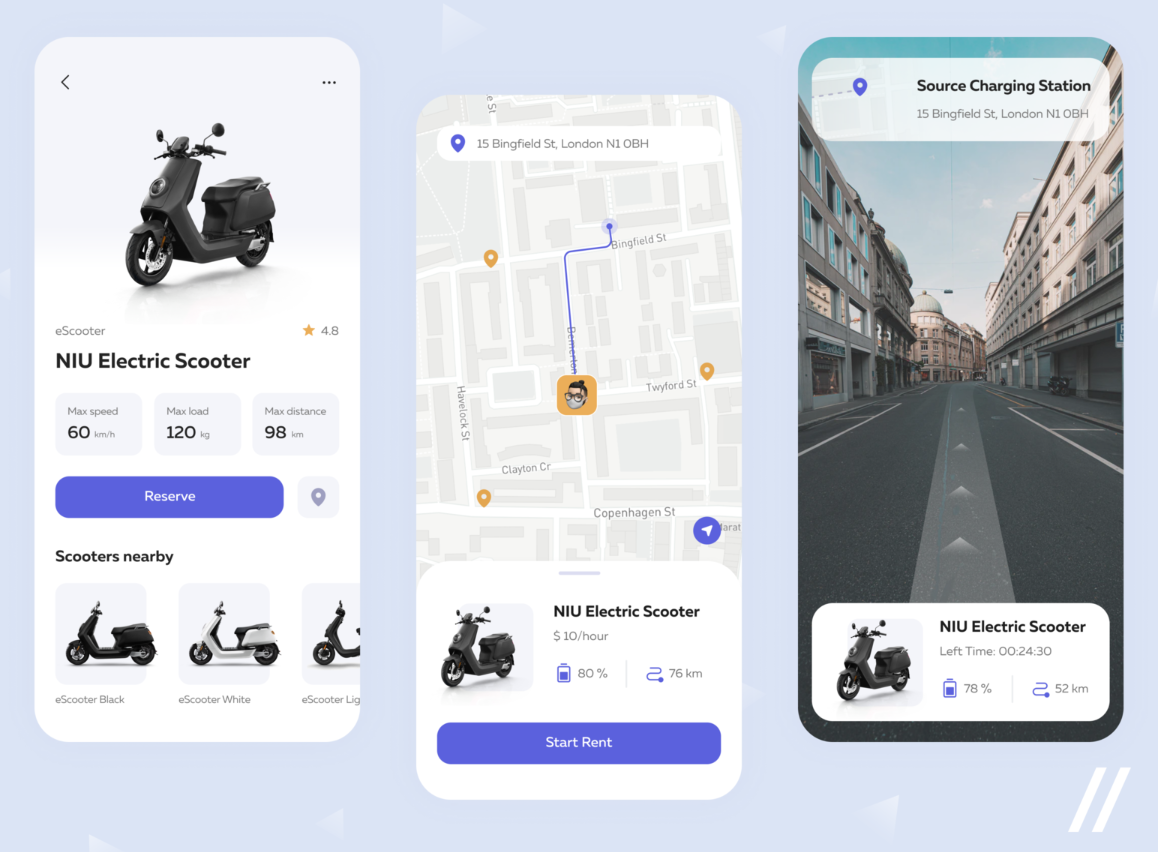
3. E-Scooter Rental: Users can view nearby e-scooter stations on a map and check the availability of e-scooters at each station. They can select a station, view e-scooter details (e.g., battery level, type), and rent an e-scooter for a specified duration.

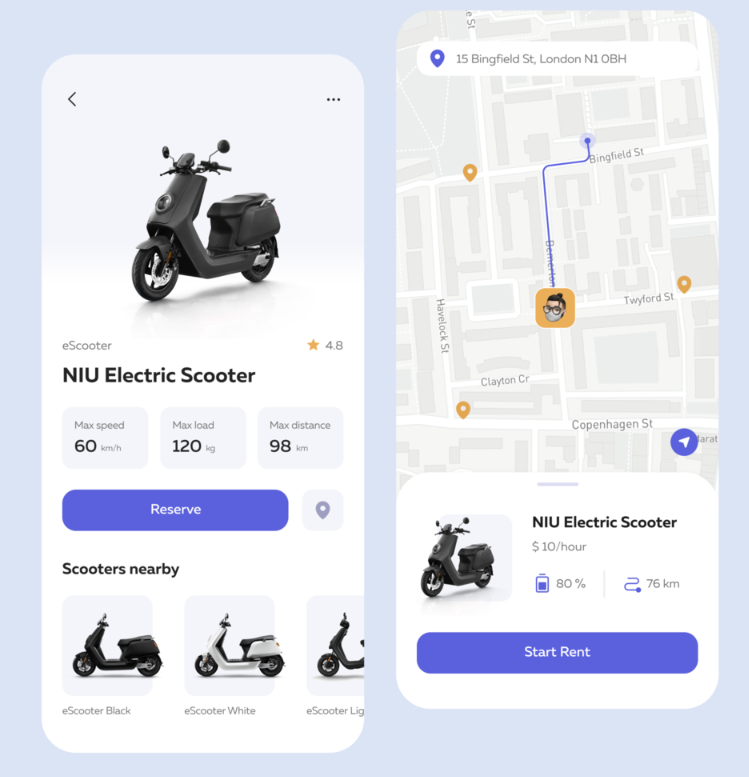
4. Task Management: After accepting a delivery task, users can manage the task details, including navigation to the pickup and delivery locations, task completion status, and submitting proof of delivery (e.g., photos).

5. Profile Settings: Users can create and manage their profiles, updating personal information, payment methods, and communication preferences. They can also view their task history, earnings, and ratings from completed tasks.

6. Checkout and Payment: The checkout process includes a summary of the rental charges for the e-scooter and any additional fees (e.g., helmet rental). Users can choose from various payment options (credit/debit cards, digital wallets) and complete transactions securely.

7. Customer Support: Accessible from within the app, the customer support interface allows users to seek assistance, report issues, or contact support representatives via chat or email for any queries related to delivery tasks, e-scooter rentals, or app functionality.





### Hardware Interfaces

Server:

* RAM: 8GB
* Storage: 1 TB SSD
* Processor: Intel Pentium 4 processor or later that’s SSE2 capable
* GPU: Nvidia GTX 1050

User Device:

* RAM: 6 GB
* Storage: 128 GB Storage
* GPS Sensor

Server Interface:  
The server interface for VAMS facilitates communication between the software application and the server hardware. It manages data storage, retrieval, and processing, ensuring efficient handling of user requests and interactions. The server must meet specified hardware requirements, including 8GB of RAM, 1TB of SSD storage, an Intel Pentium 4 processor or later that’s SSE2 capable, and an Nvidia GTX 1050 GPU for enhanced processing capabilities. This interface may utilize standard communication protocols such as HTTP or HTTPS to ensure secure data transmission between the software and server components.

User Device Interface:  
VAMS’s user device interface enables interaction between the software application and various user devices, including smartphones, tablets, laptops, and desktop computers. It ensures compatibility across different device types and operating systems, allowing users to access the application seamlessly. The interface leverages device-specific features such as touchscreens, GPS sensors, and other hardware components to provide a rich and interactive user experience. VAMS may utilize platform-specific development frameworks or libraries, such as React Native or Flutter, to build native or cross-platform applications optimized for different device environments. Additionally, the interface may incorporate standard communication protocols such as TCP/IP or Bluetooth to enable data exchange between the software and user devices.

### Software Interfaces

VAMS is designed to be compatible with a variety of operating systems to ensure widespread accessibility across different devices and platforms. Supported operating systems include:

* Windows
* MacOS
* iOS
* Android
* Linux

The interface with the operating system is crucial for VAMS to ensure seamless compatibility and functionality across different platforms. By supporting a wide range of operating systems, including Windows, MacOS, iOS, Android, and Linux,

VAMS aims to reach a diverse user base and provide a consistent user experience regardless of the device or platform used. The software components of VAMS interact with the underlying operating system to access system resources, manage user interface elements, and perform platform-specific tasks such as file operations, network communication, and device integration. This interface allows VAMS to leverage the capabilities and features of each supported operating system, enabling optimal performance and usability for users across various devices and environments.

### Communications Interfaces

VAMS relies on efficient communication protocols to facilitate seamless interaction between the software components, user devices, and external services. The application utilizes HTTP protocols for servicing requests and transmitting data, typically in JSON format, to ensure compatibility and interoperability across different platforms and systems. This enables secure and reliable communication between the client-side application and the server-side infrastructure, allowing for the exchange of information related to user requests, delivery tasks, e-scooter rentals, and payments.

To ensure data security during transmission, VAMS implements encryption using the AES (Advanced Encryption Standard) protocol to protect sensitive information from unauthorized access or interception. By encrypting data before transmission, VAMS enhances the confidentiality and integrity of user data, mitigating the risks associated with potential security threats or breaches. Additionally, the application may incorporate mechanisms for data synchronization to maintain consistency across distributed systems and ensure real-time updates and access to the latest information for users and stakeholders.

## Functional Requirements

1. User Authentication:

* Login/Signup: Users can register for a new account or log in to an existing account to access VAMS's features. The system securely stores and manages user credentials, ensuring data confidentiality and integrity.

2. Delivery Task Management:

* Task Assignment: Users can view available delivery tasks and choose to accept or reject them based on their preferences and availability.
* Task Details: Users can access detailed information about each delivery task, including pickup and drop-off locations, package details, and delivery instructions.
* Task Status Tracking: Users can track the status of their assigned delivery tasks in real-time, receiving updates on pickup, transit, and delivery.
* Task Completion: Users can mark delivery tasks as completed once they have successfully delivered the package to the intended recipient.

3. E-Scooter Rental:

* Rental Booking: Users can book e-scooters for rental, selecting pickup locations and rental durations.
* Rental Details: Users can view rental details, including e-scooter specifications, rental fees, and terms of use.
* Rental Status: Users can track the status of their e-scooter rentals, including pickup and return times, and receive notifications and updates throughout the rental period.
* Rental Return: Users can mark e-scooter rentals as returned once they have finished using the e-scooter and returned it to the designated location.

4. Payment Processing:

* Payment Options: Users can choose from various payment methods, including credit/debit cards, digital wallets, and cash on delivery, to complete transactions securely.
* Payment Confirmation: Users receive confirmation of successful payments for delivery tasks and e-scooter rentals.

5. User Profile Management:

* Profile Settings: Users can update personal information, delivery addresses, payment methods, and communication preferences in their user profiles.
* Task History: Users can view past delivery tasks and e-scooter rentals in their task history.

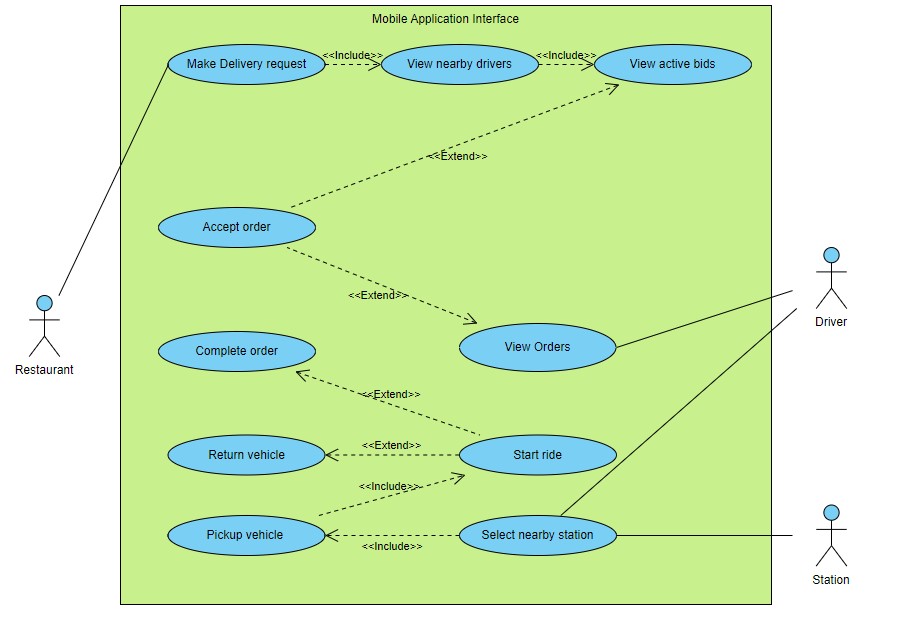
6. Customer Support:

* Contact Support: Users can access customer support options, including live chat, email, or phone support, to seek assistance, report issues, or provide feedback.

These functional requirements cover the core operations and features of VAMS, ensuring a comprehensive and user-friendly experience for its users across different use cases and scenarios.

## Behaviour Requirements

### Use Case View

**

# Other Non-functional Requirements

## Performance Requirements

1. Response Time:

* Any user action within the VAMS application, such as browsing delivery tasks, accepting orders, or updating task status, should result in a response time of less than 3 seconds to ensure a smooth and seamless user experience.

2. Task Assignment Time:

* The time taken to assign delivery tasks to users should not exceed 5 seconds, including task validation, user verification, and task assignment confirmation to the user.

3. Task Status Update Time:

* Users should receive real-time updates on the status of their assigned delivery tasks with a minimum update frequency of every 30 seconds to track the progress accurately and ensure timely completion.

4. Task Completion Time:

* Users should be able to mark assigned delivery tasks as completed within 10 seconds of successful delivery to update the task status and notify the system.

5. Customer Support Response Time:

* Customer support inquiries and issues reported by users should be acknowledged and addressed within 24 hours to provide timely assistance and resolution to users' concerns.

These performance requirements are essential to ensure the efficient and effective operation of VAMS, providing users with a reliable and responsive platform for managing delivery tasks and e-scooter rentals.

## Safety and Security Requirements

**Safety Requirements:**

1. User Safety: VAMS must ensure the safety of users taking up delivery tasks and using the e-scooters provided by the app. This includes providing guidelines for safe riding practices, emphasizing the use of safety gear, and conducting periodic safety checks on e-scooters.

2. Delivery Safety Protocols: VAMS should establish safety protocols for delivery tasks to minimize the risk of accidents or harm during transportation. This may include guidelines for safe handling of packages, secure packaging practices, and training for safe navigation in urban environments.

**Security Requirements:**

1. User Authentication: Implement strong authentication mechanisms, including multi-factor authentication, to ensure that only authorized users can access the VAMS platform and its features securely.

2. Data Privacy Protection: Implement robust measures to protect user data privacy and confidentiality, adhering to relevant data protection laws and regulations. This includes encryption of sensitive information, secure storage practices, and user consent mechanisms for data processing.

3. Secure Data Transmission: Use encryption protocols such as TLS to secure data transmission between the client application and server, preventing interception or tampering of sensitive information during transit.

4. Secure Payment Processing: Ensure that all payment transactions within VAMS are processed securely, following PCI DSS compliance standards to protect users' payment information from unauthorized access or fraud.

5. Regular Security Updates: Keep the VAMS application and its underlying infrastructure updated with the latest security patches and fixes to address any potential vulnerabilities and protect against security threats.

6. Security Audits and Testing: Conduct regular security audits and penetration testing to identify and remediate security vulnerabilities proactively, ensuring the integrity and resilience of the application against potential cyber threats.

## Software Quality Attributes

**4.3.1 Reliability:**

* High Availability: VAMS should strive for a minimum uptime of 99.9%, ensuring that the service is consistently available to users without significant downtime or interruptions.
* Accurate Data: The information provided by VAMS, including delivery task details, e-scooter availability, and delivery status, should be accurate and reliable, with a maximum error rate of 1% to maintain user trust and satisfaction.

**4.3.2 Usability:**

* Intuitive User Interface: VAMS should prioritize ease of use and intuitive design, ensuring that users can navigate the application effortlessly and perform tasks such as taking up delivery tasks and accessing e-scooters without extensive training.
* Accessible Design: The application should adhere to accessibility standards, ensuring that users with disabilities can access and use VAMS's features effectively.

**4.3.3 Security:**

* Data Protection: VAMS must implement robust security measures to protect user data from unauthorized access, tampering, or data breaches. This includes encryption of sensitive information, secure storage practices, and access controls to ensure data confidentiality and integrity.
* Secure Payment Processing: Ensure that all payment transactions within VAMS are processed securely, following PCI DSS compliance standards to protect users' payment information from fraud and theft.

**4.3.4 Maintainability:**

* Modular Design: VAMS should adopt a modular architecture that allows for easy maintenance and updates. Components should be loosely coupled, facilitating independent development, testing, and deployment of new features or bug fixes.
* Documentation: Provide comprehensive documentation for developers, including code comments, architecture diagrams, and API documentation, to facilitate understanding and maintenance of the system over time.

**4.3.5 Performance:**

* Optimized Response Time: VAMS should aim for a maximum response time of 2 seconds for key user interactions such as taking up delivery tasks, accessing e-scooters, and tracking deliveries, to ensure a smooth and responsive user experience.
* Scalability: The application should be designed to scale horizontally and vertically to accommodate increasing user traffic and load demands without sacrificing performance or reliability.

**Appendix A – Data Dictionary**

| **Field Name** | **Data Type** | **Field Size** | **Description** | **Example** |
| --- | --- | --- | --- | --- |
| user\_id | Integer | 10 | Unique identifier for users | 12345 |
| username | String | 20 | User's username | user123 |
| password | String | 20 | User's password | \*\*\*\*\*\*\*\*\* |
| email | String | 30 | User's email address | user@example.com |
| role | String | 15 | User's role (e.g., customer, admin) | customer |
| address | String | 50 | User's postal address | 123 Main St, City, Country |
| phone\_number | String | 15 | User's phone number | +1234567890 |
| order\_status | String | 20 | Status of user's food order | Delivered |
| payment\_status | String | 20 | Status of user's payment | Completed |
| parcel\_status | String | 20 | Status of user's parcel delivery | In transit |
| reservation\_id | Integer | 10 | Unique identifier for reservations | 98765 |
| order\_id | Integer | 10 | Unique identifier for food orders | 123456 |
| parcel\_id | Integer | 10 | Unique identifier for parcel delivery | 54321 |
| restaurant\_id | Integer | 10 | Unique identifier for restaurants | 67890 |
| delivery\_time | Time | 10 | Estimated delivery time for food orders | 19:00 |
| delivery\_date | Date | 10 | Estimated delivery date for food orders | 2024-01-28 |
| parcel\_type | String | 20 | Type of parcel being delivered | Small |
| parcel\_weight | Float | 5 | Weight of parcel being delivered (in kg) | 1.5 |
| reservation\_time | Time | 10 | Reservation time at restaurant | 20:00 |
| reservation\_date | Date | 10 | Reservation date at restaurant | 2024-02-01 |
| payment\_method | String | 15 | Payment method used for transaction | Credit Card |
| payment\_amount | Float | 10 | Amount paid by user | 50.00 |
| feedback | String | 100 | User's feedback on service | Excellent service! Will order again |

**Appendix B - Group Log**

| **Date** | **Actors** | **Work Done** |
| --- | --- | --- |
| 23/02/2024 | Mohit | Analysed Requirements |
| 23/02/2024 | Aradhya | Analysed Requirements |
| 23/02/2024 | Smit | Analysed Requirements |
| 23/02/2024 | Varun | Analysed Requirements |
| 01/03/2024 | Mohit | Prepared SRS |
| 01/03/2024 | Aradhya | Prepared SRS |
| 01/03/2024 | Smit | Prepared SRS |
| 01/03/2024 | Varun | Prepared SRS |