POLITECNICO
MILANO 1863

Requirement Analysis and Specification Document RASD

Best Bike Paths

Cristian Summa Mathias Rodigari

Software Engineering II

Academic Year 2025–2026

1 Introduction

1.1 Purpose

In recent years, cycling has become an increasingly popular way to move through cities, whether for commuting, leisure, or sport. As more cyclists take to the streets, the need for safe, well-maintained and easily accessible bike paths has grown accordingly. Yet, riders often struggle to find reliable information about the quality of a route, its suitability for cycling or the obstacles they may encounter along the way. At the same time, cyclists wishing to keep track of their activities lack a unified environment where their personal trips and community-generated insights can coexist and support one another.

The purpose of Best Bike Paths (BBP) is to create a shared platform where cyclists can record their rides, contribute meaningful information about the state of bike paths, and benefit from the knowledge collected by the community. Through manual reporting and detailed trip logging, users can help build a rich, continuously updated map of cycling conditions. BBP also enables anyone to explore optimal routes between two locations, selecting paths that are not only efficient but also pleasant and safe.

By bringing together personal activity tracking and collaborative path information, BBP aims to improve the cycling experience as a whole, empowering users with the clarity and confidence needed to choose the routes that best match their preferences and needs.

1.1.1 Goals

G1: **Support trip recording:** The system shall allow registered users to record their cycling trips and store them for future consultation.

G2: **Provide meaningful trip insights:** The system shall analyse recorded trips and present users with relevant statistics, such as distance, duration, and performance metrics.

G3: **Enable sharing of bike path information:** The system shall allow users to contribute information about bike paths and make it available to the community.

G4: **Support automated path acquisition:** The system shall allow users to start an automated tracking session during a bike ride, during which the system acquires path data through the device's sensors.

G5: **Report obstacles and path conditions:** The system shall identify, store, and display potential obstacles or hazards along bike paths to inform and alert users.

G6: **Support manual path insertion:** The system shall allow users to manually enter information about bike paths, including their status and characteristics.

G7: **Provide path discovery between two locations:** The system shall help users find all available bike paths connecting a given origin and destination.

1.2 Scope

The core purpose of Best Bike Paths (BBP) is to support cyclists in documenting their rides, contributing information about bike paths, and accessing reliable routes that suit their needs. The system acts as a mediator between users and the information they generate, collecting data from individual cyclists and making it accessible to the broader community.

The external users who interact with BBP fall into two categories: registered users, who can record trips and insert information about bike paths, and unregistered users, who can freely browse available routes. Registered users access the system through personal accounts, which allow them to store their activity history and contribute to the shared repository of path data. When recording a ride, users may either manually insert the characteristics and status of the paths they traverse or, if enabled, start an automated tracking session that collects sensor data to reconstruct the travelled route and detect potential obstacles. Before such automatically gathered information becomes public, users must review and confirm its correctness, ensuring that the shared data remains trustworthy.

BBP also provides functionalities for any user, registered or not, to search for routes between two locations. The system analyzes all available path information and returns one or more viable bike paths, prioritizing those that are safer, more reliable, or better maintained based on community-provided data.

Through these capabilities, the system offers a unified environment where cyclists can both contribute and benefit from up-to-date knowledge about bike paths, improving navigation, safety awareness, and overall cycling experience.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

Path
A segment of road or bike infrastructure considered safe for cycling. It may consist of a dedicated bike lane, a road where motor vehicles are restricted, or a street where traffic conditions (e.g., low speed limits) allow safe coexistence with cars.

Trip
A cycling activity performed by a user, consisting of movement across one or more paths or streets. A trip has a clearly defined start and end time and may include additional recorded information such as distance, average speed, weather, etc.

User
Any individual interacting with the system. Depending on the type of interaction, the system distinguishes between registered and unregistered users.

Registered User
A user who has created an account. Registered users can record trips, insert path-related information, initiate tracking sessions, and access personal data history.

Unregistered User
A user who interacts without an account. They can browse paths, search routes, and record trips locally, but cannot save trips or contribute path information.

Path Status
A qualitative evaluation of a path's condition using categories (optimal, medium, sufficient, requires maintenance). Reflects surface quality, safety, and usability.

Obstacle
Any element along a path that may impede or endanger a cyclist, like potholes, bumps, debris, or irregularities. Can be reported manually or inferred via sensors.

Tracking Session
A user-initiated recording mode collecting sensor data (GPS, accelerometer, gyroscope) to reconstruct the route and detect obstacles.

Sensor Data
Information collected during a tracking session, including GPS location, speed, acceleration, and orientation, used to reconstruct paths and detect obstacles.

1.3.2 Acronyms

BBP Best Bike Paths

GPS Global Positioning System

API Application Programming Interface

DB Database

UI User Interface

UX User Experience

OS Operating System

SSO Single Sign On

1.4 Revision History

Version 1.0: 23/12/2025

1.5 Reference Documents

This document and the information contained herein refer to the following sources:

- Requirement Engineering and Design Project Specification, Software Engineering II course, Academic Year 2025/26.
- Course slides and materials available on the WeBeep page of the Software Engineering II course.

1.6 Document Structure

This document is organized as follows:

1. **Introduction:** Provides an overview of the product, highlighting its goals, purpose, and scope.
2. **Overall Description:** Offers a more detailed view of the product, including real-world scenarios, context, and a technical analysis of the system.
3. **Specific Requirements:** Lists all the functional and non-functional requirements necessary to achieve the product goals, along with their relationships to domain assumptions.
4. **Formal Analysis Using Alloy:** Presents a formal description of a portion of the real-world phenomena, analyzed through Alloy to identify inconsistencies between assumptions and modeled phenomena.
5. **Effort Spent:** Details the contribution and time spent by each team member in the preparation of this document.
6. **References:** Lists all documents, tools, and resources used in the creation of this document.

2 Overall Description

2.1 Product Perspective

2.1.1 Scenarios

User Searches for a Bike Path to Reach a Specific Destination User **U** intends to ride their bicycle to a particular destination. They open the BBP client on their smartphone and log in if not already authenticated. Next, they navigate to the search field and enter the desired destination.

BBP then displays a set of paths leading toward the destination. While some paths may not reach the destination precisely, all are presented as potential options. The available paths are displayed both on a map preview and as a list. BBP clearly highlights the most optimal path, considering factors such as distance, path quality, presence of obstacles, and estimated duration.

User **U** selects a preferred path and presses the "Go" button, which redirects them to the Navigation screen. Here, a map and a textual/visual representation of turn-by-turn instructions guide **U** to the destination while they ride their bike with their smartphone secured in a bike holder.

Note: In this scenario, multiple paths exist to the destination. If no path is available, BBP will prompt the user to add the missing path, and the process continues as described in the next section.

User Adds a Missing Path to BBP

When User **U** discovers a path not yet recorded in BBP, they can add it to the system by selecting the "Add Path" button and choosing a method for submitting the path data:

- **Automated Mode:** In Automated Mode, **U** is redirected to a screen similar to the Navigation screen. The map is displayed to provide orientation, but turn-by-turn navigation instructions are not provided. **U** then rides their bike along the missing path while BBP records sensor data, including GPS and other device sensors, to capture path information automatically.
- **Manual Mode:** In Manual Mode, **U** is redirected to a dedicated interface where they can manually input path data. The screen includes a map for selecting streets by tapping, as well as a search field for entering the street name. Once the street is selected and BBP confirms its suitability for inclusion as a path, the "Next" button becomes active, allowing **U** to proceed.

In the following step, **U** can record obstacles along the path, specifying their type (e.g. pothole) and location on the map.

Upon completion, the path is saved as a **Draft**, and a summary screen is displayed. The user may review and edit the data or click "Publish" to submit the

path to BBP's Review Team. The team verifies the correctness of the information and, once approved, adds the path to the BBP database.

2.1.2 Domain Class Diagram

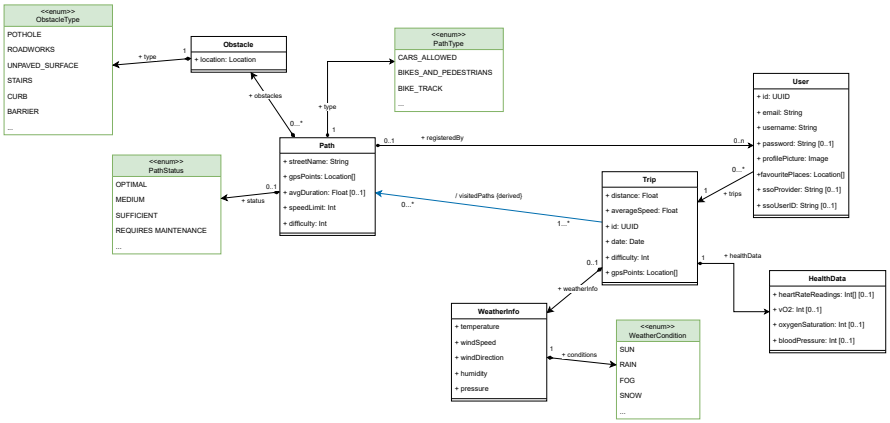


Figure 1: UML representation of the Domain Class Diagram

The Domain Class Diagram illustrates the main entities involved in the system domain and the relationships among them. It provides a high-level conceptual view of the data model, intentionally abstracting away implementation-specific details in order to remain independent of the target programming language or framework.

Location, Coordinates, and Geo-Referenced Data

BBP makes extensive use of geographic data, such as coordinates and geotagged information. In this diagram, we assume that the implementation language and/or framework provides a suitable data structure or class to represent such data. This abstraction is generically referred to as the **Location** class.

If the chosen technology stack does not provide native support for this kind of data, implementing a custom Location class would be straightforward and would not affect the overall design.

Optional Attributes and Nullability

The diagram abstracts from concrete implementation details related to nullable and non-nullable fields. While collections naturally allow empty states when modeling relationships between classes, some attributes within classes may also be optional.

These cases are explicitly indicated using the UML multiplicity notation [0..1], meaning that the corresponding attribute may be absent (i.e., null). Attributes

without this notation are considered mandatory. The way this constraint is enforced will depend on the implementation language, in particular on whether it provides native support for optional or nullable types.

Paths as Real-World Streets

In this design, a **Path** corresponds to a real-world street. This modeling choice simplifies global coverage and route computation: rather than requiring a predefined path between every pair of locations, routes can be dynamically computed by connecting multiple streets, similarly to standard navigation systems.

This approach improves scalability and flexibility while aligning with established geographic information system practices.

Separation Between Trips and Paths

A key design decision is to avoid storing Paths directly within Trips. Although saving the list of Paths traversed by a User inside a Trip would be simpler, BBP cannot enforce user behavior. Users may deviate from suggested routes, ride on unmapped streets, or traverse areas considered unsafe by BBP.

To ensure robustness and system stability, both **Trip** and **Path** independently store their own sequences of Location points. This allows Trips to safely handle out-of-scope or unknown geographic data, while the association between Trips and Paths is computed dynamically when needed.

An additional benefit of this choice is improved temporal robustness: BBP can correctly handle situations where Paths change over time, become unsafe, or cease to exist altogether, without invalidating previously recorded Trips.

Health Data Collection

Trips may include health-related data acquired through sensors on connected accessories or devices. This information is encapsulated in the **HealthData** class, which is associated with each Trip.

Given the wide variety of possible devices, sensors, and supported capabilities, all attributes within HealthData are modeled as optional. This design ensures maximum flexibility and allows BBP to safely handle partial or incomplete datasets without compromising correctness or stability.

Path Registration and User Association

Paths may be associated with the User who registered them. This relationship is optional [0..1], since not all Paths managed by BBP are expected to be user-generated.

Because BBP already relies on a map API for geographic data, the system database is pre-populated, prior to public release, with all Paths (i.e., real-world streets) that satisfy BBP's quality and safety standards. As a result, user

registration of Paths is not the primary mechanism through which BBP acquires path data.

Nevertheless, Users retain the ability to register Paths through the two mechanisms specified in the requirements. These interactions are treated as "missing data" reports, allowing Users to enrich or correct the existing dataset rather than acting as the main source of path acquisition. Leveraging an already extensive and well-structured street database significantly improves coverage, data consistency, and overall system reliability.

However, Users can, and are encouraged to, provide updated information regarding the status of all Paths, as they are often the most accurate source of real-world road conditions and similar contextual data.

2.1.3 State Diagram

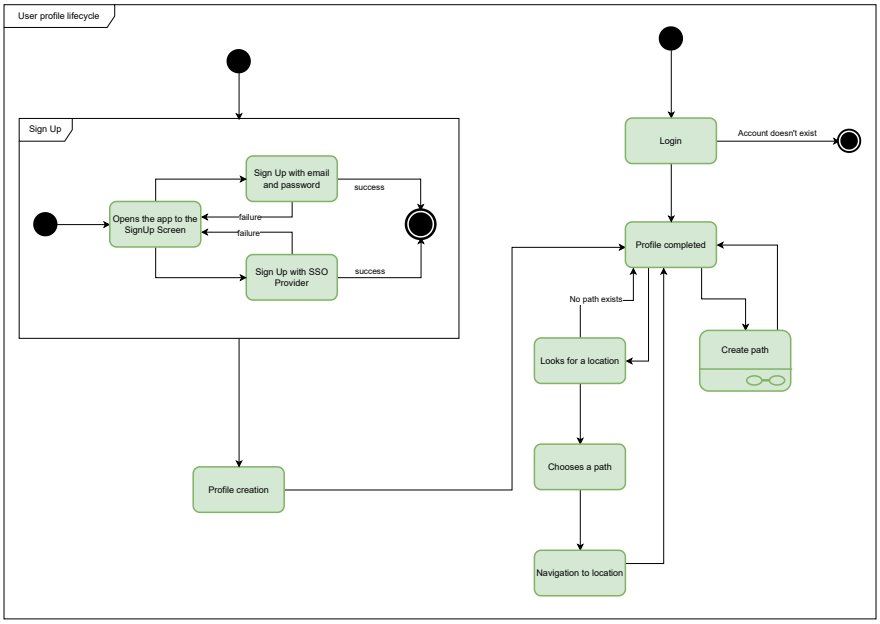


Figure 2: State Diagram representing the User profile's lifecycle

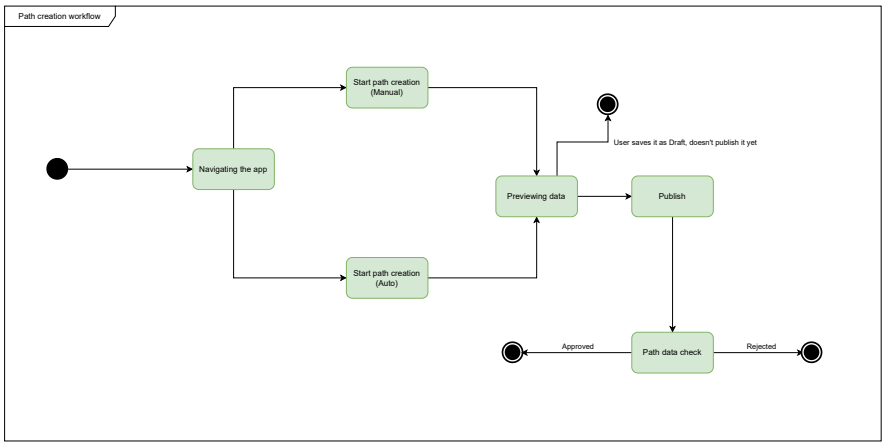


Figure 3: State Diagram representing the internal 'Create Path' subprocess

2.2 Product Functions

Sign Up and Login The sign-up process is straightforward. Users may register using a traditional email-and-password method or via Single Sign-On (SSO). The system should support the main SSO providers, particularly those natively integrated with mobile operating systems, such as Google and Apple. Optionally, third-party providers, such as Meta (Sign in with Facebook), may also be integrated.

To log in, users enter their email address and password and select the "Login" button. The system then verifies the credentials and, if correct, authenticates the user. Alternatively, users may select an SSO provider, in which case they are redirected to the provider's authentication page. Upon successful authentication, the user is redirected back to BBP, which grants access to the application.

Searching for a Path

Users can search for a destination by typing its name into the search field and selecting from a list of suggestions. Alternatively, users may navigate the map and manually select the destination.

Once a destination is selected, BBP displays a list of possible paths from the user's current location to the destination, ranked by effectiveness. Paths are displayed both on the map and in a list format.

Navigating to a Destination

After selecting a path, the user can initiate navigation by pressing the "Start Navigation" button. This action begins a trip recording event during which BBP records sensor data and provides turn-by-turn navigation instructions. The user

can then follow the path to reach the destination.

Automated Mode

Users may add a path by selecting the "Add Path" button and choosing "Automated Mode." In this mode, BBP initiates a trip recording event similar to standard navigation; however, turn-by-turn instructions are not provided.

While riding, BBP displays a map to assist with orientation and records health data as well as real-time GPS tracking. Sensor data from the device, such as accelerometer and gyroscope readings, are collected to identify potential obstacles along the path.

At the conclusion of the trip, BBP presents an overview of the recorded data and prompts the user to confirm or edit it. The path may then be published and made publicly accessible if the user chooses.

Importantly, BBP does not automatically start tracking in the background; the session begins only when initiated by the user. After the session, the user reviews and confirms or corrects any detected obstacles before the data is published to the community, ensuring accuracy and avoiding false positives.

Manual Mode

Users may also add a path manually by selecting "Manual Mode" after pressing the "Add Path" button. In this mode, users are directed to a dedicated interface where they can input relevant data.

Users can specify the path location by selecting it on a map or by entering the street name and choosing from suggestions. Additional details, such as whether cars and pedestrians are allowed, the path's status, and the presence of obstacles, must also be provided. Obstacles can be recorded by touching the relevant point on the map or dragging and dropping an obstacle icon to its location.

At the end of the process, BBP presents a summary of the entered data and prompts the user to confirm or edit it. The path may then be published and made publicly accessible if the user chooses.

2.3 User Characteristics

BBP distinguishes between two main types of users: Registered Users and Unregistered Users. Each type has specific capabilities and limitations within the system.

Registered Users

Registered Users have created an account in BBP and can access the full functionality of the system. Their characteristics include:

- Ability to record trips and store them for future reference.
- Ability to add new bike paths, either manually or using automated tracking.
- Ability to report obstacles along paths and update path status information.
- Access to personal trip history and statistics, such as distance traveled, average speed, and other performance metrics.
- Ability to search for routes and navigate between locations using the system.
- Responsibility to confirm or edit any automatically collected data before it is published.

Unregistered Users

Unregistered Users interact with BBP without an account and have limited capabilities:

- Can browse and view available bike paths.
- Can search for routes between two locations.
- Can record trips locally on their device, but these trips are not saved to the system.
- Cannot contribute path information, report obstacles, or access historical data in the system.

General Characteristics

All users, regardless of registration status, are expected to:

- Interact with the system via the BBP client on a smartphone.
- Follow the system's interface and workflow to ensure data consistency and reliability.
- Be aware that contributions from Registered Users are shared with the community after verification, while unregistered interactions remain local.

2.4 Assumptions, Dependencies and Constraints

2.4.1 Regulatory Policies

Data Privacy and Protection
All personal information collected from users, including account details, trip records, and contributed path or obstacle data, must be handled in accordance with the General Data Protection Regulation (GDPR) and other relevant privacy laws.

Non-Commercial Use
User-contributed data shall not be used for commercial purposes beyond the operation and improvement of BBP services.

Data Confidentiality
Any sensitive information, including health data from sensors or user location history, must be stored securely and accessed only by authorized components or personnel.

Compliance Verification
Processes involving user-generated content, such as path publication or obstacle reports, must include review mechanisms to ensure data accuracy, reliability, and adherence to platform policies.

2.4.2 Domain Assumptions

D1: User Authentication Availability
Users who perform actions involving data creation or modification that affect the shared system state (e.g., inserting or publishing Paths, contributing Obstacle information) are registered and authenticated in the system.

D2: Anonymous User Capabilities
Unregistered users can use the system to browse available bike Paths, search for routes between locations, and record Trips. Trips recorded by unregistered Users are stored only locally on the user's device.

D3: Isolation of Anonymous User Data
Data generated by unregistered Users, including recorded Trips and related statistics, is not stored in the system's database and does not contribute to shared Path information, Obstacle detection, or Path status evaluation.

D4: User-Initiated Tracking Sessions
Trip recording and automated tracking sessions are explicitly started by the User. The system does not autonomously initiate tracking activities.

D5: Availability of Device Sensors
The User's mobile device provides access to location and motion sensors (e.g., GPS, accelerometer, gyroscope) required for Trip recording, Path reconstruction, and Obstacle detection.

D6: Accuracy of Sensor Data
Sensor data collected during a tracking session is assumed to be sufficiently accurate to allow meaningful reconstruction of Paths and identification of potential Obstacles.

D7: User Data Review and Confirmation
Users review, confirm, and, if necessary, correct automatically collected or inferred data (e.g., detected Obstacles, reconstructed Paths) before such data is published and made available to the community.

D8: Availability of External Services
External services providing map data, map rendering, and weather information are available and respond within acceptable time constraints during normal system operation.

D9: Network Connectivity
Users have a reliable network connection when interacting with the system functionalities that require data transmission to or from the backend.

3 Specific Requirements

3.1 External Interface Requirements

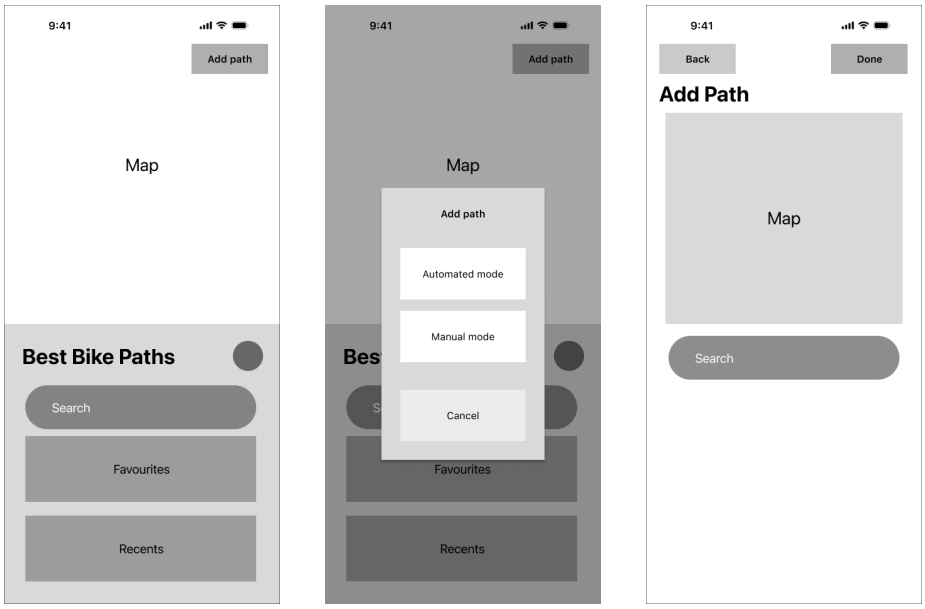
3.1.1 Users Interfaces

BBP's client will be a mobile application built with native technologies for both target OSes, allowing direct access to system APIs to access sensors data, health and fitness integrations and possibly better background execution compared to cross platform and web-based solutions.

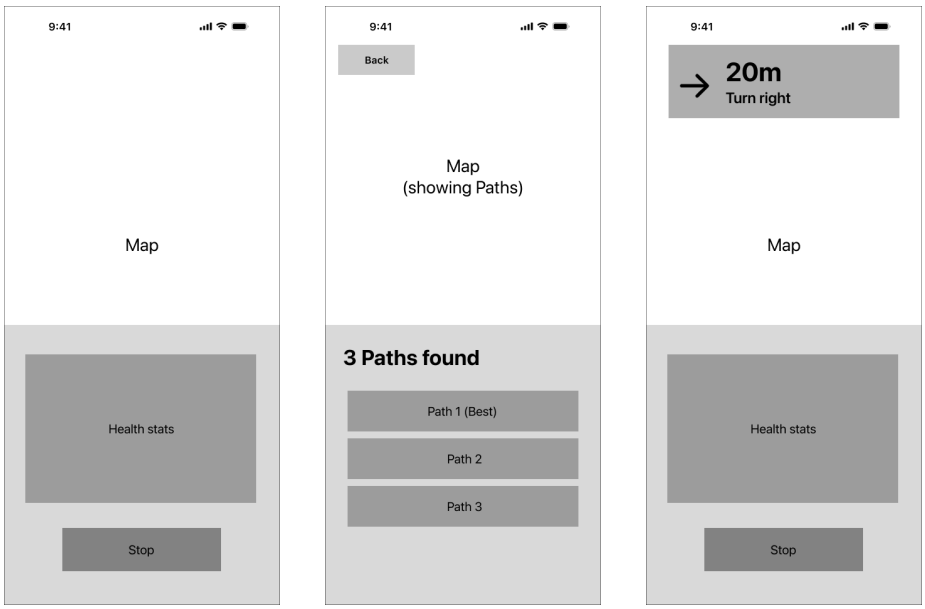
Being native, BBP will attempt as much as possible to offer a User Experience that is both pleasurable and consistent with what the userbase from the respective OS expects, both in terms of design and layout.

While the different clients will implement platform-specific design paradigms (Liquid Glass in iOS, Material Design 3 in Android), the foundation of its layout will be shared between both, providing a similar-enough experience on both sides.

Here are some wireframe design mockups for the layout of BBP, platform-agnostic at the moment. High level mockups will be provided in the following Design Document.



(a) Main screen (b) Add Path mode (c) Add Path - Manual



(d) Add Path - Auto (e) Search paths (f) Trip and Navigation

Figure 4: Mockup screens of the BBP mobile app UI layout

3.1.2 Hardware Interfaces

Our platform is a mobile application designed for Android (version 9 and above) and iOS (version 18 and above) devices. Users need a smartphone with an active internet connection to access all core functionalities. No additional hardware is required; however, optional devices such as smartwatches or fitness trackers can be integrated to provide enhanced tracking and health-related data.

3.1.3 Software Interfaces

Weather Information Provider
Provides weather forecasts and real-time weather information required by the system. This information is used, for example, to display expected weather conditions before a trip and to store such information as part of the recorded trip for future reference.

Map Data Provider
Provides geographic and street-level data, including road networks and bike paths. Such data is used, for example, to populate known paths, compute routes, analyze path characteristics, and support the validation of user-generated paths.

Map Rendering Provider
Renders maps and geographic information and supports user interaction with the map. This functionality is used, for example, to display routes on a map, visualize turn-by-turn navigation instructions, and allow the user to explore the map through zooming and panning.

Health and Fitness System Services
Provides access to fitness-related data and tracking capabilities exposed by the underlying operating system. This data is used to record physical activity metrics during a trip and to integrate trip information with system-level fitness tracking features.

3.1.4 Software Interfaces

The BBP mobile application communicates with users and external services through secure APIs. All interactions, including data transmission and location information, must be encrypted using HTTPS.

User authentication is supported via two mechanisms:

- **Traditional email and password:** Users can register and log in using their email and password.
- **Single Sign-On (SSO):** Users may authenticate using major SSO providers.

For real-time features, such as turn-by-turn navigation, the platform must use an appropriate protocol (e.g., WebSocket) to ensure low-latency communication.

Additionally, the application shall support push notifications to deliver timely updates and alerts to users.

3.2 Functional Requirements

R1 The system shall provide statistics about each trip, including weather information (if available), distance, average speed, and related metrics.

R2 Users shall be able to insert and publish information about bike paths, including their status and the presence of obstacles.

R3 Users shall be able to manually insert paths by specifying street names.

R4 The system shall translate street names into GPS coordinates and store the path information.

R5 Users shall be able to start a tracking activity on their mobile device prior to a bike ride, instructing the system to record the path of their trip.

R6 The system shall record data from device sensors during a tracking session.

R7 The system shall reconstruct a path based on the GPS data collected during the tracking session.

R8 The system shall detect sudden anomalies in sensor data and associate them with potential obstacles along the path.

R9 The system shall present the recorded data to the user at the end of the trip, requesting confirmation or correction.

R10 The system shall allow users to publish the path data after it has been recorded and confirmed.

R11 Users shall be able to query available paths given an origin and destination.

R12 The system shall visualize all available paths on a map, given two locations (origin and destination).

R13 The system shall compute a score to evaluate the effectiveness of each path in reaching the destination.

R14 The system shall rank multiple plausible paths based on their computed score.

R15 During an automated tracking session, BBP shall analyze speed and sensor data to detect cycling activity, reconstruct the path, and identify potential obstacles. The session must be explicitly started by the user; no background automatic tracking is performed. Detected obstacles require user confirmation before publication.

3.2.1 Use Case Diagram

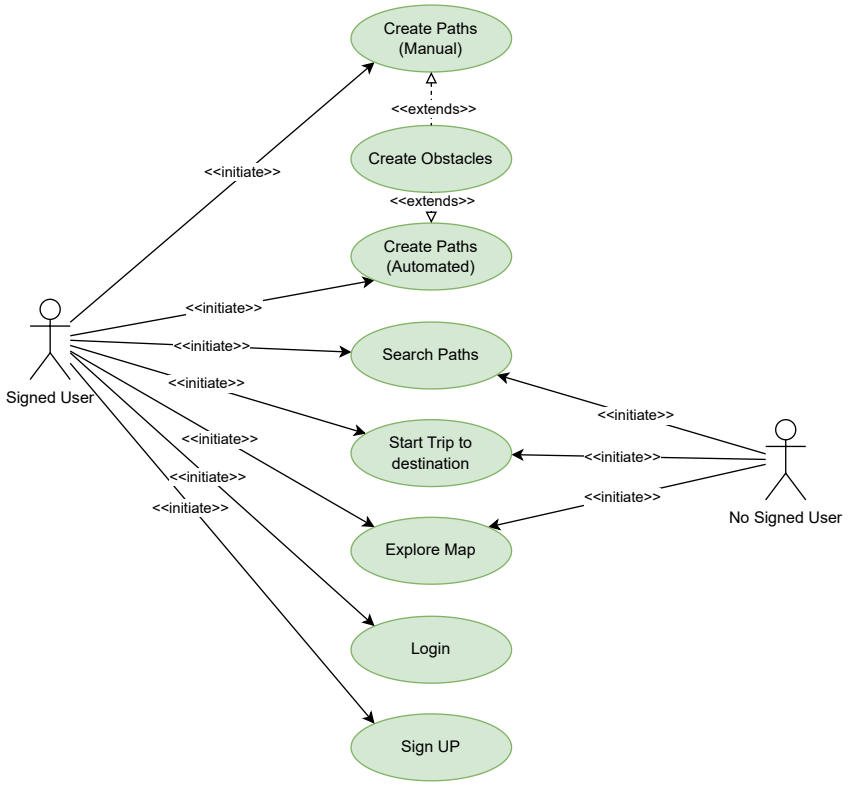


Figure 5: Connection between user categories and their associated use cases.

3.2.2 Use Cases

[UC1] User Registration

Name	User Registration
Actors	Unregistered User
Entry Condition	User opens the BBP client
Event Flow	(a) The User selects the 'Sign Up' option (b) The System asks the User to choose a registration method (either email/password or SSO) (c) The User provides the required information and confirms the registration (d) The System validates the provided data and creates a new User account
Exit Condition	The User is successfully registered and authenticated in the System
Exception	(d) The provided information is invalid or already associated with an existing account

[UC2] User Login

Name	User Login
Actors	Registered User
Entry Condition	User opens the BBP client
Event Flow	(a) The User selects the 'Login' option (b) The System asks the User to choose an authentication method (either email/password, or any of the SSO providers) (c) The User provides the required credentials or completes the authentication with the selected SSO provider (d) The System verifies the provided information, directly or through the selected authentication provider, and authenticates the User
Exit Condition	The User is successfully authenticated and gains access to the System
Exception	(d) The provided credentials are invalid or the selected authentication provider is unavailable

[UC3] Search for Bike Paths between Two Locations

Name	Search for Bike Paths
Actors	Registered User and Unregistered User
Entry Condition	User opens the BBP client
Event Flow	(a) The User enters or selects an origin and a destination (b) The System analyzes available Path data (c) The System computes one or more possible bike Paths between the two locations (d) The System displays the resulting Paths on a map and as a ranked list
Exit Condition	Available bike Paths between the selected locations are displayed
Exception	(c) No viable Path is available between the selected locations

[UC4] Start Navigation and Record Trip

Name	Navigate and Record Trip
Actors	Registered User and Unregistered User
Entry Condition	User has selected a bike Path
Event Flow	(a) The User presses the 'Go' button (b) The System starts a Trip recording session (c) The System provides turn-by-turn navigation instructions (d) The System collects sensor and GPS data during the Trip (e) The User reaches the destination and ends the Trip
Exit Condition	The Trip is recorded. If the User is registered, the Trip is stored in the System.
Exception	(b) Required permissions for sensor or location access are not granted

[UC5] Add Path using Automated Mode

Name	Add Path (Automated Mode)
Actors	Registered User
Entry Condition	User opens the BBP client
Event Flow	(a) The User selects the 'Add Path' button and chooses 'Automated Mode' (b) The System starts a tracking session without navigation instructions (c) The User rides along the Path while the System collects sensor and GPS data (d) The System reconstructs the Path and detects potential obstacles (e) The System presents the collected data to the User for review and confirmation
Exit Condition	The Path data is confirmed and saved by the User
Exception	(d) Sensor data is insufficient to reconstruct the Path correctly

[UC6] Add Path using Manual Mode

Name	Add Path (Manual Mode)
Actors	Registered User
Entry Condition	User opens the BBP client
Event Flow	(a) The User selects the 'Add Path' button and chooses 'Manual Mode' (b) The System displays an interface for manual Path insertions (c) The User selects or searches for the street on the map, or types its name in the search field and picks it from the available options (d) The User inserts Path characteristics and records Obstacles (e) The System shows a summary of the inserted data for confirmation
Exit Condition	The Path is saved as a Draft by the User
Exception	(c) The selected street cannot be validated as a bike Path, or is already registered as such

[UC7] Publish Path Information

Name	Publish Path
Actors	Registered User
Entry Condition	User opens one of his own Draft Paths
Event Flow	(a) The User presses the 'Publish' button (b) The System validates the submitted data (c) The System makes the Path information publicly available
Exit Condition	The Path is published and accessible to all Users
Exception	(b) The submitted data is incomplete or inconsistent

[UC8] Review Trip Statistics

Name	Review Trip Statistics
Actors	Registered User and Unregistered User
Entry Condition	User has completed at least one Trip
Event Flow	(a) The User opens the Trip history section (b) The User selects a recorded Trip (c) The System displays Trip statistics such as distance, duration, speed, the map Path, weather information and recorded health data, if available
Exit Condition	Trip statistics are displayed to the User
Exception	(c) Some trip data (e.g. health metrics, GPS points, or weather) is unavailable

3.2.3 Sequence Diagrams

The following Sequence Diagrams provide a high-level representation of the interactions between the User and the BBP System.

Some Sequence Diagrams describe the execution of multiple Use Cases, as certain system functionalities are typically performed consecutively or alternatively within the same interaction flow.

The diagrams are intentionally abstract and simplified. They focus on the externally observable interactions between the User and the System and do not model internal implementation details of the client or server components, except where a generic description is required to clarify the interaction.

[UC1] User Registration

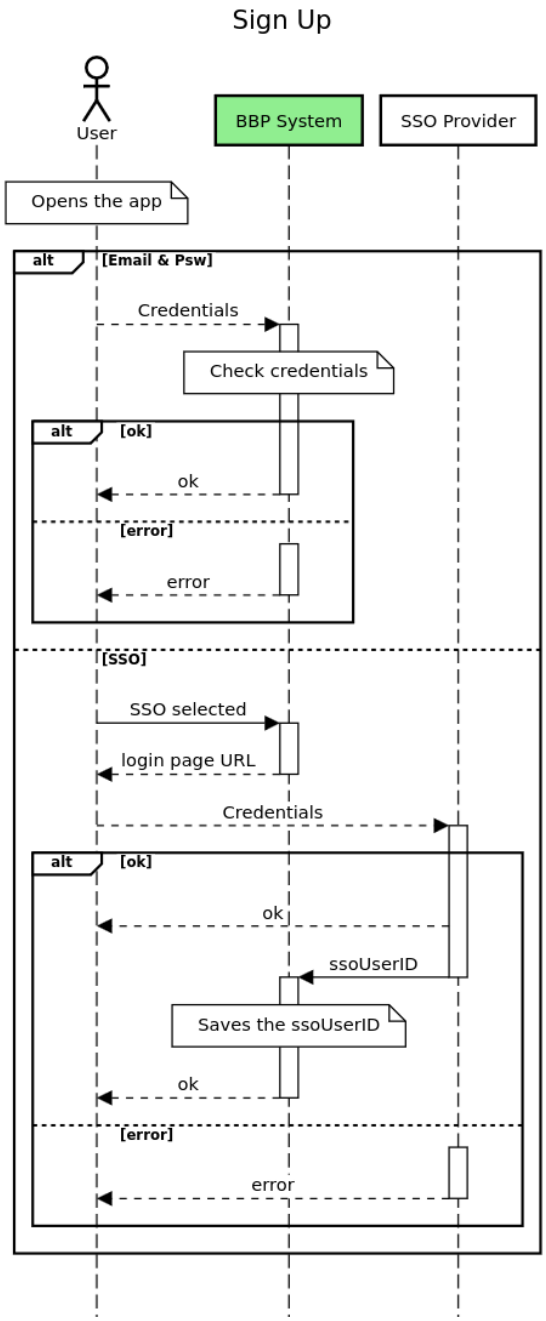


Figure 6: Sequence Diagram of the SignUp process

[UC2] User Login

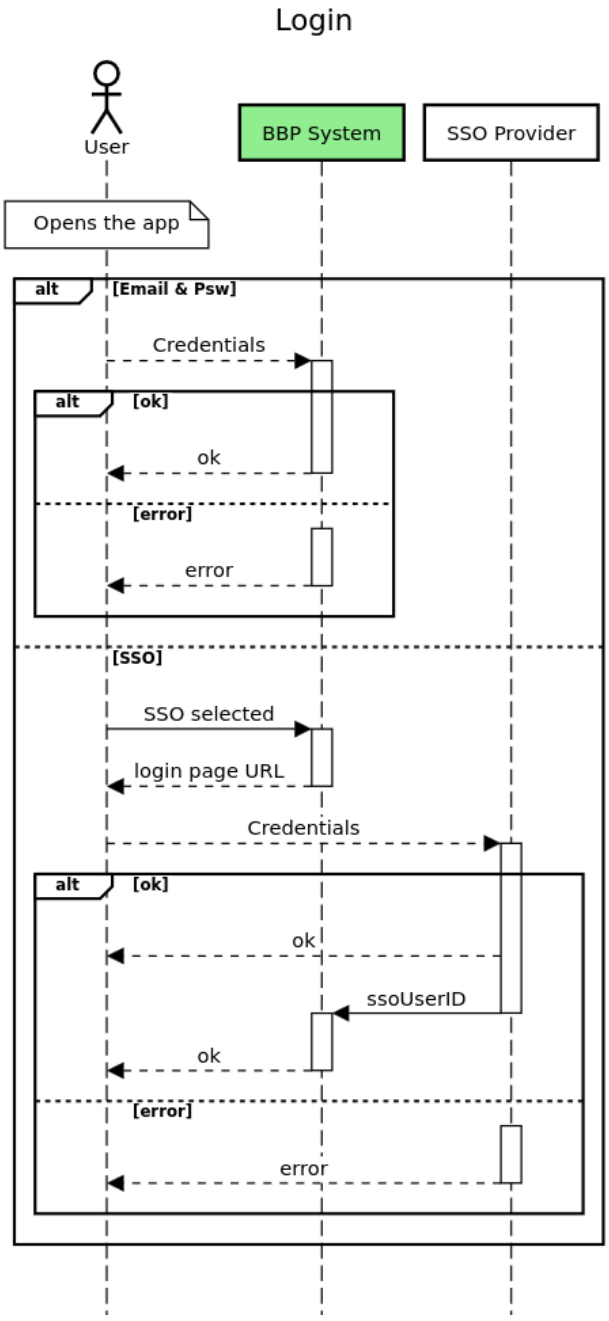


Figure 7: Sequence Diagram of the Login process

[UC3] Search for Bike Paths between Two Locations
[UC4] Start Navigation and Record Trip

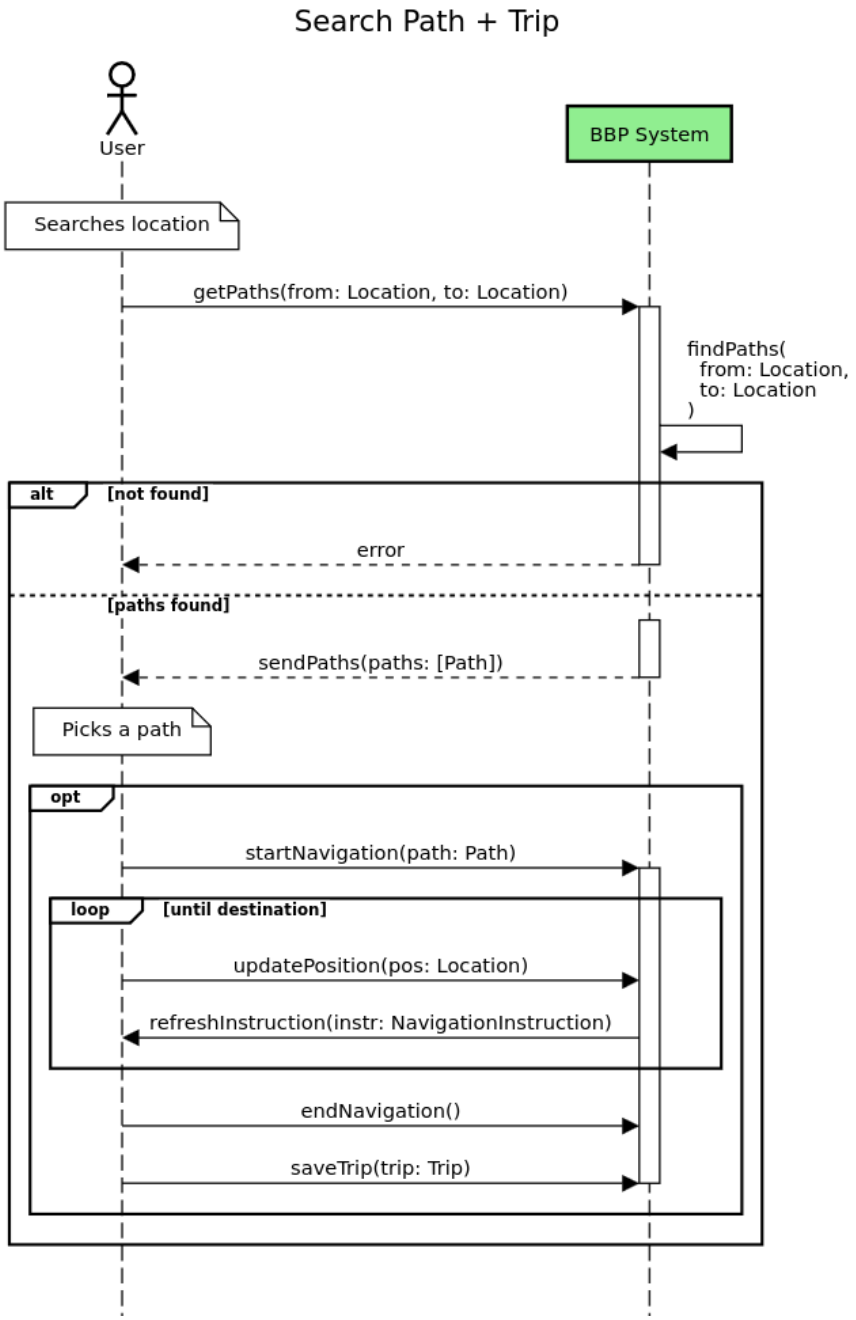


Figure 8: Sequence Diagram of the Search Path process plus optional Navigation

[UC5] Add Path using Automated Mode
[UC6] Add Path using Manual Mode
[UC7] Publish Path Information

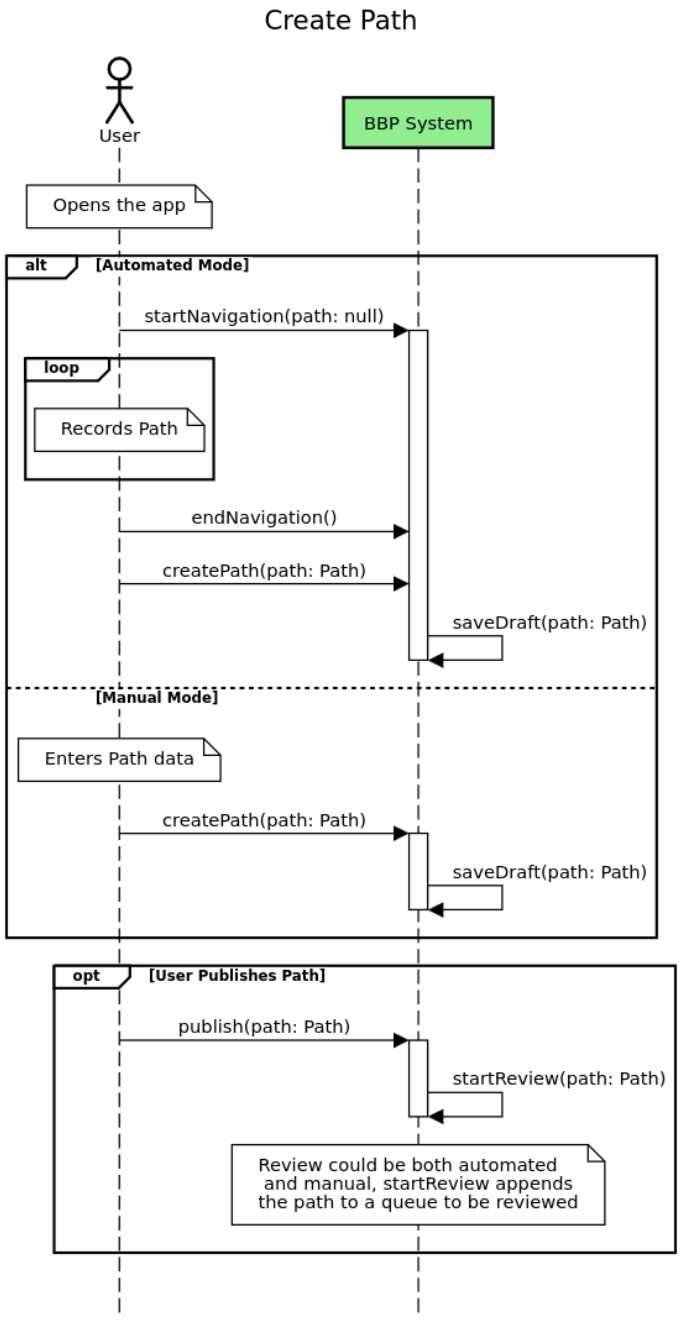


Figure 9: Sequence Diagram of the Path creation process

[**UC8**] **Review Trip Statistics**

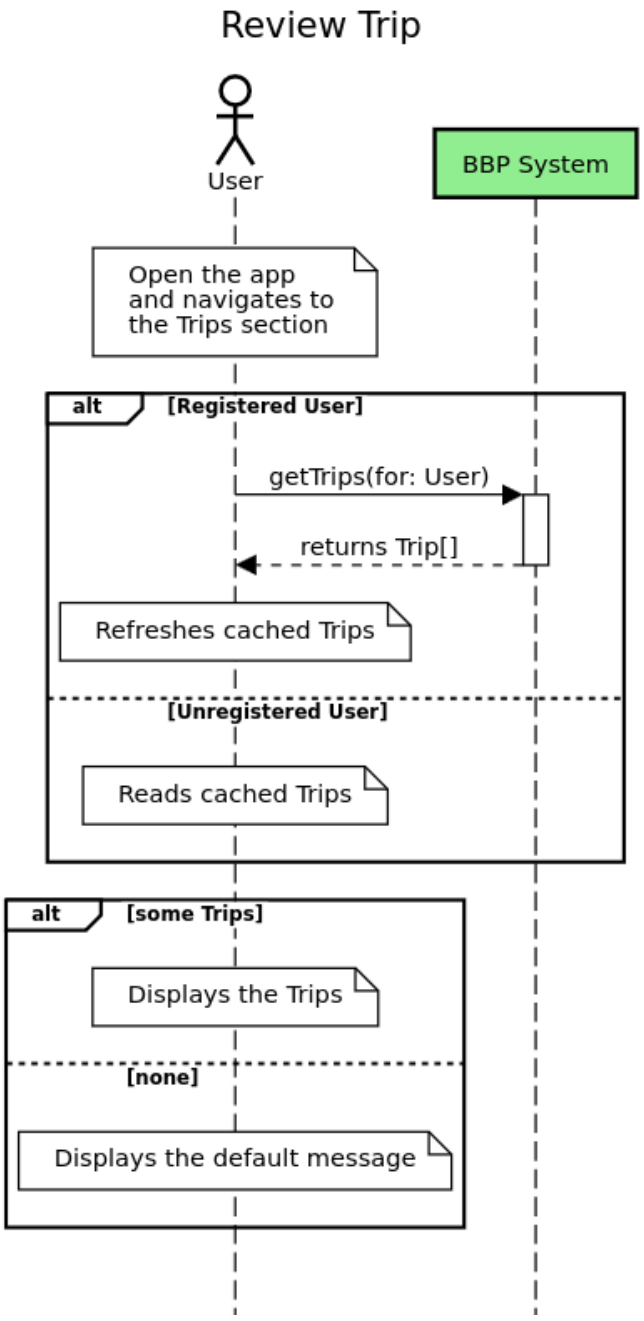


Figure 10: Sequence Diagram of the User reviewing one of his older Trips

3.2.4 Requirement Mapping

G1: Support trip recording The system shall allow registered and unregistered users to record their cycling trips and store them for future consultation.	
R5: Users should be able to start a tracking activity on their mobile device before going for a bike ride. **R6**: The System should keep track of sensors' data. **R7**: The System should be able to reconstruct a path based on the GPS data retrieved in this way.	**D2**: Unregistered users can record Trips, which are stored locally on their device. **D4**: Trip recording sessions are explicitly started by the User. **D5**: The User's mobile device provides access to the required sensors.

G2: Provide meaningful trip insights The system shall analyse recorded trips and present users with relevant statistics, such as distance, duration, and performance metrics.	
R1: The System should provide statistics about each trip, including distance, average speed, and weather information if available. **R9**: The System should present the data to the User at the end of the trip, asking for confirmation or correction of the recorded data.	**D6**: Sensor data collected during a tracking session is sufficiently accurate to derive meaningful statistics. **D7**: Users review and confirm recorded trip data before it is finalized. **D8**: External services providing weather information are available.

G3: Enable sharing of bike path information The system shall allow users to contribute information about bike paths and make it available to the community.	
R2: Users should be able to insert and publish information about bike paths, including their status and the presence of obstacles. **R10**: The System should allow to publish the path's data after being inserted by the User.	**D1**: Users contributing shared Path information are registered and authenticated. **D7**: Users review and confirm Path data before publication. **D9**: Users have a reliable network connection when publishing data.

G4: Support automated path acquisition The system shall allow users to start an automated tracking session during a bike ride, during which the system acquires path data through the device's sensors.	
R5: Users should be able to start a tracking activity on their mobile device. **R6**: The System should keep track of sensors' data. **R7**: The System should be able to reconstruct a path based on GPS data. **R15**: During an Automated Tracking session, the System shall analyze sensor data to reconstruct the path and identify potential obstacles.	**D4**: Automated tracking sessions are explicitly initiated by the User. **D5**: The User's mobile device provides access to the required sensors. **D6**: Sensor data is sufficiently accurate to reconstruct Paths.

G5: Report obstacles and path conditions The system shall identify, store, and display potential obstacles or hazards along bike paths to inform and alert users.	
R8: The System should detect sudden jumps in sensors' data and associate them with potential Obstacles. **R9**: The System should present detected Obstacles to the User for confirmation or correction. **R2**: Users should be able to insert information about Obstacles along Paths.	**D6**: Sensor data allows the identification of potential Obstacles. **D7**: Users confirm or correct detected Obstacles before publication. **D1**: Only authenticated users can contribute shared Obstacle information.

G6: Support manual path insertion The system shall allow users to manually enter information about bike paths, including their status and characteristics.	
R3: Users should be able to insert Paths manually by inputting street names. **R4**: The System should be able to translate street names into GPS coordinates and store Path information. **R2**: Users should be able to insert Path characteristics and status information.	**D1**: Manual Path insertion is performed by registered and authenticated Users. **D8**: External services providing map and geocoding data are available.

G7: Provide path discovery between two locations The system shall help users find all available bike paths connecting a given origin and destination.	
R11: Users should be able to query for Paths given an origin and a destination. **R12**: The System should visualize available Paths on a map. **R13**: The System should compute a score based on the effectiveness of a Path. **R14**: The System should rank Paths based on their score.	**D2**: Both registered and unregistered Users can search for Paths. **D8**: External services providing map data and rendering are available. **D9**: Users have a reliable network connection when querying Paths.

3.3 Performance Requirements

Real-Time Sensor Data Processing

The mobile app shall:

- Process accelerometer and gyroscope signals in real time.
- Apply filtering techniques, such as Kalman or low-pass filters.
- Identify candidate potholes with low latency (recommended < 100 ms).

Internet Access

The mobile app shall:

- Be able to send trip data to the backend.
- Request weather information from external services.
- Fetch map tiles for visualization.

Additionally:

- Offline trips must be supported, with data stored locally and uploaded later.

Map Rendering Performance

The mobile device shall:

- Display maps smoothly, using Google Maps, Mapbox, or a custom solution.
- Handle multiple possible paths simultaneously.
- Support polylines and markers for obstacles.

Typical hardware capable of this includes Android or iPhone devices from the last 5–6 years.

3.4 Design Constraints

3.4.1 Standards Compliance

Conformity Standards

GDPR: The system manages users' personal and administrative data; therefore, it must comply with the General Data Protection Regulation in accordance with European guidelines.

ISO/IEC 27001: The system must adopt a structured and comprehensive information security management approach to protect sensitive information and adapt to evolving security risks.

ISO 9241: The system must comply with usability and human–system interaction principles to ensure that navigation and turn-by-turn instructions are presented clearly and safely, minimizing user distraction during use.

Development Standards

ISO/IEC 12207: The system must comply with the software life cycle management standard by applying best practices throughout all main processes leading to its realization.

3.4.2 Hardware Limitations

Minimum Requirements Users will access the system via a native mobile application developed for their OS. The client performs minimal local computation, primarily retrieving data and rendering maps. The devices must run a supported OS version:

- Android 9 or newer
- iOS 18 or newer

Based on these specifications and comparisons with similar map-based applications, the minimum hardware requirements are:

- Snapdragon 820 or newer (or equivalent)
- Apple A15 or newer
- 4 GB of RAM or more

Automated Tracking Sessions

Due to operating system restrictions, battery consumption considerations, and privacy concerns, BBP does not perform automatic background tracking to detect cycling activity. Instead, Automated Tracking sessions must be explicitly started by the user.

During these sessions, the system can:

- Access GPS and device sensors (accelerometer and gyroscope) to reconstruct paths.
- Identify potential obstacles along the route.

Limiting tracking to user-initiated sessions ensures:

- Efficient resource usage
- Compliance with OS background execution policies
- User control over personal data collection

3.5 Software System Attributes

3.5.1 Reliability

The BBP system is designed to provide dependable operation and ensure the integrity of user and path data. Key reliability considerations include:

Uptime and Availability

The system shall be accessible to users 24/7 for core functionalities such as trip recording, map viewing, and path searches. Planned maintenance windows shall be minimal and communicated in advance to users.

Data Integrity

- Recorded trips, path data, and obstacle information shall not be lost under normal operation.
- Offline trip data must be stored locally on the user's device and uploaded to the backend when a network connection is available.
- Updates to shared path or obstacle data shall be atomic to prevent partial or inconsistent information being stored.

Fault Tolerance and Error Handling

- The mobile client shall handle network interruptions gracefully, retrying uploads or storing data offline when necessary.
- Sensor failures (GPS, accelerometer, gyroscope) shall not crash the application; users shall be notified if data collection is incomplete.
- Backend services shall handle concurrent requests safely to prevent data corruption or conflicts.
- The system shall handle location coordinates that fall outside registered paths gracefully. This includes:
  - Avoiding crashes or application freezes when GPS points are outside mapped areas.

- Recording these coordinates appropriately (e.g., as part of the user's trip, flagged as "unmapped").
- Providing visual or textual feedback to the user if necessary, without interrupting navigation or trip recording.

Consistency

- Shared path and obstacle information shall remain consistent across users, even when multiple users contribute or update data simultaneously.
- Conflicting updates shall be prevented or resolved through review mechanisms before final publication.

Performance Considerations While primarily a functional concern, responsiveness of tracking, map rendering, and path searches affects perceived reliability. The system shall ensure these operations are performed promptly to maintain a smooth and reliable user experience.

3.5.2 Availability

The S&C platform should deliver a highly available service that can be accessed quickly and efficiently by its users. Core functionalities of the system should incorporate redundancy and failover mechanisms to maintain service continuity. A significant increase in users could cause the system to scale rapidly, adding complexity and potentially reducing service availability. This aspect should be carefully considered during the initial design phase. Generally, the design should guarantee at least 99,9% of uptime. It should also implement circuit breakers to prevent cascading failures, and administrators should be able to respond quickly and effectively to issues

3.5.3 Security

The system stores sensitive data like passwords and health metrics, therefore the system must guarantee privacy and discretion.

All stored data must be protected with encryption, and data in transit must also be encrypted to ensure security. The system may be particularly susceptible to Injection and Denial of Service attacks, hence it should incorporate measures to prevent or reduce their impact.

3.5.4 Maintainability

The components should be designed to be easily scalable, maintainable, reusable and testable.

To achieve this, the system should implement modular design principles and ideally adopt a microservice architecture.

The code should be readable, maintainable and fully documented. Documentation should include detailed and well analyzed API definitions, a comprehensive description of the system architecture and should define the testing strategies employed.

3.5.5 Portability

The system should be compatible with most smartphones still in use, as we discussed previously (paragraph 3.4.2).

4 Formal analysis using Alloy

```
// --- Signatures ---

// -- Users
sig Email {}

sig User {
    email: one Email,
}

// -- Coordinates and Obstacles
sig Coordinates {}

sig Obstacle {
    coord: one Coordinates
}

// -- Paths
sig Path {
    start: one Coordinates,
    end: one Coordinates,
    points: set Coordinates,
    registeredBy: lone User,
    obstacles: set Obstacle,
} {
    #points >= 2
    start != end
    start in points
    end in points
}

// -- Trip Modes
abstract sig TripMode {}
one sig ManualMode, AutomatedMode extends TripMode {}

// -- Trips
sig Trip {
    rider: one User,
    route: seq Coordinates,
    startingTime: one Int,
    endingTime: one Int,
    mode: one TripMode,
} {
```

```
    endingTime > startingTime
    #route >= 2
}

// --- Facts (Structural Constraints) ---

fact UniqueEmails {
    all u1, u2: User | u1 != u2 implies u1.email != u2.email
}

fact NoPathInsideAnother {
    all p1, p2: Path |
        p1 != p2 implies not (p1.points in p2.points and
            p1.points != p2.points)
}

fact ObstaclesInsidePath {
    all p: Path, o: Obstacle | o in p.obstacles implies o.coord
        in p.points
}

fact UniqueObstacleCoordinates {
    all o1, o2: Obstacle | o1 != o2 implies o1.coord != o2.coord
}

fact NoTripOverlapForUser {
    all t1, t2: Trip |
        t1 != t2 and t1.rider = t2.rider implies
            t1.endingTime <= t2.startingTime or t2.endingTime
                <= t1.startingTime
}

fact TripRouteUniqueCoordinates {
    all t: Trip |
        all i, j: Int |
            i >= 0 and i < #t.route and
            j >= 0 and j < #t.route and
            i != j implies t.route[i] != t.route[j]
}

// --- Functions ---

fun pathsUsed[t: Trip]: set Path {
    { p: Path | some i: Int | i >= 0 and i < #t.route and
        t.route[i] in p.points }
}

fun pathScore[p: Path]: Int {
    #p.points - #p.obstacles
}

// --- Assertions (Functional Checks) ---

// 1. Every trip covers at least one path
assert TripCoversPath {
    all t: Trip | some p: Path | some i: Int | i >= 0 and i <
        #t.route and t.route[i] in p.points
```

```
}

// 2. Trips detect obstacles along their route
assert TripHitsObstacle {
  all t: Trip |
      all o: Obstacle |
          (some p: Path | p.registeredBy = t.rider and o in
              p.obstacles) implies
              some i: Int | i >= 0 and i < #t.route and
                  t.route[i] = o.coord
}

// 3. Trip chooses best path in its route
assert TripChoosesBestPath {
    all t: Trip | some p: Path | p in pathsUsed[t] and all q:
        Path | q in pathsUsed[t] implies pathScore[p] >=
        pathScore[q]
}

// 4. Paths have unique start/end points
assert UniqueStartEndPaths {
    all p1, p2: Path | p1 != p2 implies p1.start != p2.start or
        p1.end != p2.end
}

// --- Example Generation Predicates ---

pred displayData {
    some t1, t2: Trip | t1 != t2
        some p1, p2: Path | p1 != p2
        some o1, o2: Obstacle | o1 != o2
        some u1, u2: User | u1 != u2
}

// --- Commands ---

// Run a scenario with sample bounds
run displayData for
    3 User, 3 Path, 10 Trip, 50 Obstacle, 3 Email, 2 TripMode,
        10 Coordinates

// Check functional requirements formally
check TripCoversPath for 3 User, 3 Path, 100 Coordinates, 10
    Trip, 50 Obstacle, 3 Email, 2 TripMode
check TripHitsObstacle for 3 User, 3 Path, 100 Coordinates, 10
    Trip, 50 Obstacle, 3 Email, 2 TripMode
check TripChoosesBestPath for 3 User, 3 Path, 100 Coordinates,
    10 Trip, 50 Obstacle, 3 Email, 2 TripMode
check UniqueStartEndPaths for 3 User, 3 Path, 100 Coordinates,
    10 Trip, 50 Obstacle, 3 Email
```

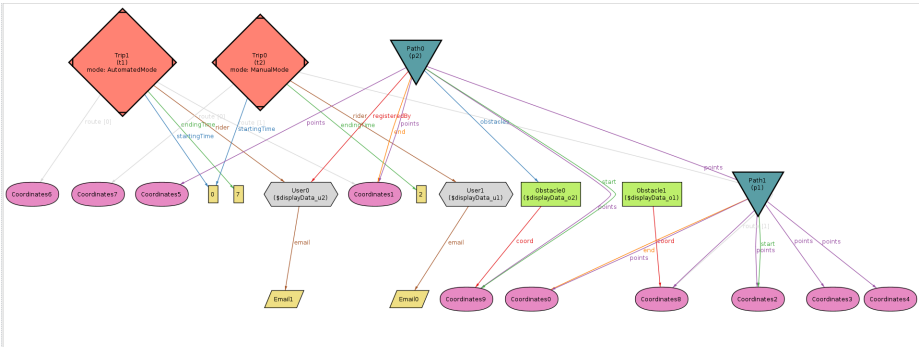


Figure 11: Visualization of Alloy's model found after simulation

5 Effort spent

	Chapter 1	Chapter 2	Chapter 3	Chapter 4
Rodigari M.	4	7	6	5
Summa C.	4	5	7	5

6 References

- Software Engineering II course slides and assignment specification.
- Overleaf, used as the collaborative LaTeX authoring environment.
- Google Docs, used for early collaborative drafting.
- Figma, used for the creation of UI wireframes and mockups.
- draw.io, used for UML and domain diagrams.
- sequencediagrams.org, used for modeling sequence diagrams.
- Alloy Analyzer, used for formal specification and verification.