POLITECNICO
MILANO 1863

Travlendar+ project Haipeng Zhu

Requirement Analysis and Specification Document

Deliverable:	RASD
Title:	Requirement Analysis and Verification Document
Authors:	Haipeng Zhu
Version:	1.0
Date:	December 24, 2025
Download page:	LINK TO YOUR REPOSITORY
Copyright:	Copyright © 2024, Haipeng Zhu – All rights reserved

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The purpose of this document is to provide a Requirement Analysis and Specification for the Best Bike Paths (BBP) system.

BBP is a software system designed to support cyclists in recording their personal biking activities and in managing the route information about bike paths. The system allows users to manually insert information about bike paths, such as their status and the presence of obstacles, and to visualize possible bike paths between a origin starting point and a destination on a map.

This document defines the goals, assumptions, and requirements of the BBP system. It serves as a contractual reference between stakeholders and developers and as a baseline for subsequent design and implementation activities.

1.1.1 Goals

G1: The user wants to record a biking trip, in order to keep track of personal biking activities.

G2: The user wants to review previously recorded biking trips, in order to recall and organize past experiences.

G3: The user wants to visualize possible biking paths between a specified origin and destination, in order to support route selection.

G4: The user wants to manually add information about a bike path based on personal experience, in order to document relevant conditions and obstacles.

G5: The user wants to decide whether the information added about a bike path should be made visible to other users, in order to control its sharing.

1.2 Scope

The scope of the document is limited to the functionalitiThis section describes the application domain of the Best Bike Paths (BBP) system by distinguishing between phenomena that occur in the real world and those that are shared between the system and its users.

The scope of this document is limited to the functionalities assigned to single-student groups, namely the recording of personal biking trips, the manual insertion of information about bike paths, and the visualization of bike paths between a specified origin and destination. Advanced functionalities such as automatic data collection, aggregation of reports from multiple users, and conflict resolution are explicitly excluded from the scope.

1.2.1 World Phenomena

World phenomena are events and conditions that occur independently of the BBP system and are not directly controlled by it. These phenomena belong to the environment in which the system operates.

- WP1: The user starting and stopping the recording of a biking trip through the system.
- WP2: The user manually inserting information about a bike path, including its status and possible obstacles.
- WP3: The system storing recorded biking trips.
- WP4: The system displaying recorded trips and bike paths on a map.
- WP5: The user specifying an origin and a destination to visualize possible biking paths.
- WP6: The user deciding whether the information added about a bike path is visible to other users.

1.2.2 Shared Phenomena

Shared phenomena are events and information that are observable and managed both by the users and the BBP system. These phenomena define the interaction between the system and its environment.

- SP1: The user signals the system to start and stop the recording of a trip.
- SP2: The system retrieves meteorological data from an external weather service to enrich trip records.
- SP3: The user manually inputs street names, path status, and obstacle descriptions into the system.
- SP4: The user sets a specific trip or path report as "publishable" or "private" within the application.
- SP5: The system provides the user with trip statistics, including total distance and average speed.
- SP6: The user inputs an origin and a destination to request path options.
- SP7: The system visualizes bike paths on a map, displaying their computed scores to the user.
- SP8: The system provides a list/inventory of the user's previously recorded trips for review.

1.3 Definitions, Acronyms, Abbreviations

- BBP: Best Bike Paths.
- User: An individual who interacts with the BBP system to record biking activities and manage bike path information.
- Trip: A biking performed by a user and recorded through the BBP system.
- Bike Path: A designated route for biking, which can be recorded and managed within the BBP system.
- Publishable Information: Information about bike paths that users can choose to share with others through the BBP system.

1.4 Revision History

Version	Date	Description
1.0	December 24, 2025	Initial version

1.5 Reference Documents

- Sotfware Engineering 2 - Requirement Engineering and Design Assignment Description
- IEEE/ISO/IEC 29148-2018 - Systems and software engineering – Life cycle processes – Requirements engineering

1.6 Document Structure

This document is organized as follows.

Section 2 provides an overall description of the BBP system, including scenarios, user characteristics, and domain assumptions.

Section 3 detailes the functional and non-functional requirements of the system.

Section 4 presents a formal analysis of selected aspects of the system using Alloy.

Section 5 reports the effort spent on this document.

2 Overall Description

2.1 Product Perspective

The Best Bike Paths (BBP) system is a standalone mobile application that interacts with external services (Maps and Weather) to provide value to individual cyclists. The system manages the lifecycle of two main entities: the biking **Trip** and the manually inserted **Path Information**.

2.1.1 Scenarios

Scenario 1: User records a personal biking trip User Alice is about to start her daily commute to work and wants to track her performance. She opens the Best Bike Paths (BBP) application on her smartphone and logs in. She navigates to the "My Trips" section and taps the "Start Recording" button. As she cycles, the system tracks her location via GPS. Once she arrives at her office, she taps "Stop Recording". The system automatically retrieves the current weather conditions (e.g., temperature and wind speed) from an external service and saves the trip . Alice can now view the summary of her ride, including total distance and average speed, stored in her personal history.

Scenario 2: User manually inserts path information User Bob is riding through "Oak Street" and notices that the bike lane is full of potholes, making it dangerous. Later, when he is safe at home, he decides to report this to the community. He logs into BBP and selects the "Add Path Info" feature. He manually enters the street name "Oak Street" and sets the status to "Requires Maintenance". He also adds a note specifying "Deep potholes near the intersection." He decides to make this information public so other cyclists can be warned, toggling the "Publishable" option. The system saves this report and associates it with the specific path location.

Scenario 3: User visualizes bike paths between origin and destination User Charlie wants to bike from his home to the central train station but is unsure of the safest route. He opens the BBP application and enters his home address as the "Origin" and the station address as the "Destination". The system searches for existing bike paths connecting these two points. It finds two possible routes and displays them on a map. The system highlights the first route as the recommended option because it has a higher score, calculated based on recent user reports indicating "Optimal" status . Charlie selects this route to view the details before starting his ride.

2.1.2 Domain Model

The Domain Model shown in Figure 1 illustrates the static structure of the BBP system, highlighting the complexity of data management regarding trips and path reports.

The model is structured around the following key concepts:

- **User**: The active agent in the system. Each user maintains a collection of `Trips` and authored `PathInformation`.
- **Trip and Geolocation**: A `Trip` is not merely a summary record; it is modeled as a composite entity containing a sequence of `GeoPoints` (representing the raw GPS trace) and a `WeatherData` object (representing the meteorological context captured at the time of the trip).
- **Path Information Hierarchy**: To distinguish between different types of user contributions, `PathInformation` is modeled as an abstract class specialized into:
  - **StatusReport**: For assessing the general condition of the path (e.g., `OPTIMAL`, `SUFFICIENT`).
  - **ObstacleReport**: For flagging specific hazards (e.g., potholes).

- **Bike Path**: Represents the logical road segment, which is geometrically defined by a set of coordinates.

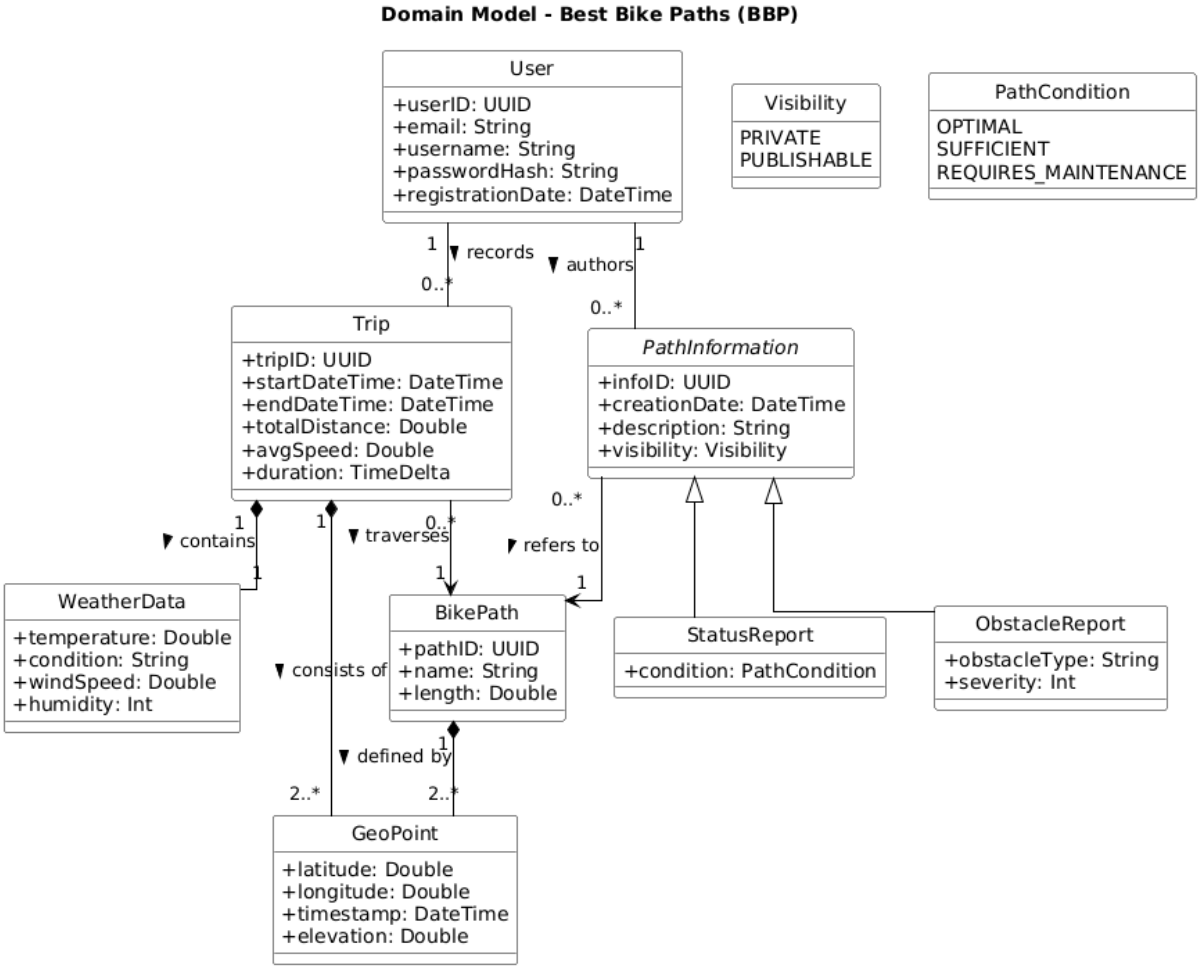

Figure 1: Detailed Domain Model Class Diagram

2.1.3 Trip Lifecycle

The recording of a trip is the central functionality of the application. Figure 2 illustrates the state transitions of a Trip entity. The process begins in the *Idle* state. When the user initiates a recording, the system transitions to the *Recording* state, utilizing GPS sensors. The user may temporarily *Pause* the activity. Upon completion, the system enters a transient state to fetch meteorological data from an external provider before permanently persisting the trip in the *Saved* history.

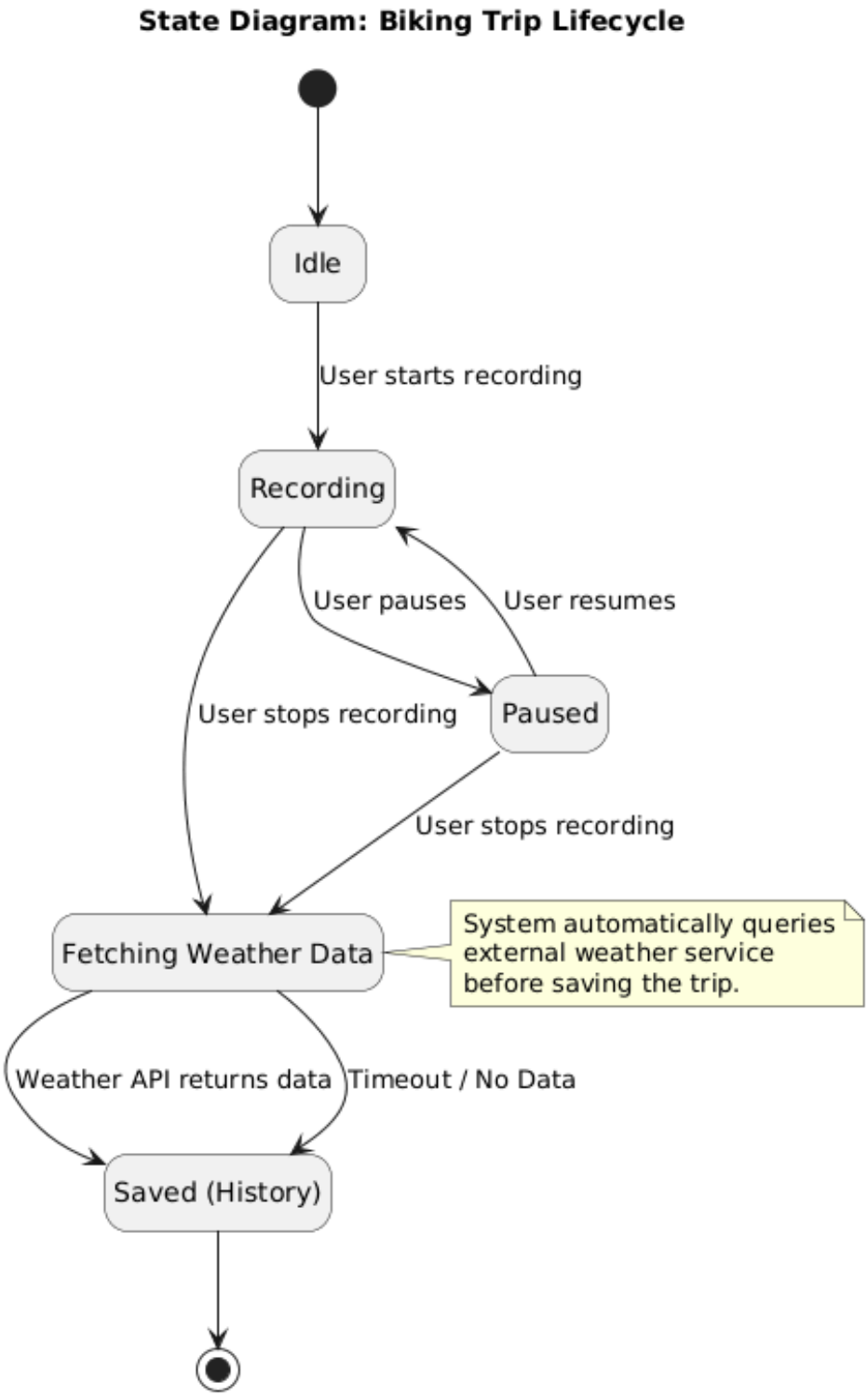

Figure 2: State Diagram: Lifecycle of a Biking Trip

2.1.4 Path Information Visibility

Users can contribute to the system by reporting path conditions. To ensure privacy and user control, the system maintains a visibility state for each report. As shown in Figure 3, a report can be stored as *Private* (visible only to the creator) or *Published* (shared with the community). The user can toggle this state at any time after creation.

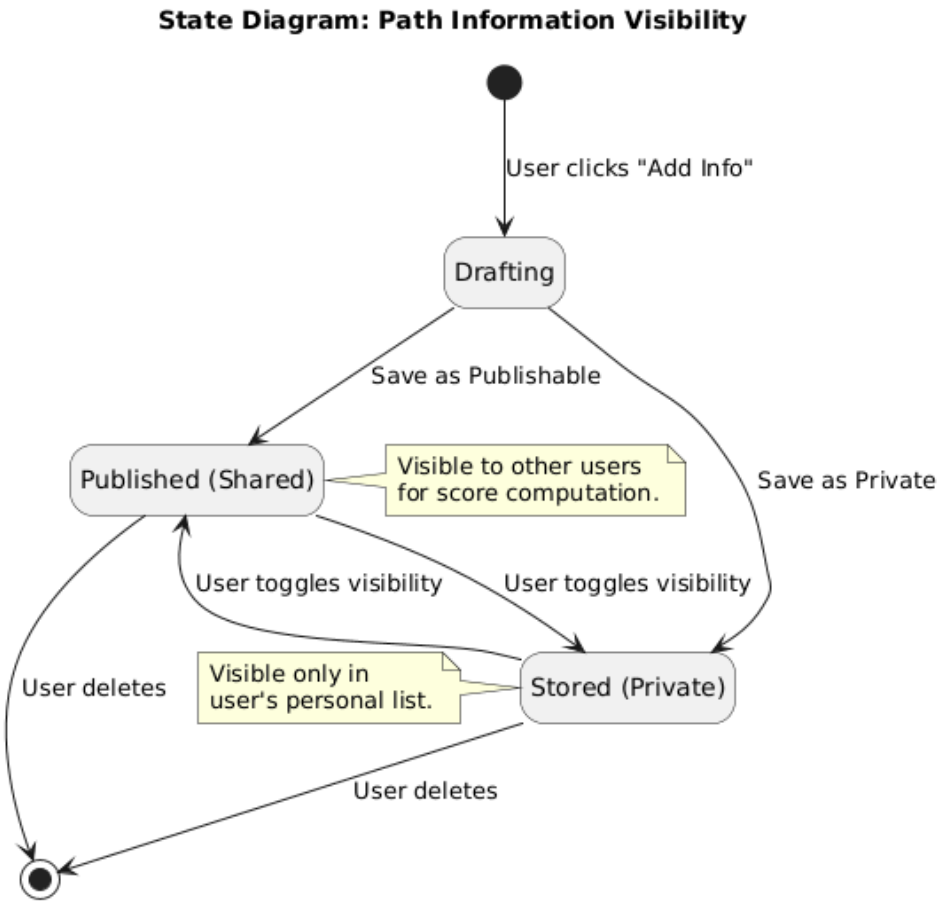

Figure 3: State Diagram: Path Information Visibility States

2.2 Product Functions

The BBP system provides a set of functionalities aimed at supporting individual cyclists in managing their biking activities and route-related information.

The main functions of the system include:

- Recording personal biking trips initiated and terminated by the user.
- Storing recorded trips and associating them with the user profile.
- Allowing users to manually insert information about bike paths based on personal experience.
- Managing the visibility of inserted bike path information according to user preferences.
- Visualizing possible biking paths between a specified origin and destination on a map.

The BBP system does not perform automatic data collection, route optimization, or aggregation of information from multiple users, as these functionalities are outside the scope of the current system.

2.3 User Characteristics

The intended users of the BBP system are individual cyclists with basic familiarity with mobile or web-based applications.

Users are expected to:

- Be able to interact with graphical user interfaces.

- Manually provide information related to bike paths based on personal experience.
- Understand basic map representations, including origin and destination points.

No advanced technical skills are required. The system is designed for occasional and personal use rather than professional or large-scale data analysis.

2.4 Assumptions, Dependencies and Constraints

2.4.1 Domain Assumptions

The correct functioning of the BBP system relies on the following assumptions about the domain and the operational environment. If these assumptions do not hold, the system may not be able to fulfill its requirements.

- **D1 - GPS Availability and Accuracy**: It is assumed that the Global Positioning System (GPS) or GNSS is available and provides geolocation coordinates with an accuracy sufficient to distinguish between adjacent streets (typically within 10-15 meters).
- **D2 - Map Service Reliability**: It is assumed that the external Map Service Provider (e.g., Google Maps, OpenStreetMap) provides up-to-date and topologically correct representations of the road network. The system assumes that if a coordinate exists on the map, the corresponding physical location allows passage.
- **D3 - Network Connectivity**: While the recording can happen offline, it is assumed that the user's device has at least intermittent internet connectivity (4G/5G/Wi-Fi) to synchronize data, retrieve weather information, and load map tiles.
- **D4 - User Trustworthiness**: It is assumed that users act cooperatively and provide truthful information when manually inserting path status or reporting obstacles. The system cannot physically verify the existence of a pothole reported by a user.
- **D5 - External Weather Service**: It is assumed that the third-party Weather Service API is operational and maintains historical or real-time weather data for the locations where trips occur.
- **D6 - Device Capability**: It is assumed that the user's smartphone has sufficient battery capacity to maintain GPS tracking and screen activity for the duration of an average biking trip (approx. 1-2 hours).

2.4.2 Dependencies

The BBP system depends on:

- **External Map APIs**: For rendering maps and calculating routes.
- **Weather Data Providers**: For enriching trip records.
- **App Stores**: For distribution and updates of the mobile application.

3 Specific Requirements

3.1 External Interface Requirements

3.2 External Interface Requirements

3.2.1 User Interfaces

The user interface of the BBP application shall be designed to be intuitive, responsive, and accessible, considering that users might interact with it while outdoors.

- **Dashboard Map View**: The main screen shall display a map centered on the user's current location, showing nearby bike paths and their status codes (e.g., Green for optimal, Red for maintenance required).
- **Trip Recording Interface**: A simplified high-contrast screen displaying real-time metrics (Duration, Distance, Current Speed) with large "Pause" and "Stop" buttons to facilitate interaction during stops.
- **Reporting Form**: An input form for manually adding path information, featuring dropdown menus for "Status" and "Obstacles" to minimize typing effort.
- **Trip History**: A chronological list of past trips, allowing users to tap and view detailed statistics and weather data.

All UI components shall adhere to the Material Design (Android) and Human Interface Guidelines (iOS) standards.

3.2.2 Hardware Interfaces

The system requires direct interaction with the following hardware components of the mobile device:

- **GPS/GNSS Sensor**: The system shall interface with the device's location services to obtain geolocation coordinates (latitude, longitude, altitude) with a minimum accuracy of 10 meters.
- **Storage**: The system shall access the device's internal storage to cache map data and temporarily save trip logs before synchronization.
- **Network Interface**: The system shall utilize the device's Wi-Fi or Cellular Data (4G/5G) interface to communicate with the backend server and external APIs.

3.2.3 Software Interfaces

- **Map Service API**: The system shall interface with an external map provider (e.g., OpenStreetMap or Google Maps API) to render map tiles and perform geocoding.
- **Weather Service API**: The system shall communicate with a third-party meteorological service (e.g., OpenWeatherMap) via REST API to retrieve weather conditions based on timestamp and location.
- **Mobile Operating System**: The application shall be compatible with Android 12+ and iOS 15+, utilizing native OS APIs for background location tracking.

3.2.4 Communication Interfaces

- **Protocol**: All client-server communication shall be encrypted using TLS 1.3 (HTTPS) to ensure data confidentiality.
- **Data Format**: Data exchange regarding trips, paths, and user profiles shall be formatted in JSON (JavaScript Object Notation).

3.2.5 User Interfaces

The BBP system shall provide a mobile application interface for cyclists. Key screens include:

- **Dashboard/Map View**: The main screen displaying the user's current location and nearby bike paths on a map.
- **Trip Recording Interface**: Displays real-time statistics (speed, duration) and controls to start/stop recording.
- **Trip History**: A list of previously recorded trips with summary data.
- **Path Information Form**: A form to manually insert data about a path (status, obstacles) and set visibility.

The UI shall be designed to be intuitive and usable on touch-screen devices.

3.2.6 Hardware Interfaces

The system requires interaction with the following hardware components of the user's mobile device:

- **GPS Receiver**: To track the user's geolocation (latitude, longitude) during trip recording.
- **Touch Screen**: For user input and interaction.
- **Network Interface**: (4G/5G/Wi-Fi) To communicate with the backend server and external services.

3.2.7 Software Interfaces

- **Map Service API**: The system shall interface with an external map provider (e.g., Google Maps API or OpenStreetMap) to render maps and visualize paths.
- **Weather Service API**: The system shall interface with an external meteorological service to retrieve weather conditions (temperature, wind, rain) associated with a recorded trip.
- **Mobile Operating System**: The application shall be compatible with major mobile OS platforms (e.g., Android, iOS).

3.2.8 Communication Interfaces

- The client application shall communicate with the BBP backend server using **HTTPS** to ensure data security.
- Data exchange format shall be **JSON** for structured information transfer.

3.3 Functional Requirements

The following is a list of functional requirements derived from the system goals.

Trip Recording

- **R1**: The system shall allow the user to start and stop the recording of a trip.
- **R2**: The system shall track the user's geolocation via GPS during the trip.
- **R3**: The system shall calculate trip statistics, including total distance and average speed.
- **R4**: The system shall retrieve weather data from an external service upon trip completion.
- **R5**: The system shall store the recorded trip details and statistics in the user's history.

History Management

- **R6**: The system shall allow users to view a list of their past trips.
- **R7**: The system shall allow users to view detailed information for a specific past trip.

Path Information Management

- **R8**: The system shall allow users to manually insert information about a bike path (street name, status, obstacles).
- **R9**: The system shall allow users to set the visibility of their inserted information to "Private" or "Publishable".

Path Visualization

- **R10**: The system shall allow users to input an origin and a destination address.
- **R11**: The system shall identify and visualize possible bike paths between the origin and destination on a map.
- **R12**: The system shall compute and display a score for each path based on its status and user reports.

3.3.1 Use Cases

Name	**UC1: Record a Biking Trip**
Actors	Registered User, External Weather Service
Entry Condition	The user is logged into the BBP application, and the device's GPS is active.
Event Flow	1. The User navigates to the "Trip" section and presses "Start Recording". 2. The System begins tracking the user's location and duration. 3. The User rides their bike to the destination. 4. The User presses "Stop Recording". 5. The System calculates trip statistics (distance, average speed). 6. The System requests current weather data from the External Weather Service. 7. The System saves the trip with statistics and weather data to the user's history. 8. The System displays the trip summary to the User.

Exit Condition	The trip is successfully stored in the database and visible in the user's history.
Exceptions	(6) **Weather Service Unavailable**: If the external service does not respond, the System saves the trip without weather data and notifies the user. (2) **GPS Signal Lost**: If GPS is lost during the ride, the System pauses recording and prompts the user.

Name	**UC2: Insert Path Information (Manual)**
Actors	Registered User
Entry Condition	The user is logged into the BBP application.
Event Flow	1. The User selects "Add Path Info". 2. The User manually enters the name of the street(s) included in the path. 3. The User selects the status of the path (e.g., optimal, sufficient, requires maintenance). 4. The User optionally describes obstacles (e.g., potholes). 5. The User toggles the visibility setting (Publishable or Private). 6. The User confirms the submission. 7. The System validates the input and stores the information.
Exit Condition	The path information is stored. If marked "Publishable", it becomes available for system-wide path scoring.
Exceptions	(2) **Invalid Street Name**: The System cannot verify the street name against the map database and asks the User to correct it.

Name	**UC3: Visualize Bike Paths**
Actors	User (Registered or Unregistered)
Entry Condition	The user has the application open on the map view.
Event Flow	1. The User specifies an "Origin" and a "Destination". 2. The User requests to visualize paths. 3. The System computes available paths between the points. 4. The System calculates a score for each path based on path status and effectiveness. 5. The System displays the paths on the map, highlighting the one with the highest score. 6. The User browses the path details.
Exit Condition	The best available bike paths are visualized on the map.
Exceptions	(3) **No Path Found**: The System cannot find a viable bike path between the points and notifies the User.

Name	**UC4: User Registration**
Actors	Unregistered User
Entry Condition	The user has installed the application but does not have an account.

Event Flow	1. The User launches the application and selects "Sign Up". 2. The User enters required details (email, password, username). 3. The System validates the input format. 4. The System creates a new account in the database. 5. The System confirms successful registration and logs the user in.
Exit Condition	A new user profile is created, and the user is authenticated.
Exceptions	(3) **Email Already Exists**: The System notifies the user that the email is already in use.

Name	**UC5: User Login**
Actors	User
Entry Condition	The user opens the application and is not authenticated.
Event Flow	1. The User enters email and password. 2. The User presses the "Login" button. 3. The System verifies the credentials against the database. 4. The System grants access and redirects the User to the main dashboard.
Exit Condition	The User is authenticated and can access restricted features (e.g., Record Trip).
Exceptions	(3) **Invalid Credentials**: The System displays an error message ("Incorrect email or password") and asks the user to retry.

Name	**UC6: View Trip History**
Actors	Registered User
Entry Condition	The user is logged in and has previously recorded at least one trip.
Event Flow	1. The User navigates to the "My Trips" (History) section. 2. The System retrieves the list of past trips associated with the user account. 3. The System displays the list, showing summary data (date, distance) for each trip. 4. The User selects a specific trip to view details. 5. The System displays detailed statistics (average speed, duration, weather conditions) for the selected trip.
Exit Condition	The User successfully reviews their past cycling activities.
Exceptions	(2) **No History Available**: If the user has no recorded trips, the System displays a "No trips recorded yet" message.

3.3.2 Sequence Diagrams

The following diagram illustrates the interaction flow for recording a biking trip.

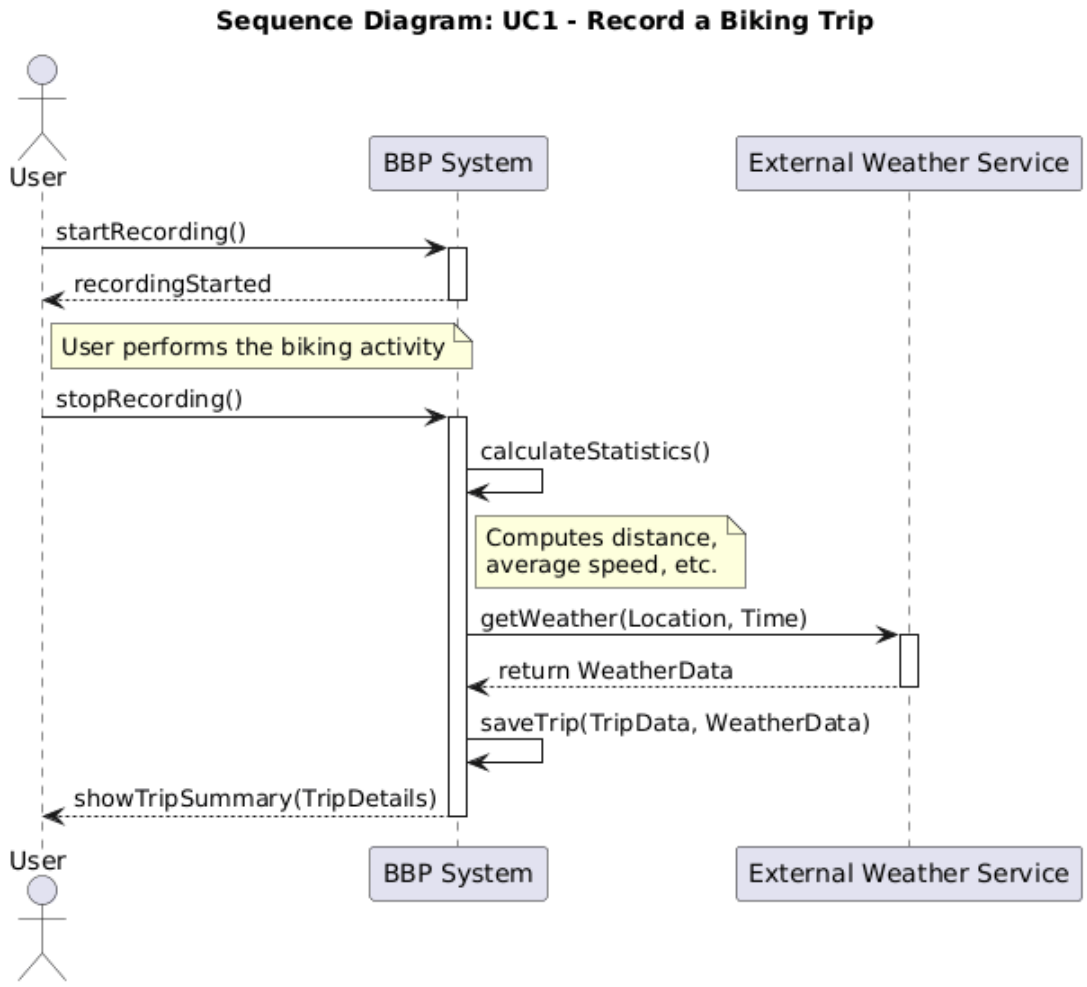

Figure 4: Sequence Diagram: UC1 - Record a Biking Trip

The process begins when the user initiates the recording...

The following diagram describes how a user manually inserts information regarding a bike path.

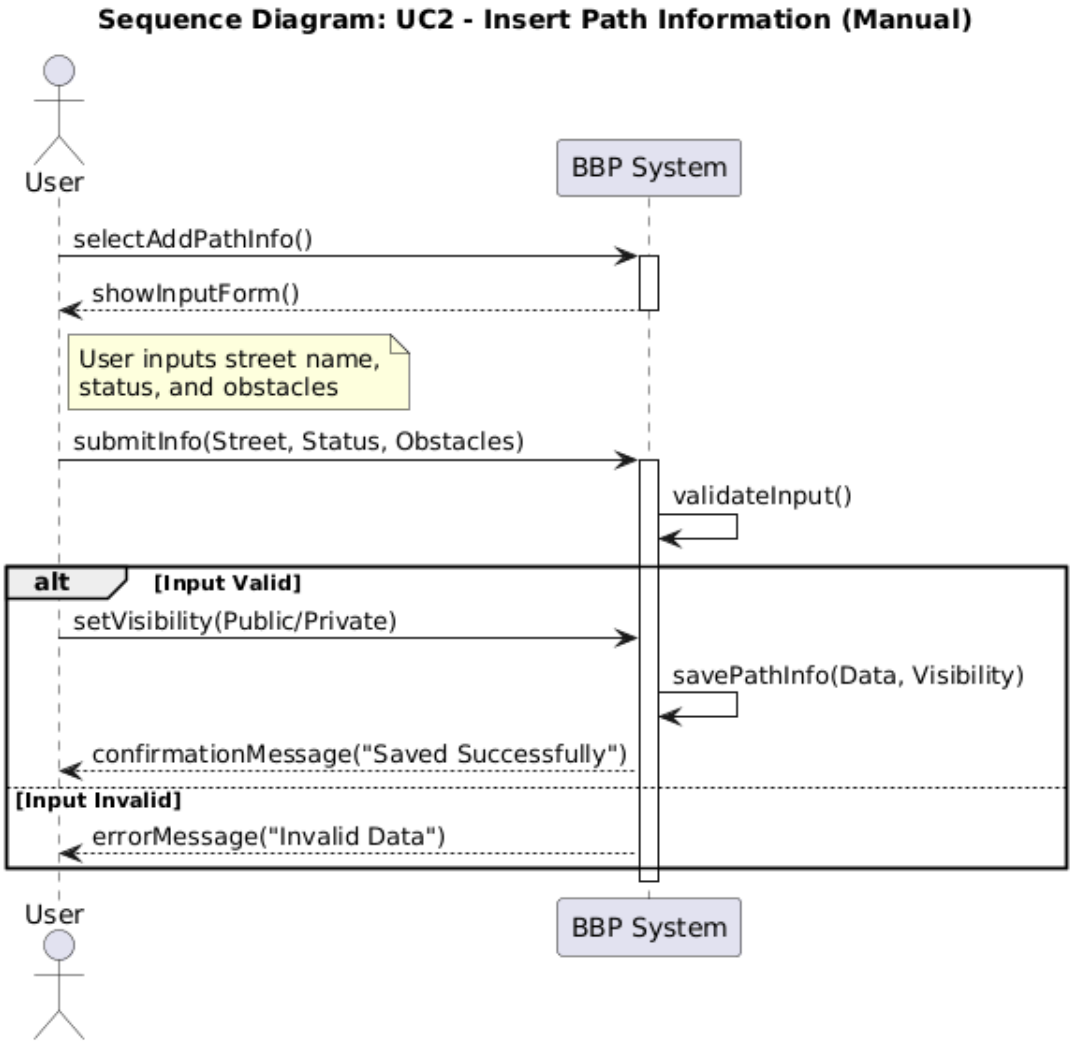

Figure 5: Sequence Diagram: UC2 - Insert Path Information

This diagram illustrates the process of requesting and visualizing the best bike path between two points.

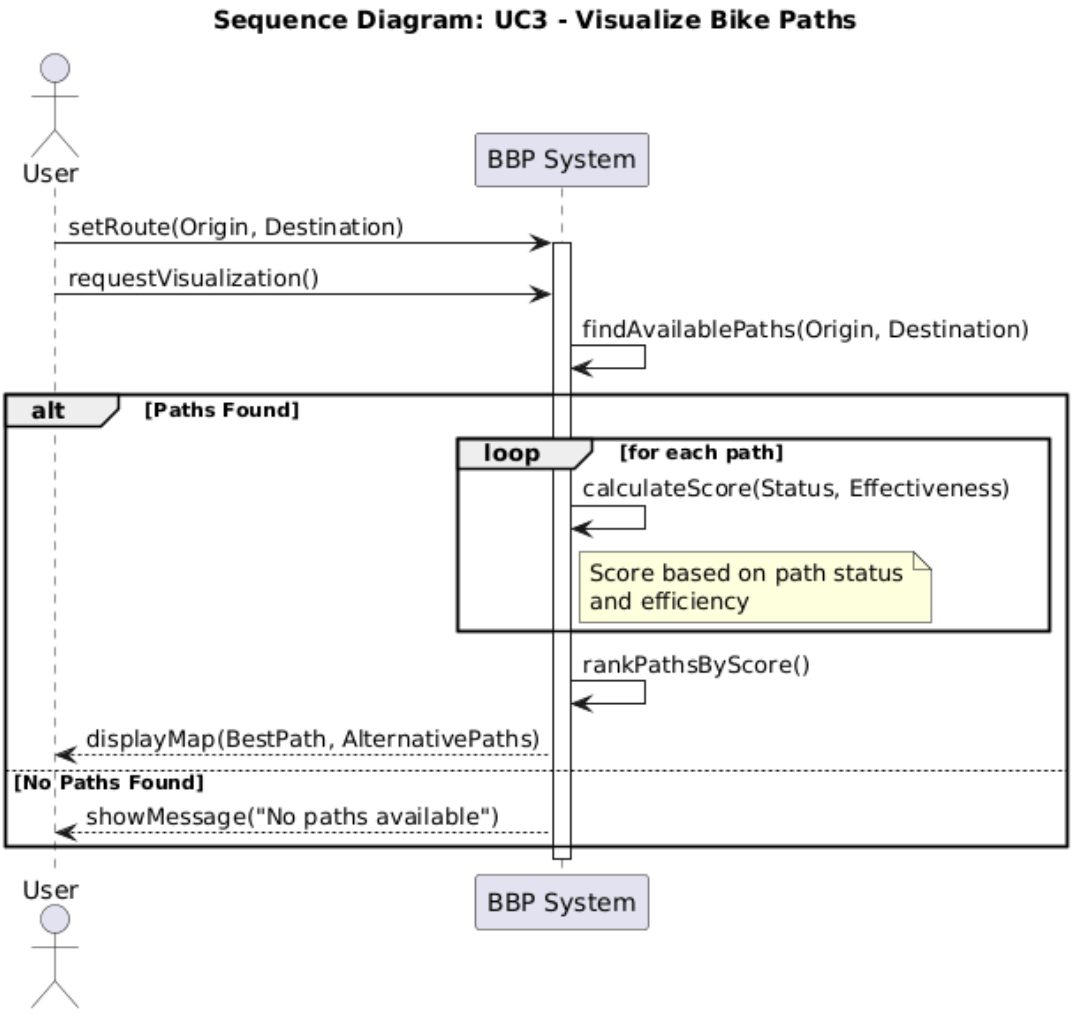

Figure 6: Sequence Diagram: UC3 - Visualize Bike Paths

The registration process flow, including validation checks, is shown below.

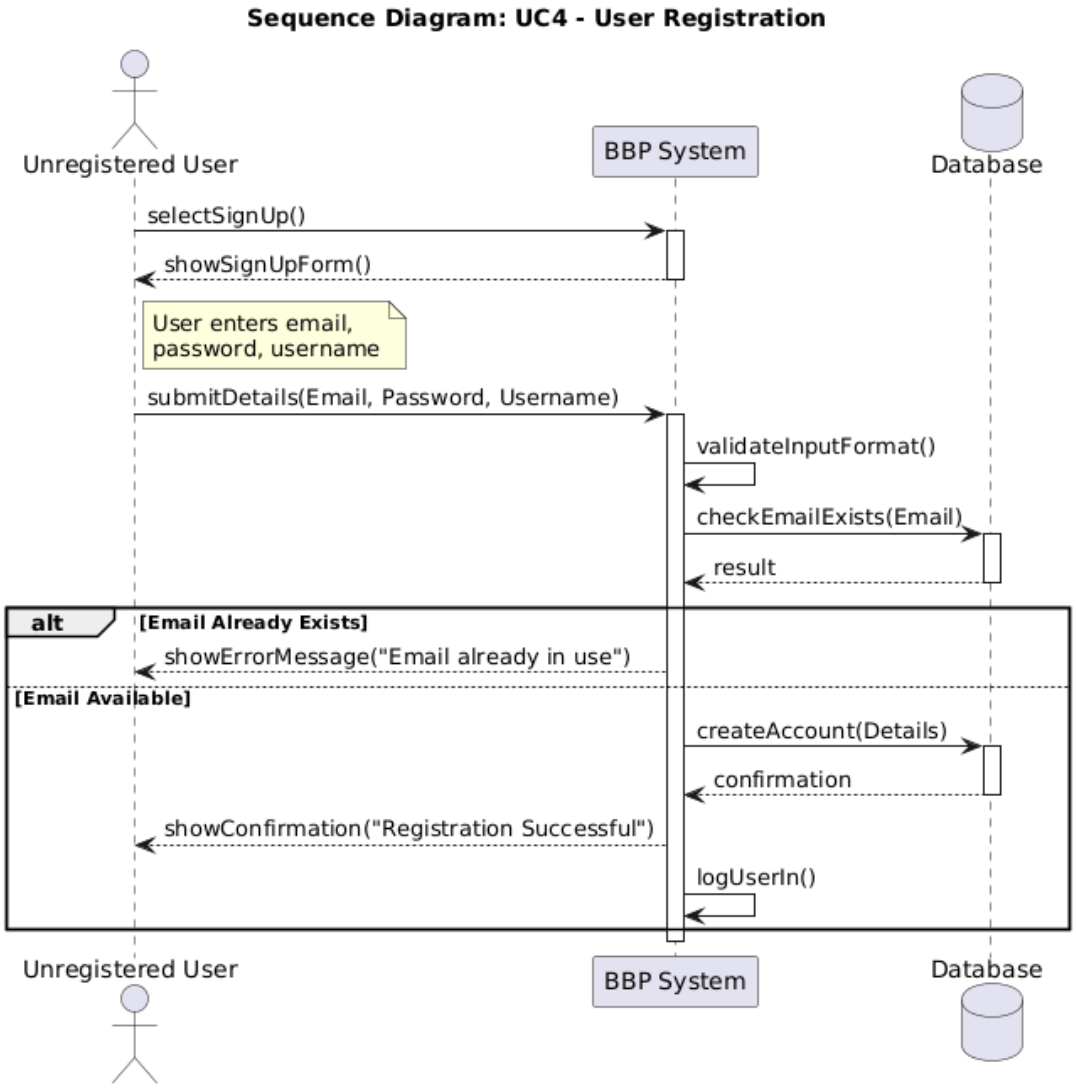

Figure 7: Sequence Diagram: UC4 - User Registration

The authentication flow for user login is depicted in the following diagram.

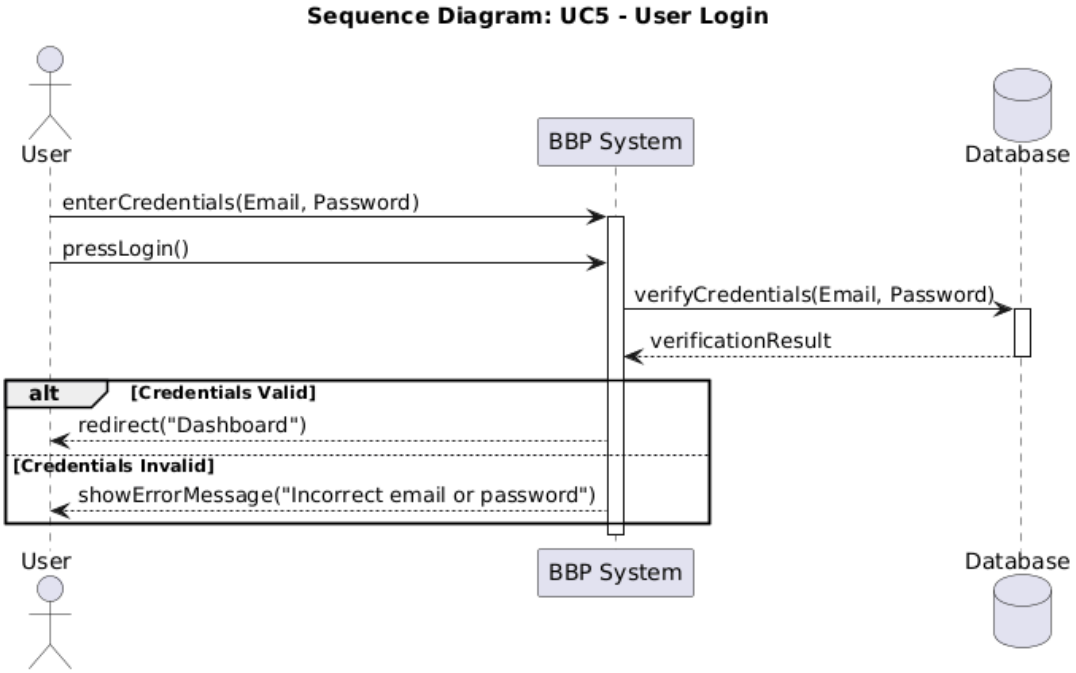

Figure 8: Sequence Diagram: UC5 - User Login

The following diagram describes the interaction for retrieving and viewing past trip records, including statistics and weather data.

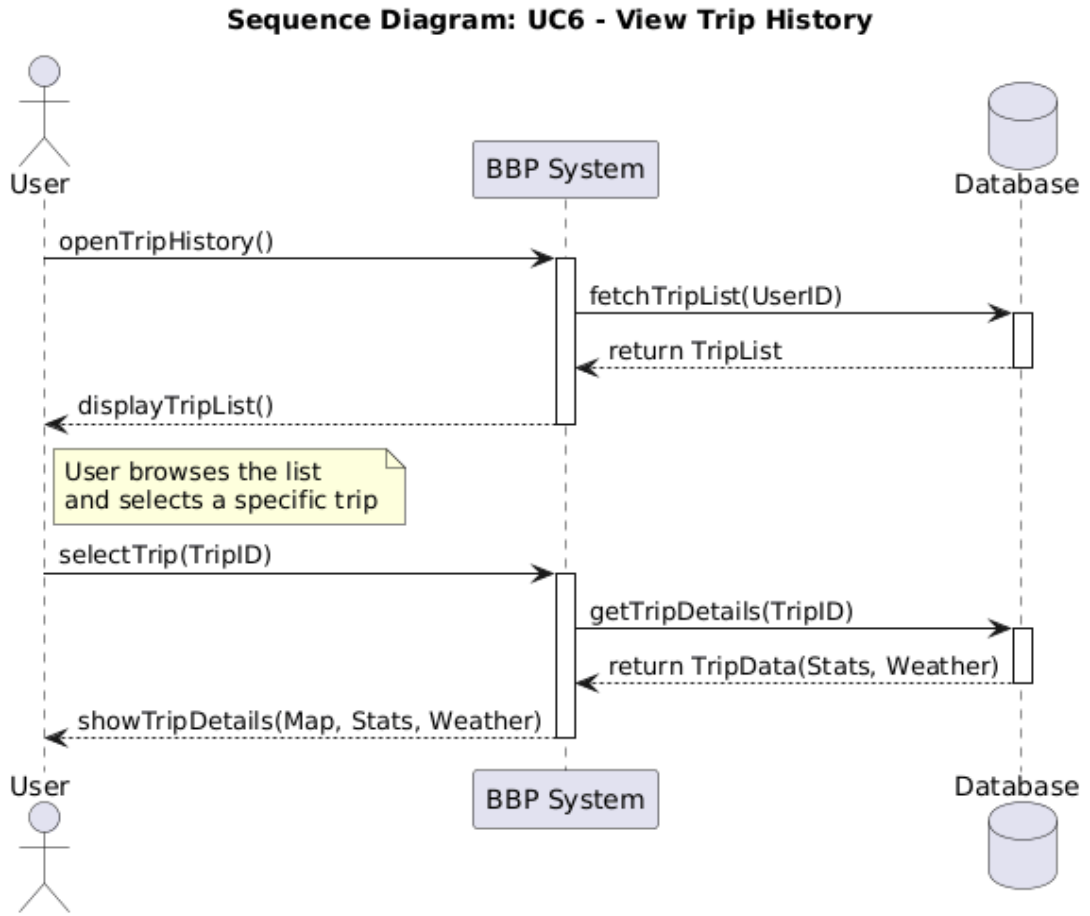

Figure 9: Sequence Diagram: UC6 - View Trip History

3.3.3 Requirement Mapping

The following table maps the Goals (G) identified in the Introduction to the Functional Requirements (R) and Domain Assumptions (D).

Goal	Related Requirements	Assumptions
G1 (Record)	R1, R2, R3, R4	D1, D2, D5
G2 (History)	R5	D3
G3 (Visualize)	R10, R11, R12	D3, D6
G4 (Add Info)	R6, R7, R8	D4
G5 (Share)	R9	D4

3.4 Performance Requirements

The BBP system shall provide responses to user interactions within a reasonable time frame.
In particular:

- Trip recording activation and termination shall be acknowledged within a few seconds.
- Visualization of bike paths shall be completed within an acceptable delay, depending on network conditions.

3.5 Design Constraints

3.5.1 Standards Compliance

- **GDPR Compliance**: Since the system collects personal data (location history, email), it must comply with the General Data Protection Regulation (GDPR) to ensure user privacy.
- **Map Data Standards**: The system handles geospatial data and should adhere to standard formats (e.g., GeoJSON) for compatibility with map services.

3.5.2 Hardware Limitations

- **GPS Accuracy**: The system's tracking precision is limited by the accuracy of the mobile device's GPS sensor, which may vary in urban canyons or bad weather.
- **Battery Consumption**: Continuous GPS tracking is battery-intensive. The application must be optimized to minimize battery drain during long trips.

3.5.3 Other Constraints

- **Network Dependency**: Core features like map visualization and weather retrieval require an active internet connection. Offline capabilities are limited to local recording (if supported by design).

3.6 Software System Attributes

3.6.1 Reliability

The BBP system shall ensure that recorded trips and inserted path information are not lost once confirmed by the user.

3.6.2 Availability

The system shall be available during normal operation times, subject to network connectivity and external service availability.

3.6.3 Security

The system shall restrict access to personal data and ensure that only authenticated users can access their recorded trips and private path information.

3.6.4 Maintainability

The BBP system shall be designed to allow future extensions, such as additional path attributes or enhanced visualization features.

3.6.5 Portability

The system shall be deployable on commonly used platforms supporting web or mobile applications.

4 Formal Analysis Using Alloy

In this section, we present a formal specification of critical aspects of the Best Bike Paths (BBP) system using Alloy. Following the project requirements, we focused on two distinct models to validate different properties of the system:

- **Static Structural Model:** This model defines the relationships between users, bike paths, trips, and path information, focusing on data consistency and visibility constraints.
- **Dynamic Behavioral Model:** This model describes the lifecycle of a trip recording, ensuring that the state transitions (from Idle to Recording to Recorded) occur in a valid temporal sequence.

4.1 Static Structural Model

The first model captures the static entities of the domain. We modeled `User`, `BikePath`, and `Trip` to represent the core data. Additionally, `PathInformation` was introduced to handle the visibility settings (`Private` or `Shared`) of user-generated content.

```
// ------------------------------------------------------
// MODEL 1: STATIC STRUCTURE
// ------------------------------------------------------

sig User {}
sig BikePath {}

sig Trip {
    user: one User,
    path: one BikePath
}

sig PathInformation {
    author: one User,
    path: one BikePath,
    visibility: one Visibility
}

enum Visibility {
    Private,
    Shared
}

run {} for 5
```

4.1.1 Visualization of Static Model

We executed the static model to generate instances that verify the relationships between the entities. Figure 10 illustrates a generated world showing the connections between Users, Trips, and the Path Information they author.

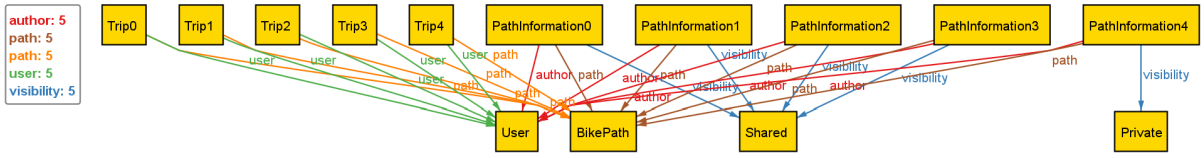

Figure 10: Alloy Instance: Static relationships between Users, Trips, and Path Information

4.2 Dynamic Behavioral Model

The second part of the analysis focuses on the time-dependent behavior of the `Trip` entity. We utilized the `util/ordering[Time]` module to simulate the progression of states. We defined a `TripState` enumeration containing `Idle`, `Recording`, and `Recorded` values, and we modeled the specific constraints allowing transitions between these states over time.

```
// -------------------------------------------------------
// MODEL 2: DYNAMIC BEHAVIOR (Trace)
// -------------------------------------------------------

module bbp_trace

open util/ordering[Time]

enum TripState { Idle, Recording, Recorded }

sig Time {}

sig User {}
sig BikePath {}

sig Trip {
    user: one User,
    path: one BikePath,
    state: Time -> one TripState
}

fun st[t: Trip, tm: Time]: one TripState {
    t.state[tm]
}

// F1: Initial State - All trips start as Idle
fact InitialState {
    all t: Trip | st[t, first] = Idle
}

// F2: Lifecycle Constraints
// Defines valid transitions between time steps
fact TripLifecycle {
    all t: Trip, tm: Time - last | {
        let s  = st[t, tm] |
        let s2 = st[t, tm.next] |
        (s = Idle      implies (s2 = Idle or s2 = Recording)) and
```



```
        (s = Recording implies (s2 = Recording or s2 = Recorded)) and
        (s = Recorded  implies (s2 = Recorded))
    }
}

// ASSERTION: Consistency Check
// Verifies that a trip cannot be 'Recorded' without having been 'Recording'
assert RecordedImpliesWasRecording {
    all t: Trip |
        (some tm: Time | st[t, tm] = Recorded) implies
        (some tm: Time | st[t, tm] = Recording)
}

check RecordedImpliesWasRecording for 6 but exactly 4 Time

// PREDICATE: Simulation
// Generates a trace where a trip successfully reaches the Recorded state
pred EventuallyRecorded {
    some t: Trip | st[t, last] = Recorded
}

run EventuallyRecorded for 6 but exactly 4 Time
```

4.2.1 Analysis Results

We executed the model using the Alloy Analyzer to validate the system logic.

Assertion Checking The assertion `RecordedImpliesWasRecording` was checked within a scope of 6 atoms and 4 time steps.

```
Executing "Check RecordedImpliesWasRecording for 6 but exactly 4 Time"
No counterexample found. Assertion may be valid.
```

This result confirms that, under the modeled constraints, it is impossible for a trip to bypass the recording phase and appear directly in the saved history (Recorded state).

Simulation To visually verify the model behavior, we ran the `EventuallyRecorded` predicate. Figure 11 demonstrates a valid execution trace generated by the analyzer, showing the state of a Trip object evolving over distinct time steps.

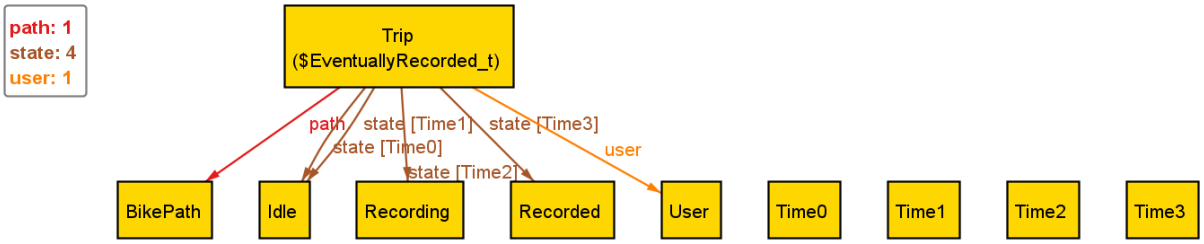

Figure 11: Alloy Trace: Dynamic evolution of the Trip State over Time

5 Effort Spent

Name	Task	Hours
Haipeng Zhu	RASD preparation	30

References

