Software Requirements Specification

for

“Talk2Metro”

Let the voice be the guide of your metro journey from booking to boarding

Prepared by

Md. Sajib Uddin

ID: ASH2125001M

Tarekul Islam Tusher

ID: ASH2125003M

Md. Imran Hasan

ID: MUH2125020

Nowshin Fabiha Ibnat

ID: BKH2125031F

Institute of Information Technology

Noakhali Science and Technology University

Table of Contents

1. Introduction

The Software Requirements Specification (SRS) introduction contains the SRS's policy, scope, references, and summary. This document's goal is to gather information about proposed system as we name it as "Talk2Metro" and is to give readers a greater understanding of it by outlining the issue statement in great detail. While defining the qualities of a high-quality product, it also emphasizes the advantages and requirements of the participants. Details on the "Talk2Metro" can be found on this document.

1.1 Problem Statement

Urban commuters often face significant challenges with traditional metro-rail ticket booking methods, such as long queues, limited ticket availability, and a lack of real-time journey updates. In response to these issues, there is a pressing need for a more efficient, user-friendly solution. The problem is to develop a web-based mobile application that allows users to seamlessly book metro-rail tickets using voice commands or manual input, displays nearby stations based on the user's location, provides route maps to the selected station, facilitates secure online payments via various gateways (including cards, Bkash, and Nagad), and delivers real-time journey notifications until the destination is reached. This system must address key challenges such as ensuring accurate voice recognition, precise location services, reliable real-time updates, secure transactions, and a scalable, intuitive user interface.

1.2 Purpose

The purpose of this project is to enhance the metro-rail ticket booking experience by developing a web-based mobile application that leverages modern technologies to provide a seamless, efficient, and user-friendly process. By incorporating voice control, real-time location detection, route mapping, and secure online payments, the application aims to eliminate the inconveniences of traditional ticket booking methods. It will provide users with the convenience of booking tickets from anywhere, receiving instant confirmation through QR codes, and staying informed with real-time journey updates. This project seeks to improve overall commuter satisfaction, reduce wait times, and streamline the ticketing process, ultimately contributing to a more efficient and accessible urban transportation system.

1.3 Project Scope

User Authentication and Profile Management

- Registration and login system
- Profile management for users

Voice Control System

- Integration of voice recognition for searching stations and booking tickets
- Voice command functionality for various tasks

Location-Based Services

- Automatic detection of the user's current location
- Display of nearby metro stations based on user's location

Route Mapping

- Visualization of the route from the user's current location to the selected station
- Integration with mapping services for accurate directions

Payment Gateway Integration

- Secure online payment processing
- Support for multiple payment methods including cards, Bkash, and Nagad

Ticket Booking & QR Code Generation

- Generation of QR codes for ticket validation post-payment
- Storage and retrieval of ticket information

Real-Time Notifications

- SMS and in-app notifications for payment confirmation
- Real-time updates on journey status until the destination is reached

User Interface Design

- Responsive and intuitive UI/UX design for mobile devices
- Seamless navigation and user-friendly interface

System Security and Scalability

- Ensuring data security and privacy for users
- Designing the system to handle high user volumes efficiently

Testing and Quality Assurance

- Unit testing, integration testing, and user acceptance testing
- Continuous monitoring and feedback for improvements

Deployment and Maintenance

- Deployment of the app on relevant app stores and web hosting platforms
- Ongoing maintenance and updates based on user feedback and technological advancements.

1.4 Glossary

This section provides definitions for all document names, acronyms, and abbreviations. The application domain's terms and concepts are defined.

HTML – Hyper Text Markup Language

JS – JavaScript

CSS – Cascading Style Sheets

SRS – Software Requirement Specification

UI – User Interface

Bootstrap (Front-end framework)

API – Application Programming Interface

Django – Django is a Python framework

MySQL – My Structured Query Language

Angular – opensource JavaScript framework

1.5 References

IEEE. IEEE Std. 830 – 1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

1.6 Overview

In "Talk2Metro" project, we are excited to introduced some innovative features here. Experience the future of metro rail with our Natural Language Guidance System and QR Code-Based Ticket Validation. Get real-time assistance in Bangla or English, skip lines with QR code scanning, and stay updated with our Notification System. Enjoy seamless travel with online ticket purchases and fine payments.

2. Stakeholders and Characteristics

2.1 Commuters (End Users)

Characteristics: Individuals who frequently use metro-rail services, including students, professionals, and tourists. They seek convenience, reliability, and ease of use in booking tickets and receiving journey updates.

Needs: User-friendly interface, quick and easy booking process, secure payments, real-time notifications, and accurate route information.

2.2 Metro-Rail Authorities

Characteristics: Organizations responsible for managing metro-rail operations and services. They are interested in improving service efficiency and passenger satisfaction.

Needs: Efficient ticketing system, reduced congestion at ticket counters, accurate passenger data, and improved customer service.

2.3 Developers

Characteristics: Technical team responsible for designing, developing, testing, and maintaining the application. They require clear requirements, robust development tools, and efficient collaboration.

Needs: Well-defined project scope, access to necessary APIs and SDKs, support for integration with payment gateways, and clear communication with stakeholders.

2.4 Payment Gateway Providers

Characteristics: Companies providing secure online transaction services, such as Visa, Mastercard, Bkash, Rocket, and Nagad. They ensure smooth and secure financial transactions within the app.

Needs: Secure integration, compliance with financial regulations, transaction monitoring, and fraud prevention.

2.5 Telecommunications Providers

Characteristics: Companies providing SMS gateway services, such as Twilio, to send notifications to users. They ensure reliable delivery of messages.

Needs: Reliable API integration, secure data transmission, and scalability to handle a high volume of messages.

2.6 Customer Support Team

Characteristics: Team responsible for assisting users with issues related to the application, payments, or travel updates. They require tools to manage user queries and feedback efficiently.

Needs: Access to user data and transaction history, efficient communication tools, and training on the application's features and functionalities.

2.7 App Store Platforms

Characteristics: Platforms like Apple App Store and Google Play Store where the application will be hosted and made available for download. They ensure the app meets certain quality and security standards.

Needs: Compliance with app submission guidelines, regular updates, and security compliance.

3. Design and Implementation Constraints

In order to ensure the project's success, we used design and implementation limitations. It can also refer to a tool that enables testers and developers to view and interact with the user interface (UI) components of an application.

3.1 Language

User interface Design, usually known as UI Design, is the visual organization of the parts of a website or technological product that a user could interact with. In other words, it is the visual layout of a website. On the other hand, the code that enables a computer program or application to run and cannot be viewed by a user is referred to as the back end. The back end of a computer system is where the majority of data and operating syntax are kept and accessed. Typically, the code is comprised of one or more programming languages.

3.1.1 HTML

HTML (Hypertext Markup Language) is the code that is used to structure a web page and its content. Precisely, the coding that organizes a web page's content is called HTML (Hypertext Markup Language). With the help of HTML, you can tell a web page whether it should be recognized as a paragraph, list, heading, link, image, multimedia player, form, or any other of the many other components that are now supported, or even a new element that you design. It is the programming language for formatting web pages that is widely accepted. Small and medium-sized businesses are the main users, as they do not actually require extensive functionality on their websites. The option to utilize HTML to design the structure of my web pages was made since it is free, works with all browsers on the client's machine, and is simple to use and understand.

3.1.2 CSS

CSS is a stylesheet language used to describe the presentation of a document written in HTML or XML. CSS specifies how items should be shown in various media, including speech, paper, screens, and other media. One of the fundamental languages of the open web, CSS is defined by the W3C (World Wide Web Consortium) specification and is supported by all major browsers.

3.1.3 Bootstrap (Front-end framework)

Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It includes optional JavaScript extensions along with HTML and CSS-based design templates for navigation, buttons, forms, and other interface elements. It only addresses front-end development, unlike many web frameworks. Along with CSS, Bootstrap would be utilized to create the application's styling. Bootstrap is important in the application for the following reasons:

- **Easy to use:** Anyone can begin using Bootstrap with just a basic Knowledge of HTML and CSS.
- **Responsive features:** The responsive CSS in Bootstrap adapts to mobile devices, tablets, and desktops.
- **Mobile-first approach:** The fundamental Bootstrap framework provides mobile-first styling.

- **Browser compatibility:** All current browsers are compatible with Bootstrap (Chrome, Firefox, Internet Explorer, Safari, and Opera).

3.1.4 JavaScript

JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. JavaScript adds interactive elements to online pages that keep users engaged, whereas HTML and CSS are languages that give web pages structure and style. The prototype is a built-in property that every JavaScript object has. The prototype is itself an object, so the prototype will have its own prototype, making what's called a prototype chain. When we get to a prototype that contains null for its own prototype, the chain comes to an end.

3.1.5 Angular

Angular is a popular open-source front-end web application framework developed by Google. It is designed for building dynamic, single-page applications (SPAs) with a structured and modular approach. Angular employs TypeScript, a superset of JavaScript, which enhances code quality and maintainability with strong typing, object-oriented programming, and other advanced features.
Key features of Angular include a component-based architecture, which allows developers to build encapsulated and reusable UI components; a powerful templating syntax for creating dynamic views; dependency injection to manage service dependencies efficiently; and comprehensive tooling for testing, debugging, and optimizing applications. Angular also supports reactive programming through RxJS, making it suitable for handling asynchronous data streams and complex user interactions.

4. Requirement Specification

4.1. Functional Requirements

4.1.1 User Authentication and Profile Management

- Users must be able to register with their email and password.
- Users must be able to log in with registered credentials.
- Users must be able to manage their profile information.

4.1.2 Voice Control System

- The application must support voice commands for searching stations and booking tickets.
- The voice recognition system must be able to handle different accents and speech patterns.

4.1.3 Location-Based Services

- The application must detect the user's current location automatically.
- The application must display a list of nearby metro stations based on the user's location.

4.1.4 Route Mapping

- The application must provide a visual route map from the user's current location to the selected station.

- The route map must update in real-time as the user moves.

4.1.5 Payment Gateway Integration

- The application must support secure online payments through various methods, including credit/debit cards, Bkash, and Nagad.
- The application must handle transaction errors and provide appropriate user feedback.

4.1.6 Ticket Generation and QR Code

- Upon successful payment, the application must generate a QR code containing ticket details.
- The QR code must be unique and valid for the selected journey.

4.1.7 Real-Time Notifications

- The application must send SMS notifications upon successful payment.
- The application must provide real-time journey updates and notifications through the app.

4.1.8 User Interface

- The application must have a responsive and intuitive user interface suitable for mobile devices.
- The interface must support both portrait and landscape orientations.

4.2. Non-Functional Requirements

4.2.1 Performance Requirements

- The application must respond to user inputs within 2 seconds.
- The system should handle up to 10,000 concurrent users without performance degradation.

4.2.2 Security Requirements

- User data must be encrypted in transit and at rest.
- The application must comply with relevant data protection regulations.
- Payment transactions must be processed through secure, PCI-compliant gateways.

4.2.3 Usability Requirements

- The application must be easy to navigate with clear instructions and help options.
- The voice control system must have a high accuracy rate (95%+) in recognizing commands.

4.2.4 Scalability Requirements

- The application must be scalable to handle increasing user loads.
- The system architecture must support easy addition of new features and services.

4.2.5 Reliability Requirements

- The application must have an uptime of 99.9%.
- The system must recover gracefully from failures and provide meaningful error messages to users.

4.2.6 Compatibility Requirements

- The application must be compatible with major mobile operating systems (iOS and Android).
- The application must work on various screen sizes and resolutions.

5. Requirement Engineering Process

5.1 Requirement Elicitation Techniques

Requirements elicitation involves the process of researching and gathering system requirements from various stakeholders, including users, customers, and administrators. Different techniques are employed to effectively gather these requirements for the "Talk2Metro" project.

5.1.1 Hold Interviews

We hold discussions that can be held individually or with a small group of participants. They are an effective way to access services without spending a lot of time with participants because we meet with people to discuss only certain important requirements of this program. Negotiations are useful for obtaining individual requirements for members in organizing workshops where those members of the program come together to resolve any issues or conflicts. We mainly perform our interview based on some specific criteria.

Sample of Requirements Collection:

Requirement Elicitation Techniques	Interviews, Surveys, Focus Groups
Collected From	Users of Metro rail
Findings	**1. User Authentication and Profile Management:** Users must register with their email and a secure password. The system will validate the email format and enforce password strength policies. Users must view and update their profile information (name, email, phone number, password) with ensured data integrity and security. **2.Voice Commands for Searching Stations and Booking Tickets:** The application must support voice commands that allow users to search for metro stations and book tickets. The system should be able to process commands like "Find nearest station" or "Book a ticket to [station name]." **3.Automatic Detection of User's Current Location:**

	The application must automatically detect the user's current location using GPS or other location services. It should request the necessary permissions from the user and provide accurate location data. **4.Display of Nearby Metro Stations:** Based on the detected location, the application must display a list of nearby metro stations. The list should be updated in real-time as the user moves. The route map must update in real-time as the user moves, providing continuous guidance. It should recalculate the route if the user deviates from the suggested path. **5.Online Payment:** The application must support secure online payments through multiple methods, including credit/debit cards, Bkash, and Nagad. The system should comply with industry standards for transaction security. **6.Handling Transaction Errors and User Feedback:** The application must handle transaction errors gracefully, providing clear and actionable feedback to the user. It should include retry mechanisms and customer support contact information. **7.Ticket Generation and QR Code:** Upon successful payment, the application must generate a unique QR code containing the ticket details. The QR code should be scannable at the metro station for validation. The generated QR code must be unique and valid only for the selected journey. It should include information such as the journey date, starting station, destination, and ticket class. **8.Real-Time Notifications:** The application must send an SMS notification to the user upon successful payment. The SMS should contain the payment confirmation and ticket details. The application must provide real-time updates and notifications about the journey status through the app. This includes departure reminders, arrival updates, and any delays or changes. **9.Responsive and Intuitive User Interface:** The application must have a responsive and intuitive user interface optimized for mobile devices. It should support smooth navigation and user-friendly interactions.

5.1.2 Perform Document Analysis

Document analysis is crucial for understanding existing systems, processes, and requirements. Here's how we can perform document analysis effectively for the "TALK2METRO" project:

- **Gather Existing Documentation** This step involves collecting any existing documentation related to the project or system under review. It may include technical specifications, user manuals, system architecture diagrams, requirements documents, and any other relevant materials. Gathering existing documentation provides insights into the project's background, functionality, and features, serving as a foundation for the review process.

- **Review Textual Content** In this step, the textual content of the documentation is carefully examined and analyzed. This includes reviewing descriptions, instructions, specifications, and other textual elements to understand the project's functionality, processes, and requirements. Reviewing textual content helps identify key information, clarify ambiguities, and ensure consistency and accuracy in documentation.

- **Identify Functionalities and processes** Here, the focus is on identifying the various functionalities and processes described in the documentation. This involves breaking down the system's capabilities and operations into distinct components, such as user registration, data management, authentication, ordering process, delivery logistics, etc. Each functionality and process are analyzed to understand its purpose, interactions, and dependencies within the system.

- **Document Findings** Finally, the findings from the review process are documented systematically. This includes summarizing key points, highlighting significant observations or discrepancies, and recording any recommendations or actions needed to address identified issues. Documenting findings ensures that insights gained from the review process are captured effectively and can be used to guide future development, updates, or improvements to the project or system.

5.2 Requirement Validation

Requirement validation ensures the accuracy and quality of requirements for the "TALK2METRO" platform. To validate requirements effectively:

5.2.1 Review the Requirements

Conduct rigorous peer reviews to identify ambiguities and gaps in the requirements. Utilize a diverse team of reviewers to examine written needs, analysis models, and relevant information thoroughly.

5.2.2 Test the Requirements

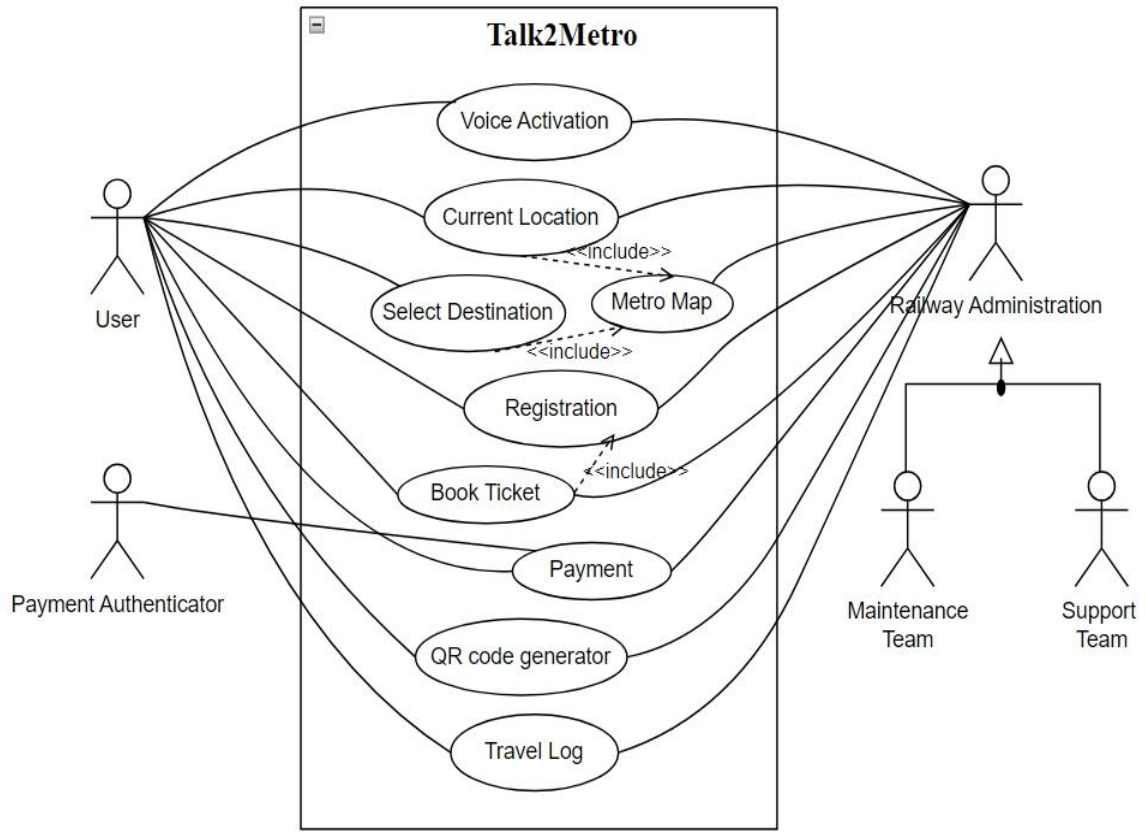
Create test cases to validate requirements and ensure they meet expected performance standards. Perform writing tests to verify the expected behavior of the "TALK2METRO" platform under

specified conditions, including user needs and system functionality.

5.2.3 Simulate the Requirements

Utilize simulation tools to simulate the proposed system and add detail to written requirements. Prototyping and simulation techniques elevate requirement validation by providing a tangible representation of the "TALK2METRO" platform's functionality.

6. Use Case Diagram



7. Use Case Descriptions

Use Case No.	01
Use Case	Voice Activation
Goal	Enable users to interact with the application using voice commands to book tickets, find stations, and get updates.
Preconditions	- User must have the application installed on their mobile device. - User must be logged in.

	- Voice recognition service must be functional.	
Success End Condition	Voice commands are accurately recognized and executed, allowing users to perform tasks hands-free.	
Failed End Condition	Voice commands are not recognized, or incorrect actions are performed.	
Primary Actors:	User	
Secondary Actors:	None	
Trigger	User initiates the voice command feature.	
Main Success Flows	**Step**	**Action**
	1	User activates the voice command feature.
	2	System prompts user for a command.
	3	User states the command (e.g., "Book a ticket to [destination]").
	4	System processes the voice input.
	5	System performs the requested action (e.g., initiates ticket booking process).
Alternative Flows	**Step**	**Branching Action**
	1	If the system does not understand the command: - System requests the user to repeat the command.
		If voice recognition fails multiple times: - System suggests switching to manual input.
Quality Requirements	**Step**	**Requirement**
	1	Accuracy of voice recognition must be above 90%.
	2	Response time for processing commands should be under 2 seconds.

Use Case No.	2	
Use Case	Current Location	
Goal	Automatically detect the user's current location to suggest nearby metro stations.	
Preconditions	User must have the application installed and be logged in. Location services must be enabled on the user's device Application has permission to access the user's location.	
Success End Condition	User's current location is accurately detected and nearby metro stations are displayed.	
Failed End Condition	User's location is not detected, or incorrect stations are suggested.	
Primary Actors:	User	
Secondary Actors:	None	
Trigger	User opens the application or requests location-based services.	
Main Success Flows	**Step**	**Action**
	1	User opens the application.
	2	System requests location data from the device.

	3	Device provides current location to the system.
	4	System displays nearby metro stations based on the user's location.
Alternative Flows	**Step**	**Branching Action**
	1	If location services are disabled: - System prompts the user to enable location services.
	2	If location cannot be determined: - System displays an error message and suggests manual input of the location.
Quality Requirements	**Step**	**Requirement**
	1	Location detection should be accurate within 50 meters.
	2	Location detection should occur within 5 seconds.

Use Case No.	3	
Use Case	Select Destination	
Goal	Allow users to select their destination station for ticket booking.	
Preconditions	User must be logged in. Application must have access to the list of all metro stations.	
Success End Condition	User successfully selects a destination station.	
Failed End Condition	User is unable to select a destination station.	
Primary Actors:	User	
Secondary Actors:	None	
Trigger	User initiates the ticket booking process.	
Main Success Flows	**Step**	**Action**
	1	User initiates the ticket booking process.
	2	System displays a list of available metro stations.
	3	User selects the desired destination station.
	4	System confirms the selection and proceeds to the next step in booking.
Alternative Flows	**Step**	**Branching Action**
	1	If the list of stations fails to load: - System displays an error message and retries loading the list.
	2	If the selected station is not available: - System notifies the user and asks for a new selection.
Quality Requirements	**Step**	**Requirement**
	1	Station list should load within 3 seconds.
	2	The system should handle up to 1000 stations without performance issues.

Use Case No.	4	
Use Case	Metro map	
Goal	Provide users with a visual map of the metro system and route information.	
Preconditions	- User must be logged in. - System must have access to up-to-date metro map data.	
Success End Condition	User views the metro map and route information without issues.	
Failed End Condition	Metro map fails to load or displays incorrect information.	
Primary Actors:	User	
Secondary Actors:	None	
Trigger	User requests to view the metro map.	
Main Success Flows	**Step**	**Action**
	1	User selects the option to view the metro map.
	2	System retrieves and displays the metro map.
	3	User interacts with the map to view different routes and stations.
Alternative Flows	**Step**	**Branching Action**
	1	If the metro map fails to load: - System displays an error message and retries loading the map.
	2	If the map data is outdated: - System prompts for an update and reloads the map.
Quality Requirements	**Step**	**Requirement**
	1	Metro map should be interactive and responsive.
	2	Map data should be updated regularly to ensure accuracy.

Use Case No.	5	
Use Case	Registration	
Goal	Allow users to register for an account within the application.	
Preconditions	- User must have the application installed. - User must have access to a valid email address or phone number.	
Success End Condition	User successfully registers and creates an account.	
Failed End Condition	User is unable to complete the registration process.	
Primary Actors:	User	
Secondary Actors:	Railway Administration	
Trigger	User initiates the registration process.	
Main Success Flows	**Step**	**Action**
	1	User selects the registration option.
	2	System prompts for necessary information (e.g., name, email, password).

	3	User provides the required information.
	4	System validates the information.
	5	System creates a new account and confirms registration.
Alternative Flows	**Step**	**Branching Action**
	1	If the email is already in use: - System notifies the user and prompts for a different email.
	2	If the validation fails: - System displays an error message and requests correction.
Quality Requirements	**Step**	**Requirement**
	1	Registration process should take less than 2 minutes.
	2	Validation should be accurate and secure.

Use Case No.	6	
Use Case	Book Ticket	
Goal	Enable users to book metro-rail tickets through the application.	
Preconditions	- User must be logged in. - User must have a payment method set up.	
Success End Condition	User successfully books a ticket and receives confirmation.	
Failed End Condition	User is unable to complete the booking process.	
Primary Actors:	User	
Secondary Actors:	Payment Authenticatior, Railway Administration	
Trigger	User initiates the ticket booking process.	
Main Success Flows	**Step**	**Action**
	1	User selects the option to book a ticket.
	2	System prompts for journey details (e.g., starting station, destination).
	3	User provides the required details.
	4	System calculates fare and displays it.
	5	User confirms the booking and initiates payment.
	6	System processes the payment.
	7	System generates a ticket and sends confirmation to the user.
Alternative Flows	**Step**	**Branching Action**
	1	If payment fails: - System notifies the user and prompts to retry or select a different payment method.
	2	If the selected journey is fully booked: - System notifies the user and suggests alternative times or routes.
Quality Requirements	**Step**	**Requirement**
	1	Ticket booking process should take less than 3 minutes.

	2	Payment processing should be secure and reliable.

Use Case No.	7	
Use Case	Payment	
Goal	Facilitate secure and efficient payment for ticket booking.	
Preconditions	- User must be logged in. - User must have a valid payment method.	
Success End Condition	Payment is processed successfully and ticket is booked.	
Failed End Condition	Payment fails and ticket is not booked.	
Primary Actors:	User, Payment Authentication	
Secondary Actors:	Railway Administration	
Trigger	User confirms ticket booking and initiates payment.	
Main Success Flows	**Step**	**Action**
	1	User initiates payment process.
	2	System presents available payment methods.
	3	User selects a payment method and provides necessary details.
	4	System processes the payment through the selected gateway.
	5	Payment is confirmed and system generates a ticket.
Alternative Flows	**Step**	**Branching Action**
	1	If payment authentication fails: - System notifies the user and prompts to retry or use a different method.
	2	If payment gateway is down: - System suggests alternative payment methods or retry later.
Quality Requirements	**Step**	**Requirement**
	1	Payment processing time should be under 10 seconds.
	2	Payment details should be securely encrypted.

Use Case No.	8
Use Case	QR Code Generator
Goal	Generate QR codes for ticket validation post-payment.
Preconditions	- User must have successfully booked a ticket. - System must have QR code generation capability.
Success End Condition	QR code is generated and sent to the user.
Failed End Condition	QR code generation fails.
Primary Actors:	User
Secondary Actors:	Railway Administration
Trigger	Successful completion of ticket booking and payment.

Main Success Flows	**Step**	**Action**
	1	User completes ticket booking and payment.
	2	System generates a QR code for the ticket.
	3	System sends the QR code to the user via the app and/or email.
Alternative Flows	**Step**	**Branching Action**
	1	If QR code generation fails: - System retries the process and notifies the user of any issues.
	2	If QR code cannot be sent: - System provides alternative ways to access the ticket
Quality Requirements	**Step**	**Requirement**
	1	QR code generating time should be less than 109 second
	2	QR code generator processing should be secure and reliable.

Use Case No.	9	
Use Case	Travel log	
Goal	Allow users to view and manage their travel history, including past trips and ticket details.	
Preconditions	-User must be logged in. -System must store travel history data for the user.	
Success End Condition	-User successfully views and manages their travel history.	
Failed End Condition	User is unable to access or manage their travel history.	
Primary Actors:	User	
Secondary Actors:	None	
Trigger	User requests to view their travel log.	
Main Success Flows	**Step**	**Action**
	1	User navigates to the travel log section in the application.
	2	System retrieves the user's travel history from the database.
	3	System displays the travel history, including details of past trips and tickets.
	4	User views details of a specific trip.
	5	User has the option to delete or export travel history data.
Alternative Flows	**Step**	**Branching Action**
	1	If the system fails to retrieve travel history: -System displays an error message and suggests retrying.
	2	If there are no travel records: -System displays a message indicating that there are no travel logs available.
Quality Requirements	**Step**	**Requirement**
	1	Travel log should load within 3 seconds.
	2	Data displayed should be accurate and up-to-date.

8. Activity Diagram

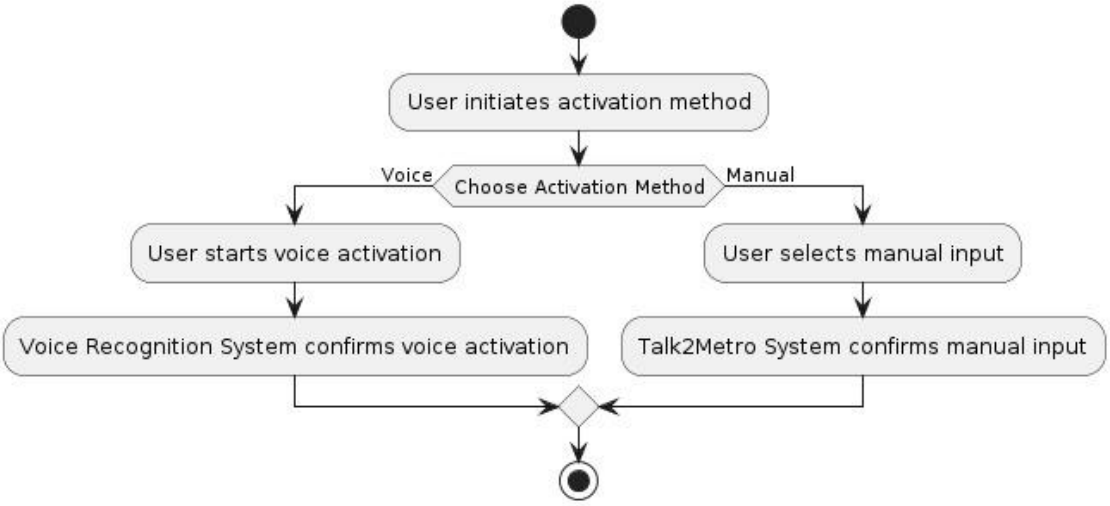


Figure:8.1 (Voice Activation)

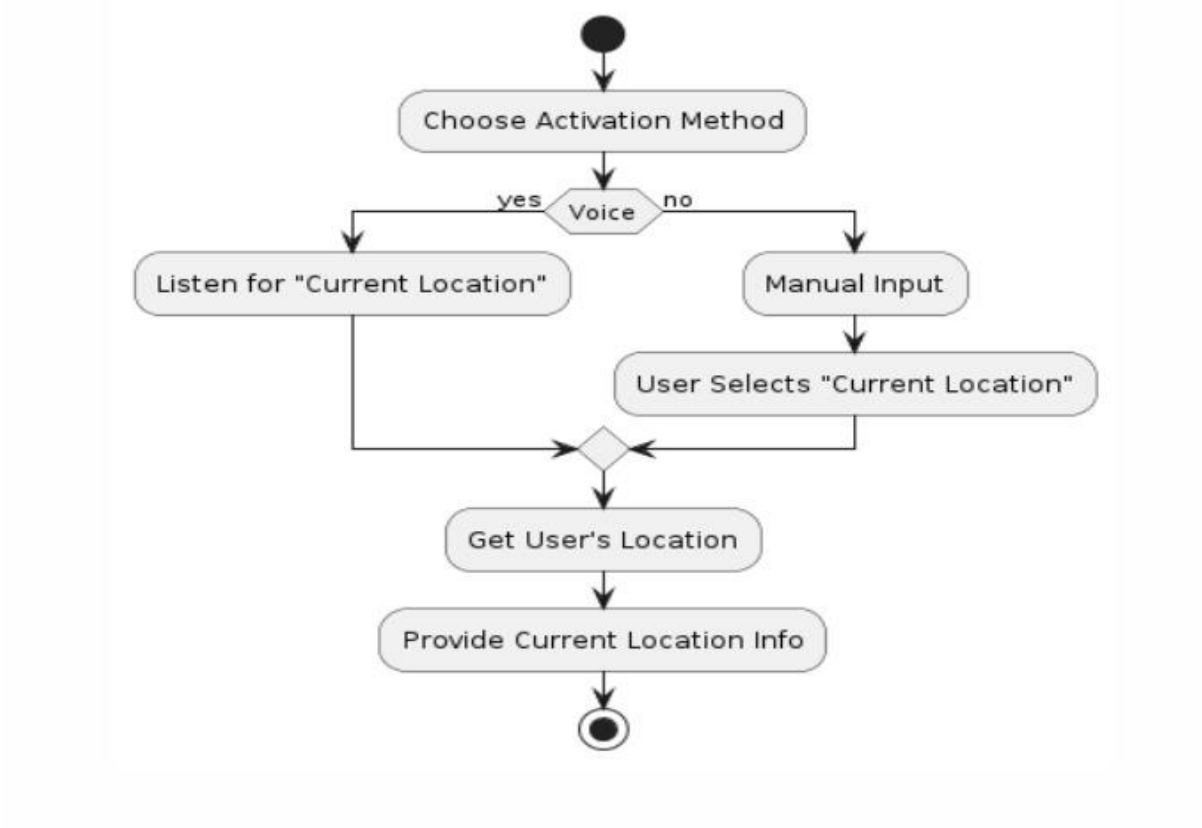


Figure:8.2 (Current Location)

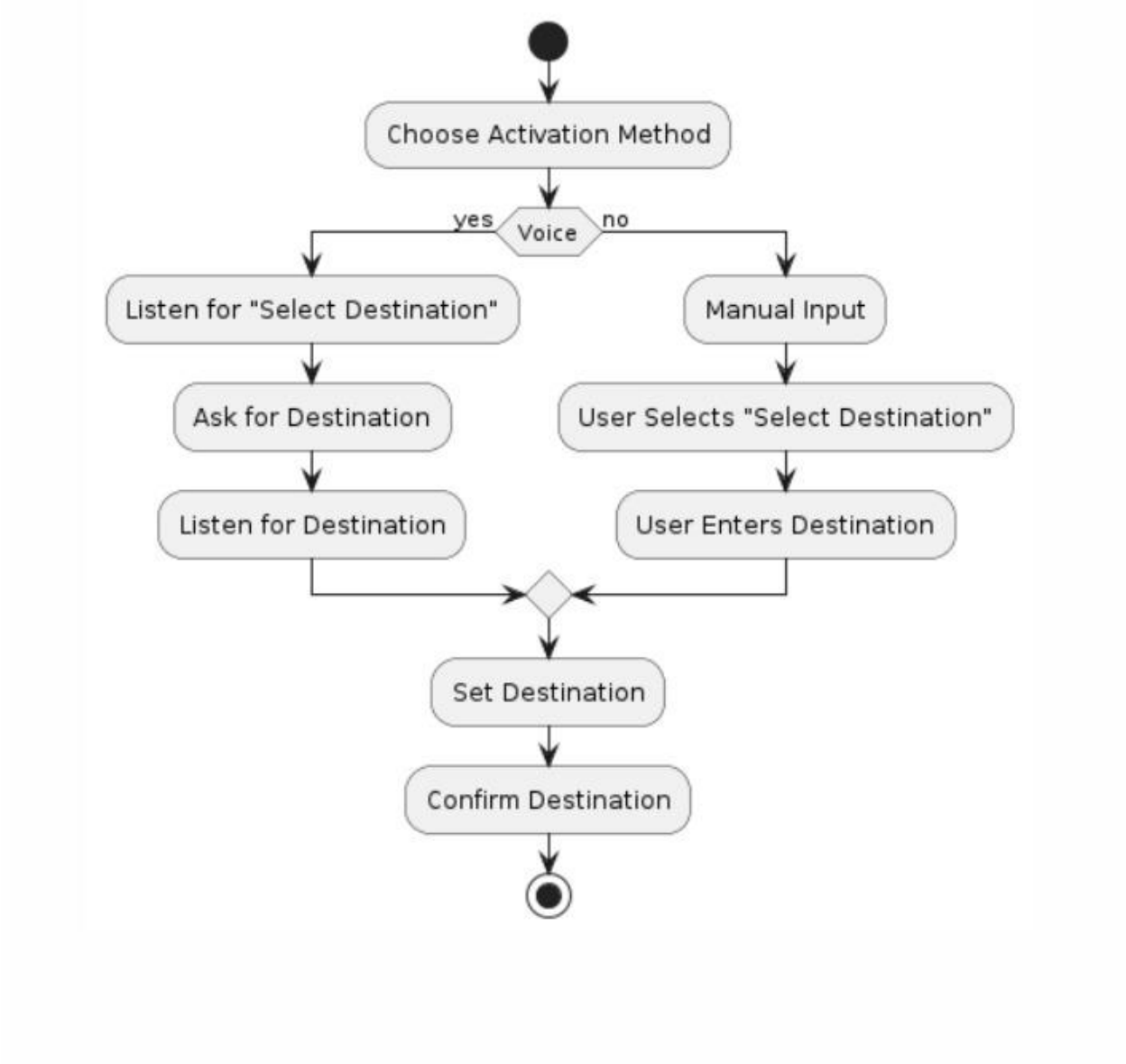


Figure:8.3 (Select Destination)

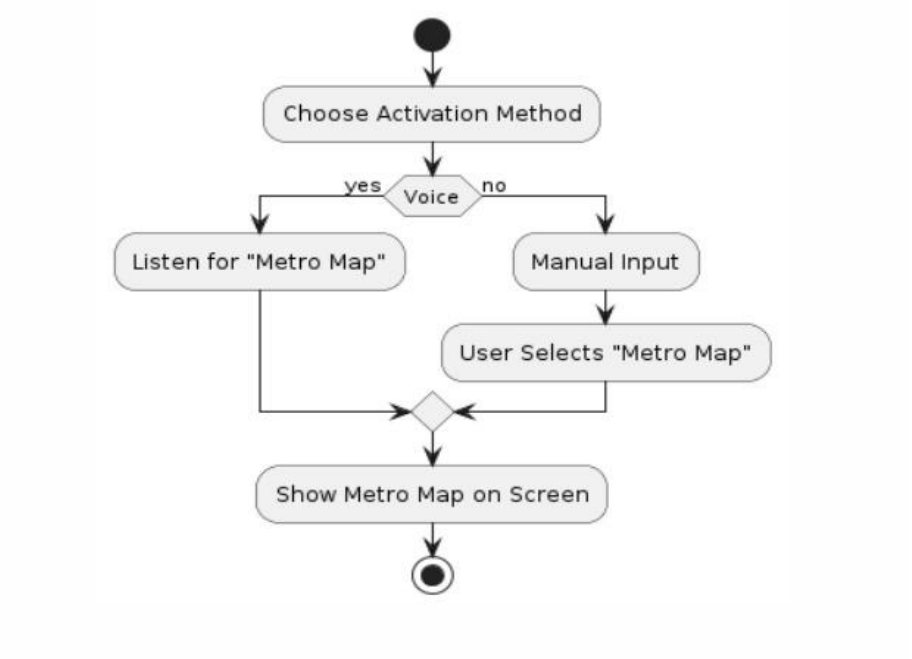


Figure:8.4(Metro Map)

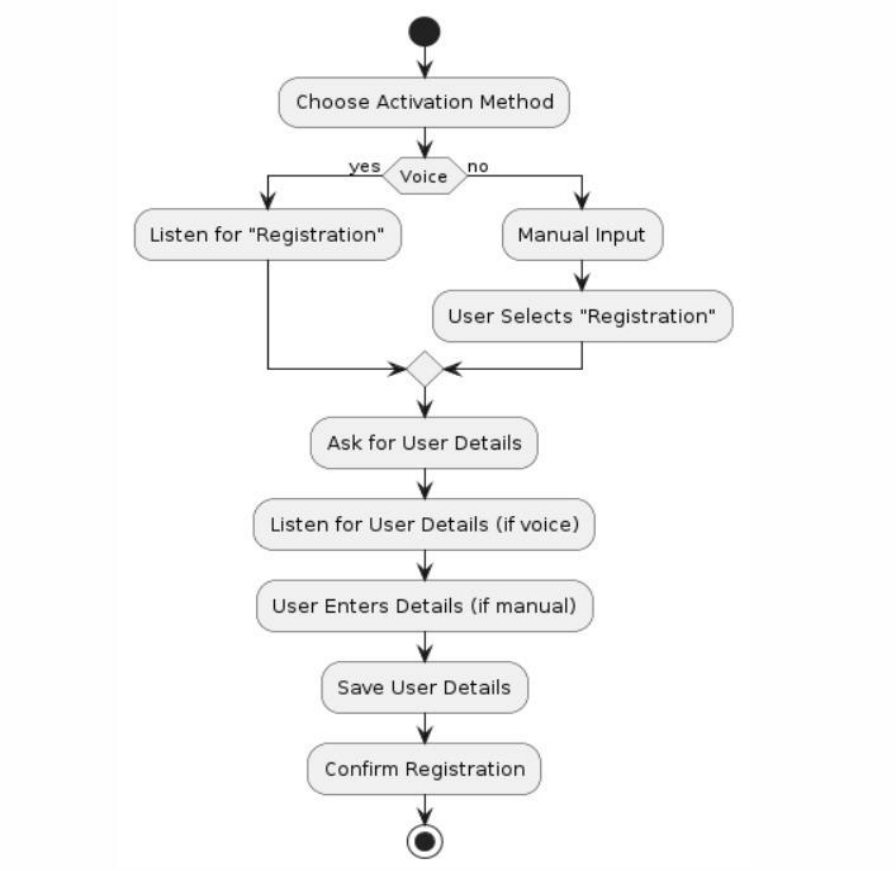


Figure:8.5 (Registration)

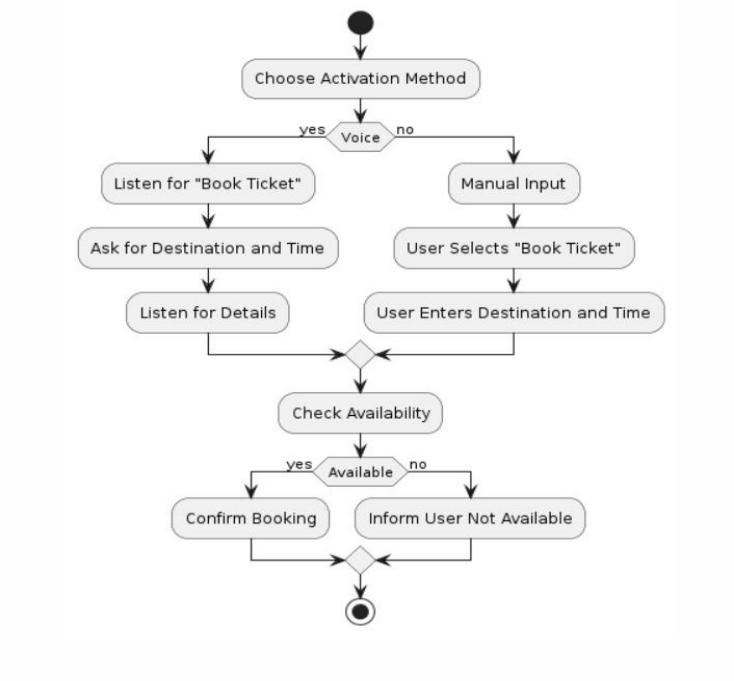


Figure:8.6 (Book Ticket)

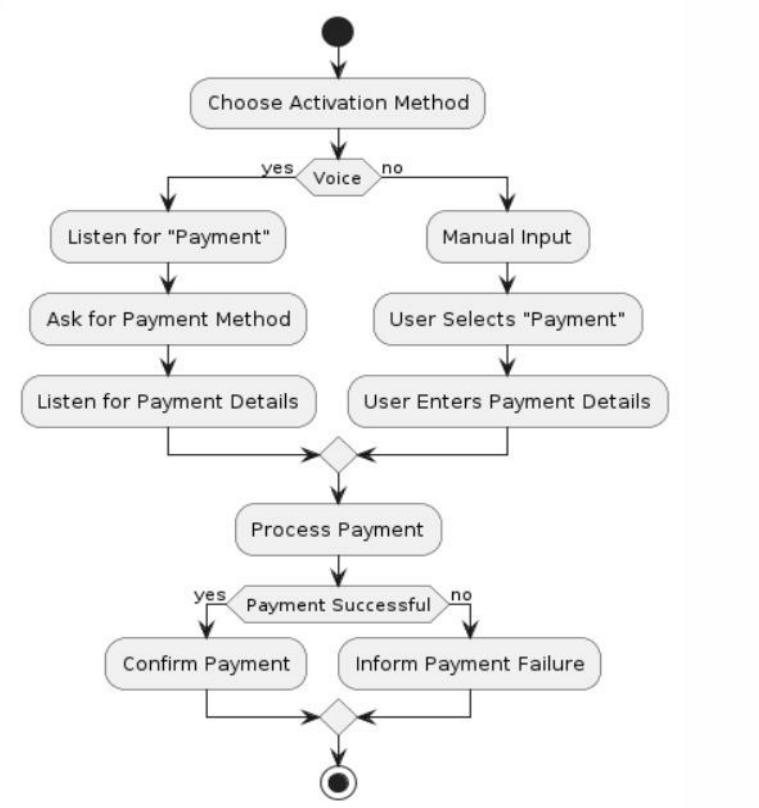


Figure:8.7 (payment)

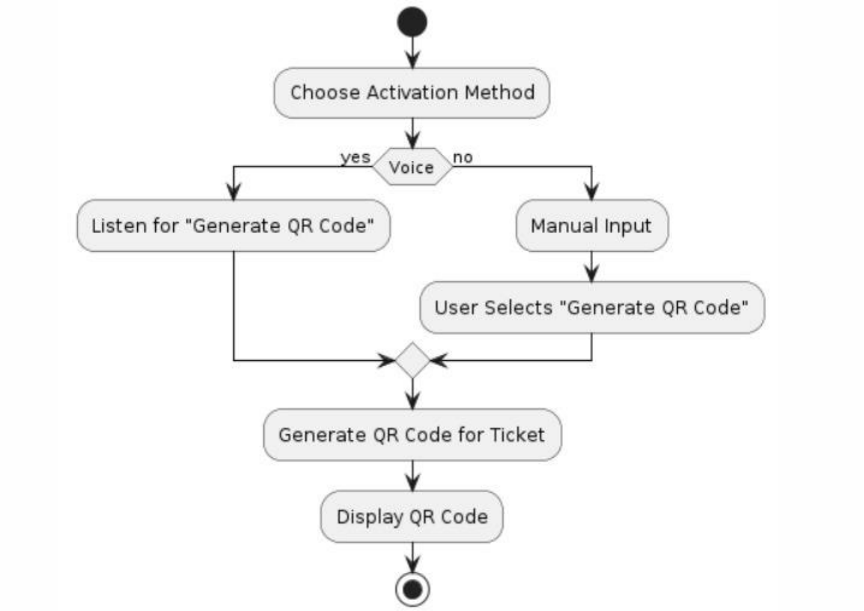


Figure:8.8(QR Code Generator)

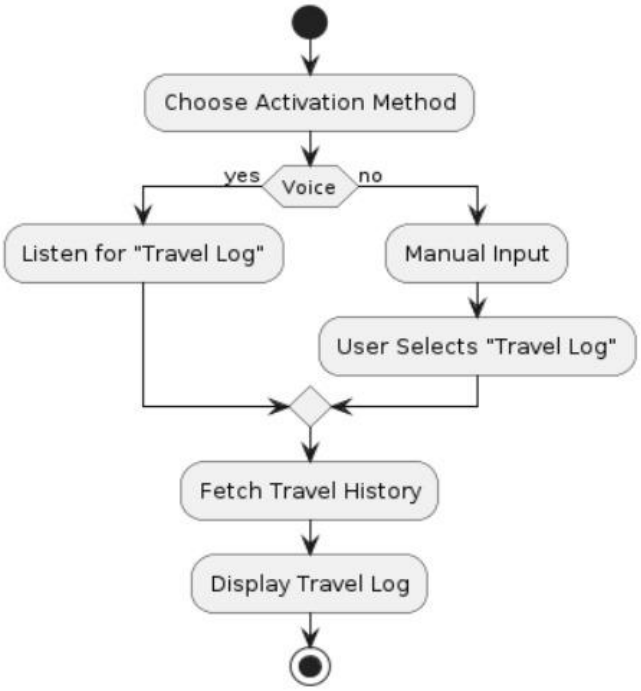


Figure:8.9(Travel Log)

9. Sequence Diagram

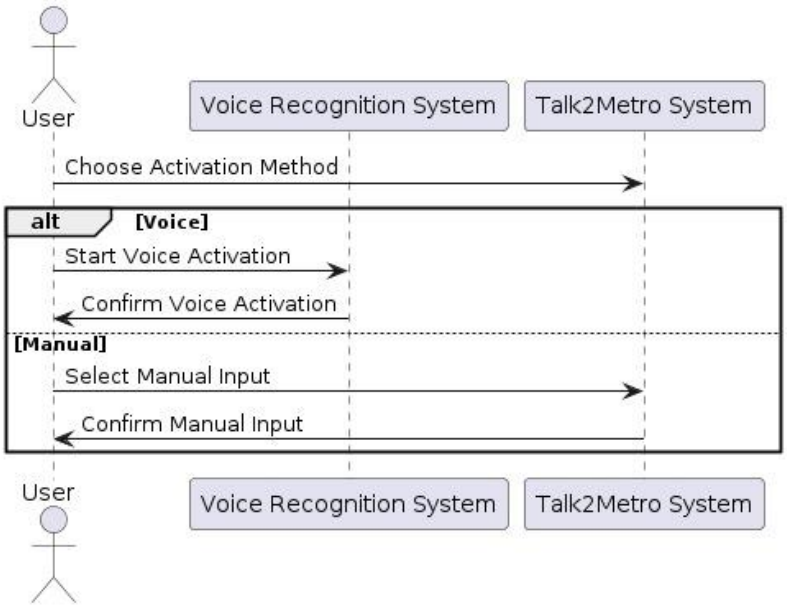


Figure:9.1 (Voice Activation)

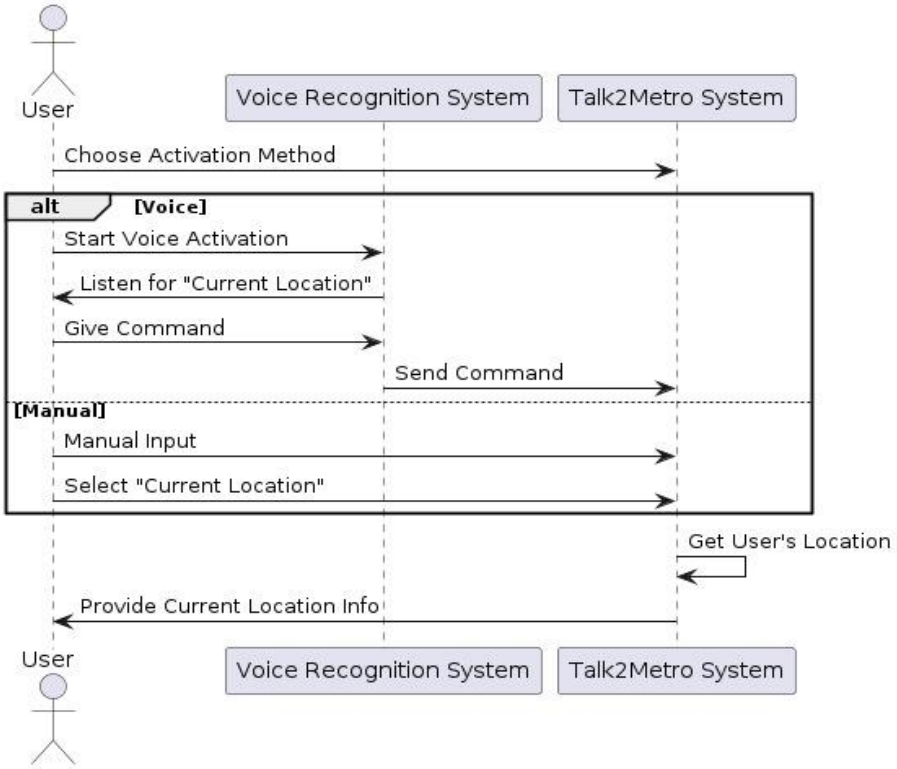


Figure:9.2 (Current Location)

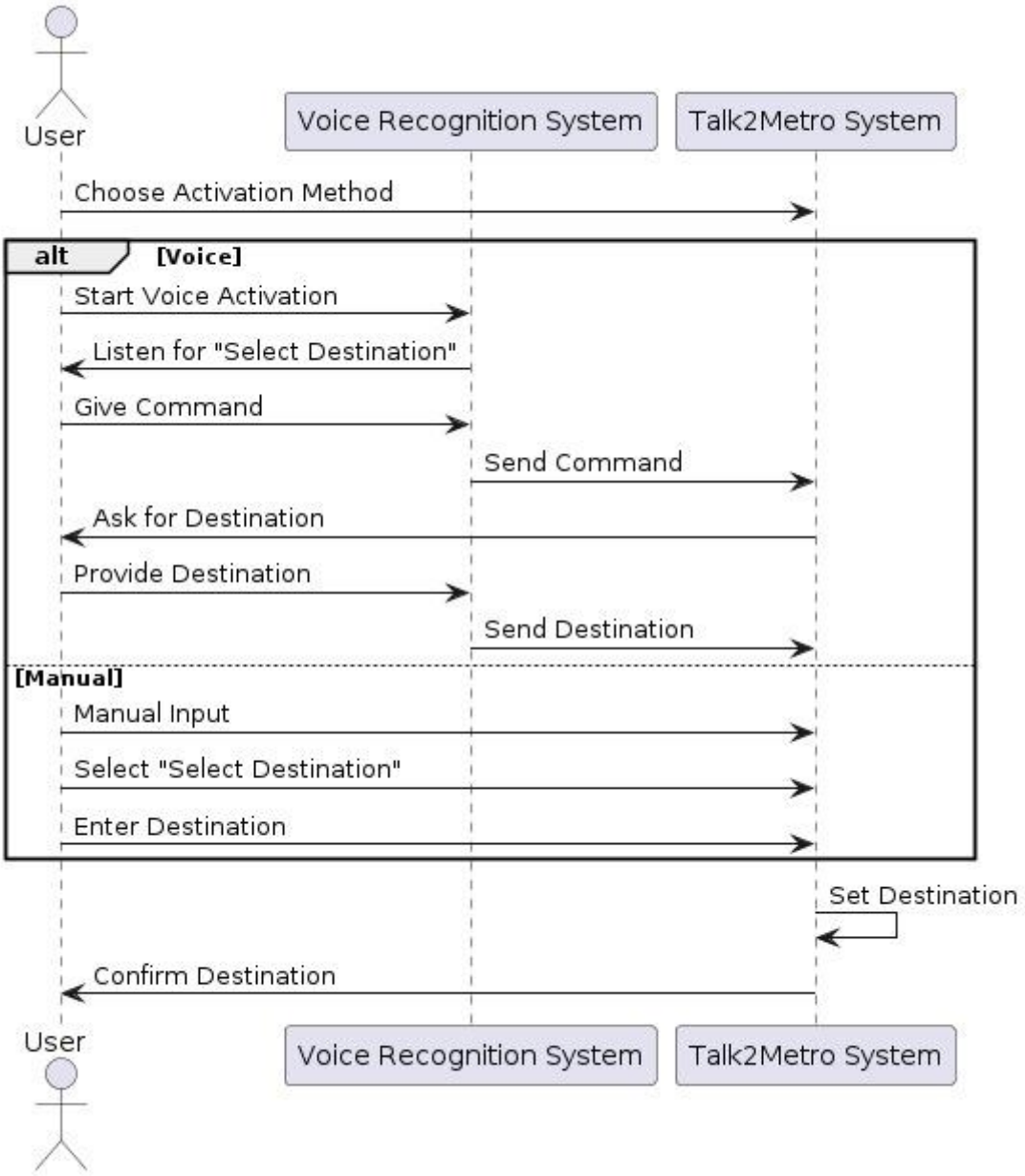


Figure:9.3(Select Destination)

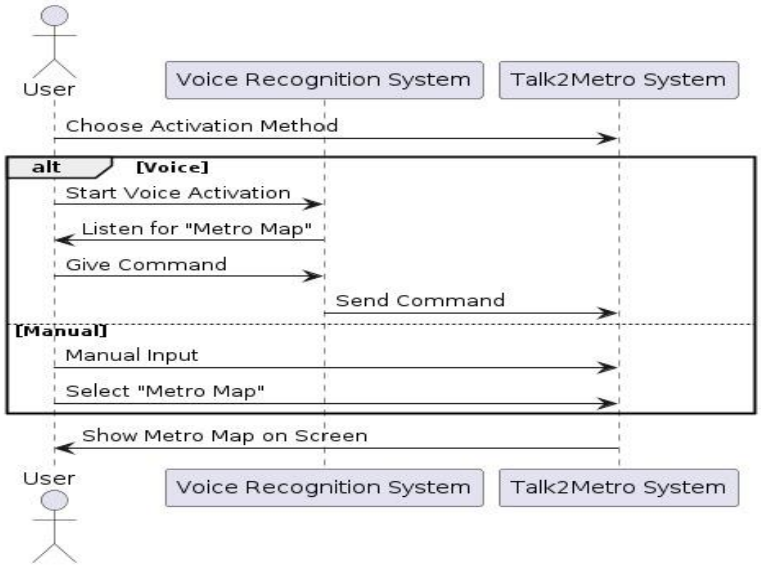


Figure:9.4 (Metro Map)

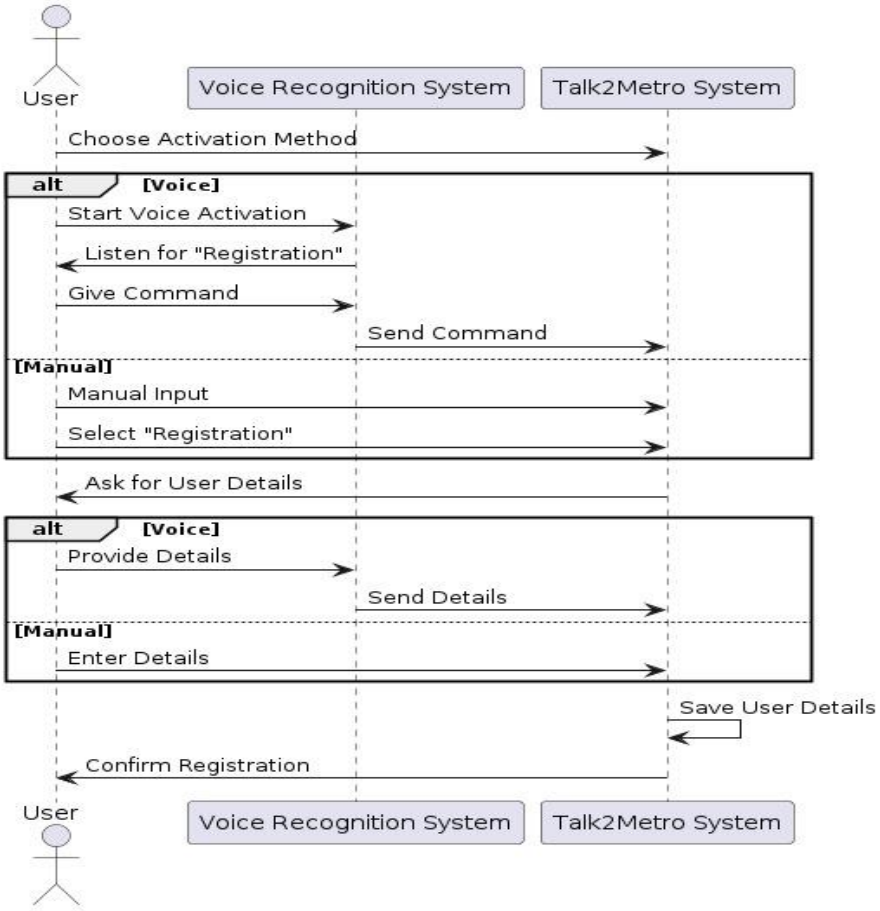


Figure:9.5 (Registration)

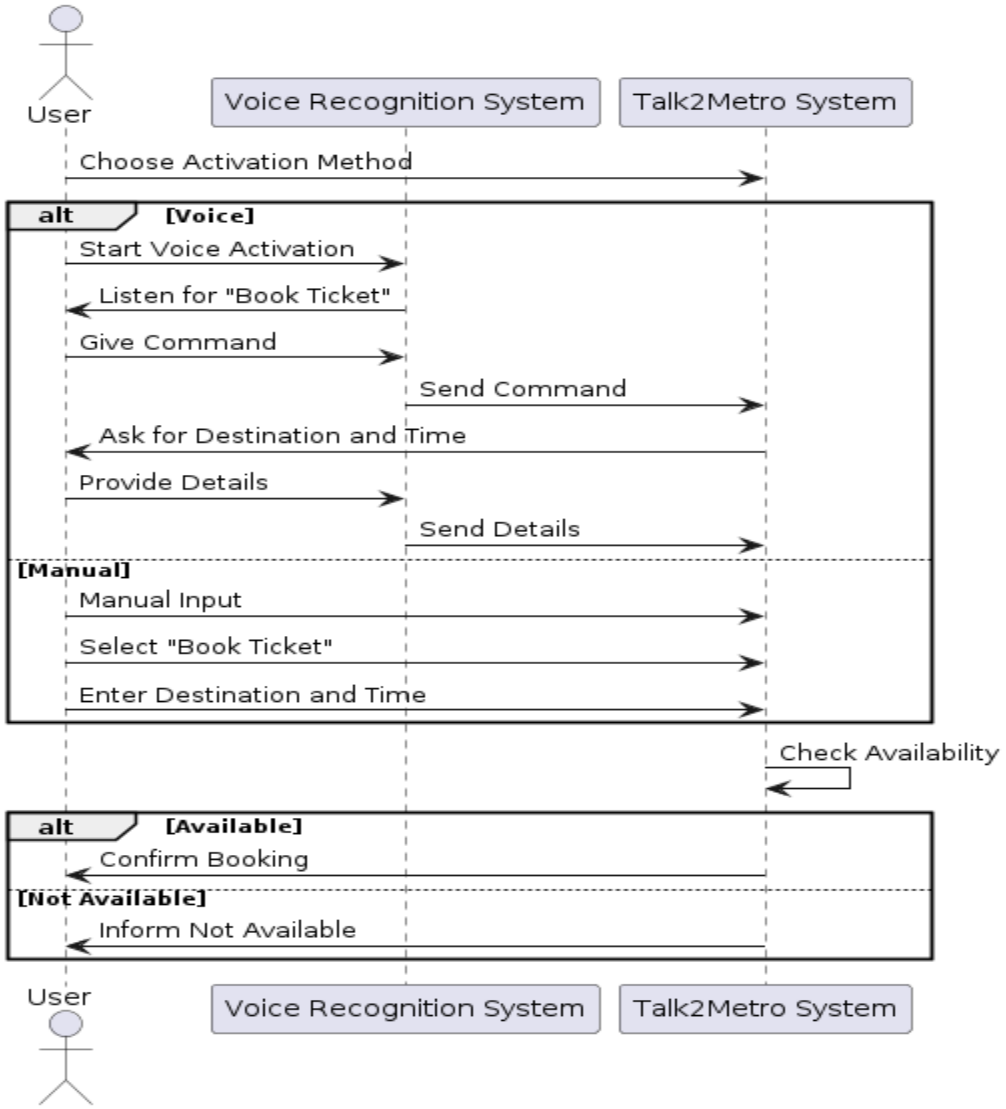


Figure:9.6 (Book Ticket)

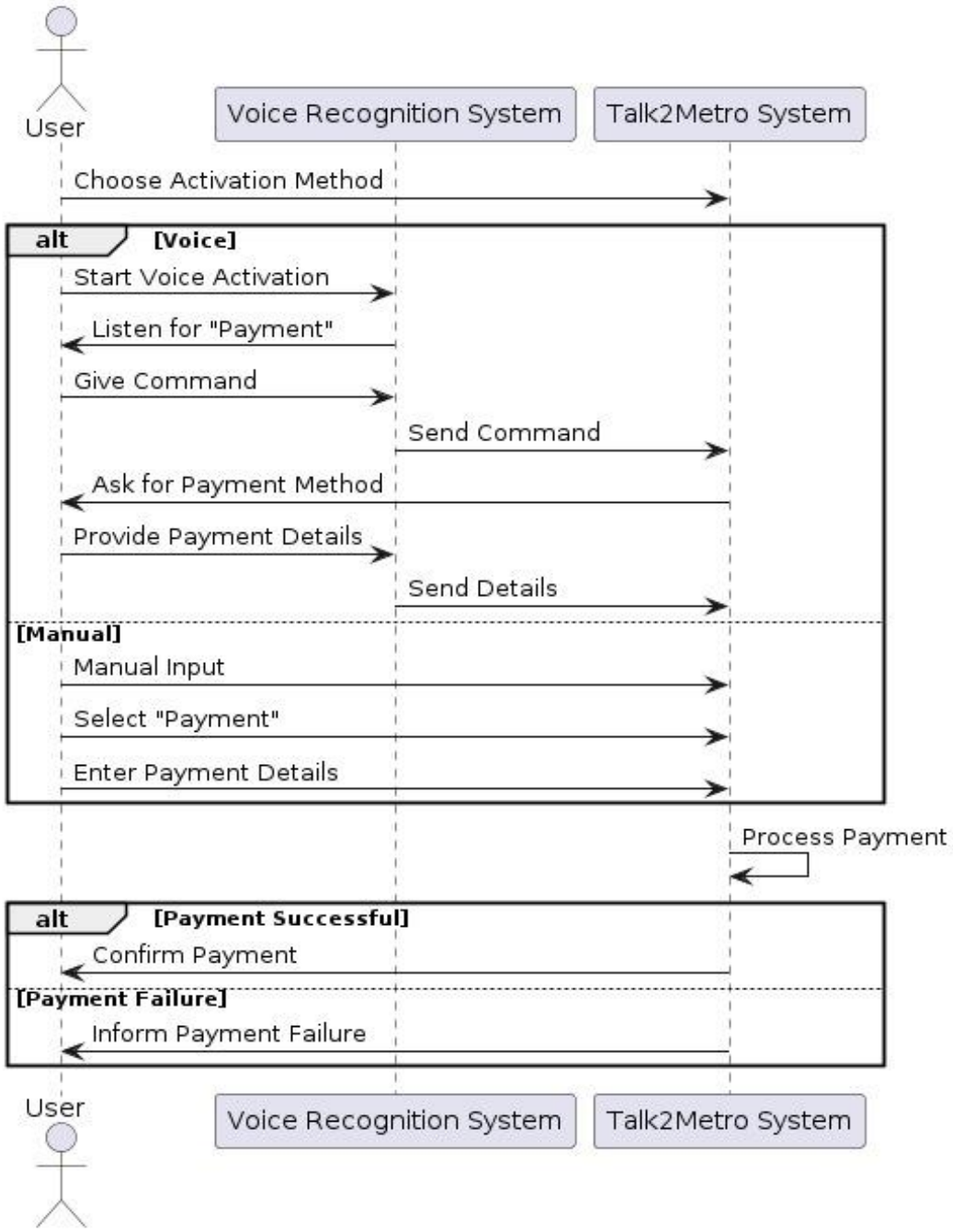


Figure:9.7 (Payment)

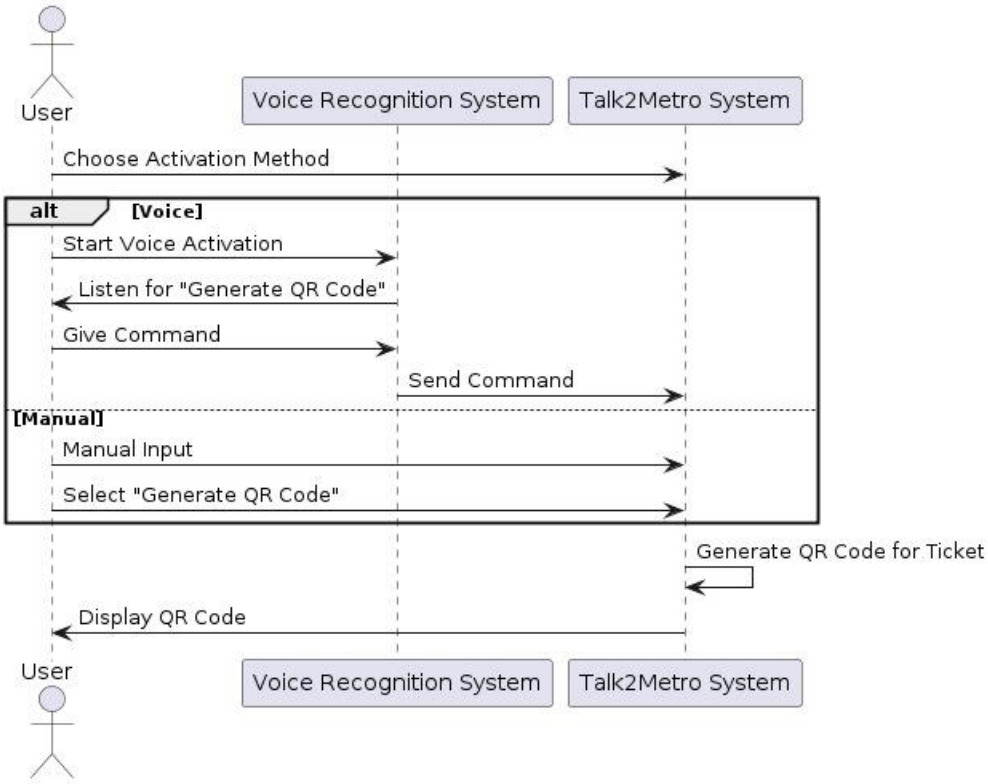


Figure:9.8 (QR Code Generator)

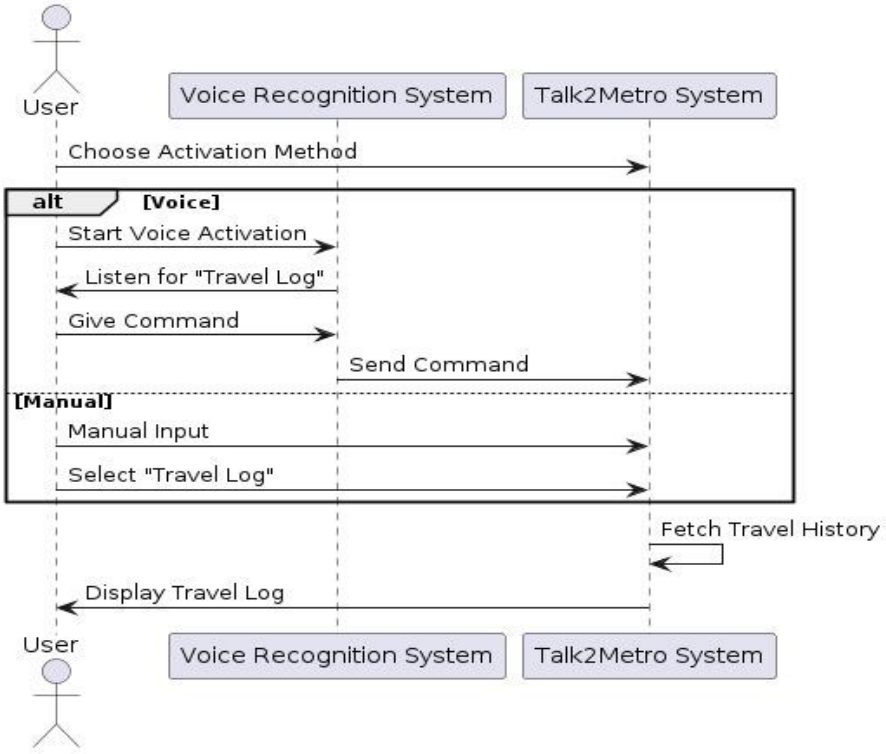


Figure:9.9 (Travel Log)

10. Swimlane Diagram

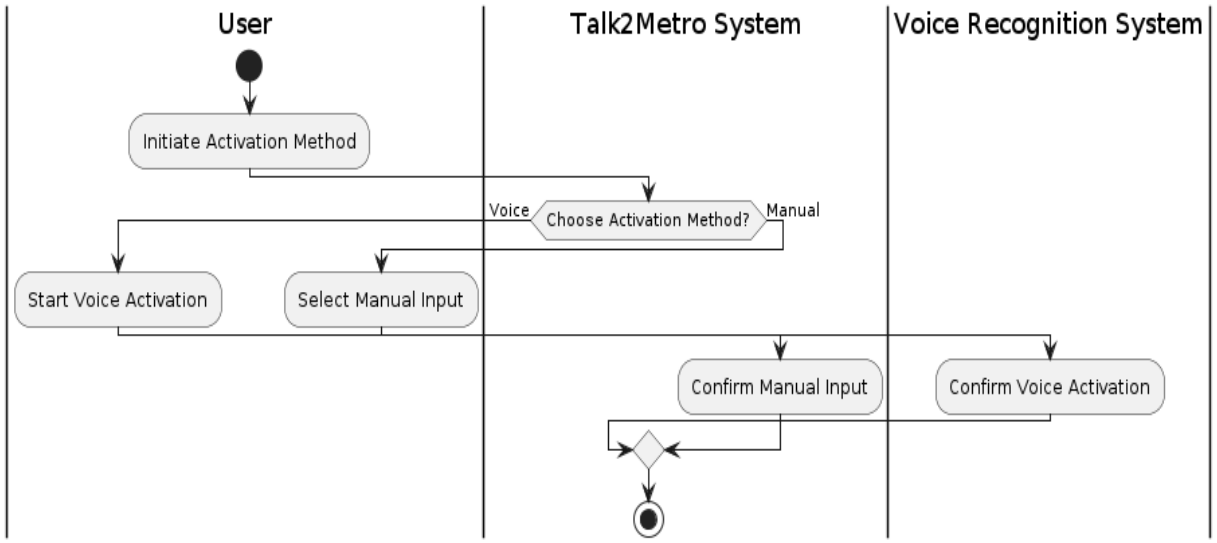


Figure:10.1 (Voice Activation)

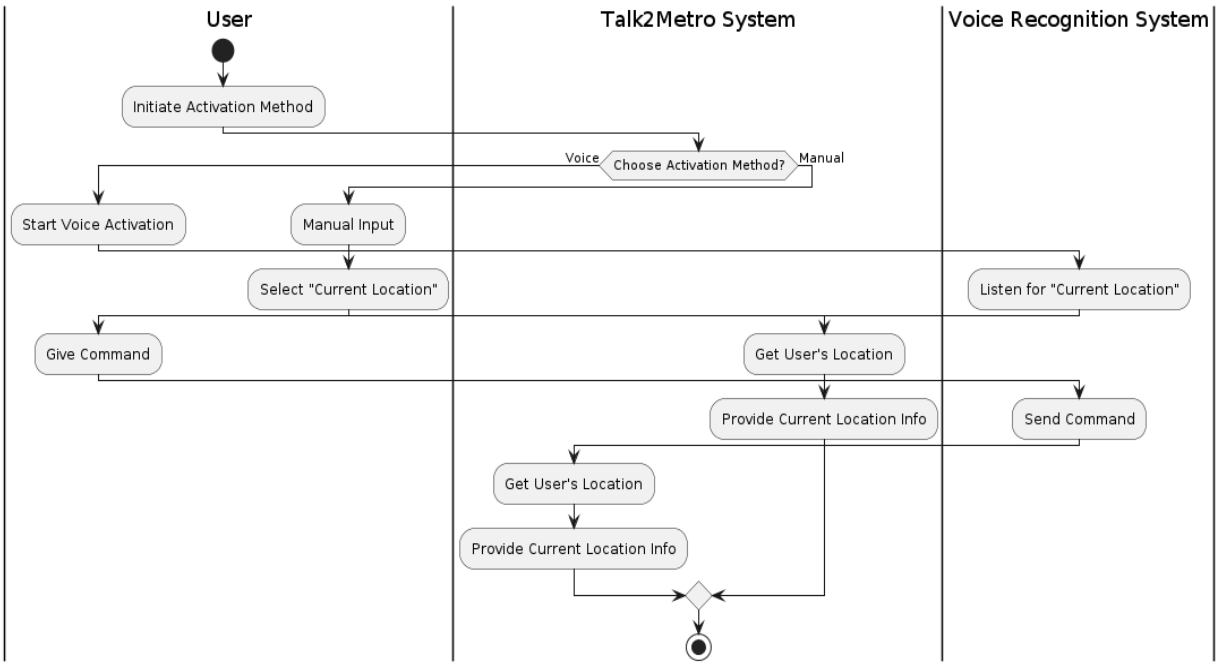


Figure:10.2 (Current Location)

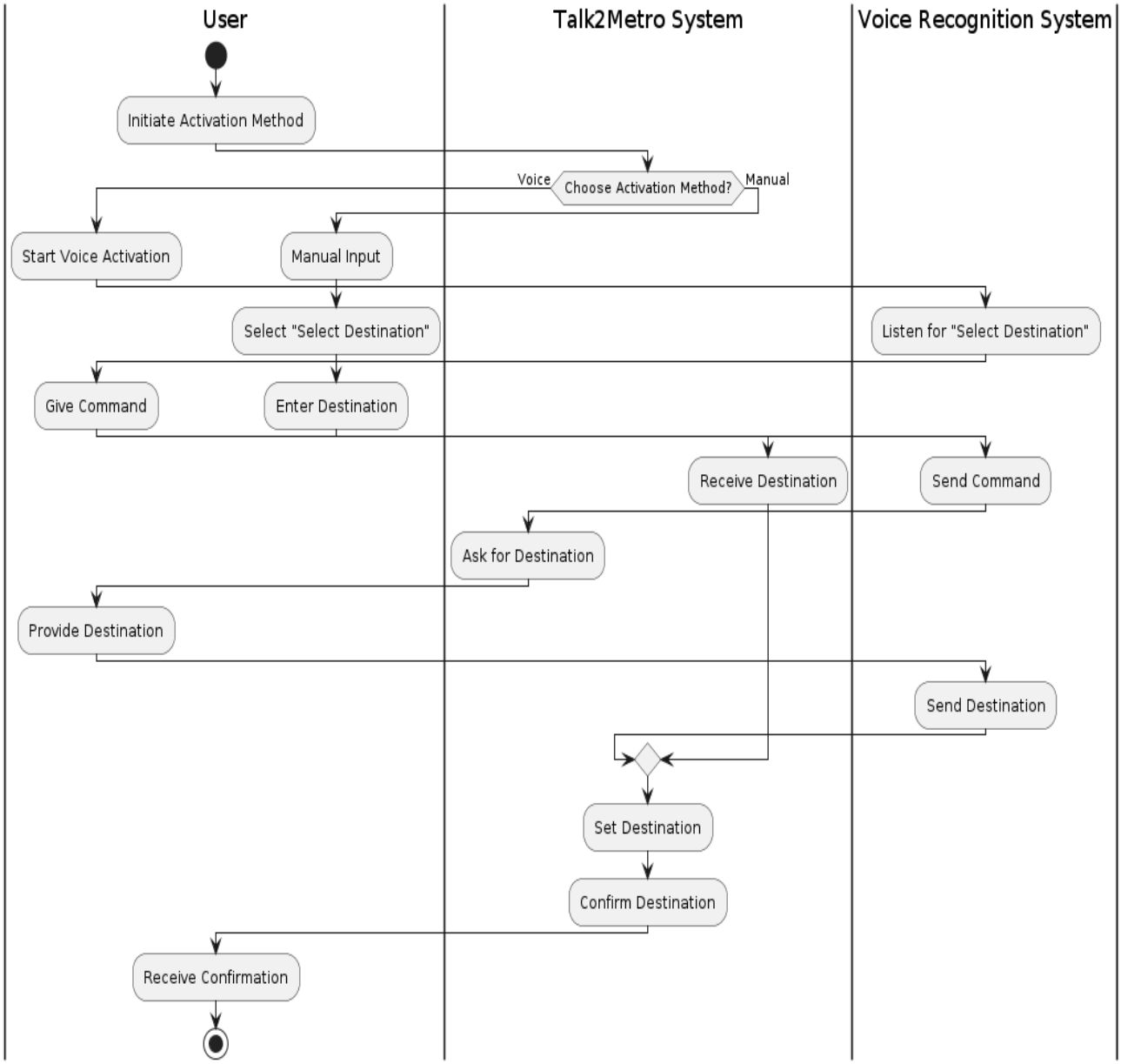


Figure:10.3 (Select Destination)

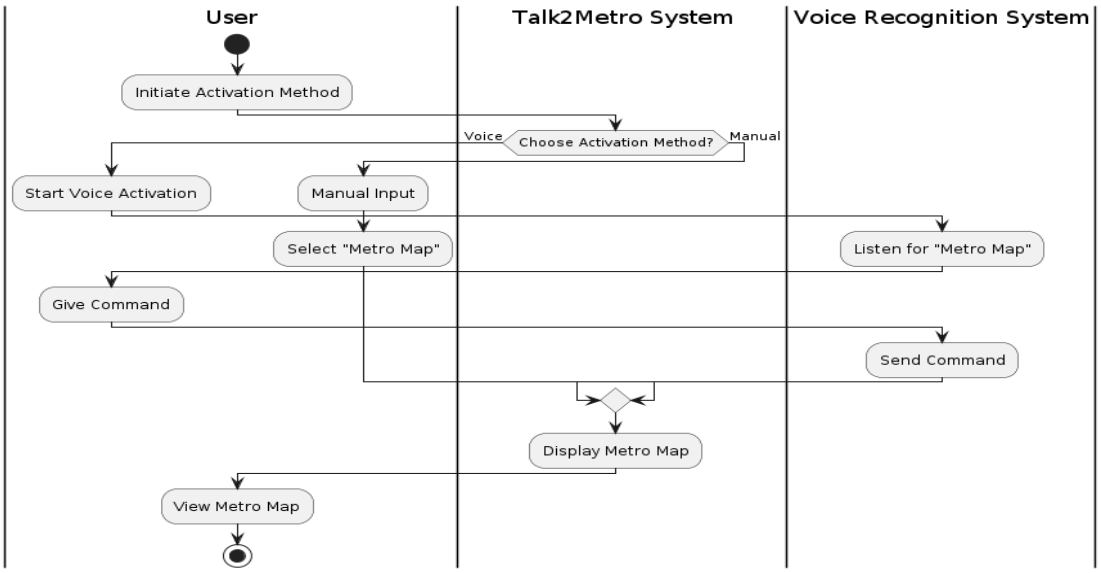


Figure:10.4 (Metro Map)

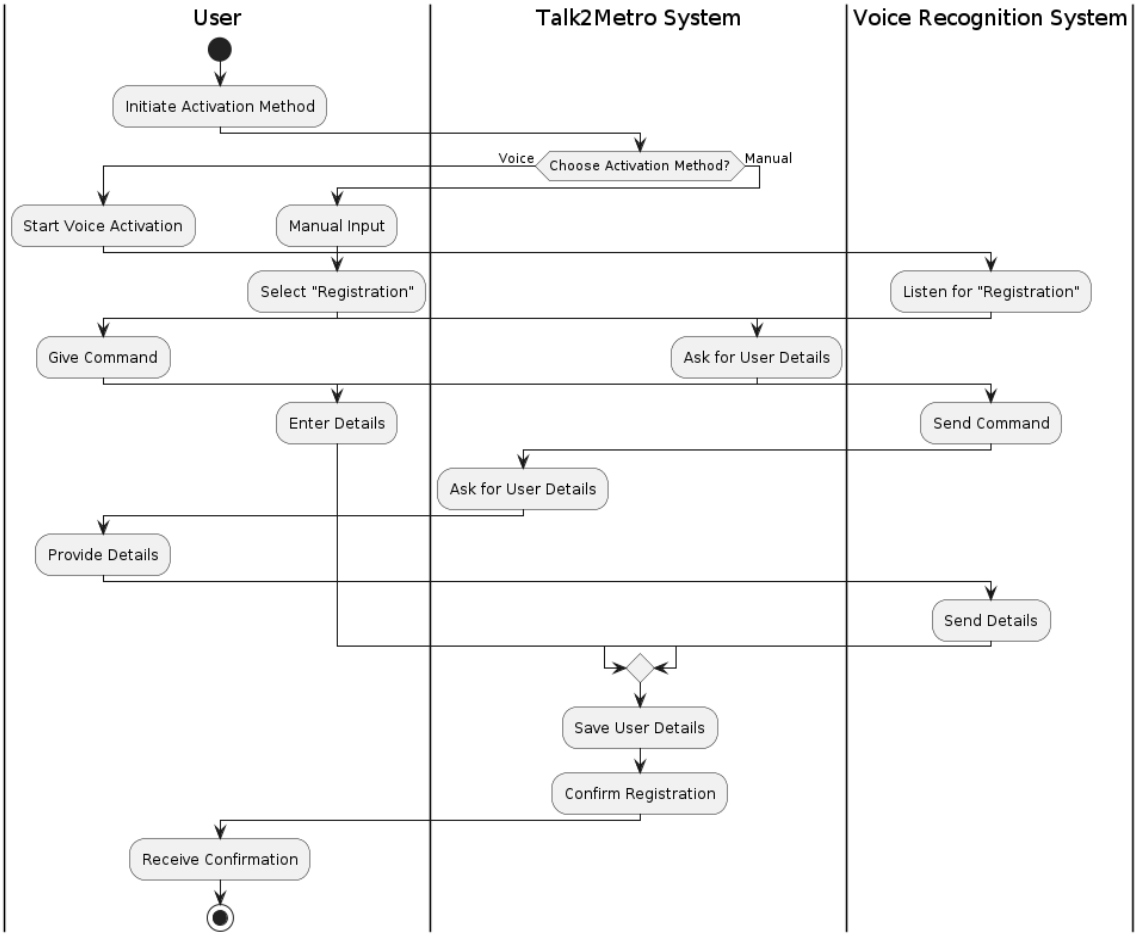


Figure:10.5 (Registration)

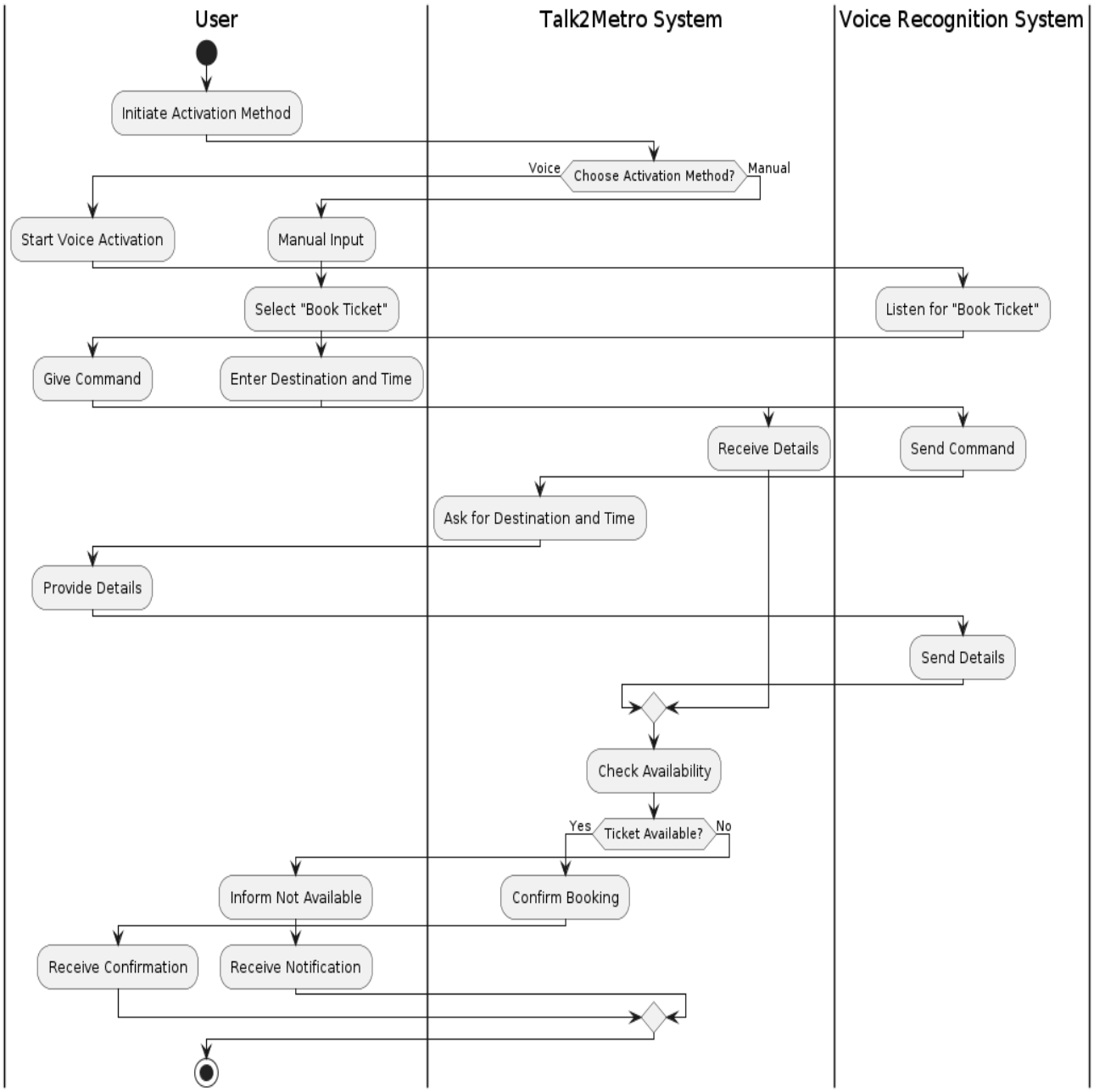


Figure:10.6 (Book Ticket)

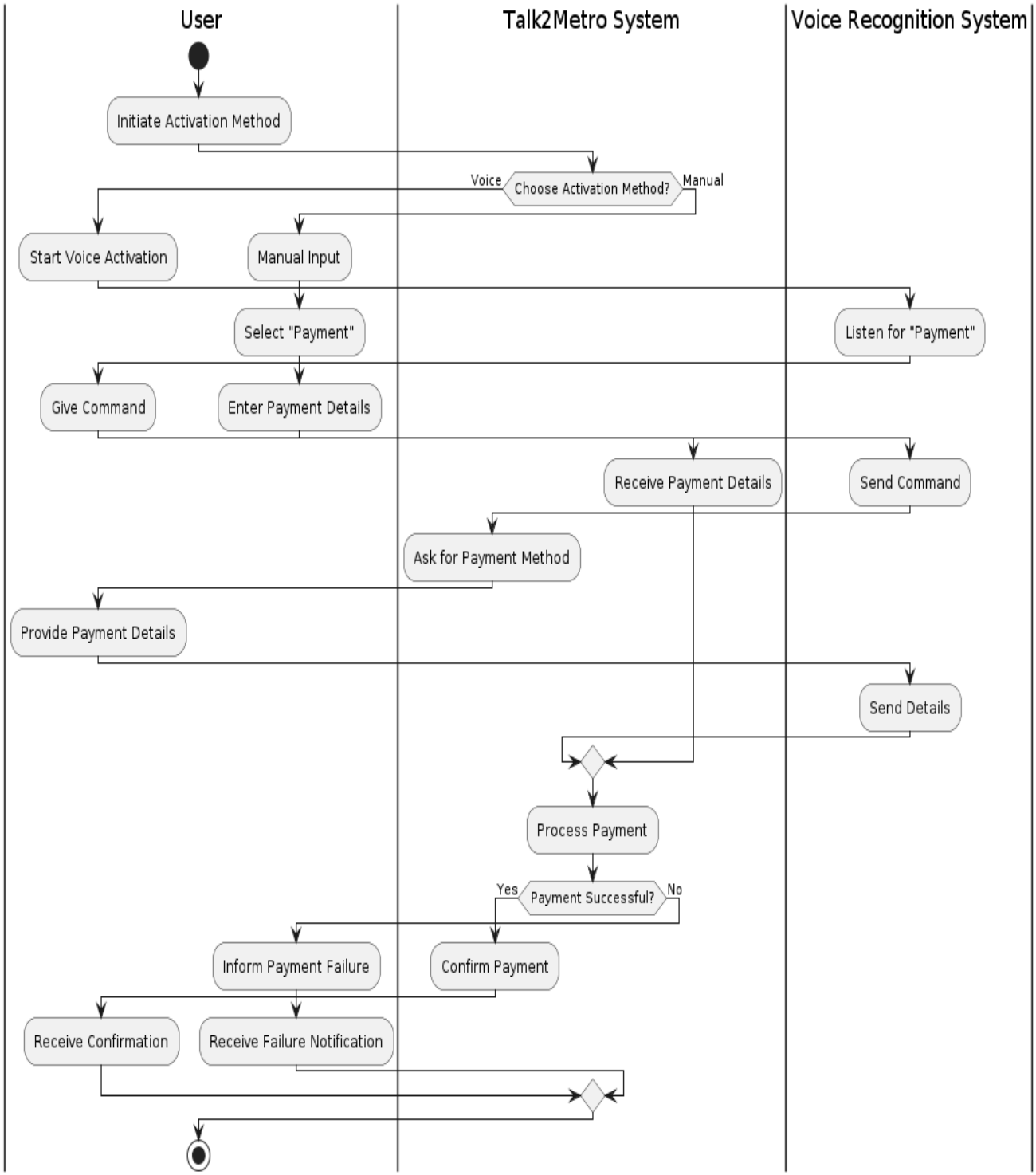


Figure:10.7 (Payment)

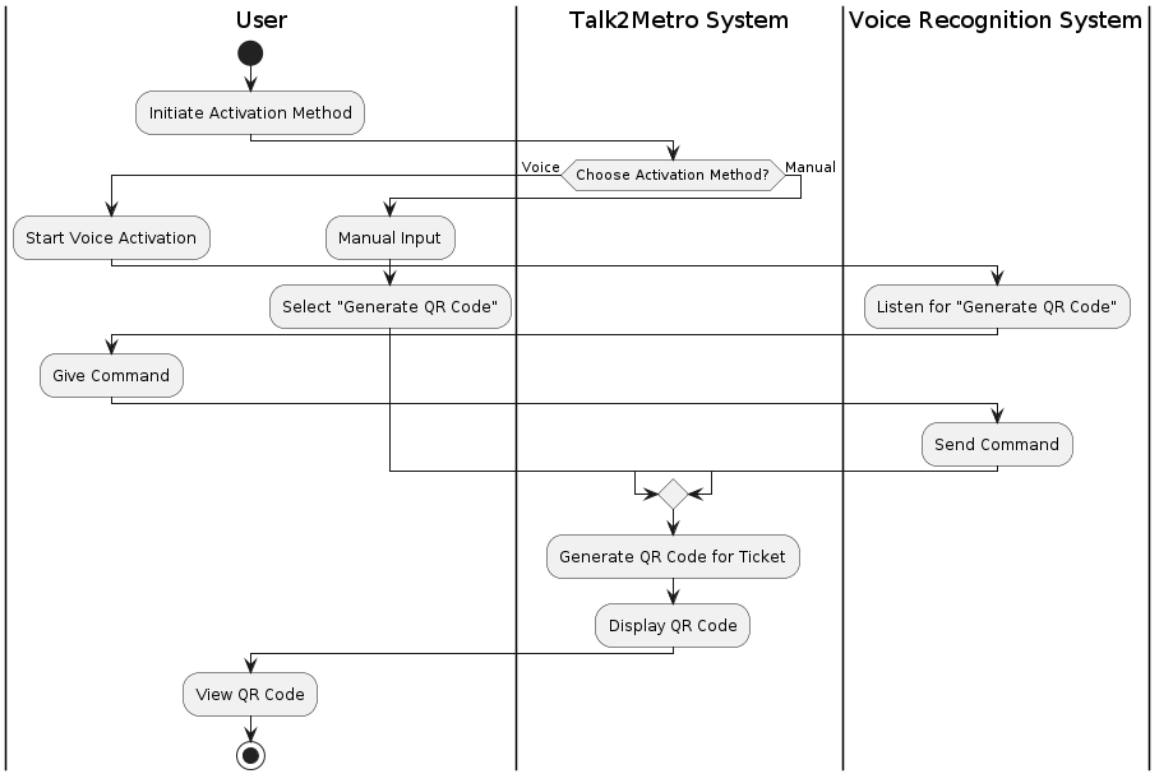


Figure:10.8 (QR Code Generator)

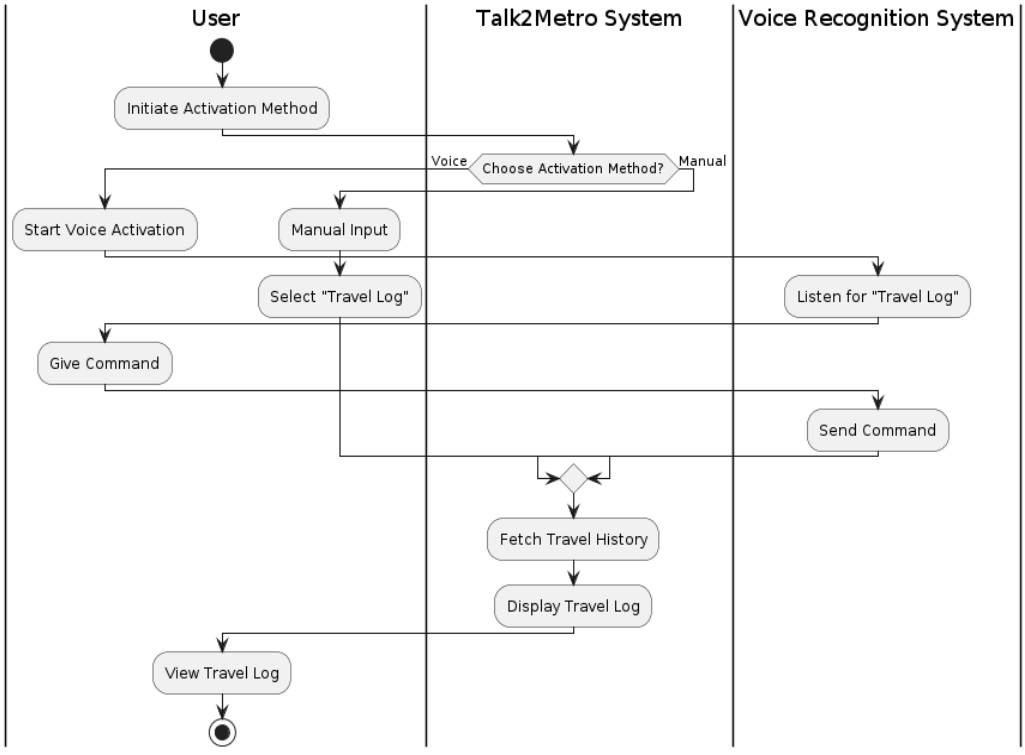


Figure:10.9 (Travel Log)

11. Appendix

11.1 Prioritization of requirements

We've prioritized the functional requirements by following Three-level Scale technique.

11.1.1 Three-level Scale

Three-level scale techniques are a type of measurement scale that uses three levels to measure a variable. The three levels are typically labeled as "low," "medium," and "high." This type of scale is often used when the variable being measured is subjective or difficult to quantify.

11.1.2 Prioritization of the requirements of Talk2Metro

F_R1 – High priority: Users should have the capability to create accounts on the platform, providing necessary information such as email address and password.

F_R2 – High priority: Users must be able to search for metro stations using various criteria such as station name or proximity, facilitating easy access to desired routes.

F_R3 – High priority: Users must be able to book metro-rail tickets seamlessly through the platform, specifying the origin, destination, and desired travel options.

F_R4 – High priority: The platform should offer secure payment processing mechanisms, enabling users to complete transactions using preferred payment methods such as credit/debit cards, bkash, and Nagad securely.

F_R5 – High priority: The platform must generate a unique QR code for each booked ticket, ensuring valid and secure ticket validation at metro stations.

F_R6 – Medium priority: Enable users to edit and update their account information, including personal details, contact information, and preferences.