POLITECNICO
MILANO 1863

COMPUTER SCIENCE AND ENGINEERING
SOFTWARE ENGINEERING II
2025 - 2026

RASD

Requirement Analysis and Specification Document

Authors: Cristhian Mejia and Sravan Yerranagu

Version: 1.0

Date: 23-December-2025

Download page: `https://github.com/ReGaL24/MejiaYerranagu`

Contents

1 Introduction

1.1 Purpose

Urban mobility is increasingly moving towards sustainable transport modes, and cycling is becoming a key strategy. However, cyclists often lack reliable information on the quality, safety, and suitability of bike routes, as existing data are often scattered, outdated, or missing altogether. At the same time, cyclists continuously produce valuable data through their daily trips, which could be used to significantly improve knowledge about cycling infrastructure.

The purpose of Best Bike Paths (BBP) is to provide a software system that enables cyclists to record and analyze their personal trips and build and search an inventory of bike paths enriched with community-provided information. BBP supports both manual and sensor-based acquisition of path information, ensuring that automatically detected issues are reviewed and confirmed by users before being shared. Additionally, the system assists any user in identifying and visualizing suitable bike paths between a given origin and destination, ranking alternatives according to path conditions and route effectiveness, even if the user is not registered.

1.1.1 Goals

G1: Allow registered users to log personal rides and view summary stats (distance, duration, average speed, and key performance metrics).

G2: Let registered users manually create and maintain bike path data by defining route segments and tagging conditions and obstacles.

G3: Let registered users automatically record bike path data during rides via GPS-based path reconstruction and sensor-based anomaly detection.

G4: Require user review before publishing automatically collected path data, and let contributors choose whether their submissions are shared with the community.

G5: Let any user view and compare bike paths between an origin and destination on a map, ranking options by a score combining route effectiveness and path conditions.

1.2 Scope

Best Bike Paths (BBP) is a software system that supports cyclists in recording personal biking trips and in managing an inventory of bike paths enriched with user-contributed information. The system enables registered users to record trips and access related statistics, optionally including meteorological information retrieved from an external service.

Registered users can also publish bike path information through two modes: *manual mode*, where users explicitly define the path segments and associate route status and obstacles, and *automated mode*, where the system reconstructs the followed path through GPS data and detects potential obstacles through signals acquired from the mobile device sensors. Since automated detection may produce inaccurate results, BBP requires users to review and confirm acquired information before it can be made publishable, and it allows contributors to control whether their information is shared with the community. Both registered and unregistered users can search in the system by specifying an origin and a destination and view one or more possible bike paths on a map, ranked according to a path score reflecting both path conditions and route effectiveness.

BBP includes the functionalities for trip storage and statistics computation, acquisition and management of bike path information, user confirmation, publication control, map-based visualization and ranking of paths. BBP relies on external services and device sensors as data sources but does not control them.

1.2.1 World Phenomena (WP)

The system operates within a world where:

WP1: Cyclists physically traverse bike paths in urban and suburban environments.

WP2: Mobile devices generate raw positioning data (e.g., GPS coordinates) with inherent accuracy and coverage limitations.

WP3: Mobile devices generate sensor data (e.g., accelerometer and gyroscope signals) reflecting physical movements during biking.

WP4: Road infrastructure elements (e.g., bike lanes, potholes, obstacles) exist as physical entities and may change over time.

WP5: Weather conditions vary over time and location, and external meteorological services maintain authoritative datasets.

WP6: Different cyclists may traverse the same paths at different times and under different conditions.

WP7: Users form subjective assessments of bike path quality based on personal experience.

WP8: Network connectivity may be intermittent during outdoor biking activities.

1.2.2 Shared Phenomena (SP)

SP1: The system acquires location data transmitted by a user's mobile device during biking activities.

SP2: The system infers biking activity based on observed movement characteristics (e.g., speed).

SP3: The system computes and stores statistics related to recorded biking trips.

SP4: The system retrieves meteorological information from an external weather service associated with a recorded trip.

SP5: The system presents recorded trip data enriched with contextual information to registered users.

SP6: Registered users manually insert information about bike paths, including involved segments, qualitative status, and obstacles.

SP7: The system acquires potential bike path information through automated sensing during biking activities.

SP8: Registered users review, confirm, or correct automatically acquired bike path information.

SP9: Registered users decide whether their contributed bike path information is made publishable.

SP10: Users specify an origin and a destination to query bike paths.

SP11: The system visualizes one or more candidate bike paths on a map, ordered according to a computed path score.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- **Bike path**: A route suitable for cycling, either characterized by the presence of a dedicated bike track or by low vehicular traffic and speed limits compatible with average cycling speed.
- **Path status**: A qualitative assessment of the condition of a bike path, expressed using predefined categories such as Optimal, Medium, Sufficient, or Requires Maintenance.
- **Path score**: A numeric value used to rank alternative bike paths between a given origin and destination, reflecting both path condition and route effectiveness.
- **Publishable information**: User-contributed bike path information that has been explicitly marked by its owner as available for consultation by other users.
- **Trip**: A complete cycling journey recorded by the system, defined by a start time, an end time, and associated measured and computed data.
- **Trip statistics**: A set of computed metrics derived from a recorded trip, including distance, duration, average speed, maximum speed, and elevation gain.
- **Obstacle**: A physical condition or element along a bike path that may negatively affect cycling safety or comfort, such as potholes or surface irregularities.
- **Weather enrichment**: The association of meteorological information (e.g., temperature, wind speed, weather conditions) retrieved from an external service with a recorded trip.
- **Manual mode**: A mode of interaction in which a user explicitly inserts bike path information without relying on automated sensing or inference.
- **Automated mode**: A mode of interaction in which the system acquires bike path information during a biking activity by analyzing data collected from the user's mobile device, such as GPS and motion sensors.
- **Registered user**: A user who has created an account in the system and is authorized to record trips and contribute bike path information.
- **Unregistered user**: A user who accesses the system without authentication and is limited to querying and visualizing bike paths.

1.3.2 Acronyms

- **BBP**: Best Bike Paths
- **GPS**: Global Positioning System
- **API**: Application Programming Interface
- **HTTPS**: Secure HTTP protocol for encrypted communications
- **RASD**: Requirements Analysis and Specification Document

1.3.3 Abbreviations

- **G**: Goal
- **WP**: World Phenomenon
- **SP**: Shared Phenomenon

1.4 Reference Documents

The assignment for this document and all the information included refer to the following documentation:

- The specification for the 2025/26 Requirement Engineering and Design Project for the Software Engineering II course.
- The slides on the WeBeep page of the Software Engineering II course.

1.5 Document Structure

- **Section 1 (Introduction)**: Problem context, scope boundaries, terminology, document metadata.
- **Section 2 (Overall Description)**: High-level system perspective through scenarios and domain models.
- **Section 3 (Specific Requirements)**: Detailed interfaces, functional requirements, performance specifications, quality attributes.
- **Section 4 (Formal Analysis)**: Alloy formal specification of critical system properties.
- **Section 5 (Effort Spent)**: Time allocation and effort tracking by team member.
- **Section 6 (References)**: Source materials and documentation.

2 Overall Description

2.1 Product Perspective

2.1.1 Scenarios

Scenario 1: Personal Trip Recording

Giovanni, an avid cyclist, opens the BBP app on his smartphone and starts a trip record before starting his 10 km commute to work. The app displays a real-time dashboard showing the current speed, elapsed time and also the distance traveled. Giovanni's location is continuously captured using his smartphone GPS module and the system calculates the distance in real-time. When he stops recording the trip after completing his commute, the system automatically computes statistics such as the total distance covered, total time spent - the duration of the trip, average speed, maximum speed and elevation gain. The system further queries the weather service and retrieves the conditions for the trip - Temperature, Humidity, Cloud cover and Wind speed. Since Giovanni is a registered user, the trip is stored in his personal account with full statistics along with the weather enrichment data. This trip can now be viewed by him along with a map visualization of his GPS tracking.

Scenario 2: Manual Path Entry

Thomas is a registered user with a deep understanding of the local biking routes. He wants to contribute information regarding the "Milan and Pavia route along the Navigli". For this, he opens the BBP app and selects "Add Path Information", He can now enter the path name, tracing the route on the map manually or by typing in the street names. He can now specify the path status to be "Optimal" based on his experience and also make a note about the presence of dedicated bike lane infrastructure. Thomas can now review the information on the map preview and then publish it. The path now becomes immediately visible to all users querying path in that geographical area.

Scenario 3: Origin - Destination Path Query

Paul is a visitor from Germany, he is captivated by the beauty of Milan and want to further explore the city with the help of a bike. Paul downloads the BBP app form the app store and opens the app without registering. Paul is presented with a searching tool where he enter the Origin and Destination, the system then geocodes the entered address, queries the database of published paths and returns thee possible routes for Paul to choose. The system computes a Ride-ability score for each of these routes based on path status and routing effectiveness.

For Example, Paul wants to go from Milano Centrale (Origin) to Piazza Sempione (Destination), the system now geocodes the locations and gives the following results:

1. via Via S. Marco - Score 0.92, Status Optimal , 4.2 km
2. via C.so di Porta Nuova - Score 0.85, Status Optimal , 4.0 km
3. via Via Vittor Pisani - Score 0.78, Status Medium , 5.0 km

The paths scores are color coded either Green, Yellow, Orange and Red for easier understanding.

Paul can now review the paths on the results page with the help of an interactive maps where the paths are color-coded for better visualization and understanding helping him select the best optimal route for his circumstances.

Scenario 4: Weather Service Unavailability

During Nicolas's trip recording, the weather service becomes temporarily unavailable due to network issues on the service provider's end, BBP system detects the weather services connection outage after 5 seconds and gracefully degrades : The trip is recorded normally with all the riding statistics, except the weather information which is marked as "Unavailable". When Nicolas finishes the trip, the app displays "Trip recorded successfully (Weather data unavailable - try refreshing later)". Nicolas' trip is stored on his account and can be accessed immediately albeit without the weather information. As the weather service becomes available, the app provides Nicolas with an option to retroactively fetch the missing weather information for his trip.

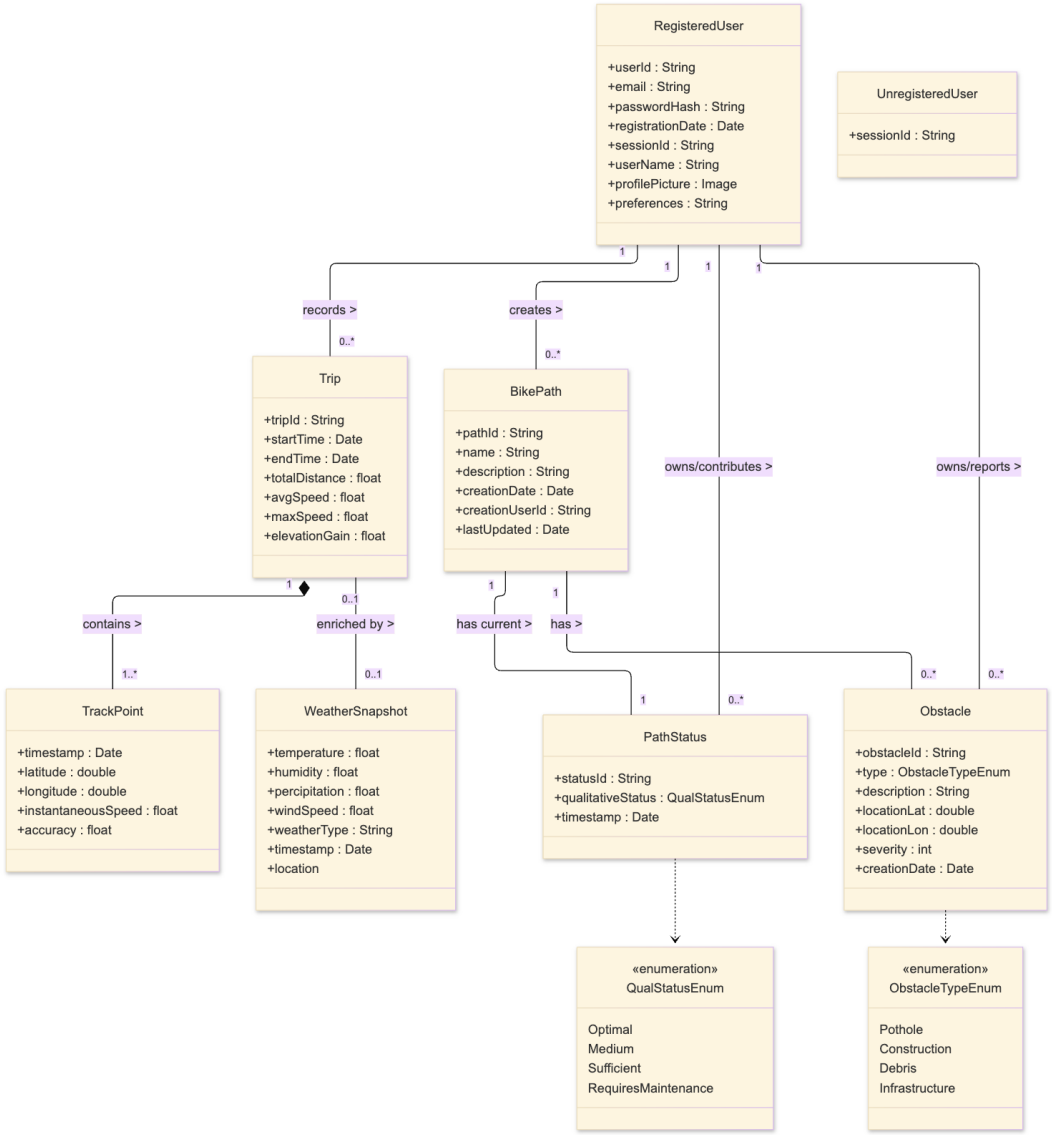
Scenario 5: Viewing Personal Trip History

Marco opens the BBP app and navigates to his trip history. The system display the trips in a ever refreshing list where each trip is previewed with date, distance, average speed, the starting location of the trip and the endpoint along with a small thumbnail of the trip map. When Marco selects a trip, the system opens the trip showing the data in depth: a map with his GPS tracking, the riding statistics : Distance covered, Duration of the trip, Average speed, Maximum speed and Elevation gain and also weather conditions at that time : Temperature, Humidity, Cloud cover, Wind speed and Precipitation.

Scenario 6: Path Query Failure

If the system route computing services experiences a temporary outage lasting 5 to 8 seconds during the path query operation, the users request times out. The system then gracefully degrades - instead of crashing, the app lets the user know that the route searching is taking longer than expected, asking the user to try again or to adjust the query. The user can then try again after the service recovers.

2.1.2 Domain Class Diagram

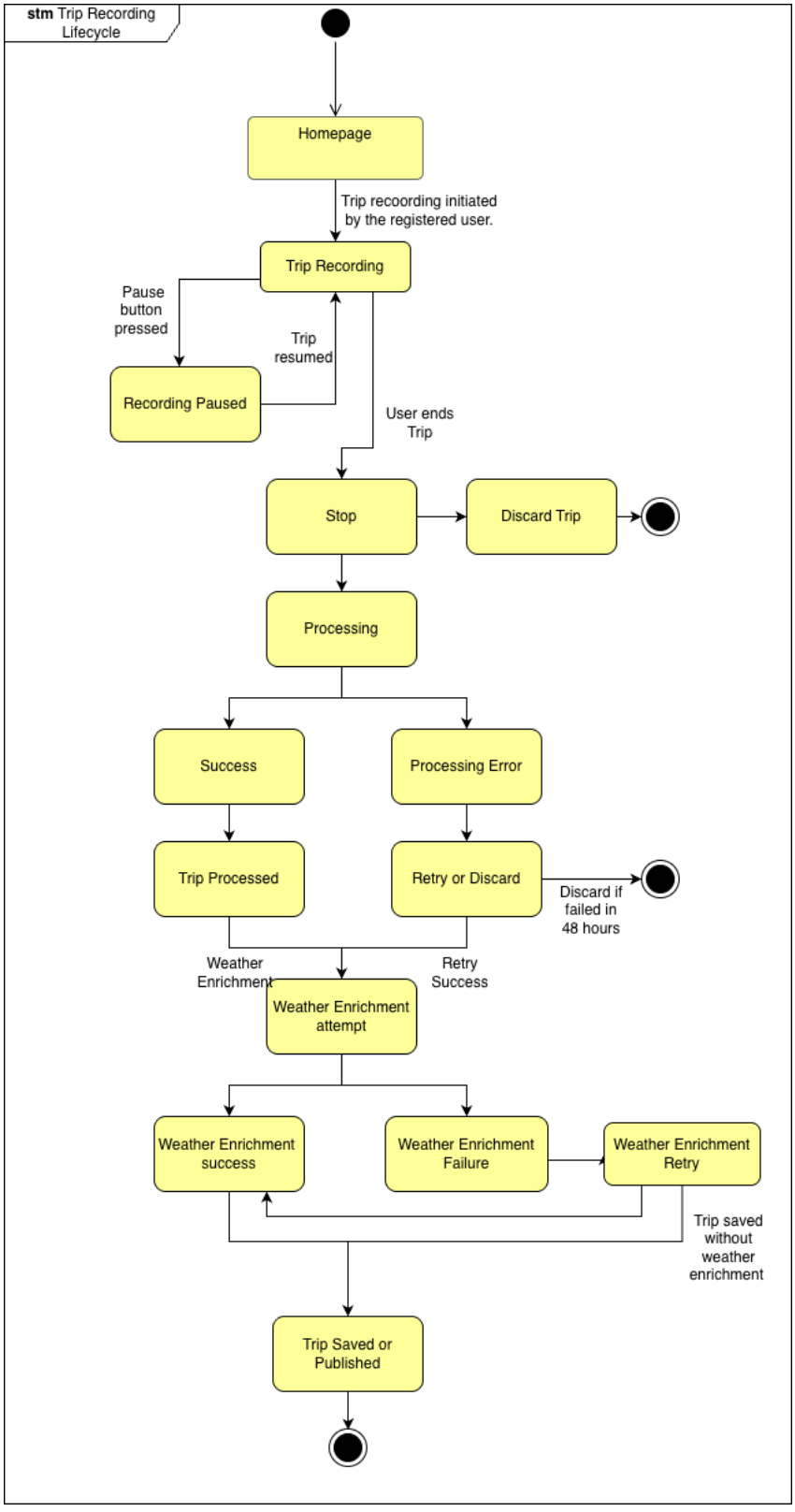


The Domain Class Diagram represents all the entities that are involved in the system and highlights the relationships between them. It is important to note that:

- The `UnregisteredUser` only has the ability to search for existing bike paths and in no shape or form has the power to grade an trip.
- Only the `RegisteredUser` has a persistent profile and has the ability to add any obstacles, publish new routes, adjourn paths with statues. `UnregisteredUser` has no stored data and doesn't posses any ability to change the parameters of a route in any sense.
- Once a trip is finished and recorded, to ensure data integrity, the trip data can't be modified.
- The `WeatherSnapshot` is best-case and best-effort, its availability or unavailability is not a metric for validation of a trip.
- Each `BikePath` is associated only with the current `PathStatus` for visualization purposes. Previous status information is not explicitly managed within the system scope and does not affect path discovery and visualization.

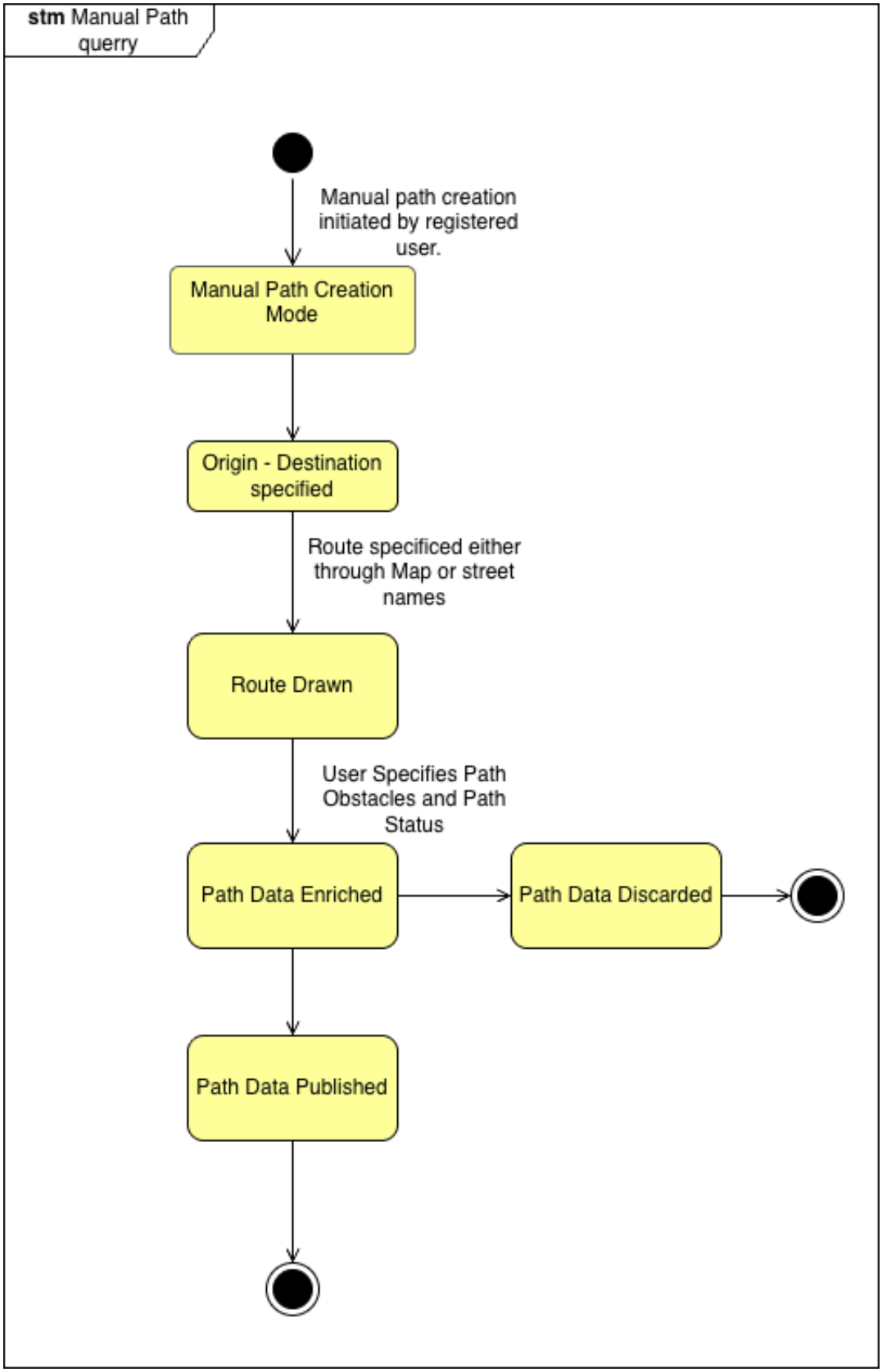
2.1.3 State Diagrams

Trip Recording Lifecycle



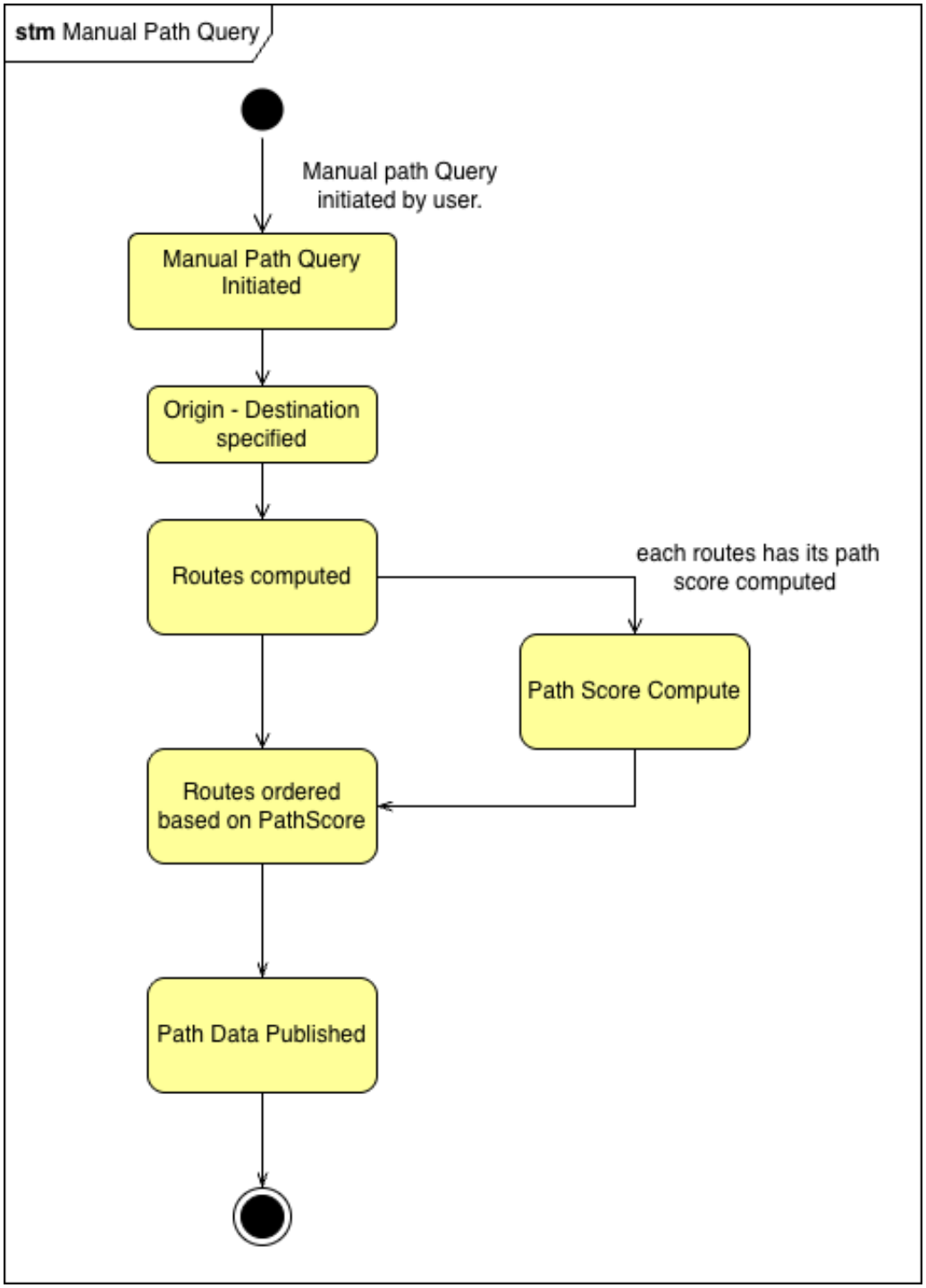
The state diagram describes the different states of a trip recording. The processing of trip information must completed in 60 seconds or else the trip will be saved as a draft for upto 48 hours after which the draft is deleted permanently. The trip recording can be cancelled at any time before the trip is saved, after which it becomes immutable. Weather enrichment is purely optional and its unavailability doesn't affect the trip recording process.

Manual Path Creation



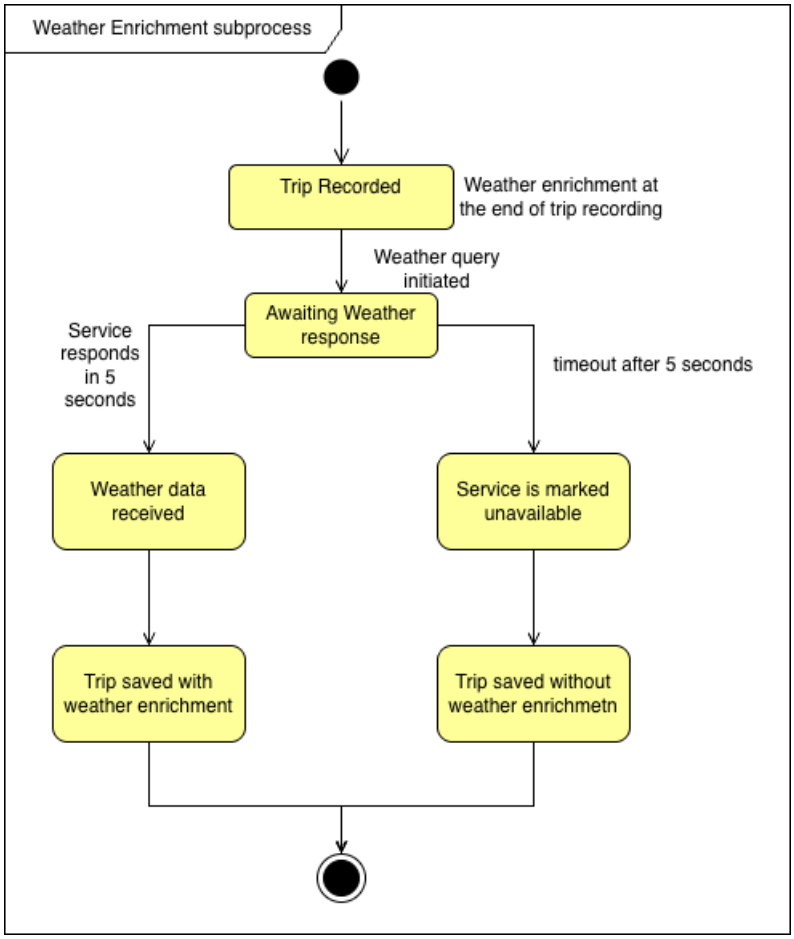
The state diagram describes the different states of a manual path creation. The `RegisteredUser` can enter the path information by specifying the Origin - Destination pair or by specifying the street names and can edit information about the path such as the obstacles and the route status before publishing. Once published, the path information becomes immutable and can't be edited or deleted.

Manual Path Query



The state diagram describes the different states of a manual path query. Both `RegisteredUser` and `UnregisteredUser` can make use of the path query functionality. The system geocodes the origin - destination pair and queries the database for viable routes. The resulting paths are scored and ranked before being presented to the user. In case of service unavailability, the system gracefully degrades and prompts the user to try again.

Weather Enrichment Subprocess



The state diagram describes the flow of the weather enrichment subprocess. The weather enrichment is triggered after a trip is recorded. The system queries an external weather service with the trip location and time frame to retrieve the weather data. In case of service unavailability or timeout, the system gracefully degrades and marks the weather data as "Unavailable". The user can then manually refresh the weather data within 48 hours of trip completion.

2.2 Product Functions

Sign up and Login

The BBP applications landing page provides the user with 3 options on opening the application for the first time. The three options are as follows :

- Sign Up
- Login
- Continue as Unregistered User

With the Sign up options, the user has to provide a valid E-mail address and a strong password to create a new account. Once the user has done so, the system takes the user on a path to fill the basic necessary information like the name, age, sex, phone number, preferred username. The system also offers the user an option to upload a profile picture and to save their height and weight to the newly created profile. This functionality is purely optional and the user can opt out of this anytime in the profile tab.

To login the user enters the Email address or the Username in the corresponding field along with Password to log in and access the system to its full potential.

When an user clicks on "Continue as Unregistered User", the system creates a new

sessionID and allows the user to enter which limited functionality. I.e. only route searching.

Profile enhancements

A `RegisteredUser` can enhance their profiles by adding a profile picture as well as adding their body measurements. The body measurements are visible only to the user and are present to further motivate the user towards a healthy living. `RegisteredUser` can also add their equipment details like the make and model of the bike, the make of the helmet, tracksuit, boots and any modifications done to the bike to the profile. These Equipment details will show up on the users landing pages when searched by another user.

Trip Recording and Statistics Computation

`RegisteredUser` initiate a trip recording by pressing the ever present START button, upon starting a trip the system acquires a GPS lock and the system continues to acquire GPS coordinates with a frequency of 1 or more per second along with the accuracy metadata. The system detects biking activity based on sustained velocity over a small period of time accounting for unforeseen halts or stoppages. Upon completion of a trip where the user presses the STOP button on the screen, the system computes distance, duration, average speed, maximum speed and elevation gain from the acquired information. This information after computation is displayed to the user immediately.

Weather Data Enrichment

Once a trip is recorded/completed, the system then queries external weather service with the trip location and time frame. If the service is responsive, the system retrieves temperature, humidity, cloud cover, wind speed and precipitation. This enriched data is then attached to the trip report which is visible in the trip detail view. In case the externally managed weather services is offline or takes >5 seconds to respond, the trip is stored without the weather data. Users can then trigger a manual update to retrieve the weather information for a trip within 48 hours of the trips completion.

Manual Path Information Entry

`RegisteredUser` can also create new paths by manually entering the origin and destination specifying a path name, street name. The user can further add path status - Optimal, Medium, sufficient or Requires maintenance, to the routes created manually. User can then specify zero or more obstacles along the path. The obstacles being pothole, construction, debris or Infrastructure. The user is provided with an interactive map which shows the path with all the information entered, further editing when necessary. When the user is happy with the details, he can publish the path information, making the path visible to all users in the path queries.

Trip Storage and Personal History

Each trip that a `RegisteredUser` completes, is stored on profile along with full GPS track, ride statistics, weather enrichment (if available) and timestamps. `RegisteredUser` can browse through their personal trip history which updates through dynamic scrolling function. the system provides a preview with date, distance, duration along with a small thumbnail of the trip map, which when clicked opens an in-detail trip summary showing detailed ride statistics, map data with GPS tracking and weather data. The trip data is stored permanently to the user account and can never be edited.

Path Query and Discovery

Any `User` (Registered or Unregistered) can make use of path query function. the user can specify the origin and destination address and the system geocodes the address, queries published routes and identifies the viable options. The system computes the `PathScore` and results are ranked by the score in descending fashion along with the color-coded status. The system visualizes the paths on an interactive map with the status coloring. Green = Optimal, Yellow = Medium, Orange = sufficient and Red = Needs Maintenance.

The discovery section of the app allows the user to search for routes that were submitted by other users. The section also displays top 10 trending routes for the region for the month allowing user to explore new places. The user can click on a route to see the route specifics and also click on the user account who created the route to see their account landing page and also other routes created by them.

Graceful Degradation

In case the externally governed services fail, the system provides graceful degradation so that the functioning doesn't bricks. If the weather service is unavailable, the trip is saved to the account without the weather enrichment. The weather is marked "Unavailable" giving the user to manually refresh/update the trip with the weather information within 48 hours of trip completion. If map geocoding fails, users can enter the raw coordinates to set up a route.

2.3 User Characteristics

2.3.1 Registered User

A Registered User can be a regular cyclist who uses the BBP app for activity tracking and/or path contribution and discovery. They are able to enhance their profile with information regarding them and their equipment for the masses to see. The Registered User is expected to be somewhat technically proficient and is comfortable with mobile apps, map interfaces and basic navigation. The registered user is someone who can be experienced with local cycling routes and infrastructures, capable of assessing the path conditions and in describing obstacles. A Registered User is capable of :

- recording personal trips with GPS data
- access trip history with detail statistics, interactive map and weather enrichment
- manually input and publish path information
- query paths between an origin and destination

Registered Users can be on the platform for personal activity tracking, health monitoring, discovering of new routes and contributing to local cycling knowledge.

2.3.2 Unregistered User

An Unregistered User can be a traveler who is visiting a new place, occasional user seeking new paths to explore or a one-time cyclist who want to get from point A to point B. An unregistered user can be someone with very low technical proficiency but is able to understand basic maps and navigation apps. An Unregistered User is capable of :

- querying paths between an origin and a destination
- view resulting path's information and status
- view publicly available paths
- interacting with map for navigation

An unregistered user can't record trips or publish new path information. The main motivation here lies in route discovery and route planning.

It is to be noted that both the Registered and Unregistered Users need a modern smartphone with GPS and internet connectivity to utilize the functioning of the system.

2.4 Assumptions, Dependencies and Constraints

2.4.1 Regulatory Policies

The system requests the user to provide their email, username, profile picture, location services. It is important that the data is handled with absolute privacy and the practices are in compliance with the General Data Protection Regulations (GDPR)

2.4.2 Domain Assumptions

- **D1**: Mobile devices provide GPS-based location data with accuracy sufficient for trip recording and path reconstruction; accuracy may degrade in dense urban environments and indoors.
- **D2**: The system can distinguish biking activity using observed movement characteristics (speed patterns), recognizing possible misclassification in edge cases or artificial triggers.
- **D3**: An external weather service is available to provide meteorological data for a given location and time; temporary outages or timeouts are handled by the system.
- **D4**: Registered users contributing bike path information act in good faith and do not intentionally submit misleading data.
- **D5**: Mobile devices can provide location samples at a rate sufficient to reconstruct the user's traveled path for the purposes of BBP.
- **D6**: A geocoding service is available to resolve user-provided street names and locations into geographic coordinates; the service may be temporarily unavailable.
- **D7**: A routing/maps service is available to compute candidate routes between an origin and a destination and to support map-based visualization.
- **D8**: Users' mobile devices have access to motion sensors required for anomaly detection, subject to OS permissions and hardware capability.
- **D9**: When network connectivity may be intermittent, trip recording can continue temporarily, while features requiring external services may be delayed or unavailable.

2.4.3 Dependencies

- **DP1**: Externally managed weather service API like that of OpenWeather API or equivalent have an availability of 95 % along with <5 second response time. Even when the weather service is unavailable, the trip is processed.
- **DP2**: The system depends on the accuracy of smartphone based GPS receiver and accelerometers. a typical accuracy $\pm 5 - 15$ meters and sampling rate of $\geq$ 1Hz is expected and the system works within these constraints.
- **DP3**: The system relies heavily on a good network connectivity for user authentication, trip uploads, path queries, weather retrieval. System can cache data fro 48 hours in case the connectivity is lost. once the connection is established the sync is restored.
- **DP4**: System requires access to public map data, map tiles and geocoding services from providers like OpenStreetMap or Google Maps. Geocoding from both $address \rightarrow coordinates$ and $coordinates \rightarrow address$ are essential steps in path querying.

2.4.4 Constraints

- **C1**: The system stores and processes only the data necessary to provide its functionalities. Raw sensor data not required for these purposes are not retained.

- **C2**: The system does not provide live location sharing, real-time tracking or location broadcasting to other users or third parties.
- **C3**: The system is intended for modern smartphones that provide the required sensor and OS capabilities (iOS 14+ and Android 10+). Older devices or feature phones are not supported.

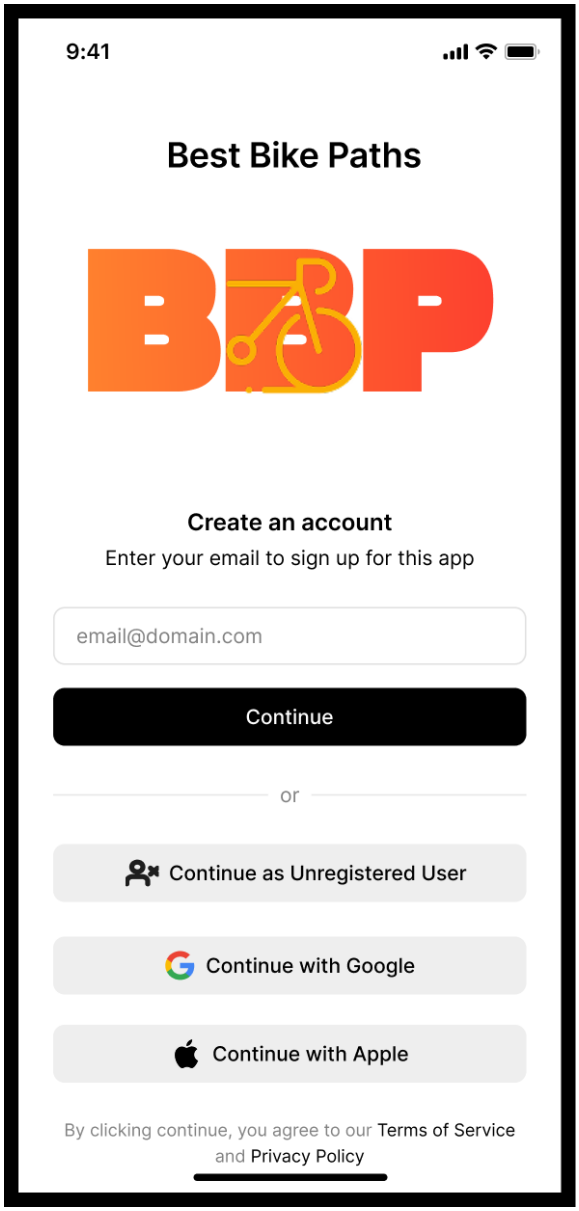
3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The below show the mock ups of the mobile application User Interface. This sections outlines the user interface of the most important screens of the application from the POV of the user. The user experience changes quiet a bit depending on the user - if he/she is a registered one or not.

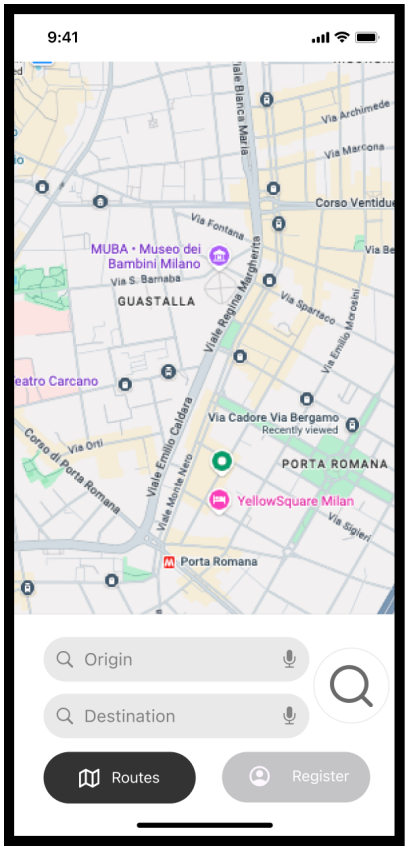
Sign-up and Login screen



Once the user opens the BBP app, they are greeted with this homepage, which offers the user to either login to an existing profile or sign-up and register as a new user or continue to the app as an unregistered user.

Home screens

Once the user logins to the app, the home screen differs based on the type of user - a registered or an unregistered one. The two mock-ups emulating the two scenarios are shown below:

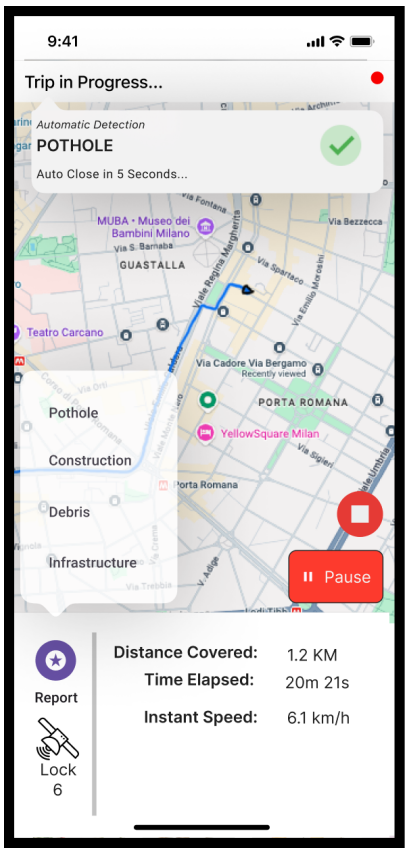


Registered User
The registered users are greeted with the home screen on the left. Based on the actions that a registered user can perform, the main home screen offers the user with 3 tab at the bottom -

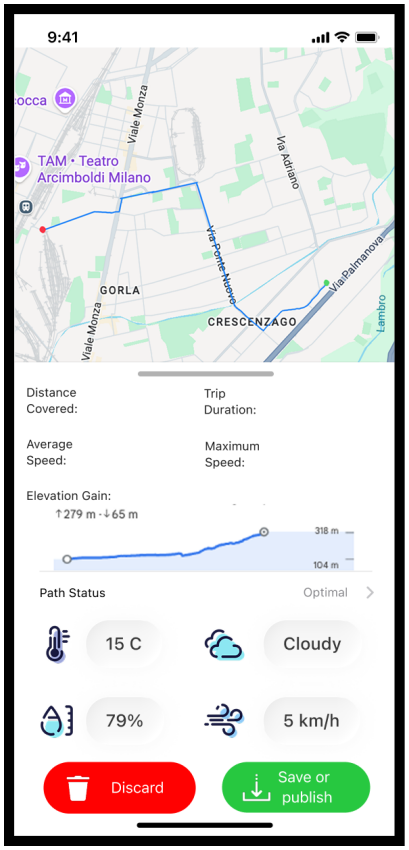
- Trip tab to initiate a new trip recording.
- Routes tab to do a manual path query.
- Profile tab to look at past trips and account overviews and settings

Unregistered User
An unregistered user's home screen on the other hand has access to just the Manual Path Query tab and an additional tab where the user can register an account if they wish to do so.

Trip Information



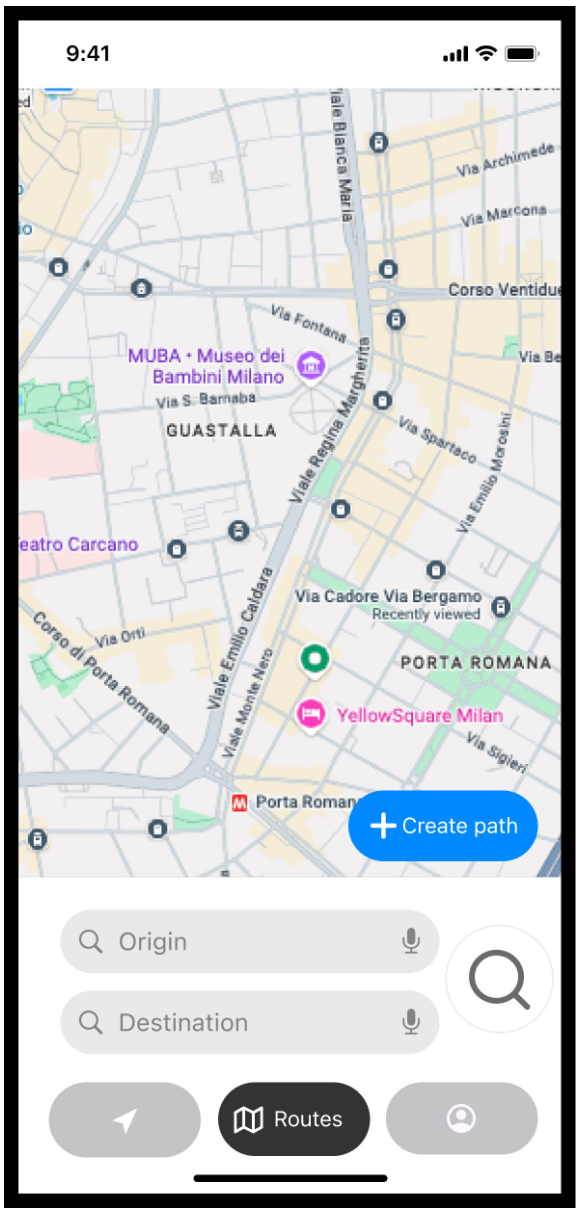
This screen depicts a trip in progress. The screen show the user, the distance covered, the time elapsed and the instantaneous speed. The application also gives the user to report any obstacles present along the path. In addition to this, the system makes use of live notifications for automatic obstacle recognition. The notification shows up on the screen and stays on for 5 seconds for the user to accept and add it to the path. This is done so that false notifications can be ignored.



Trip summary screen show the user the summary with the ride statistics computations along with the weather enrichment if present. The user has the following 3 options:

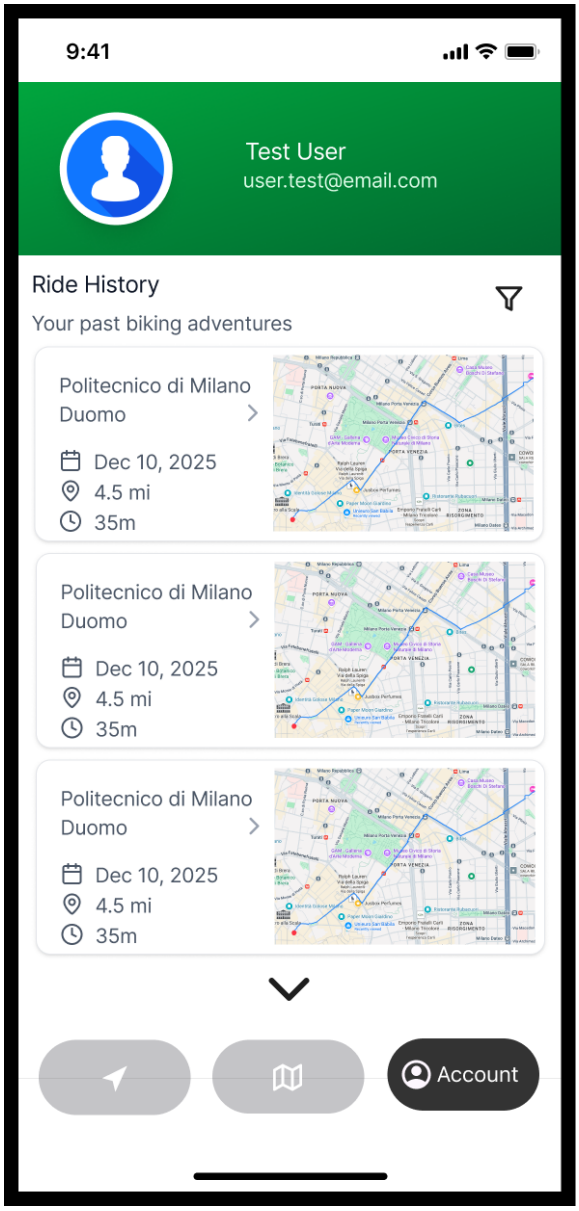
- Discad the trip.
- Save the trip to their profile but not publish the Trip.
- Publish the Trip which also saves it to their profile.

Manual Path Query



This Manual Path Query screen is essentially the same for both the users, offering them the option of manually searching the required path between origin-destination pair. The only additional UI element that the registered user has is the create path button, which allows the registered user to create paths manually.

Account Overview



The Account overview tab allows the registered users to see their past trip and its summaries by click on a particular trip. The account tab also houses the user banner which shows their profile picture along with the user information.

3.1.2 Hardware Interfaces

The system relies a lot on personal devices that can be carried with the user that have the ability to record the change in GPS data, motion along with internet connectivity.

GPS Module

The GPS data for the system is mainly accessed via smartphone operating systems' GPS API mainly

- CLLocationManager on iOS
- LocationManager on Android

An accuracy of ± 10 meters and sampling rate of $\geq$ 1Hz during active recording is expected. Metadata from the GPS such as latitude, longitude, altitude, accuracy and speed are used by the system to provide the user with ride statistics

Device Motion Sensor

Standard smartphone motion sensor APIs are used to get information from the accelerometer to better detect the biking motion.

Network Interfaces

System uses internet connectivity through WiFi or cellular standards like 5G, 4G or LTE for processing of data. HTTPS is used for encrypted communications. the data transmission happens with the help of JSON payloads with gzip compression for large files.

3.1.3 Software Interfaces

Weather Service API

The system uses the Weather Service API to get the weather information for the location. The weather service has a timeout set at 5 seconds after which the system retries to fetch the weather a max of 3 times with exponential back-offs in the background (1s, 2s, 4s). Failure to get the weather information before the timeout leads to a graceful fallback where the system completes the trip and saves it to the profile without weather information. A sample API request is shown below:

```
Request:

GET https://api.openweathermap.org/data/2.5/weather
    ?lat={latitude}
    &lon={longitude}
    &appid={API_KEY}
    &units=metric
```

Geocoding Service Interface

Similarly the system uses OpenStreetMap APIs or other equivalent for geocoding from both *address* $\rightarrow$ *coordinates* and *coordinates* $\rightarrow$ *address* during path querying. The geocoding service has a timeout of 10 seconds after which the system gives the user an error message saying the operation is taking longer than expected. In this case, the user has the option of entering raw coordinates for the query to go through.

```
Request (Forward Geocoding):

GET https://nominatim.openstreetmap.org/search
    ?q={address}
    &format=json
    &limit=1



Request (Reverse Geocoding):

GET https://nominatim.openstreetmap.org/reverse
    ?lat={latitude}
    &lon={longitude}
    &format=json
```

Mobile Device Sensor APIs

The GPS data and the motion sensor data for the system are mainly accessed via smartphone operating systems' GPS and Device Sensor APIs

iOS:

- CLLocationManager for GPS location and Heading
- CMMotionManager for accessing the data from accelerometer/gyroscope for activity detection.

Android:

- FusedLocationProviderClient for GPS locations and Heading

- SensorManager for accessing data from the motion sensors

Both iOS and Andoid apps require permissions from the users to access these data.

3.1.4 Communication Interfaces

BBP application relies on its internet connectivity for a happy sail. The interactions on the app must be secure and encrypted using HTTPS/TLS Protocol. The platform also makes use of REST API for communicating with the server for the trip creations, path queries and route updates.

3.2 Functional Requirements

3.2.1 Functional Requirements Specification

User Access and Profile Management

- **R1**: The system shall allow a `User` to sign up and create a new account.
- **R2**: The system shall allow a `User` to continue as an `UnregisteredUser`.
- **R3**: The system shall allow a `RegisteredUser` to log in.
- **R4**: The system shall allow a `RegisteredUser` to update and modify profile information.
- **R5**: The system shall restrict an `UnregisteredUser` to path querying and visualization functionalities only.

Trip Recording and Sensing

- **R6**: The system shall request `User` permission to access GPS and motion sensors on the device.
- **R7**: The system shall detect biking activity based on a established minimum sustained velocity.
- **R8**: The system shall allow a `RegisteredUser` to initiate trip recording.
- **R9**: The system shall display real-time trip data during recording.
- **R10**: The system shall allow a `RegisteredUser` to pause and resume trip recording.
- **R11**: The system shall allow a `RegisteredUser` to discard a trip before final upload.
- **R12**: The system shall compute trip statistics.
- **R13**: The system shall store a completed trip and its data if the `RegisteredUser` confirms saving it.
- **R14**: The system shall allow a `RegisteredUser` to view past trips and access trip history.
- **R15**: The system shall prevent GPS data loss during network unavailability.

Weather Enrichment

- **R16**: The system shall retrieve meteorological data for a completed trip from an external service.
- **R17**: The system shall attach weather information with the trip if available.
- **R18**: The system shall allow the `User` to manually refresh weather enrichment.
- **R19**: The system shall gracefully degrade if the weather service is unavailable.

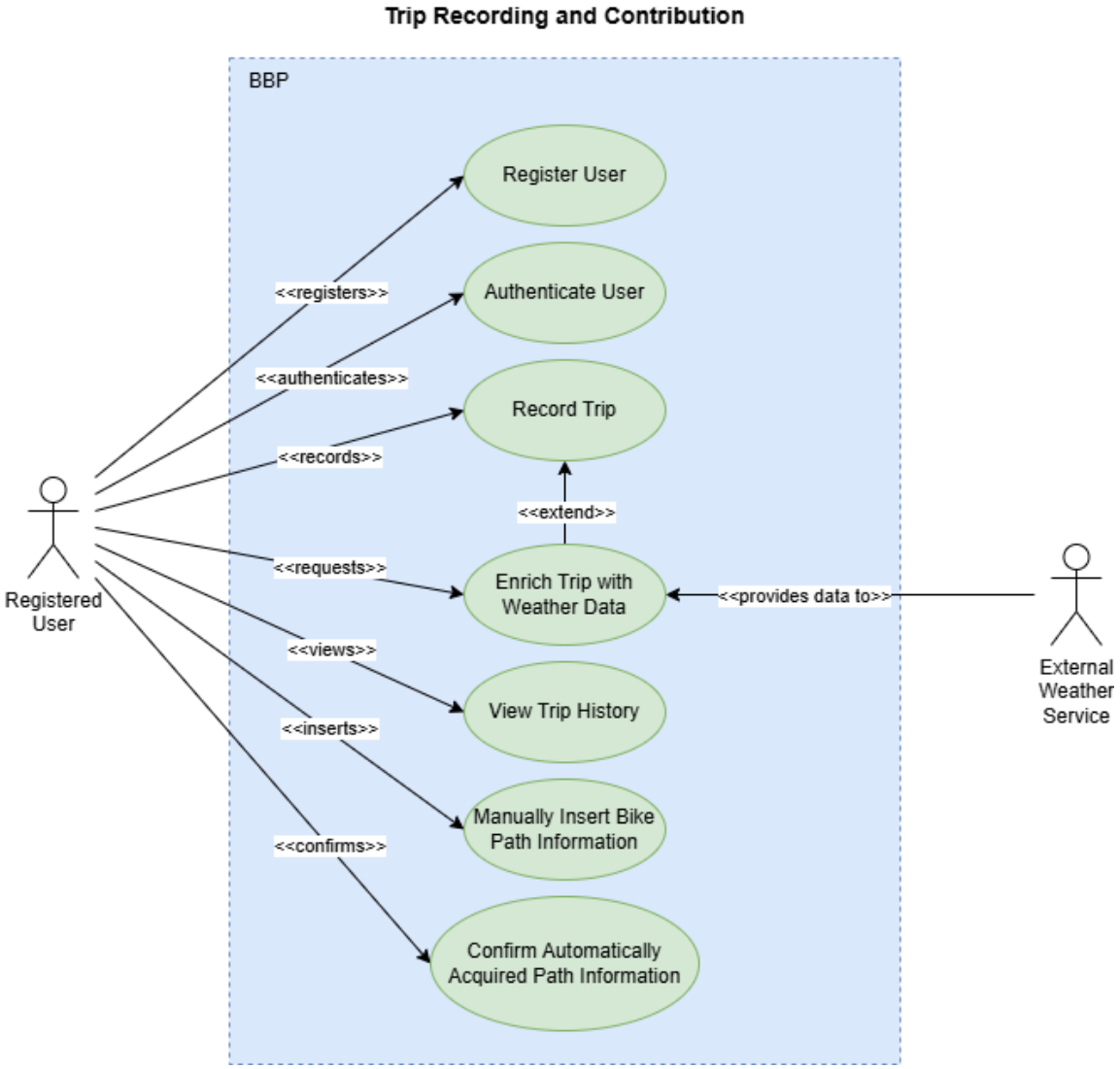
Manual and Automated Path Contribution

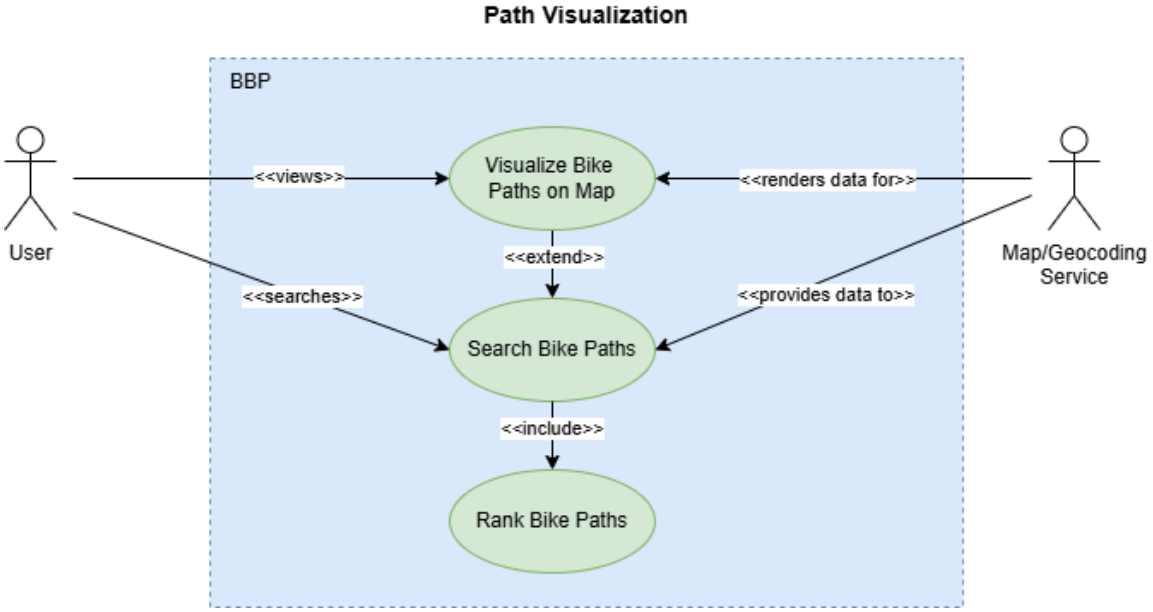
- **R20**: The system shall allow a `RegisteredUser` to manually insert bike path information.
- **R21**: The system shall allow a `RegisteredUser` to review and confirm automatically acquired path information.
- **R22**: The system shall publish path information when the `RegisteredUser` makes it publishable.
- **R23**: The system shall allow a `RegisteredUser` to control the publication of contributed path information.
- **R24**: The system shall gracefully degrade if the geocoding service is unavailable.

Path Discovery and Visualization

- **R25**: The system shall compute a `PathScore` for each path based on `PathStatus` and obstacles present on path.
- **R26**: The system shall list routes ordered by descending `PathScore`.
- **R27**: The system shall allow any `User` to visualize routes on an interactive map.
- **R28**: The system shall allow any `User` to query paths between an origin and a destination.

3.2.2 Use Case Diagrams





3.2.3 Use Cases

In the use case diagrams and specifications, the actor `User` is used as a generic role representing any individual interacting with the BBP system. Depending on the context, a `User` may be either a `RegisteredUser` or an `UnregisteredUser`. Use cases that require authentication explicitly mention the `RegisteredUser` actor, while use cases that can be performed without authentication use the generic `User` actor. Registration-related use cases describe the transition of a `User` into a `RegisteredUser` upon successful authentication.

[UC1] Register User

Name	**Record Trip**
Actors	`User`
Entry Condition	When this use case starts, the following condition is true: The `User` accesses the registration functionality of BBP.
Event Flow	(a) The `User` selects the option to register and create a new account. (b) The system prompts the `User` to provide the required registration information. (c) The `User` submits the requested information. (d) The system validates the `User` information. (e) If the information is valid, the system creates a new `User` account. (f) The system confirms successful registration and grants the `User`, a `RegisteredUser` status.
Exit Condition	This use case terminates when the `User` has either been successfully registered and when the registration process is aborted.
Exceptions	(d) If the provided registration information is invalid or incomplete, the system notifies the `User` and requests correct input. (e) If an internal error occurs during account creation, the system informs the `User` and to try again later.

[UC2] Access System as Unregistered User

Name	**Access System as Unregistered User**
Actors	`UnregisteredUser`
Entry Condition	When this use case starts, the following condition is true: The `UnregisteredUser` accesses BBP and chooses to continue as a guest.

Event Flow	(a) The `UnregisteredUser` selects the option to continue as a guest (unregistered user). (b) The system grants the `UnregisteredUser` access to BBP with restricted privileges. (c) The `UnregisteredUser` accesses path search and visualization functionalities. (d) The system blocks the `UnregisteredUser` from accessing functionalities reserved for a `RegisteredUser`.
Exit Condition	This use case terminates when the `UnregisteredUser` either exists BBP or chooses to authenticate or register.
Exceptions	(d) If the `UnregisteredUser` attempts to access some restricted functionality, the system informs the `UnregisteredUser` that authentication is required to access.

[UC3] Authenticate User

Name	**Authenticate User**
Actors	`RegisteredUser`
Entry Condition	When this use case starts, the following condition is true: The `RegisteredUser` accesses the login functionality of BBP.
Event Flow	(a) The `RegisteredUser` provides authentication credentials. (b) The system validates and verifies the provided credentials. (c) If the credentials are valid, the system authenticates the `RegisteredUser`. (d) The system grants access to the `RegisteredUser` to functionalities reserved for authenticated users.
Exit Condition	This use case terminates when the `RegisteredUser` is successfully authenticated or when the authentication process fails.
Exceptions	(b) If the provided credentials are invalid, the system notifies the `RegisteredUser` and requests retrying with valid credentials. (c) If an internal error occurs during authentication, the system informs the `RegisteredUser` and suggests to try again later.

[UC4] Record Trip

Name	**Record Trip**
Actors	`RegisteredUser`
Entry Condition	When this use case starts, the following condition is true: The `RegisteredUser` is authenticated in BBP and has granted the system permission to access the device's GPS and sensors.
Event Flow	(a) The `RegisteredUser` begins recording a trip by pressing the "Start" button. (b) The system verifies that required permissions for GPS and sensors are granted. (c) The system starts collecting movement data from the device. (d) During the trip, the system displays real-time information to the `RegisteredUser`, for example, current speed, elapsed time and distance covered. (e) The `RegisteredUser` may decide at any moment during the trip to pause with the corresponding button. (f) If paused, the `RegisteredUser` may resume the trip recording, which triggers data collection and real-time updates again. (g) The `RegisteredUser` can stop the trip at any moment with the corresponding button. (h) If stopped, the system calculates final trip statistics, like total distance, duration, elevation gain and average speed. (i) The system prompts the `RegisteredUser` to save or discard the recorded trip. (j) If the `RegisteredUser` confirms trip saving, the system stores the data from the recorded trip. (k) Otherwise, data about the recorded trip is discarded by the system.
Exit Condition	This use case terminates when the trip has either been successfully saved and stored by the system or discarded by the `RegisteredUser`.
Exceptions	(b) If the user doesn't grant necessary permissions, the system informs the user that trip recording cannot begin and ends the use case. (c) If there is a temporary loss of connectivity during recording, the system will start collecting data locally and synchronizing whenever connection is restored. (c) If any sensor becomes unavailable during trip recording, the system will automatically pause the recording and notify the `RegisteredUser`.

[UC5] View Trip History

Name	**View Trip History**
Actors	`RegisteredUser`
Entry Condition	When this use case starts, the following condition is true: A `RegisteredUser` is authenticated in BBP and accesses the trip history within the profile section.
Event Flow	(a) The `RegisteredUser` selects the option to view past trips. (b) The system retrieves the list of trips previously recorded and saved by the `RegisteredUser`. (c) The system displays the trips history and summary information for each (e.g., date, distance, duration), presented in reverse chronological order. (d) The `RegisteredUser` selects a trip from the displayed list. (e) The system shows a detailed view for the selected trip with more information, like trip statistics, related obstacles, path status.
Exit Condition	This use case terminates when the `RegisteredUser` exits the trip history view or a specific trip detail view.
Exceptions	(b) If the `RegisteredUser` has no existing saved trips, the system informs the user that no history is available. (c) If an error occurs during the trips data retrieval, the system notifies the `RegisteredUser` and suggests to try again later.

[UC6] Enrich Trip with Weather Data

Name	**Enrich Trip with Weather Data**
Actors	`RegisteredUser`, `WeatherService`
Entry Condition	When this use case starts, the following condition is true: A `RegisteredUser` has completed and saved a trip, and the system tries to link weather information to it upon user request.
Event Flow	(a) When a recorded trip is saved, a `RegisteredUser` selects the option for "Weather Enrichment". (b) The system identifies geographic and time data associated to the trip. (c) The system requests the `WeatherService` to retrieve weather data corresponding to the trip. (d) The `WeatherService` returns relevant weather information, such as temperature, wind condition, weather status, precipitation. (e) The system saves the weather information and associates it with the recorded trip. (f) The system displays the enriched weather data to the `RegisteredUser` as an additional section in the trip summary view. (g) At any point, the `RegisteredUser` can request a manual refresh of this data, which triggers a repeat on the data retrieval/association from the `WeatherService`.
Exit Condition	This use case terminates when the weather information has been successfully saves and associated with the trip or when an error occurred during weather information retrieval.
Exceptions	(c) If the `WeatherService` is unavailable, the system notifies the `RegisteredUser` that the weather enrichment functionality is temporarily unavailable. (e) If the weather data retrieved is incomplete or inconsistent, the system displays the available correct information to the `RegisteredUser` and notifies them of incomplete enrichment. (g) If the manual refresh request fails at any point, the system notifies the `RegisteredUser` and keeps the previous data without modification, if it exists.

[UC7] Manually Insert Bike Path Information

Name	**Manually Insert Bike Path Information**
Actors	`RegisteredUser`, `GeocodingService`
Entry Condition	When this use case starts, the following condition is true: A `RegisteredUser` is authenticated in BBP and accesses functionality for manual insertion of path information.
Event Flow	(a) The `RegisteredUser` selects the option to manually insert path information. (b) The system asks the `RegisteredUser` to fill the information for the route (directions, street names, references). (c) The `RegisteredUser` inputs the required information. (d) The system validates the route information with the `GeocodingService`. (e) The `RegisteredUser` provides information regarding the path itself, including a qualitative descriptor of the path status, and any relevant obstacles and their positions in the route. (f) The `RegisteredUser` selects whether the new path should be made publicly available or kept private. (g) The system stores the inserted path information and updates the publication according to the `RegisteredUser` selection.
Exit Condition	This use case terminates when the system has successfully stored all the inserted path information after the `RegisteredUser` confirmation/discard.
Exceptions	(d) If the provided route information is invalid or can't be resolved by the `GeocodingService`, the system notifies the `RegisteredUser` and asks for a correct input. (d) If the `GeocodingService` is unavailable, the system notifies the `User` that the manual path insertion functionality is temporarily unavailable. (e) If the `RegisteredUser` exits the insertion process before completion, the system discards the data and no path information is stored.

[UC8] Confirm Automatically Acquired Path Information

Name	**Confirm Automatically Acquired Path Information**
Actors	`RegisteredUser`
Entry Condition	When this use case starts, the following condition is true: A `RegisteredUser` has completed a trip and the system has automatically collected potential path information which is yet unpublished.
Event Flow	(a) The system notifies the `RegisteredUser` that there is information that has been automatically collected during the trip, which is available for review. (b) The `RegisteredUser` accesses the interface to review the collected data. (c) The system displays the path along with every detected potential obstacle or distinctive path attributes to qualitatively review. (d) The `RegisteredUser` reviews the displayed information. (e) The `RegisteredUser` can confirm the information without changes. (f) The `RegisteredUser` also has the option to edit the path information, like removing false obstacles. (g) Once the `RegisteredUser` has the completed information, with or without changes, they can decide if the information can be made publicly available or kept private. (h) The system stores the reviewed path information and updates the publication according to the `RegisteredUser` selection.
Exit Condition	This use case terminates when the system has successfully stored all the acquired path information after the `RegisteredUser` review and confirmation/discard.
Exceptions	(a) If there is no automatically collected information for a completed trip, the system simply informs the `RegisteredUser` and the use case is terminated. (c) If there is incomplete or inconsistent information regarding an automatically acquired piece of data during the trip, the system highlights the issue and notifies the `RegisteredUser` to correct or discard the data. (d) If the `RegisteredUser` exits the review process without any confirmation, the system keeps the data as non-reviewed and does not publish it.

[UC9] Search Bike Paths

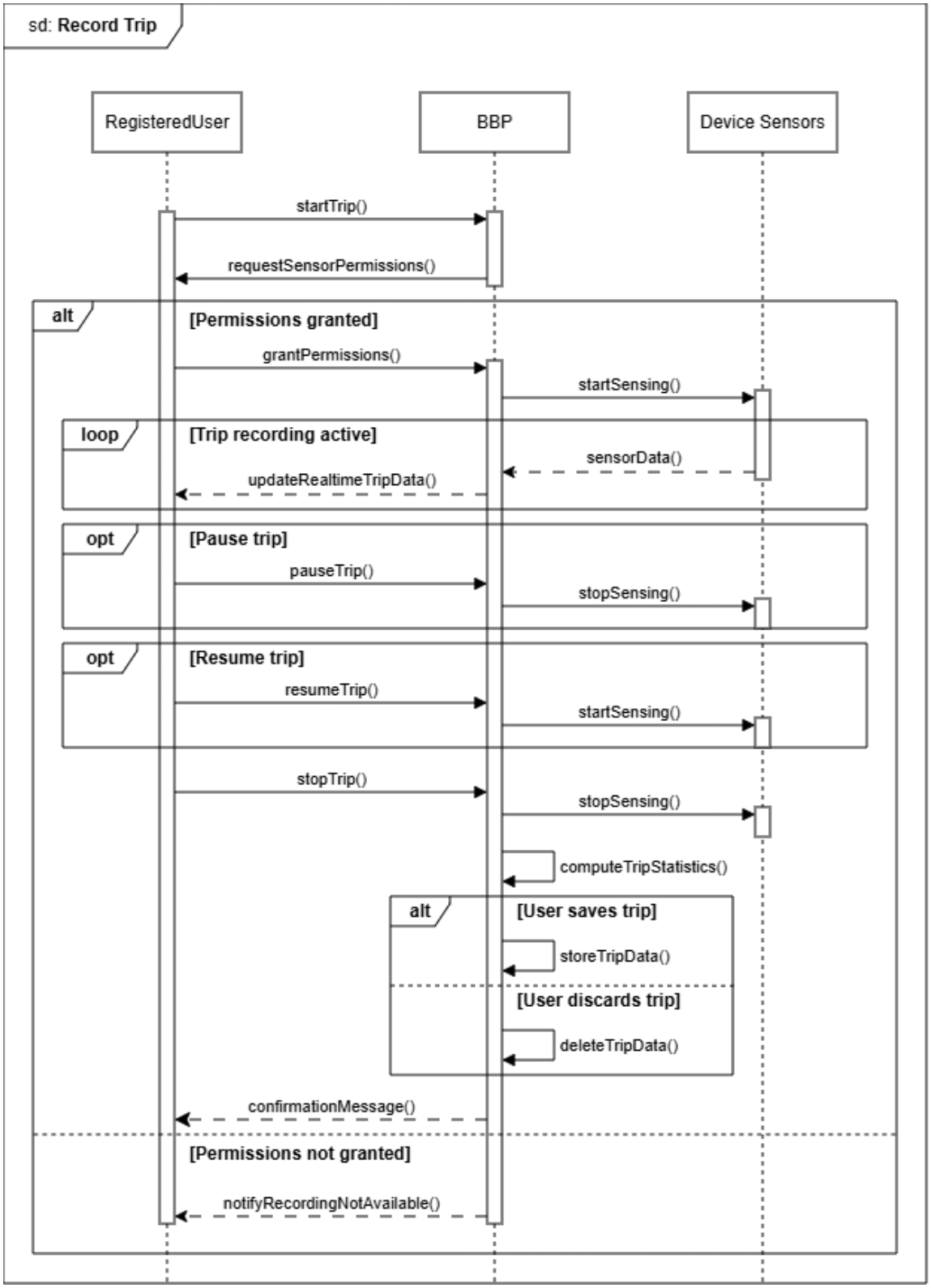
Name	**Search Bike Paths**
Actors	`User` (`RegisteredUser` or `UnregisteredUser`), `GeocodingService`
Entry Condition	When this use case starts, the following condition is true: A `User` accesses BBP path search functionality and specifies an origin and a destination.
Event Flow	(a) A `User` inserts an origin and a destination in the corresponding fields. (b) The system validates the inputted coordinates or directions. (c) The system calls the `GeocodingService` to retrieve the locations and compute the route between them. (d) The system shows one or more possible paths provided by the external service. (e) For each path, the system computes a `PathScore` based on the path conditions and recorded characteristics, if any. (f) The system presents the possible paths in descending order by the calculated `PathScore`.
Exit Condition	This use case terminates when the system has displayed an ordered list of paths between a specified origin and destination, or has notified the `User` if there are no paths available.
Exceptions	(b) If the provided origin or destination are invalid or can't be resolved, the system notifies the `User` and asks for a correct input. (c) If the `GeocodingService` is unavailable, the system notifies the `User` that the search functionality is temporarily unavailable. (d) If there are no suitable paths between the locations provided, the system notifies the `User` and asks to insert other valid coordinates.

[UC10] Visualize Bike Paths on Map

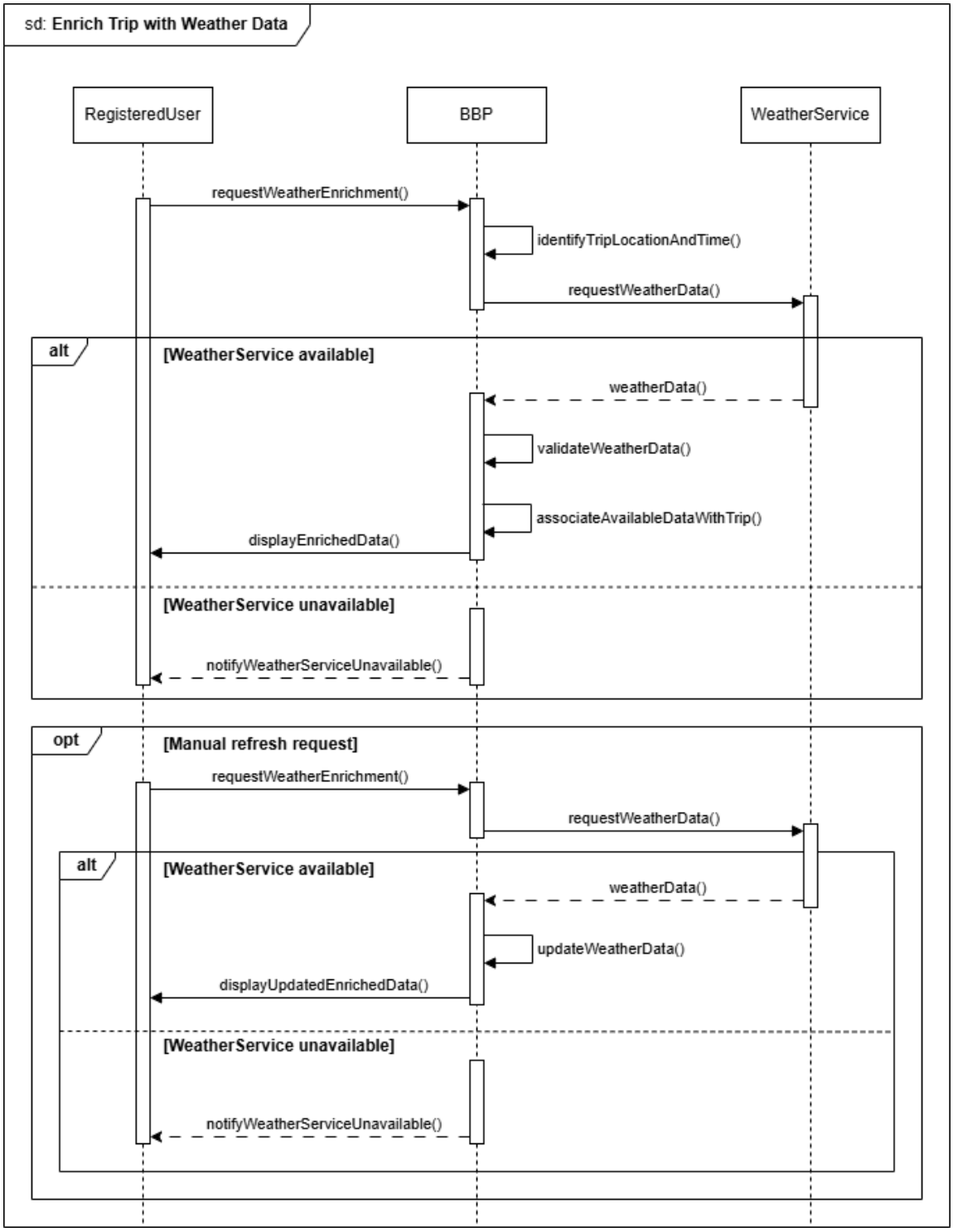
Name	**Visualize Bike Paths on Map**
Actors	`User` (`RegisteredUser` or `UnregisteredUser`), `GeocodingService`
Entry Condition	When this use case starts, the following condition is true: A `User` has performed a bike path search with at least one possible path available.
Event Flow	(a) A `User` selects one of the bike paths from the search results. (b) The system retrieves the route and map information for the selected path with the help of the `GeocodingService`. (c) The system renders the path in the interactive map on the screen. (d) The `User` can interact with the map to inspect the path details. (e) The `User` can trigger the "begin trip" action for the selected path.
Exit Condition	This use case terminates when the selected bike path is successfully displayed in the interactive map or if the `User` exits the map view.
Exceptions	(b) If the `GeocodingService` is unavailable, the system notifies the `User` that the map visualization functionality is temporarily unavailable. (c) If there is an error during path rendering on the map, the `User` is notified by the system to try again or choose another path for visualization.

3.2.4 Sequence Diagrams

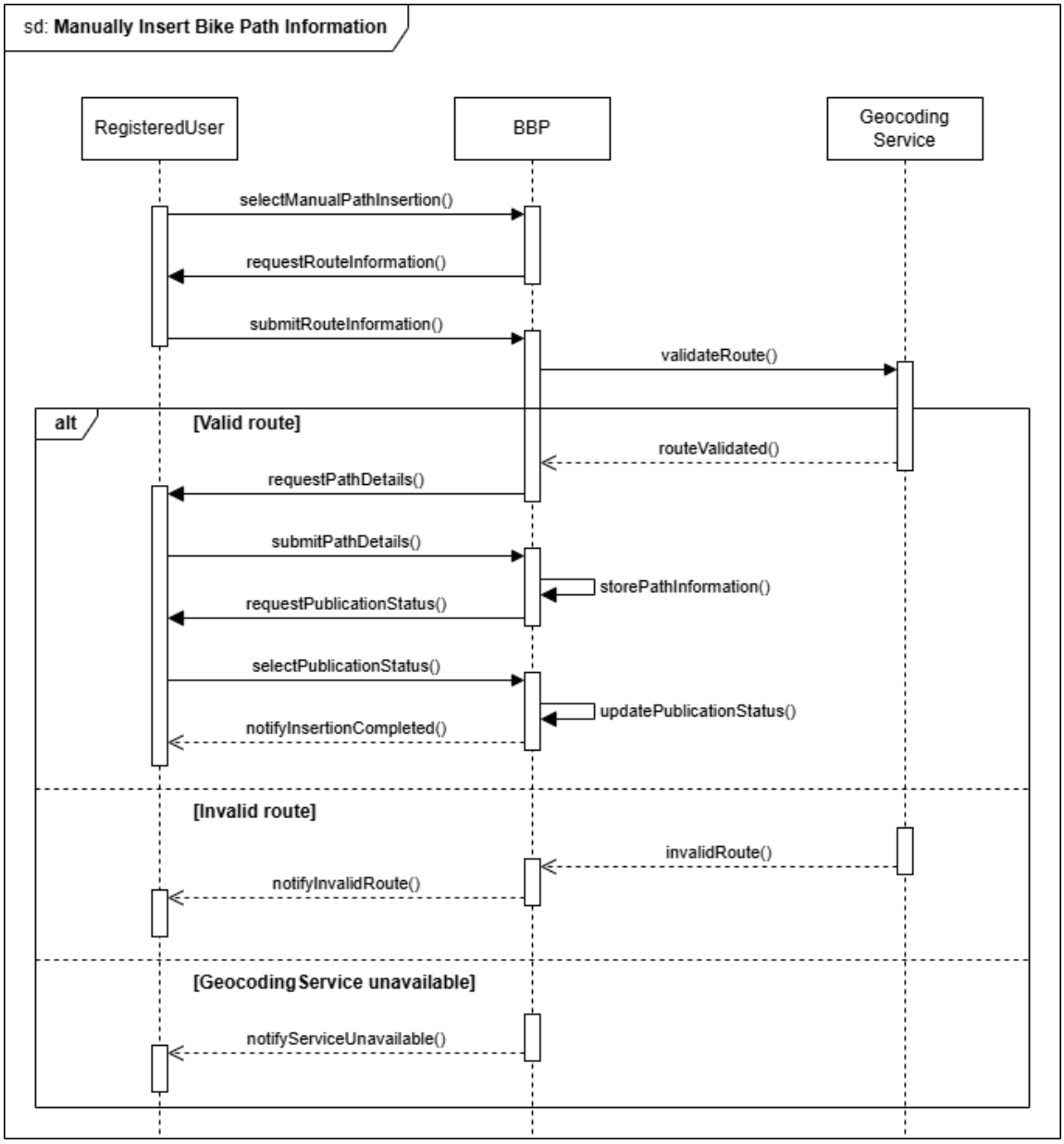
[UC4] Record Trip



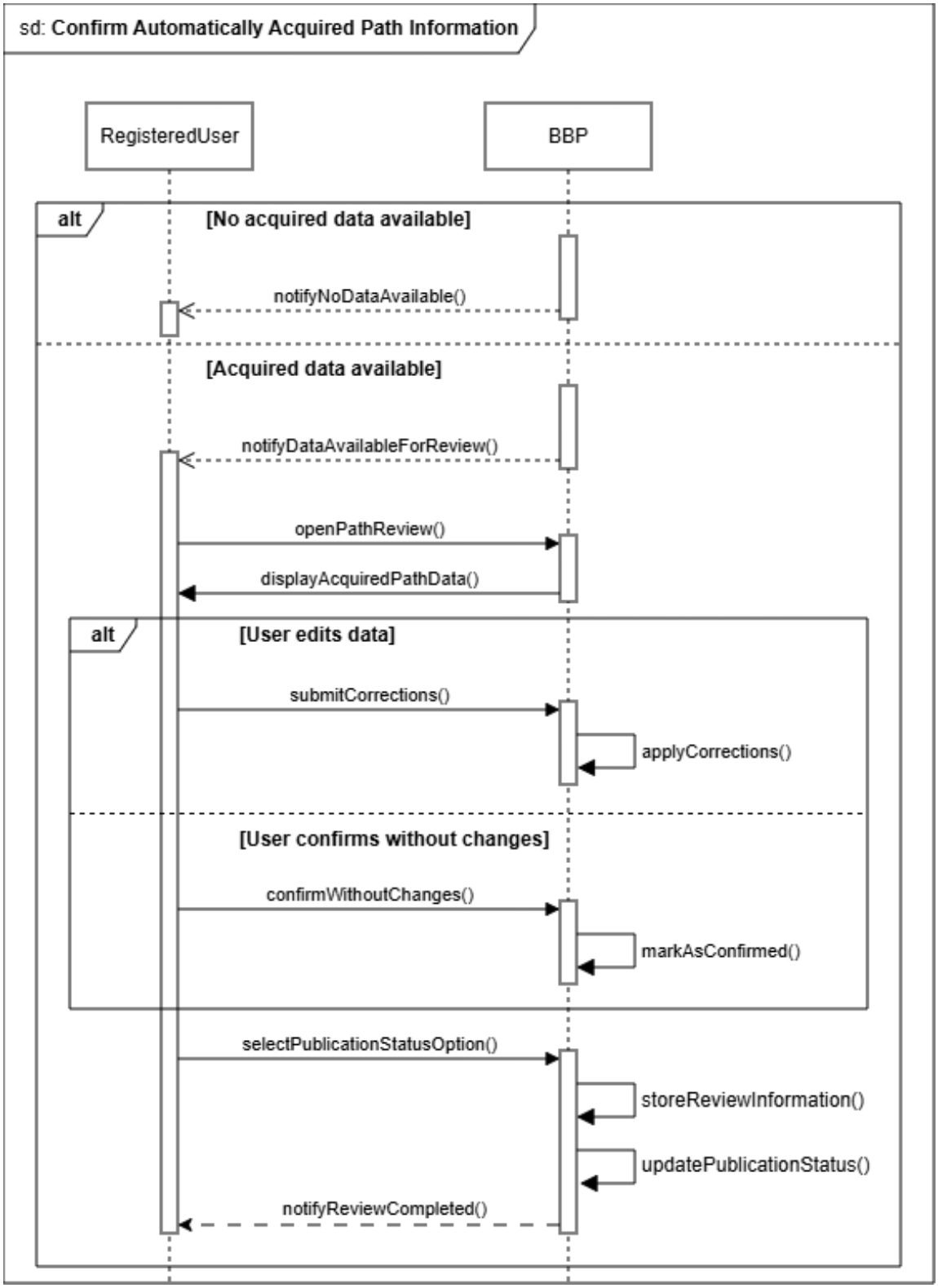
[UC6] Enrich Trip with Weather Data



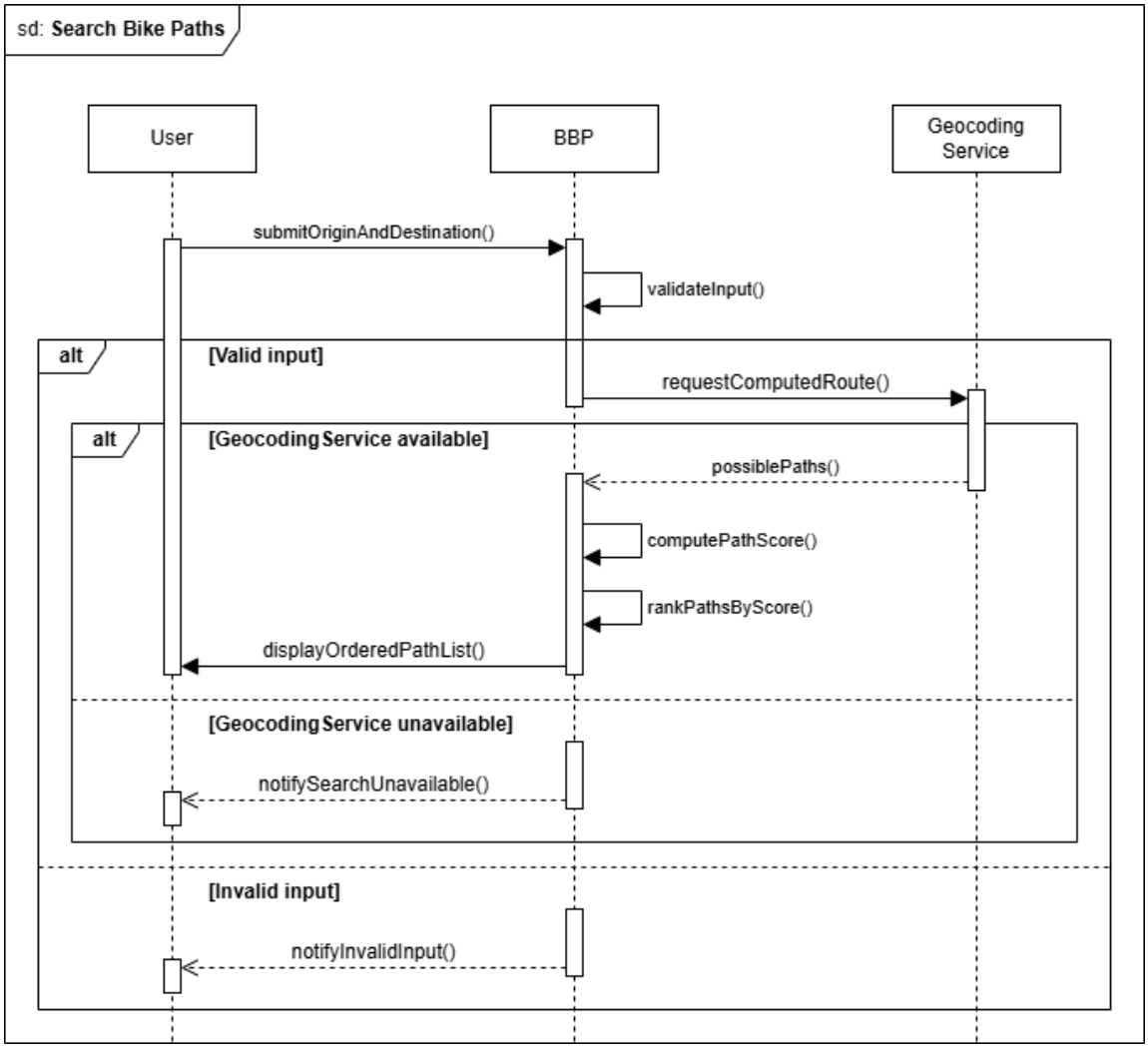
[UC7] Manually Insert Bike Path Information



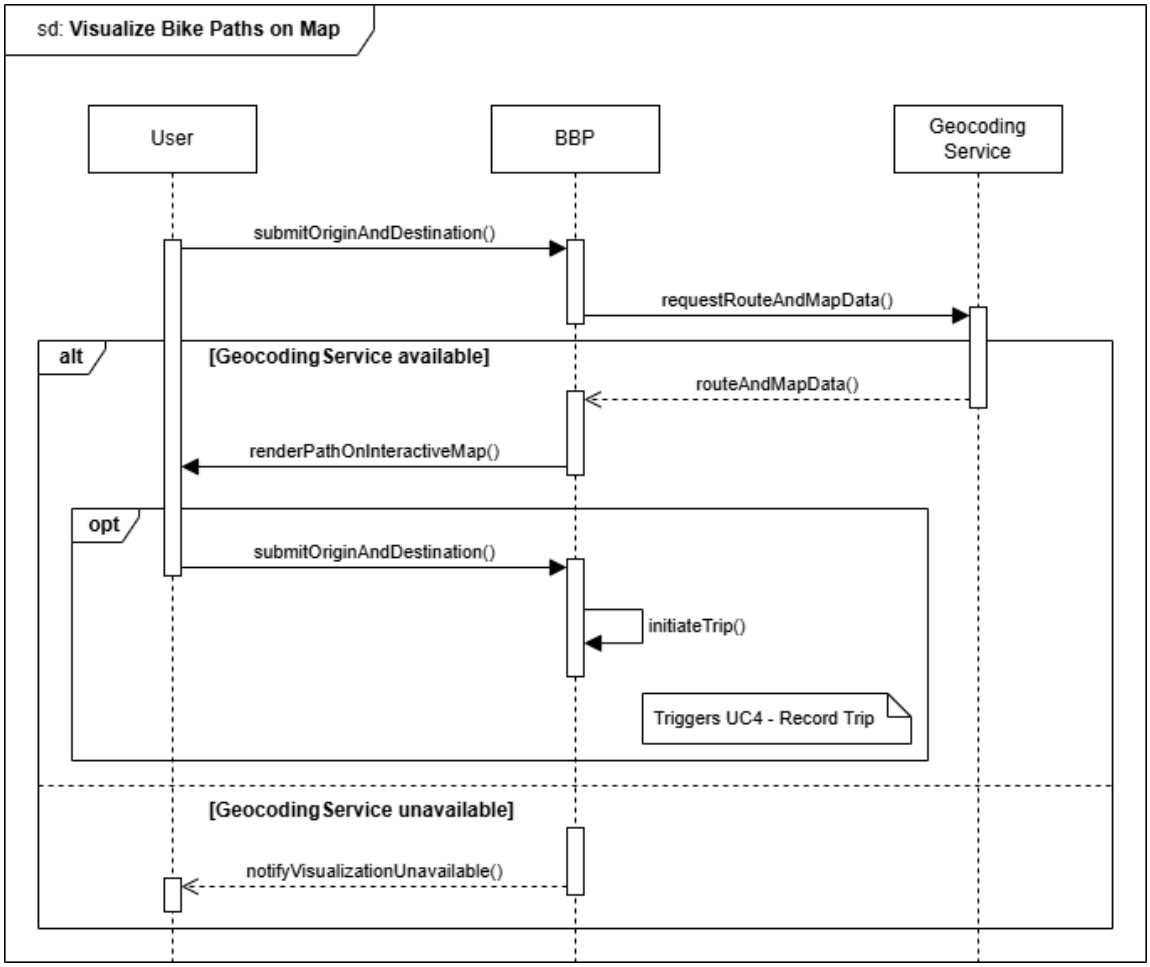
[UC8] Confirm Automatically Acquired Path Information



[UC9] Search Bike Paths



[UC10] Visualize Bike Paths on Map



3.2.5 Requirement mapping

G1: Allow registered users to log personal rides and view summary stats (distance, duration, average speed, and key performance metrics).	
R1: The system shall allow a `User` to sign up and create a new account. **R3**: The system shall allow a `RegisteredUser` to log in. **R4**: The system shall allow a `RegisteredUser` to update and modify profile information. **R6**: The system shall request `User` permission to access GPS and motion sensors on the device. **R7**: The system shall detect biking activity based on a established minimum sustained velocity. **R8**: The system shall allow a `RegisteredUser` to initiate trip recording. **R9**: The system shall display real-time trip data during recording. **R10**: The system shall allow a `RegisteredUser` to pause and resume trip recording. **R11**: The system shall allow a `RegisteredUser` to discard a trip before final upload. **R12**: The system shall compute trip statistics. **R13**: The system shall store a completed trip and its data if the `RegisteredUser` confirms saving it. **R14**: The system shall allow a `RegisteredUser` to view past trips and access trip history. **R15**: The system shall prevent GPS data loss during network unavailability. **R16**: The system shall retrieve meteorological data for a completed trip from an external service. **R17**: The system shall attach weather information with the trip if available. **R18**: The system shall allow the `User` to manually refresh weather enrichment. **R19**: The system shall gracefully degrade if the weather service is unavailable.	**D1**: Mobile devices provide GPS-based location data with accuracy sufficient for trip recording and path reconstruction; accuracy may degrade in dense urban environments and indoors. **D2**: The system can distinguish biking activity using observed movement characteristics (speed patterns), recognizing possible misclassification in edge cases or artificial triggers. **D3**: An external weather service is available to provide meteorological data for a given location and time; temporary outages or timeouts are handled by the system. **D5**: Mobile devices can provide location samples at a rate sufficient to reconstruct the user's traveled path for the purposes of BBP. **D8**: Users' mobile devices have access to motion sensors required for anomaly detection, subject to OS permissions and hardware capability. **D9**: When network connectivity may be intermittent, trip recording can continue temporarily, while features requiring external services may be delayed or unavailable.

G2: Let registered users manually create and maintain bike path data by defining route segments and tagging conditions and obstacles.	
R1: The system shall allow a `User` to sign up and create a new account. **R3**: The system shall allow a `RegisteredUser` to log in. **R6**: The system shall request `User` permission to access GPS and motion sensors on the device. **R20**: The system shall allow a `RegisteredUser` to manually insert bike path information. **R22**: The system shall publish path information when the `RegisteredUser` makes it publishable. **R23**: The system shall allow a `RegisteredUser` to control the publication of contributed path information. **R24**: The system shall gracefully degrade if the geocoding service is unavailable.	**D4**: Registered users contributing bike path information act in good faith and do not intentionally submit misleading data. **D6**: A geocoding service is available to resolve user-provided street names and locations into geographic coordinates; the service may be temporarily unavailable. **D7**: A routing/maps service is available to compute candidate routes between an origin and a destination and to support map-based visualization.

G3: Let registered users automatically record bike path data during rides via GPS-based path reconstruction and sensor-based anomaly detection.	
R1: The system shall allow a `User` to sign up and create a new account. **R3**: The system shall allow a `RegisteredUser` to log in. **R6**: The system shall request `User` permission to access GPS and motion sensors on the device. **R7**: The system shall detect biking activity based on a established minimum sustained velocity. **R8**: The system shall allow a `RegisteredUser` to initiate trip recording. **R12**: The system shall compute trip statistics. **R15**: The system shall prevent GPS data loss during network unavailability. **R21**: The system shall allow a `RegisteredUser` to review and confirm automatically acquired path information. **R23**: The system shall allow a `RegisteredUser` to control the publication of contributed path information.	**D1**: Mobile devices provide GPS-based location data with accuracy sufficient for trip recording and path reconstruction; accuracy may degrade in dense urban environments and indoors. **D2**: The system can distinguish biking activity using observed movement characteristics (speed patterns), recognizing possible misclassification in edge cases or artificial triggers. **D5**: Mobile devices can provide location samples at a rate sufficient to reconstruct the user's traveled path for the purposes of BBP. **D8**: Users' mobile devices have access to motion sensors required for anomaly detection, subject to OS permissions and hardware capability. **D9**: When network connectivity may be intermittent, trip recording can continue temporarily, while features requiring external services may be delayed or unavailable.

G4: Require user review before publishing automatically collected path data, and let contributors choose whether their submissions are shared with the community.	
R21: The system shall allow a `RegisteredUser` to review and confirm automatically acquired path information. **R22**: The system shall publish path information when the `RegisteredUser` makes it publishable. **R23**: The system shall allow a `RegisteredUser` to control the publication of contributed path information.	**D4**: Registered users contributing bike path information act in good faith and do not intentionally submit misleading data.

G5: Let any user view and compare bike paths between an origin and destination on a map, ranking options by a score combining route effectiveness and path conditions.	
R2: The system shall allow a `User` to continue as an `UnregisteredUser`. **R5**: The system shall restrict an `UnregisteredUser` to path querying and visualization functionalities only. **R24**: The system shall gracefully degrade if the geocoding service is unavailable. **R25**: The system shall compute a `PathScore` for each path based on `PathStatus` and obstacles present on path. **R26**: The system shall list routes ordered by descending `PathScore`. **R27**: The system shall allow any `User` to visualize routes on an interactive map. **R28**: The system shall allow any `User` to query paths between an origin and a destination.	**D6**: A geocoding service is available to resolve user-provided street names and locations into geographic coordinates; the service may be temporarily unavailable. **D7**: A routing/maps service is available to compute candidate routes between an origin and a destination and to support map-based visualization.

3.3 Performance Requirements

The main performance aspects of the system is its ability to get the synergy right with the different API calls, GPS and sensor data. Unavailability of non essential data should be dealt gracefully. Given that the system can have 1000's if not more concurrent at its peak, it is important that performance requirements are robust and should adhere to the said guideline with respect to timeouts.

Trip Recording Response Times

- When the Start button is pressed the system should secure a GPS lock in ≤ 2 seconds
- The GPS update rate should be ≥ 1 Hz and the real-time metrics should refresh $\leq$ 500 ms, no perceivable lag when metrics are updated on the screen.

Trip Processing and Weather Enrichment

- The trip should complete processing ≤ 10 seconds.
- A hard limit of 5 seconds is enforced for weather enrichment. This is negotiable as the trip can be saved even without weather in case the service times-out.

Route Query Response

- Origin - destination query should yield results within 3 seconds of the query.
- Geocoding should yield desired results in ≤ 5 seconds along with PathScore computation of ≤ 0.5 seconds for paths viewable on the screen.
- All in all, the worst case scenario should give an output to the user in under 10 seconds.

Map Visualization Performance

- Maps with paths rendered should support a frame-rate of ≥ 30 FPS on a mid range smartphone.
- Pan/Zoom operations should be responsive (< 200 ms latency)

Trip Upload Performance

- GPS tracking data should be uploaded in ≤ 5 seconds (20 minute trip $\rightarrow$ 3 MB data).
- *gzip* is used when necessary to achieve $60 - 80\%$ reduction in size.

Performance efficiency

- The system shall support ≥ 5000 concurrent active users without performance degradation.
- Trip data should be around 2 MB including all the GPS tracking, stats, metadata.
- Historical data >30 days should be compressed via archival.

Battery Consumption

- Trip recording with the continuous use of GPS, Motion sensors and the screen should be ≤ 8 % battery per hour on a typical mid-range smartphone with 3000mAh battery.
- Idle state should result in minimal battery drain of around <1% per hour.

3.4 Design Constraints

3.4.1 Standard Compliance

Security and Conformity Standards

- TLS 1.2+ for all HTTPS communications.
- The system must be compliant with General Data Protection Regulation in accordance with the European Guidelines.
- The Password should be a minimum of 8 characters with good complexity - Uppercase, Lowercase, Number, Special characters.
- Should prevent SQL injections and have XSS protection.
- The system shall use OAuth 2.0 or equivalent for authentication.

Data Format Standards

- Geographical coordinates should follow WGS 84 - latitudes and longitudes in decimal degrees.
- Timestamps are to be in ISO 8601 format with timezones. Eg."2025-12-09T14:30:45Z"
- Distance should be in either Metric or Imperial based on users settings.

3.4.2 Hardware Limitations

Mobile device limitations

- OS : iOS 14+ / Android 10+

- Minimum Storage : 100 MB free space (app and local buffering)
- Minimum RAM : 2 GB (Optimal $\geq$ 4 GB)
- GPS Hardware : Standard smartphone GPS
- Motion Sensors: Standard Motion Sensor suit

Network Limitations

- Minimum Bandwidth of 1 Mbps upload and download.
- The system tolerates around 20 seconds of roundtrip delays without user facing timeout.
- Trip data is buffered locally for $\leq$ 48 hours. Sync is established upon reconnection.

Database and Storage

- Estimated daily volume with around 10,000 users is around 150 GB raw trip data
- Estimated 1 year storage is around 55 TB, database can be shard by geographic region for scalability

3.4.3 Other Design Constraints

Third-Party Service Dependencies

- Weather services like OpenWeather API or Equivalent are used with ¡ 5 seconds response time.
- OpenStreetMap or Google Maps API are used for maps and geocoding

Data Retention and Privacy

- Trip data is retained for 5 years after which it is deleted per GDPR data minimization guidelines.
- Path information is retained indefinitely as its deemed a community asset.
- User profiles are retained for the account lifetime. Should the user request to delete the account, the user profile data is wiped after 180 days.
- GPS tacking is not published, only the path status and obstacles are made public.

Platform Architecture

- The backend is divided into microservices like Trip Service, Path Service, Weather Service, Notification Service enabling for independent scaling when in need.
- Frontend is native mobile development aided by web based query interface.
- Database powered by PostgreSQL.

3.5 Software System Attributes

3.5.1 Reliability

Trip Data Integrity

- The trip data shall never be lost except in times of detrimental device failure
- The trip data should remain locally on device momentarily after trip completion and cloud synchronization should be carried out within 2 attempts.
- In case of timeouts, the recovery system will retry with uploading the trip data automatically and this shall require no user intervention unless there is persistent network issues.

Weather Enrichment Robustness

- Failure in acquiring weather enrichment will never cause a trip rejection.
- In case when the weather enrichment fails, the trip will be recorded without the weather data and the weather fields in the trip summary are marked "unavailable".
- Should the weather enrichment service fail, the system provides the user with the option of fetching the weather information within 48 hours port trip submission via a manual "Retry" option.

Data Consistency

- A trip once submitted to the system is un-editable.
- Paths once published cannot be edited or deleted.
- GPS lock on the device should be continuous and should be within tolerance of $<$ 5 seconds. If at all metadata for GPS includes larger gaps, the system will flag the metadata.

3.5.2 Availability

The system shall maintain an average operational state of $\geq 99.0\%$ with a maximum scheduled maintenance of 2 hrs/month during low traffic window (1:00 - 2:00 GMT). Critical services such as User authentication, Trip storage and Path queries are to replicated across 2 or more geographic locations to aid in automatic fail-over in case the primary system fails. Caching to be used in case the route computation times-out, so that results from previous queries are displayed offering the user to retry. Graceful degradation is offered with the weather service and geocoding services where trip can be saved without weather enrichment for the former and with user entering raw coordinates for the later.

3.5.3 Security

The system shall use OAuth 2.0 token based system where users can authenticate via Google, Apple or with Emails and passwords. Personal data like emails, GPS traces are encrypted using AES-156 encryption standard, while passwords use `bcrypt` hashing and at last HTTPS with TLS 1.2+ is used for communication. Threat mitigation has been provided against SQL injection, XSS and CSRF attacks. Further, a limit of 100 request per minute per IP, a max file upload limit of 100MB and a query timeout of 30 sec is put in place for protection against DoS attacks and for reliability of the system.

3.5.4 Maintainability

The system should be readable, maintainable and be fully documented. Microservice architecture to be used to ensure modular design principles with acceptable linear independent paths and unit tests. the services are to be designed such that they are scalable, maintainable and testable. Runbooks are used for starting, stopping, debugging and for common troubleshooting. Migrations are ought to be tested in QA or staging environment before the applying them too production databases.

3.5.5 Portability

Since the system mainly runs on smartphones, the mobile platform should be equipped with iOS 14+ (iPhone 8 +) or Android 10+ (2GB+ RAM devices.) sporting a responsive design of 320px for ease of use on handhelds. The system shall have no proprietary dependencies ensuring usage of public data sources.

4 Formal Analysis using Alloy

For the formal analysis of the BestBikePath system, we utilized the Alloy modeling language and the Alloy Analyzer tool. The BestBikePath system was essentially broken into two modules :

- Trip Recording System
- Path Publication System

for easier understanding and modeling. Each module was modeled separately along with relevant signatures, facts, assertions and predicates to capture the system's behavior and constraints.

BestBikePath Trip Recording:

The BestBikePath (BBP) Trip Recording system is designed to record and manage bike trips for users. The system captures essential trip details, including start and end locations, timestamps, and other trip information. The primary goal is to ensure accurate trip recording while maintaining data integrity and user distinguishability.

```
abstract sig User {
  userId: one Int
}

sig RegisteredUser extends User {
  userName: one String,
  email: one String,
  passwordHash: one String
}

sig UnregisteredUser extends User {
  sessionId: one String
}

sig Trip {
  tripId: one Int,
  state: one TripState,
  startTime: one Int,
  endTime: one Int,
  distance: one Int,
  duration: one Int,
  avgSpeed: one Int,
  maxSpeed: one Int,
  elevationGain: one Int,
  weather: lone WeatherData
}

-- TripState represents the trip recording lifecycle
abstract sig TripState {}
one sig TripCreated, TripRecording, TripPaused, TripCompleted,
    TripProcessing,
TripProcessed, TripSaved, TripSavedEnriched extends TripState {}

sig WeatherData {
  temperature: one Int,
  humidity: one Int,
  windSpeed: one Int,
  precipitation: one Int,
  conditions: one String,
  fetchedAt: one Int
}

sig BikePath {
```

```
43    pathId: one Int,
44    createdBy: one RegisteredUser,
45    createdAt: lone Int,
46    currentStatus: one PathStatus,
47    obstacles: set Obstacle
48  }
49
50  abstract sig PathStatus {}
51  one sig Optimal, Medium, Sufficient, RequiresMaintenance extends
        PathStatus {}
52
53  sig Obstacle {
54    obstacleId: one Int,
55    type: one ObstacleType,
56    location: one String,
57    reportedBy: one RegisteredUser
58  }
59
60  abstract sig ObstacleType {}
61  one sig Pothole, Construction, Debris, Infrastructure extends
        ObstacleType {}
62
63  sig PathInformation {
64    pathId: one Int,
65    path: one BikePath,
66    trip: some Trip,
67    submittedBy: one RegisteredUser,
68    creationType: one CreationType,
69    state: one PublicationState
70  }
71
72  abstract sig CreationType {}
73  one sig ManualEntry, AutomaticEntry extends CreationType {}
74
75  abstract sig PublicationState {}
76  one sig Private, Public extends PublicationState {}
77
78  -- ===== CONSTRAINTS AND FACTS =====
79
80  -- Fact: Obstacles must be reported by a RegisteredUser
81  fact ObstaclesHaveReporters {
82    all o: Obstacle | some o.reportedBy
83  }
84
85  -- Fact: Obstacles must belong to a Path
86  fact ObstaclesBelongToPaths {
87    all o: Obstacle | one bp: BikePath | o in bp.obstacles
88  }
89
90  -- Fact: Consistent Trip Data
91  fact TripNumericValuesConsistency {
92    all t: Trip {
93      t.distance > 0
94      t.duration > 0
95      t.avgSpeed >= 0
96      t.maxSpeed >= t.avgSpeed
97      t.endTime > t.startTime
98    }
99  }
100
101
102 -- ===== ASSERTIONS =====
103
```

```
104 -- Assert 1: Check that every obstacle has a location and reporter
105 assert ObstaclesHaveData {
106   all o: Obstacle | some o.location and some o.reportedBy
107 }
108 check ObstaclesHaveData for 10 but 6 Int
109
110 -- Assert 2: Saved Trips Have Valid Statistics
111 assert SavedTripsHaveStats {
112   all t: Trip |
113     (t.state = TripSaved) => (t.distance > 0 and t.duration > 0 and
          t.avgSpeed >= 0)
114 }
115 check SavedTripsHaveStats for 10
116
117 -- Assert 3: Weather Is Optional But Consistent (review later)
118 assert WeatherConsistency {
119   all t: Trip |
120     (t.state in TripSavedEnriched) => (t.weather = WeatherData)
121 }
122 check WeatherConsistency for 10
```

Result:

The following figure shows the result of running the above assertions of the Alloy model in the Alloy Analyzer. It shows that the assertions we defined hold true.

```
Executing "Check ObstaclesHaveData for 10 but 6 int"
  Actual scopes: 10 User, 10 RegisteredUser, 10 UnregisteredUser, 10 Trip, 8 TripState,
  Solver=sat4j Bitwidth=6 MaxSeq=10 SkolemDepth=1 Symmetry=20 Mode=batch
  116851 vars. 12500 primary vars. 328449 clauses. 262ms.
  No counterexample found. Assertion may be valid. 19ms.

Executing "Check SavedTripsHaveStats for 10"
  Actual scopes: 10 User, 10 RegisteredUser, 10 UnregisteredUser, 10 Trip, 8 TripState,
  Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20 Mode=batch
  149487 vars. 16360 primary vars. 406612 clauses. 54ms.
  No counterexample found. Assertion may be valid. 47ms.

Executing "Check WeatherConsistency for 10"
  Actual scopes: 10 User, 10 RegisteredUser, 10 UnregisteredUser, 10 Trip, 8 TripState,
  Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20 Mode=batch
  181311 vars. 20220 primary vars. 481735 clauses. 60ms.
  No counterexample found. Assertion may be valid. 3ms.
```

BestBikePath Path Publication:

The second model focuses on the rules governing the review, publication, and visibility of bike path information contributed by users. These rules capture system-wide constraints that determine how information becomes accessible to other users.

```
-- Actors --
abstract sig User {}
sig RegisteredUser extends User {}
sig UnregisteredUser extends User {}

-- Paths --
sig BikePath {
  obstacles: set PathObstacle
}

-- Path contribution modes
abstract sig PathCreationMode {}
one sig ManualMode, AutomaticMode extends PathCreationMode {}

-- Review status
abstract sig PathReviewStatus {}
one sig Unreviewed, Reviewed extends PathReviewStatus {}

-- Publication status
abstract sig PathPublicationStatus {}
one sig Private, Public extends PathPublicationStatus {}

-- Optional obstacle abstraction
abstract sig PathObstacle {}
one sig Pothole, Construction, Debris, Infrastructure extends
    PathObstacle {}

-- Path information contributed by users
sig PathInformation {
  path: one BikePath,
  submittedBy: one RegisteredUser,
  creationMode: one PathCreationMode,
  reviewStatus: one PathReviewStatus,
  publicationStatus: one PathPublicationStatus
}


-- == Key Facts ==

-- Fact 1: Only registered users can submit path information
fact OnlyRegisteredUsersSubmit {
  all pi: PathInformation | pi.submittedBy in RegisteredUser
}

-- Fact 2: Manually created paths is considered reviewed by default
fact AllManualPathsAreReviewed {
  all pi: PathInformation |
    pi.creationMode = ManualMode => pi.reviewStatus = Reviewed
}

-- Fact 3: Automatically registered data must be reviewed before
    publication
fact AutomaticEntriesNeedReviewBeforePublish {
  all pi: PathInformation |
    (pi.creationMode = AutomaticMode and pi.publicationStatus =
        Public)
      => pi.reviewStatus = Reviewed
}
```

```
56
57 -- Function that returns all public paths or submitted by a specific
       user
58 fun visiblePathInformation[u: User]: set PathInformation {
59   { pi: PathInformation |
60       pi.publicationStatus = Public
61       or pi.submittedBy = u
62   }
63 }
```

Result:

The following assertions validate the core constraints governing review and publication of path information, ensuring that unsafe or unreviewed data cannot be publicly exposed.

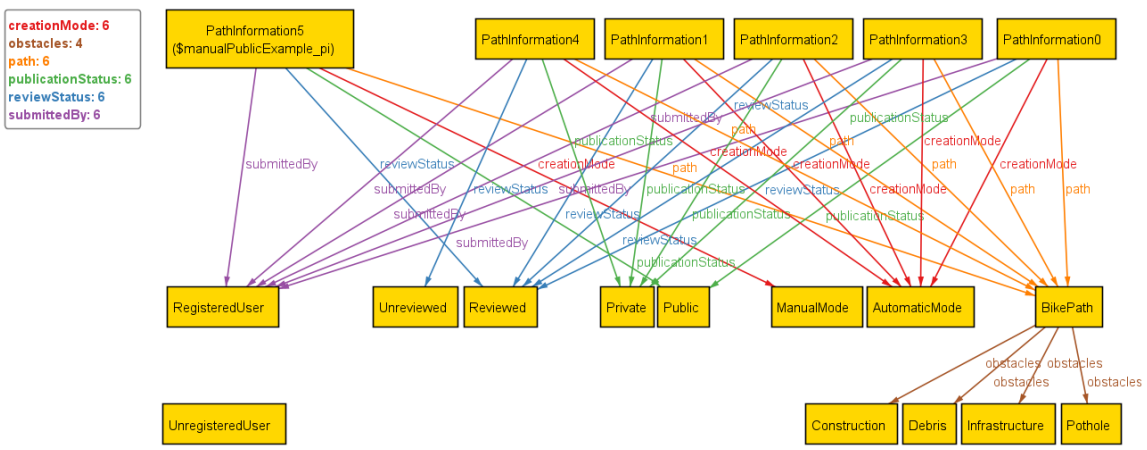
```
1
2 -- == Assertions ==
3
4 -- Assertion 1: No automatically acquired, unreviewed path
      information is publicly visible
5 assert NoUnreviewedAutomaticPublic {
6   no pi: PathInformation |
7     pi.creationMode = AutomaticMode
8     and pi.reviewStatus = Unreviewed
9     and pi.publicationStatus = Public
10 }
11 check NoUnreviewedAutomaticPublic for 10
12
13 -- Assertion 2: Any publicly visible path information has been
      reviewed beforehand
14 assert PublishedOnlyIfReviewed {
15   all pi: PathInformation |
16     pi.publicationStatus = Public => pi.reviewStatus = Reviewed
17 }
18 check PublishedOnlyIfReviewed for 10
19
20 -- Assertion 3: Manual path information is recognized as reviewed by
       default
21 assert ManualAlwaysReviewed {
22   all pi: PathInformation |
23     pi.creationMode = ManualMode => pi.reviewStatus = Reviewed
24 }
25 check ManualAlwaysReviewed for 10
26
27 -- Assertion 4: Unregistered users never see private path
      information
28 assert UnregisteredUsersSeeOnlyPublic {
29   all u: UnregisteredUser |
30     all pi: PathInformation |
31       pi in visiblePathInformation[u] =>
32         pi.publicationStatus = Public
33 }
34 check UnregisteredUsersSeeOnlyPublic for 20
35
36 -- Assertion 5: Registered users are able to see their own private
      path information
37 assert OwnersCanSeePrivatePaths {
38   all u: RegisteredUser |
39     all pi: PathInformation |
40       (pi.submittedBy = u) =>
41         pi in visiblePathInformation[u]
42 }
43 check OwnersCanSeePrivatePaths for 20
```

```
Executing "Check NoUnreviewedAutomaticPublic for 10"
   Actual scopes: 10 User, 10 RegisteredUser, 10 UnregisteredUser, 10 BikePath
   Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20 Mode=batch
   3892 vars. 350 primary vars. 6677 clauses. 31ms.
   No counterexample found. Assertion may be valid. 16ms.

Executing "Check PublishedOnlyIfReviewed for 10"
   Actual scopes: 10 User, 10 RegisteredUser, 10 UnregisteredUser, 10 BikePath
   Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20 Mode=batch
   7760 vars. 700 primary vars. 13308 clauses. 23ms.
   No counterexample found. Assertion may be valid. 4ms.

Executing "Check ManualAlwaysReviewed for 10"
   Actual scopes: 10 User, 10 RegisteredUser, 10 UnregisteredUser, 10 BikePath
   Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20 Mode=batch
   11628 vars. 1050 primary vars. 19939 clauses. 18ms.
   No counterexample found. Assertion may be valid. 2ms.

Executing "Check UnregisteredUsersSeeOnlyPublic for 20"
   Actual scopes: 20 User, 20 RegisteredUser, 20 UnregisteredUser, 20 BikePath
   Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20 Mode=batch
   24198 vars. 2170 primary vars. 42436 clauses. 36ms.
   No counterexample found. Assertion may be valid. 18ms.

Executing "Check OwnersCanSeePrivatePaths for 20"
   Actual scopes: 20 User, 20 RegisteredUser, 20 UnregisteredUser, 20 BikePath
   Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20 Mode=batch
   37188 vars. 3290 primary vars. 66534 clauses. 38ms.
   No counterexample found. Assertion may be valid. 15ms.
```

As we can see in the following examples, the three predicates defined in the model demonstrate the intended behavior of the BestBikePath Path Publication system, in which only safe and reviewed paths are made public, and unregistered users cannot contribute path information.

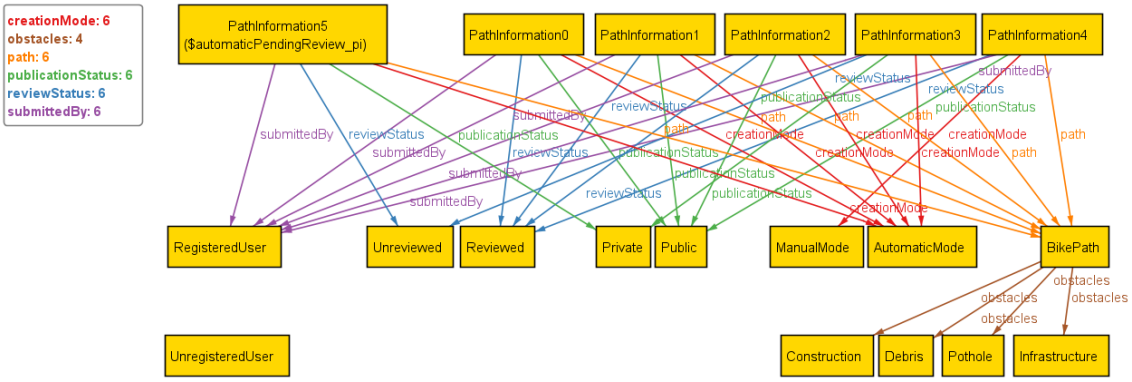
Predicate 1: In this first example, the configuration includes a path information object created in manual mode that is marked as reviewed and set to public visibility (`PathInformation5`). This instance represents a successfully reviewed and published path, as described in UC7. At the same time, the figure shows examples of other objects which still represent valid system states that coexist with the published path, demonstrating that the model allows for a variety of states while enforcing the publication rules.

```
1 pred manualPublicExample {
2   some pi: PathInformation |
3     pi.creationMode = ManualMode
4     and pi.reviewStatus = Reviewed
5     and pi.publicationStatus = Public
6 }
7 run manualPublicExample for 6
```



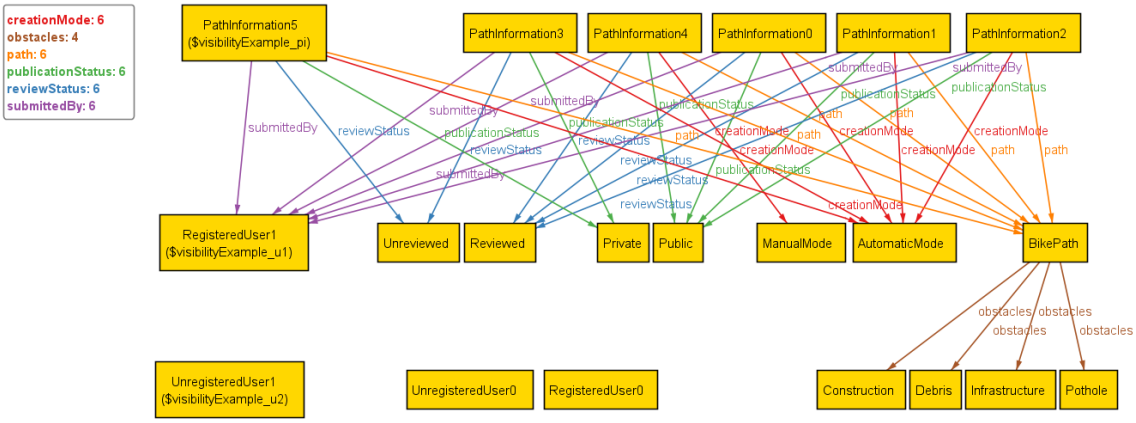
Predicate 2: Similarly, in the next example, the configuration includes a path information object created in automatic mode that is marked as unreviewed and kept private (`PathInformation5`). This instance represents the system state immediately after automatic data acquisition and before user confirmation, as described in UC8.

```
pred automaticPendingReview {
  some pi: PathInformation |
    pi.creationMode = AutomaticMode
    and pi.reviewStatus = Unreviewed
    and pi.publicationStatus = Private
}
run automaticPendingReview for 6
```



Predicate 3: Finally, in the last example, we extend the previous analyses by focusing on information visibility, showing that unreviewed automatic contributions remain private and accessible only to their registered owner.

```
pred visibilityExample {
  some u1: RegisteredUser, u2: UnregisteredUser, pi: PathInformation
      |
    pi.creationMode = AutomaticMode
    and pi.reviewStatus = Unreviewed
    and pi.publicationStatus = Private
    and pi.submittedBy = u1
    and pi in visiblePathInformation[u1]
    and pi not in visiblePathInformation[u2]
}
run visibilityExample for 6
```



5 Effort spent

The following table displays the effort spent by each team member on the different sections of this document, measured in hours. The division of work is only indicative and may not reflect the actual time spent by each member as each section requires collaboration from all members.

Team Member	Sections 1	Section 2	Section 3	Section 4
Cristhian Mejia	7	4	15	10
Sravan Yerranagu	2	12	13	10

6 References

References

[1] Software Engineering 2, *2025/26 Requirement Engineering and Design Project Document* The specification for the 2025/26 Requirement Engineering and Design Project for the Software Engineering II course.

[2] The slides on the WeBeep page of the Software Engineering II course.

[3] JGraph, *diagrams.net (draw.io)*, version 29.0.3. Available at: `https://www.diagrams.net/`.

[4] Mermaid Project, *Mermaid State Diagram Documentation*. Available at: `https://docs.mermaidchart.com/`

[5] OpenWeather, *OpenWeather API Documentation*. Available at: `https://openweathermap.org/api`

[6] OpenStreetMap, *OpenStreetMap API Documentation*. Available at: `https://wiki.openstreetmap.org/wiki/API`

[7] OWASP Foundation - Standrd Awareness Document, *OWASP Top 10 - 2021*. Available at: `https://owasp.org/Top10/2021/`

[8] D. Hardt, *The OAuth 2.0 Authorization Framework*, datatracker.ietf.org, Oct. 2012. Available at: `https://datatracker.ietf.org/doc/html/rfc6749`

[9] REST API Tutorial, *REST API Tutorial: Learn REST API Design*, Available at: `https://www.restapitutorial.com`.

[10] Alloy Team, *Alloy: A Language and Analyzer for Software Modeling,* open source language and analyzer for software modeling, version 6.2.0. Available at: `https://alloytools.org`

[11] European Union, *General Data Protection Regulation (GDPR)*, Official Journal of the European Union, 2016/679. Available at: `https://gdpr-info.eu/`.