POLITECNICO
MILANO 1863

COMPUTER SCIENCE AND ENGINEERING
SOFTWARE ENGINEERING II
2025 - 2026

RASD

Requirement Analysis and Specification Document

Best Bike Paths

Authors:
Leonardo Guglielmi, Francesco Lo Conte

Version:
1.0
(December 22, 2025)

Contents

1 Introduction

1.1 Purpose

The growing interest in cycling, whether as a recreational activity, a means of transportation, or a sport, brings with it a significant challenge: finding routes that are not only efficient, but also safe and well-maintained. Cyclists often lack reliable and up-to-date information on trail conditions, such as the presence of potholes, obstacles, or roads with little traffic. At the same time, many cyclists meticulously log their trips to monitor their performance, collecting valuable data that, however, remains siloed. This creates a gap where vital community knowledge about trail quality is not easily shared or accessible. "Best Bike Paths" (BBP) aims to provide a solution. Commissioned by a cyclists' association, BBP will be a software system designed to create and manage a community-driven inventory of cycling routes. The platform will help bridge this information gap by allowing registered users to track their trips while simultaneously submitting detailed information on the condition of their routes. Other users, registered or not, will then be able to use this collective data to find and display the best possible cycling routes between two points, ranked by a quality score.

1.1.1 Goals

- **G1:** A registered user wants to track their personal cycling activities and related performance statistics.
- **G2:** A registered user wants to contribute to the community inventory by sharing reliable information on the condition of the trails (e.g. quality, obstacles, potholes).
- **G3:** Any user (registered or not) wants to find and view the best cycling route between an origin and a destination, based on up-to-date and relevant data.
- **G4:** The cycling association aims to provide the community with a tool to create, consult, and maintain a reliable and centralized inventory of cycling routes.

1.2 Scope

The Best Bike Paths (BBP) system is designed to create, manage, and distribute a community-driven inventory of cycling paths, acting as a mediator between the physical conditions of the road network and the cyclists' need for safety. The scope of the application covers the entire lifecycle of path data: from its collection via mobile devices to its aggregation into a quality metric (Path Score) utilized for routing.

For this project, the following users interacting with the system have been identified:

- **Registered User**
- **User**

A Registered User will be able to use the application to log and store their trips, tracking their cycling activities and related statistics. When available, this data can be enriched with weather information retrieved from external services. Furthermore, this user is the primary contributor to the inventory. They can enter route information in two ways:

1. In **manual mode**, by actively specifying the route status (e.g., optimal, requires maintenance) and the presence of obstacles (e.g., potholes).
2. In **automatic mode**, by allowing the app to acquire data from GPS and mobile device sensors while cycling, in order to automatically detect potential problems such as potholes.

For automatically collected data, the system will ask the user to confirm or correct the information before making it available to the community. Once confirmed or manually entered, this information becomes public.

Any user, whether registered or not, can benefit from the collected information. The user can specify a starting point and a destination. The system leverages third-party mapping services to identify valid physical routes and overlays them with BBP's inventory data. If a route is present in the inventory, it is displayed with its Path Score; otherwise, it is displayed without it. If multiple routes exist, BBP will present them based on this score, calculated based on the route's status derived from user-confirmed data.

1.2.1 World phenomena

- **WP1:** The user fills up the registration form with its personal information.
- **WP2:** The user inserts an invalid email address in the registration form.
- **WP3:** The user inserts an email address in the registration form already used for another account.
- **WP4:** The registered user inserts the account credentials into the login form.
- **WP5:** The registered user inserts the wrong credentials into the login form.
- **WP6:** The registered user modifies an attribute.
- **WP7:** The registered user forgots the password.
- **WP8:** The registered user inserts the new password.
- **WP9:** The user searches for a path.

- **WP10:** The user inserts starting and destination points.
- **WP11:** The user stops to cycling.
- **WP12:** The registered user evaluates a path.
- **WP13:** The registered user's personal device samples sensors data.
- **WP14:** The external weather service is unavailable.
- **WP15:** The external mapping system is unavailable.
- **WP16:** The registered user encounters a problem along a path.
- **WP17:** The registered user inserts a route problem information
- **WP18:** The registered user encounters a fixup problem.
- **WP19:** The registered user changes the problem status to "Resolved".
- **WP21:** The registered user confirms automatically detected problems.
- **WP22:** The registered user denies automatically detected problems

1.2.2 Shared phenomena

World controlled

- **SP_WC1:** The user registers itself.
- **SP_WC2:** The user submits to the system the registration form.
- **SP_WC3:** The registered user sends the login form to the system.
- **SP_WC4:** The registered user sends the attribute modification to the system.
- **SP_WC5:** The registered user submits the new password.
- **SP_WC6:** The registered user confirms to the system the account deletion.
- **SP_WC7:** The user starts the search for a path in the app.
- **SP_WC8:** The user submits to the system the starting and destination points.
- **SP_WC9:** The user starts a ride.
- **SP_WC10:** The external mapping service sends to the system a list of paths
- **SP_WC11:** The user stops the ongoing ride.
- **SP_WC12:** The user resumes the stopped ride.
- **SP_WC13:** The user terminates the ride.

- **SP_WC14:** The registered user submits the path evaluation.
- **SP_WC15:** The registered user enables to automatically collect data during an activity.
- **SP_WC16:** The registered user's personal device sends to the system sampled data upon activity completition.
- **SP_WC17:** The external weather service sends to the system weather conditions for a given path and datetime.
- **SP_WC18:** The registered user submits a route problem information.
- **SP_WC19:** The registered user reports the path fixup.
- **SP_WC20:** The registered user submits detected problem confirmations.
- **SP_WC21:** The registered user submits detected problems denials.
- **SP_WC22:** The registered user sends to the user a request for its activity history.
- **SP_WC23:** The registered user asks to the system to delete an activity.

Machine controlled

- **SP_MC1:** The system sends to the user the registration form.
- **SP_MC2:** The system sends to the registered user the login form.
- **SP_MC3:** The system asks to the registered user for which account it must reset the password
- **SP_MC4:** The system asks to the user confirmation about account deletion.
- **SP_MC5:** The system shows to the user a list of paths.
- **SP_MC6:** The system asks to an external mapping service for a list of paths
- **SP_MC7:** The system asks to the registered user to evaluate a path travelled during a completed activity.
- **SP_MC8:** The system asks for weather conditions to an external weather service.
- **SP_MC9:** The system shows to the registered user the activity summary with performances.
- **SP_MC10:** The system shows to the registered user the activity summary with performances and weather conditions.
- **SP_MC11:** The system asks confirmation to the registered user about detected problem during an activity.

- **SP_MC12:** The system shows to the registered user its activity history.

1.3 Definitions, Acronyms, Abbreviations

This section contains the definitions for people that may not know what a specific concept is, acronyms and abbreviations used throughout the document.

1.3.1 Definitions

- **Bike Path:** a route deemed suitable for cycling. This includes paths with a proper bike track or roads where cars are rare and speed limits are compatible with the average speed of a bike.
- **Ride:** a cycling trip made by a not-registered user.
- **Activity:** a personal record of a registered user's cycling trip, stored by the system to track performance metrics like distance and speed.
- **Publishable Information:** data about a bike path (e.g., status, obstacles) that a registered user has either entered making it available to the wider community.
- **Path Score:** a metric computed by BBP to rank route options. It is based on the status of the path and its effectiveness in getting the user from their origin to their destination.
- **Obstacle:** any significant element or condition on a cycle path that may represent a danger or hindrance to the cyclist (e.g. pothole).

1.3.2 Acronyms

- **BBP:** Best Bike Paths.
- **GPS:** Global Positioning System.
- **API:** Application Programming Interface.

1.3.3 Abbreviations

- **G*:** Goal.
- **WP*:** World Phenomenon.
- **SP*:** Shared Phenomenon.
- **R*:** Requirement.
- **UC*:** Use Case.
- **D*:** Domain Assumption.

Note: asterisks are intended as a replacement for the number.

1.4 Revision history

- **Version 1.0 (17/11/2025)**

1.5 Reference documents

This document is based on the following materials:

- The specification of the RASD and DD assignment of the Software Engineering II course a.y. 2025/26.
- Course slides shared on WeBeep.
- Past Requirement Analysis and Specification Documents.

1.6 Document structure

1. **Introduction:** a brief description of the project. It contains the main goals and objectives that the final system wants to achieve.
2. **Overall description:** this section is a high-level representation of the system and of the interactions of the system with the other actors.
3. **Specific requirements:** a detailed list of all the requirements needed for the system to achieve the goals. It contains valuable information for developers.
4. **Formal analysis using Alloy:** a formal description of the model of the system with Alloy.
5. **Effort spent:** the time spent on each section of the document, for each member of the group.
6. **References:** reference to documents or tools used for writing this document.

2 Overhall description

2.1 Product perspective

2.1.1 Scenarios

[SC1] Registering a new account

User "Zoe" has just downloaded the BBP app in order to monitor her activities on the bicycle, and wants to create a profile. So she creates an account by entering her name, surname, email, birth date, gender, and accepting the privacy policy. Once her information is verified, she receives an email to confirm her mail address. She confirms it, and the account is succesfully created.

[SC2] Logging into account

Registered user "Monica" wants to enter in the BBP app with her account. She opens the BBP app, enters her email and password on the login screen, and submits that information. Then the account information is displayed to her, and she can use all app functionalities.

[SC3] Updating account information

Registered user "Giulio" noticed that he had selected the wrong birth date during account creation. He decides to fix it: he opens the BBP app, goes to the Profile section, and opens the edit screen. On this screen he changes the birth date with the correct one, he confirms the update and the app now displays the correct date.

[SC4] Resetting account password

Registered user "Vittorio" changed mobile device and installed the BBP app, but when he tried to log in, he realized he had forgotten his password. Then from the login page he clicks on the link to reset the password, which takes him to a form in which he enters the account email. After a few seconds, he receives an email which contains a link to reset the password. He opens it, fills out the form with the new password and submits it. Then he tries to log-in again in the app with the new password, successfully logging in.

[SC5] Account deletion

After months of inactivity, registered user "Mirko" decides he no longer wants to cycle and deletes his BBP account. He opens the BBP app, opens the Account section and from the options he selects that one to delete the account. He confirms to the app that he wants to delete his account, he receives an email containing a link to confirm his choice a second time. He opens it, reads the disclaimer and confirms that he wants to delete the account. After a few hours, he receives another email confirming account deletion.

[SC6] Intelligent route planning with successful match (Causal user)

Tourist "Diana" wants to explore the city by bike but is concerned about traffic and poor roads. She accesses the BBP website without logging in and enters "Hotel Plaza" as the origin and "Museo della Scienza" as the destination, receiving two possible paths in response. Diana notices that the shortest route (3 km) has a low "Path Score", with several "Pothole" icons along the way. The alternative, slightly longer route (3.5 km) has an excellent "Path Score" and it's marked as having excellent conditions. Diana chooses the green route, starts the trip and follows the instructions.

[SC7] Intelligent route planning with unsuccessful match (Casual user)

Casual user "Mirko" wants to find a route to reach out his friends by bicycle. He opens the BBP app and proceedes as a guest, then searches for a path but the app doesn't find a match. He selects the option to create a new path, selects one of the proposed alternatives and starts the trip.

[SC8] Intelligent route planning with successful match (Registred user)

Registered user "Giorgio" is planning his daily cycling training ride. He opens the BBP app, logs in and searches for bike paths with a starting point near his home and a length of 30 km. He receives three paths: the first path has a high "Path Score" but that passes by his ex-wife's house; the second path has a decent "Path score" with no problem marked; the third one has low "Path Score" with several potholes marked on the map. Given these options, he chooses the second one and starts the activity.

[SC9] Intelligent route planning with unsuccessful match (Registered user)

Registered user "Sara" wants to reach her hometown pharmacy by bike. She opens the BBP app, searches for a path from her home to the pharmacy but no match is found. She then selects the option to create a new path, and starts the activity following one of the suggested paths.

[SC10] Starting ride trip

User "Marco" has selected the path he wants to do by bike, starts the trip by selecting the relative option. By doing so, he's able to see the path he should follow and his position in real-time.

[SC11] Stopping and resuming ride trip

User "Tony" started an activity, but in the middle of it, he encounters his old friend "Lorenzo" and stops for a chat. He opens the activity screen and pauses it. Later, when he finished with his friends, he resumes the activity.

[SC12] Automatic activity monitoring and trip data enrichment

Registered user "Alessandro" is preparing for his weekly training session. He wants to track his performance, including its correlation with weather conditions. He starts recording his activity allowing automatic collection of data for both check path and weather conditions tracking. Once he has finished his trip, he stops the recording and after a little bit he views the trip summary on the app: the path map; the total distance traveled; the average, maximum and minimum speed; maximum, average and minimum altitude; and the weather conditions.

[SC13] Route score assignment

Registered user "Anna" started and completed her activity with the bicycle. Upon completion, she receives from the app the summary of the activity and a small form to score the route. She selects the score she wants to give and submits it.

[SC14] Automatic path information update

Registered user "Carlo" started an activity with automatic monitoring. Almost at the end of the ride, he rode over a pothole. When he arrives at work he checks the BBP app to see whether the pothole has been detected or not. He notices that there two potholes were detected: one approximately in the middle of the path, and another one near his work building. Since he didn't encountered a pothole in the middle of the path, he selects it and discards it. He then selects the pothole near his work building, confirms it and adds an optional note to be more detailed.

[SC15] Manual path information update 1

Registered user "Bianca" is riding a popular bike path when she notices that a stretch, previously marked as "Optimal", is now blocked by unreported construction. She decides to alert the community: she stops and reports the problem on the BBP app by specifying the bike path, the type of problem, the problem position. She also adds an optional textual note for more detailed, then submits the report, receiving an acknowledgement few seconds before.

[SC16] Manual path information update 2

Registered user "Edoardo" is riding along a path where a pothole had been reported the previous week. He notices that the pothole has been fixed, so he selects the pothole icon on the map and switches its status to Resolved. After

a few hours, he decides to check if the icon on the map has been removed, and finds out that the pothole mark disappeared.

[SC17] Historical performance analysis

Registered User "Alessandra", after months of using BBP, wants to analyze her performance progress. She opens the Trip History app section and looks at the list of all her saved trips. She filters the list by "Last month" and looks at the aggregated graph showing her average speed and the total distance traveled for that period. Then she searches for a specific activity she completed two months ago to check for improvements.

[SC18] Trip deletion

Registered user "Caterina" has an accident during her last recorded trip, and therefore the recorded performances are inaccurate. She opens the BBP app, goes to the Activity History section and searches for the trip she wants to delete. Once she finds it, she selects the option to delete it, she confirms that she wants to do that and then the trip is deleted.

2.1.2 Domain Class Diagram

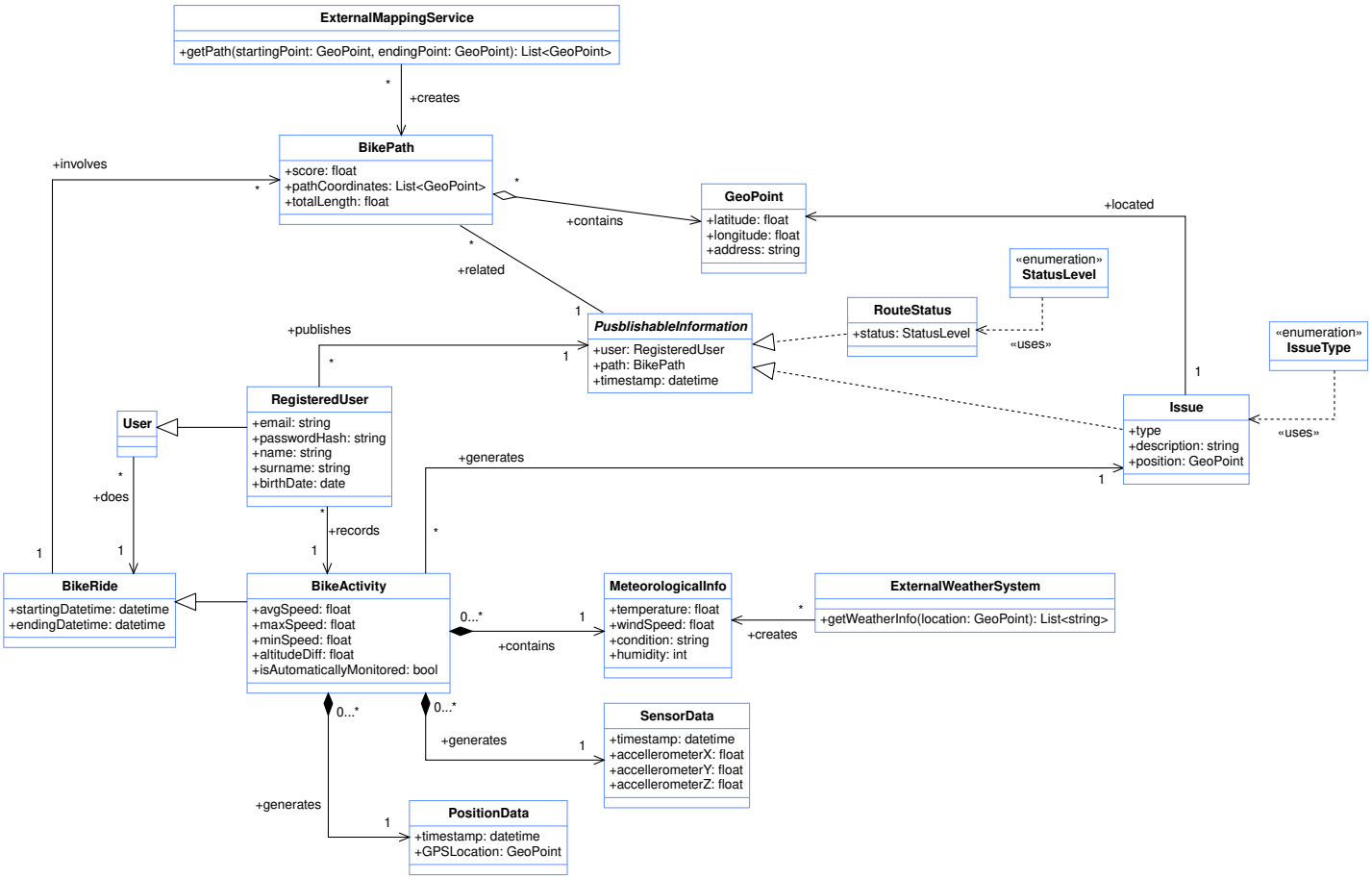


Figure 1: Domain Class Diagram of the BBP system

Figure 1 shows the domain class diagram. The following points outline some choices made while modeling the domain:

- **User & Trip generalizations:** two superclasses were introduced to standardize and encapsulate the core functionalities shared by all elements participating the generalization. By contrast, their specialization classes are responsible for capturing and highlighting the specific, differentiated relationships that these elements have with the rest of the domain components.

- **Publishable Info abstraction:** the introduction of this abstraction was motivated by the intent to make the defined set of publishable information highly adaptable, in order to maximize flexibility for future extensions about publishable informations.

2.1.3 State Diagrams

Activity Lifecycle

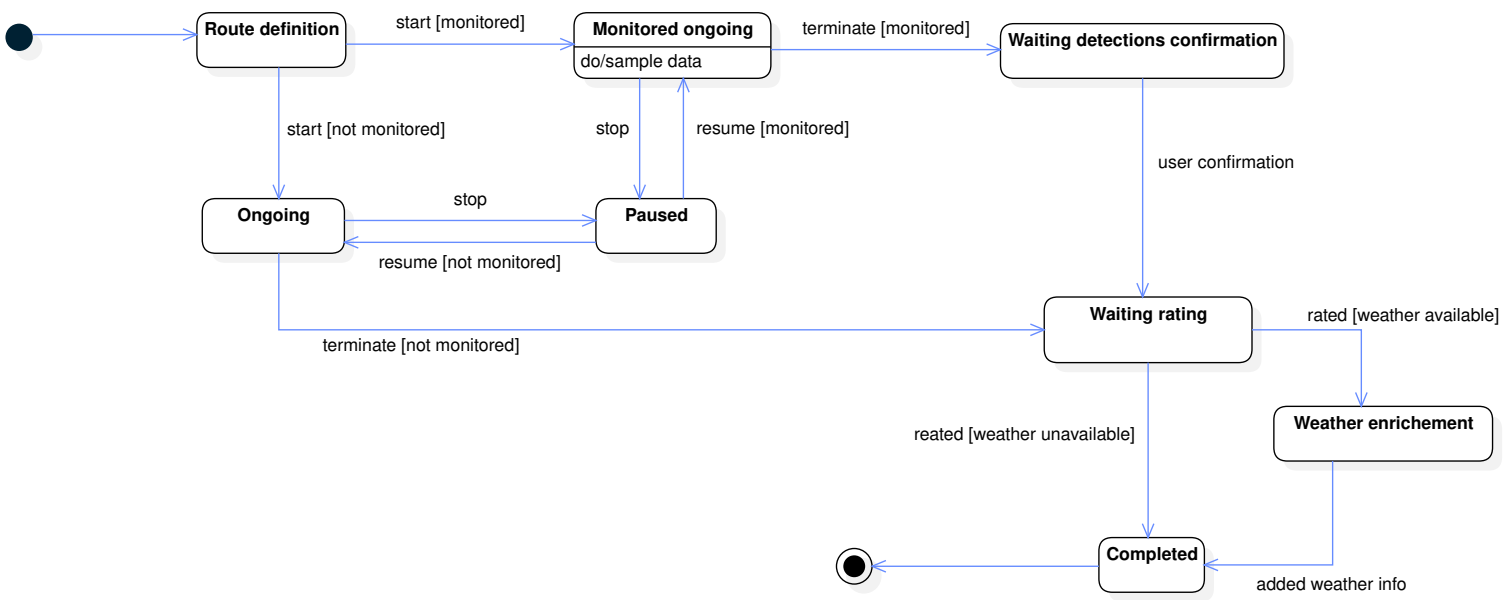


Figure 2: State Diagram of the Lifecycle of a Trip in the BBP System

The diagram in Figure 2 models the complete lifecycle of an Activity. The process begins with the `Route Definition` state, in which the route to be traveled is established. Subsequently, depending on whether the user decides to automatically monitor the trip, the Activity transitions into either the `Monitored ongoing` state or the `Ongoing` state. From either ongoing states, the **Activity** can enter the `Paused` state if the user performs the *stop* action, and it returns to the previous ongoing state if the user *resumes* it. Within the `Monitored ongoing` state, the system also performs the *sampling* of sensor data in order to detect anomalies along the path. When the user chooses to *terminate* the activity, the transition depends on the current state:

- If the Activity is in the `Ongoing` state, it transitions directly to the `Waiting rating` state.
- If the **Activity** is in the `Monitored ongoing` state, it first passes through the `Waiting detections` confirmation state, where the system waits for the user to *confirm/discard* automatically detected issues. After receiving user confirmations, it then transitions to the `Waiting rating` state.

In the `Waiting rating` state, the Activity waits a score for the path from the user. Following this input, the Activity may optionally enter the `Weather enrichmenten` state, depending on whether the External Weather Service is available or not; otherwise, it will transition directly to the `Completed` state.

Issue Lifecycle

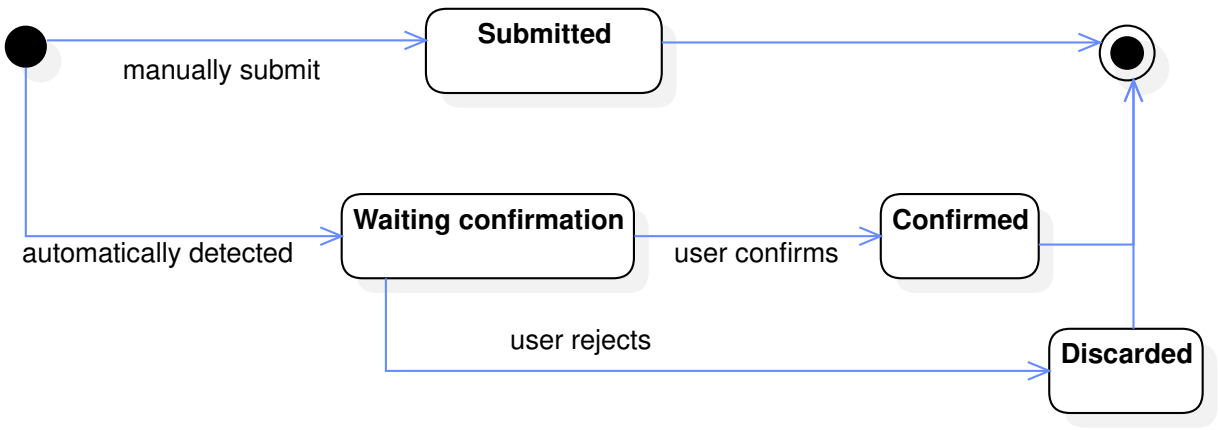


Figure 3: Data lifecycle state diagram in BBP system

The diagram in Figure 17 models the complete Issue lifecycle, from its origin to its final state. The process rigorously distinguishes the issue based on its source to direct it to the correct validation path. The flow forks immediately from the initial state:

- **Manual report**: Once submitted, the report directly becomes an entity on its own by entering in the `Submitted` state.
- **Automatic report**: Upon automatic detection, the report enters the `waiting confirmation` state, in which it waits for the user to *confirm* or *discard* it. Consequently, it transitions into the corresponding `Confirmed` or `Discarded` state. If the issue is `Discarded` then there would be no further evolution. If the issue is `Confirmed` then it transitions in the `Submitted` state, ending up in the same final state as manual reported issues.

2.2 Product functions

Sign up & Login

This feature is the entry point for any user wishing to actively contribute to the inventory. A visitor can register by providing their information and credentials, and once the account is created the user can log in to access their reserved area, view their travel history, and use the tracking features. Without authentication, the user remains in "read-only" mode, without access to all the features expected of a registered user.

User Profile Management

Registered users have access to a dedicated section for managing their personal data. Here they can update their contact information and personal details, change their password, or delete their account. These actions ensure that the user maintains full control over their digital identity within the system.

Trip Recording

This is a core feature available exclusively to registered users. Users can start a recording session at the beginning of their activity. During the trip, the system tracks their geographic location via GPS in real time. Users have the flexibility to pause and resume recording (for example, during a rest stop). Upon completion, the trip is stored.

Statistics Calculation and Data Enrichment

Upon completion of a trip, the system processes the raw data to provide detailed statistics, such as total distance traveled and average speed. Additionally, BBP automatically queries external services, if available, to retrieve weather information (temperature, wind, and weather conditions) for the area and time of the trip. This data is integrated into the trip record, providing the user with richer context for analyzing their performance.

Manual Data Entry

Registered users can actively contribute to the quality of the inventory by entering manual reports. Through a dedicated interface, users can specify the status of a road segment (e.g., "Optimal," "Requires Maintenance") or report the presence of specific obstacles. The system associates this information with the current GPS coordinates (or those selected on the map) and makes it immediately available to the community.

Automatic Detection via Sensors

If the registered user enables "Automatic Mode" while driving, the system uses the mobile device's accelerometer and gyroscope to monitor vibrations and sudden movements. Internal algorithms analyze this data to identify potential road surface anomalies, such as potholes. This process occurs in the background so as not to distract the user while riding.

Review and Confirmation of Detections

To ensure data reliability and filter out false positives, automatic detections are not published immediately. At the end of the journey, the system presents the user with a list of detected anomalies. The registered user must explicitly confirm the presence of the obstacle (validation) or discard the detection (if incorrect). Only data with a positive feedback is promoted to publishable information. The published route data is then used to calculate the Path Score.

Activity History Consultation

Registered users may consult their historical trip data via a dedicated summary view, which provides the necessary functionality to view or delete specific trip records.

Route Search

This feature is accessible to all users, regardless of registration. The user enters a starting point and a destination in the search interface. The system processes the request and calculates one or more possible cycling routes connecting the two points, if data for such routes exists in the inventory. Otherwise, the an External Mapping System will be interrogated by the system in order to create a path from the starting point to the destination submitted by the user.

Display and Path Score

The routes found are displayed on an interactive map. For each route, the system calculates and displays a `Path Score`. This summary score aggregates information about the route's status and the presence of confirmed obstacles, allowing the user to quickly assess not only the distance, but also the safety and quality of the proposed route.

2.3 User Characteristics

This section describes the general characteristics of users who interact with the BBP system. There are two main categories of users: Registered Users (the active contributors) and General Users (the passive users).

2.3.1 Registered Users

The Registered User represents the core of the BBP ecosystem. This profile typically corresponds to a regular cyclist (commuter or recreational) who wishes to monitor their performance and actively contribute to community safety.

Profile and Skills

The user must have a personal account with login credentials. It is assumed that they have moderate familiarity with the use of smartphones and GPS technology. Since the app is used in mobile contexts, the user requires a clear interface that minimizes distractions.

Needs and Interactions:

- **Tracking:** The user wants to track their trips in order to analyze statistics such as speed and distance, contextualized with weather data if available.
- **Active Contribution:** The user wants to report obstacles or assess road conditions to help other cyclists. They can do this manually or by activating automatic mode.
- **Validation:** The user is responsible for data quality. The system relies on them to confirm or discard automatic sensor detections (e.g., potholes) at the end of the trip, ensuring that only truthful information influences the Path Score.
- **Privacy:** The user wants sensitive data (such as personal travel history) to remain private, while agreeing to share anonymized road condition data publicly.
- **Trip planning**: The user needs to access to an updated archive of paths in order to plan its cycling activity; so it needs to find the most efficient or the more intriguing path from its starting point up to its destination, but avoiding those paths having some problem; it's also not interested in paths with low score, Since they won't match its expectancies.

2.3.2 Generic User

The Generic User includes anyone who accesses the platform without authenticating. This profile includes tourists, occasional cyclists, or route planners who need quick and reliable information without the commitment of registration.

Profile and Skills

They do not have a persistent profile in the system. Minimum proficiency in using digital maps and web/mobile interfaces is required. Interaction is sporadic and aimed at an immediate goal: reaching a destination.

Needs and Interactions:

- **Safety and Planning:** The primary need is to find the safest or most efficient route between two points. The user relies on the system to avoid poor or dangerous roads.
- **Immediacy:** They want to view routes and their Path Score immediately. It's not interested in contributing data or saving history, but only in consuming aggregated information generated by the community.
- **Reliability:** It expects the obstacle reports (e.g., potholes) displayed on the map to be up-to-date and verified, so it can plan its trip with confidence.

2.4 Assumptions, dependecies and constraints

2.4.1 Domain Assumptions

The following assumptions describe real-world conditions that the system considers true and necessary for the correct functioning of the intended features:

- **D1 - Hardware Equipment:** It is assumed that the user's mobile device is equipped with functioning and calibrated hardware, specifically: GPS receiver, accelerometer, and gyroscope.
- **D2 - Accuracy of user registration data:** It is assumed that the information entered by users during registration phase is correct and truthful.
- **D3 - Accuracy of route feedbacks:** It is assumed that the user's feedbacks about routes problems (either manual or automatically detected) are correct and truthful.
- **D4 - Accuracy of Basemaps:** It is assumed that third-party mapping services provide a correct topological representation of reality, that is if a road exists on the map then it's assumed that physically exists and that it'is drivable safely by bycicles (unless otherwise reported on BBP).
- **D5 - GPS Signal Availability:** It is assumed that, for most of the duration of an outdoor trip, satellite coverage is sufficient to ensure useful location accuracy.
- **D6 - Distinguishable Movement Patterns:** It is assumed that the physical characteristics of cycling are sufficiently distinct from those of other modes of transport or walking in order to allow classification algorithms to operate with an acceptable level of accuracy.

2.4.2 System Dependencies

The BBP system is not an island; it relies on external services to provide added value. Failure of these services degrades the system's functionality as follows:

- **External Weather Service:** BBP depends on third-party APIs to retrieve weather data (temperature, wind). If this service is unavailable, the system will continue to record trips, but the "Weather Enrichment" feature will not be performed, and the trips will be saved without this metadata.
- **External Mapping Services:** Route visualization and address geocoding depend on external map providers. If these are unavailable, the "Route Search" and "Trip Recording" features will be compromised.

2.4.3 System Constraints

- **GDPR and Privacy:** Since the system tracks users' physical movements (sensitive data), the management, storage, and sharing of GPS data must strictly comply with the GDPR regulation. Personal travel data must not be accessible to other users.
- **Energy Consumption:** The automatic detection algorithm must be optimized to avoid draining the mobile device's battery quickly, ensuring coverage of medium-duration trips (e.g., 2-3 hours).
- **Intermittent Connectivity:** Since cycling routes can pass through areas with poor network coverage, the mobile application must be able to store sensor data locally and synchronize it with the server as soon as the connection is re-established.

3 Specific requirements

3.1 External interface requirements

3.1.1 User interfaces

This section presents mockups of the BBP mobile application's user interface. The images illustrate the main interaction flows defined in the scenarios, demonstrating how the system meets usability and functionality requirements.

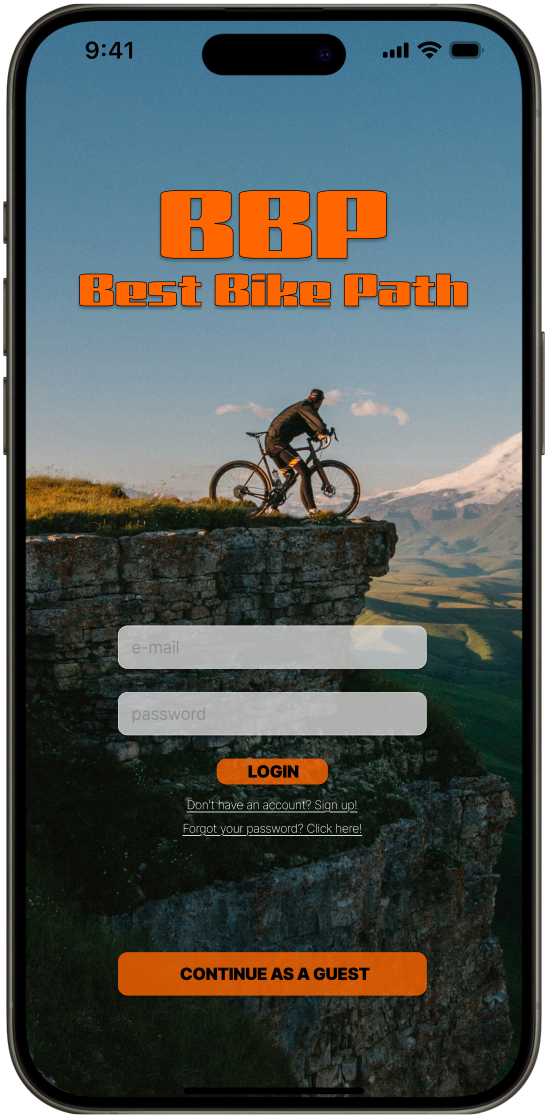


Figure 4: Login and Registration Screen

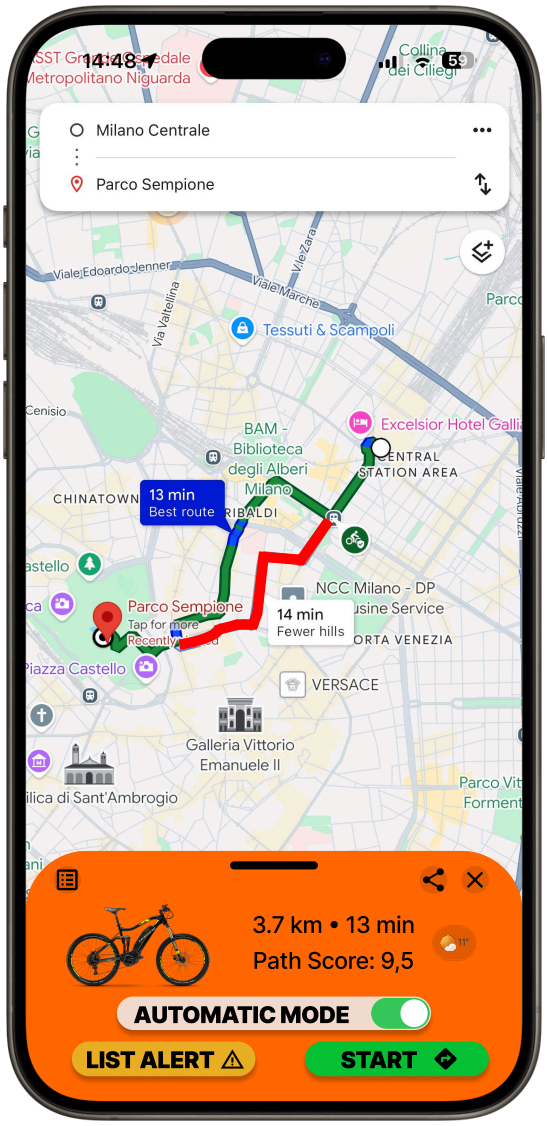


Figure 5: Route Selection Screen

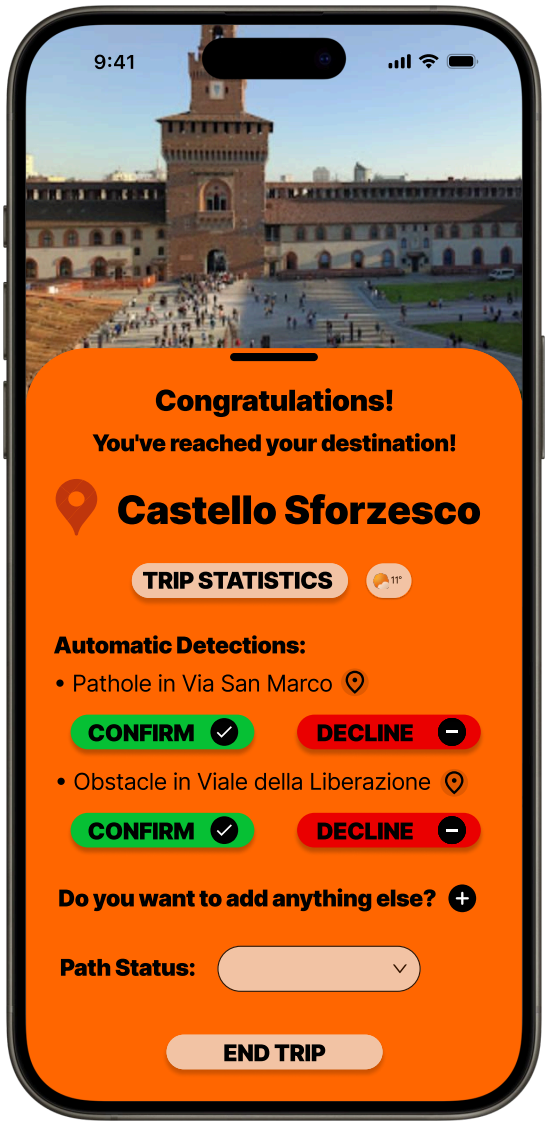


Figure 6: Post-Trip Confirmation Screen

Figure 7: Trip History Screen

3.1.2 Hardware interfaces

Since BBP is a mobile application focused on automatic tracking and detection, hardware interfaces are critical to the system's operation.

- **GPS:** The system requires access to the mobile device's GPS receiver to track the user's location in real time during travel and to geolocate alerts.

- **Inertial Sensors:** For the "Automatic Mode" feature, the application needs to interface directly with the device's motion sensors to detect vibrations and road surface anomalies.

3.1.3 Software interfaces

The system interacts with external software components to enhance its functionality.

- **External Weather Service API:** The system interfaces with a weather data provider to retrieve historical weather conditions for the time and location of the completed trip.
- **Mapping Service API:** The application uses mapping services for map rendering, route calculation, and address geocoding.
- **Mobile OS APIs:** The app interacts with native Android and iOS APIs for managing permissions and push notifications.

3.1.4 Communication interfaces

- **Network Protocols:** All communications between the mobile application and the backend server are via the **HTTPS** protocol to ensure the security and encryption of data in transit, especially for authentication information and sensitive location data.
- **Network Connectivity:** The device must have a network interface (4G/5G/Wi-Fi) to send data to the server and download maps.

3.2 Functional requirements

Authentication and Account Management

- [**R1**] The system shall allow any user to create an account.
- [**R2**] The system shall allow registered user to log in using their credentials.
- [**R3**] the system shall allow registred user to reset their account password
- [**R4**] The system shall allow registered user to update their personal profile information.
- [**R5**] The system shall allow registered user to delete their account.

Trip Recording and Monitoring

- [**R6**] The system shall allow registered user to start the recording of a new trip.
- [**R7**] The system shall allow registered user to pause and resume the recording of an active trip.

- [**R8**] During the recording, the system shall track the user's position and its performance statistics.
- [**R9**] Upon completion of a trip, the system shall automatically retrieve weather data from an external service, if available, and associate it to the saved trip.

Data Contribution and Governance

- [**R10**] The system shall allow registered user to insert manual reports regarding the status of a path.
- [**R11**] The system shall allow registered user to submit feedback regarding the Path Status.
- [**R12**] The system shall allow registered user to insert manual reports regarding problems on the path.
- [**R13**] The system shall allow registered user to enable automatic detection during a trip.
- [**R14**] When automatic detection is active, the system shall analyze data from the device's sensors to detect potential anomalies.
- [**R15**] The system shall present the list of automatically detected anomalies to the registered user at the end of the recorded trip for review.
- [**R16**] The system shall allow the registered user to confirm or discard a detected anomaly.

Path Planning and Visualization

- [**R17**] The system shall compute valid cycling routes between a specified starting point and a destination based on the available physical road network.
- [**R18**] The system shall compute and visualize one or more valid routes between the specified points on a map.
- [**R19**] The system shall compute a Path Score for each route derived from available inventory data
- [**R20**] The system shall display confirmed obstacles on the map with visual markers.
- [**R21**] The system shall allow the user to filter the search on Path properties.

Trip History

- [**R22**] The system shall allow registered user to view the list of its past trips.
- [**R23**] The system shall allow registered user to view the details of a specific past trip, including the route on the map, statistics, and weather data (if they exist).
- [**R24**] The system shall allow registered users to delete a specific trip from their history.
- [**R25**] The system shall allow the user to search a specific trip in its history.
- [**R26**] The system shall allow the user to filter the view of its history.

3.2.1 Use Case Diagram

Figure 8: Use Case Diagram

3.2.2 Use cases

[UC1] Account creation

Name	**Account creation**
Actors	User
Entry Condition	True
Event Flow	1. The person downloads the BBP, opens it and starts the creation of an account 2. The system asks to the user to fill out a form with the following personal information: name, surname, birth date, geneder, email, password; the systems also asks to accept the privacy terms 3. The person fills out the questionare and submits it 4. The system checks the information submitted and sends a verification email containing a link to verify the email address, which expires in 1 hour 5. The person receives the email and opens the link to confirm the email 6. The system sends an acknowledgement of successful account creation
Exit Condition	The account is successfully created if the information are correct. If not, an error message is sent and the account isn't created.
Exception	• Email address is not valid, therefore a warnigs is displayed in point *2.* and the form can't be submitted. • Email inserted during registration has been already used, therefore the account is not created and in the point *4.* instead of a link an informative message is sent. • The user doesn't open the confirmation link within an hour, therefore the account creation procedure is aborted and the submited data cancelled. If the confirmation link is open after 1 hour, an error message is sent.

Table 1: Refers to SC1. We highlight that the same person can create multiple accounts.

[UC2] User login

Name	**User login**
Actors	Registered user
Entry Condition	The user isn't logged in
Event Flow	1. The user opens the BBP app and the login form is shown to it 2. The user types email and password, then submits the form 3. The system receives the user's login info and checks the information 4. The system retrives the information to build the homepage for that account and sends it
Exit Condition	The user logs into its account if submits the correct information. If not, the user can-t login and an error message is displayed.
Exception	• The user sends an email not related to any account, therefore the system sends an error message to the user • The user sends a password that doesn't match with the one on the system for that account, therefore the system sends an errore message to the user

Table 2: Refers to SC2.

[UC3] Account information update

Name	**Account information update**
Actors	Registered user
Entry Condition	True
Event Flow	1. The user opens his profile page and selects the option to modify the account 2. The user selects the attribute to modify, types the new value and sends it (except email or password) 3. The system checks the new values and sends an acknowledgement
Exit Condition	The account profile is updated. If not, an error meassge is diplayed and the account isn't updated.
Exception	

Table 3: Refers to SC3

[UC4] Password account reset

Name	**Password account reset**
Actors	Registered user
Entry Condition	The user can't log into its account
Event Flow	1. The user clicks the "Reset Password" option in the login page and sends a reset password request 2. The system receives the request and sends a form in which the user must specify the account email 3. The user receives the form, enters the account email and submits it 4. The system receives the account email, checks the email address and sends an email to that address with a link to reset the password that expires in 1 hour 5. The user receives the email, opens the link, types the new password two times and sends the new password 6. The system receives the new password, updates the account password and sends an update confirmation 7. The user receives the update confirmation
Exit Condition	The user can log-in with the new password if the information submitted are truthfull and are sent within expiration time. If the informations are not correct, the systems send an error messagge and the password isn't updated.
Exception	• The user submits an invalid email address, therefore an error message would be sent to it • The user submits the email address of another account, therefore he won't receive the email to reset the password • The user submits an email not associated to any account, therefore an error messagge is sent to it. • The user doesn't submits the new password after opening the link within one hour, therefore the password wouldn't be changed

Table 4: Refers to SC4.

[UC5] Account deletion

Name	**Account deletion**
Actors	Registered user
Entry Condition	True
Event Flow	1. The user opens the Profile page, opens the options and selects to delete the account 2. The system receives the request and sends a confirmation message on the app 3. The user sends the confirmation 4. The system receives the confirmation and sends an email containing a link to delete the account, which expires in 1 hour 5. The user receives the email, opens the link and sends the second confirm. 6. The system receives the second confirmation and processes the account deletion request 7. Once the system has deleted the account, sends to the ex-account mail address a deletion message confirmation
Exit Condition	The account is deleted. If the confirmation isn't given within the expiration times, the account isn't deleted.
Exception	• The user didn't confirm the second time within one hour, therefore the request expires

Table 5: Refers to SC5

[UC6] Route planning with route match

Name	**Route planning with route match**
Actors	User
Entry Condition	True
Event Flow	1. The user opens the Search page and inserts the starting point and the destination 2. The system retrieves from its archive all paths near the starting point specified by the user and that lead toward the destination, ordered by "Path Score" and sends them to the user 3. The user explores the choices given by the system and selects one of them
Exit Condition	A path, if existing, is displayed to the user. Otherwise see UC7.
Exception	• No path between starting point and destination is found, see UC7

Table 6: Refers to SC6, SC8

[UC7] Route planning without route match

Name	**Route planning without route match**
Actors	User, External Mapping Service
Entry Condition	True
Event Flow	1. The user opens the Search page and inserts the starting point and the destination 2. The system fails to retrieve from its archive any path near the starting point specified by the user and that lead toward the destination 3. The system sends to the External Mapping Service a request to compute a path between the starting point and the ending point. 4. The External Mapping Service computes the path and returns it to the system 5. The system forwards the computed paths to the user 6. The user searches among returned paths, selects one of them and starts the trip activity for the selected path
Exit Condition	A path is displayed to the user
Exception	

Table 7: Refers to SC7, SC9

[UC8] Trip activity for unregistered user

Name	**Trip activity**
Actors	Unregistered user
Entry Condition	True
Event Flow	1. The user selects the path and starts the trip, sending an information retrieval request to the system 2. The system receives the request, retrieves the path map and sends it to the user 3. The user receives the path map The user might stop and resume the trip whenever he likes 4. The user completes the trip and closes it
Exit Condition	The user completed its trip
Exception	

Table 8: Refers to SC10, SC11

[UC9] Trip activity for registered user

Name	**Trip activity**
Actors	Registered user
Entry Condition	True
Event Flow	1. The user selects the path and starts the activity, sending an information retrieval request to the system 2. The system receives the request, retrieves the path map and sends it to the user 3. The user receives the path map The user might stop and resume the activity whenever he likes 4. The user completes the activity, and sends an "Activity Completion message" to the system 5. The system receives the notification about the activity completion and sends a form to rate the path 6. The user receives the rating form, fills it out and sends it 7. The system receives the rating form and sends an acknowledgement
Exit Condition	The user completed its trip activity and scored to the path
Exception	

Table 9: Refers to SC10, SC11, SC13

[UC10] Automatic trip monitoring

Name	**Automatic trip monitoring**
Actors	Registered user, Weather Service
Entry Condition	True
Event Flow	1. The user selects the route he wants to ride on, selects the option to automatically collect data and starts the activity 2. The user's personal device samples user position every second and stores it 3. Once finished the user stops the activity and the data collected are sent to the system 4. The system sends a request to retirve weather conditions during the activty to a third-part Weather Service 5. The Weather Service responds to the system with the requested data 6. The system sends to the user the activity summary, waether conditions plus the map showing the path traveled
Exit Condition	The user completed the activity and activity summary is shown
Exception	• Weather conditions can't be retrived from the third party service, therefore only the metrics about user performances are shown to the user • User's device loses GPS signal during the activity, therefore the system is not able to compute all the metrics about user performances and shows only the ones that can be computed with the available data, notifying the user about the partial (or total) data loss

Table 10: Refers to SC12

[UC11] Manual route problem report

Name	**Manual route problem report**
Actors	Registered user
Entry Condition	Route problem hasn't been reported
Event Flow	1. The user after noticing a problem along a route opens the Report Issue page 2. The user searches for the route and selects it, then specifies the issue type, the issue position along the route and adds a description, then submits the report 3. The system receives the issue report, updates the information about that problem and send an acknowledgement
Exit Condition	Path problem has been reported
Exception	

Table 11: Refers to SC15

[UC12] Route problem fixup report

Name	**Route problem fixup report**
Actors	Registered user
Entry Condition	Route fixup hasn't been reported
Event Flow	1. The user opens the app, selects the problem icon on the route and marks it as fixed 2. The system receives the fixup report, updates the information about that problem and sends an acknowledgement
Exit Condition	Route problem has been reported
Exception	

Table 12: Refers to SC16

[UC13] Automatic route problem detection and report

Name	**Automatic route problem detection and report**
Actors	Registered user
Entry Condition	True
Event Flow	1. The user starts an activity with automatic issue detection enabled 2. The BBP app collects data from the user's device sensors and analyzes them 3. When the user finishes the activity, the BBP app shows to the user a list of all problems detected and their location, asking the user confirmation for each one of them 4. The user confirmes whether the problems detected are real issues or false positives 5. The BPP app sends to the system the confirmed issues 6. The system sends and acknowledgement, and updates the path status
Exit Condition	Path problem has been detected and reported
Exception	

Table 13: Refers to SC14

[UC14] User's activity history consultation

Name	**User's activity history consultation**
Actors	Registered user
Entry Condition	True
Event Flow	1. The user opens the relative page on the app, optionally applies filters, and sends the request to the system 2. The system runs the query and retrieves the activities matching the request, then sends the result to the user
Exit Condition	The user consults its activity history
Exception	

Table 14: Refers to SC17

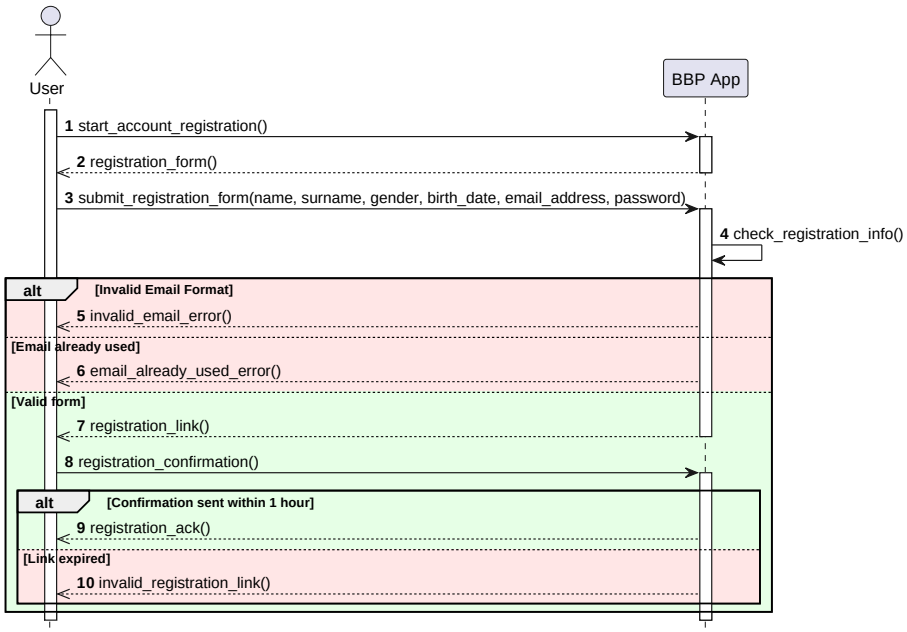
[UC15] User activity deletion

Name	**User activity deletion**
Actors	Registered user
Entry Condition	User's activity history contains N activities
Event Flow	1. The user opens the Acivity History and searches for the trip to delete 2. The user selects the trip, selects the option to delete it and confirms the deletion. 3. The system recevies the deletion request and sends an acknowledgement to the user.
Exit Condition	User's activity history contains $N - 1$ activities
Exception	

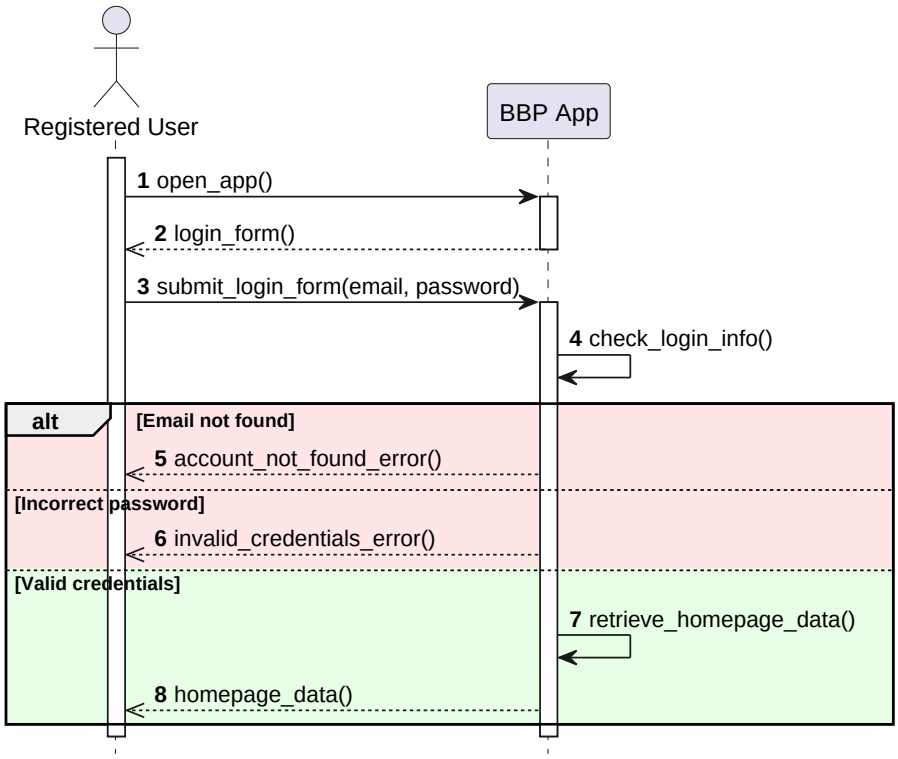
Table 15: Refers to SC18

3.2.3 Sequence Diagrams

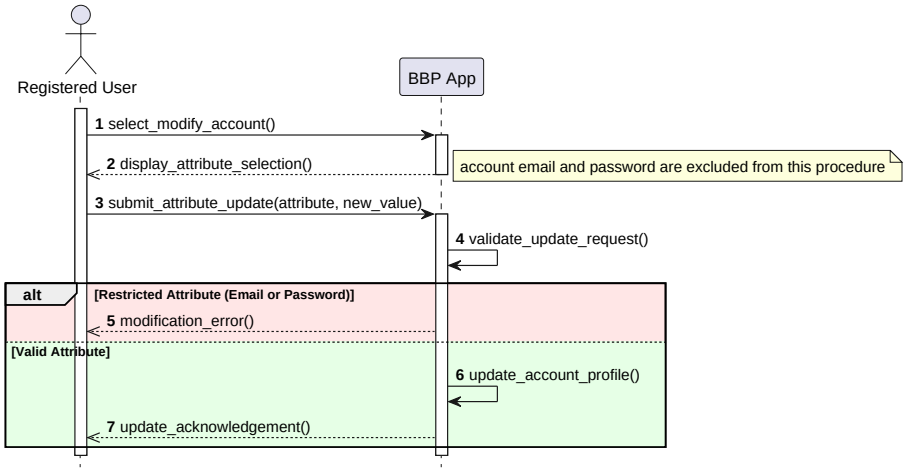
[UC1] Account creation



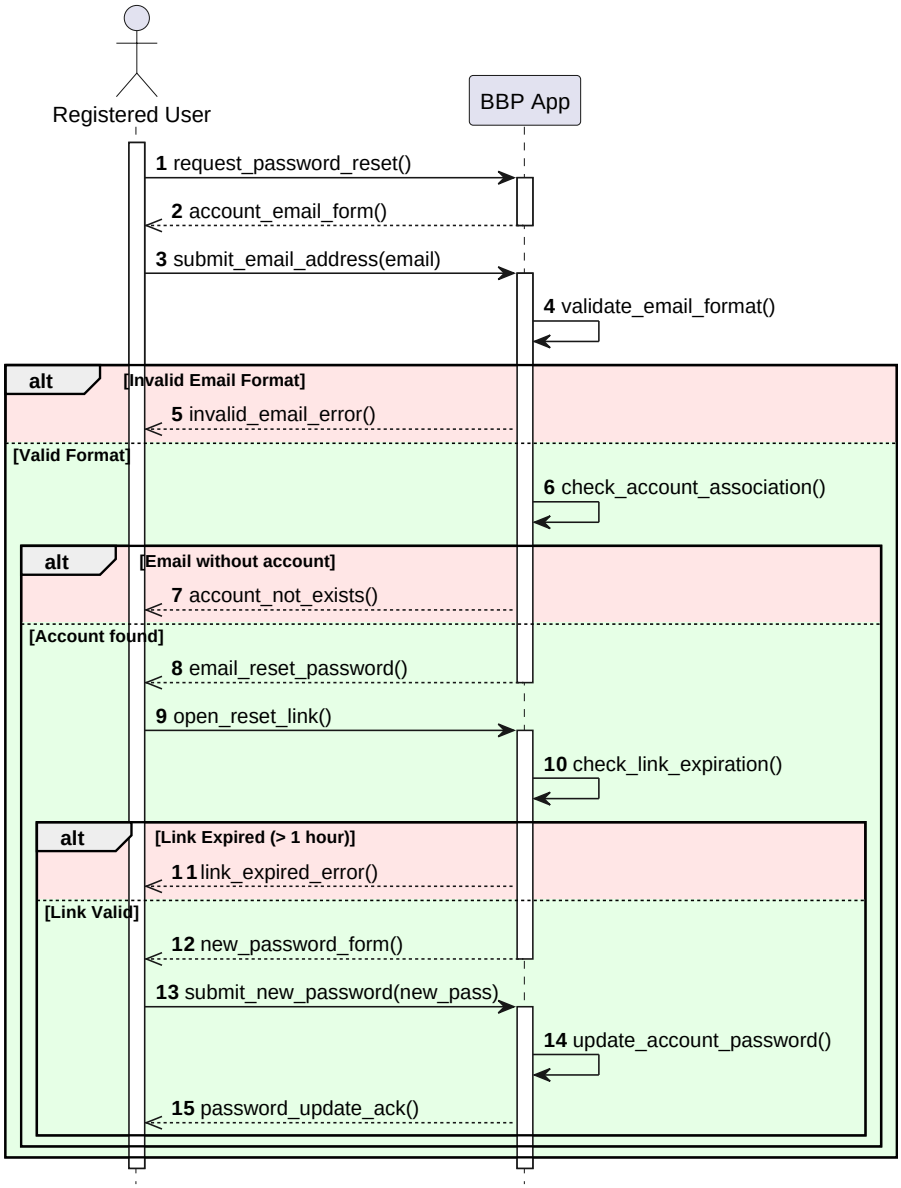
[UC2] User login



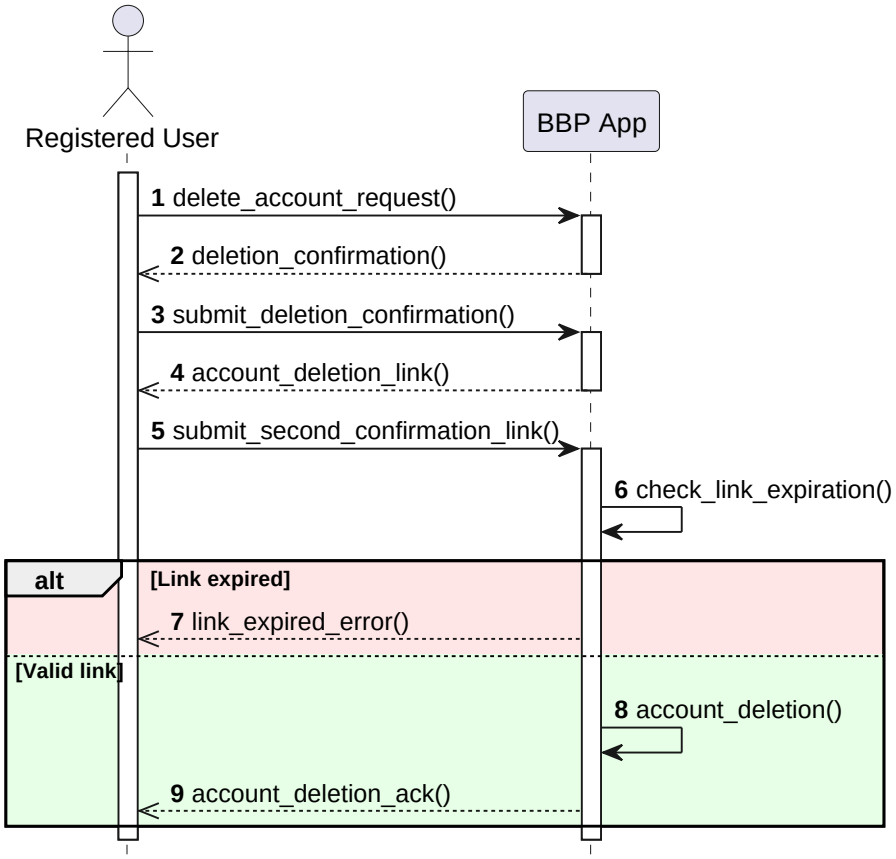
[UC3] Account information update



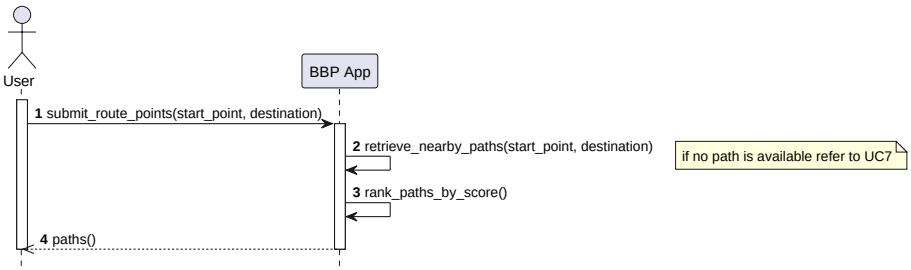
[UC4] **Password account reset**



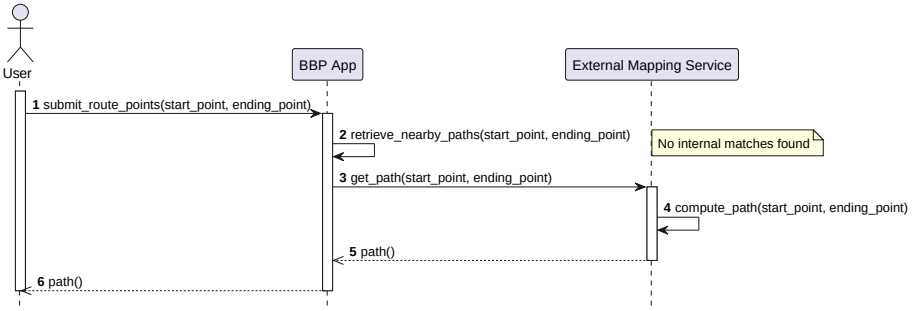
[UC5] Account deletion



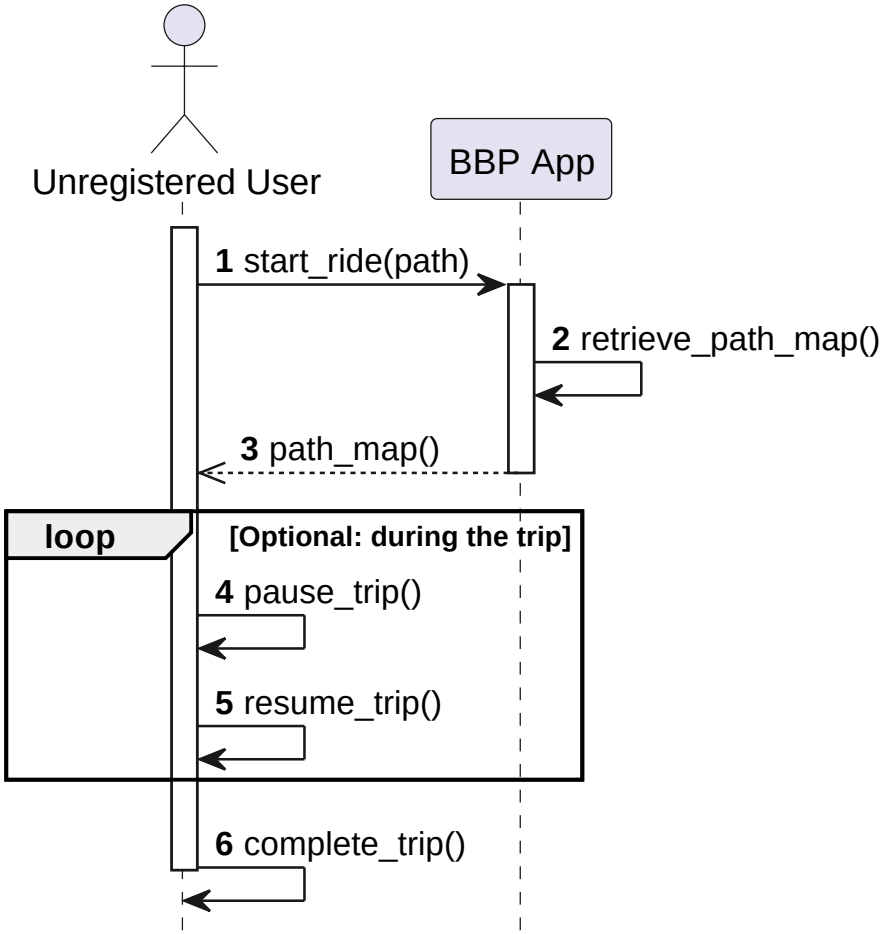
[UC6] Route planning with route match



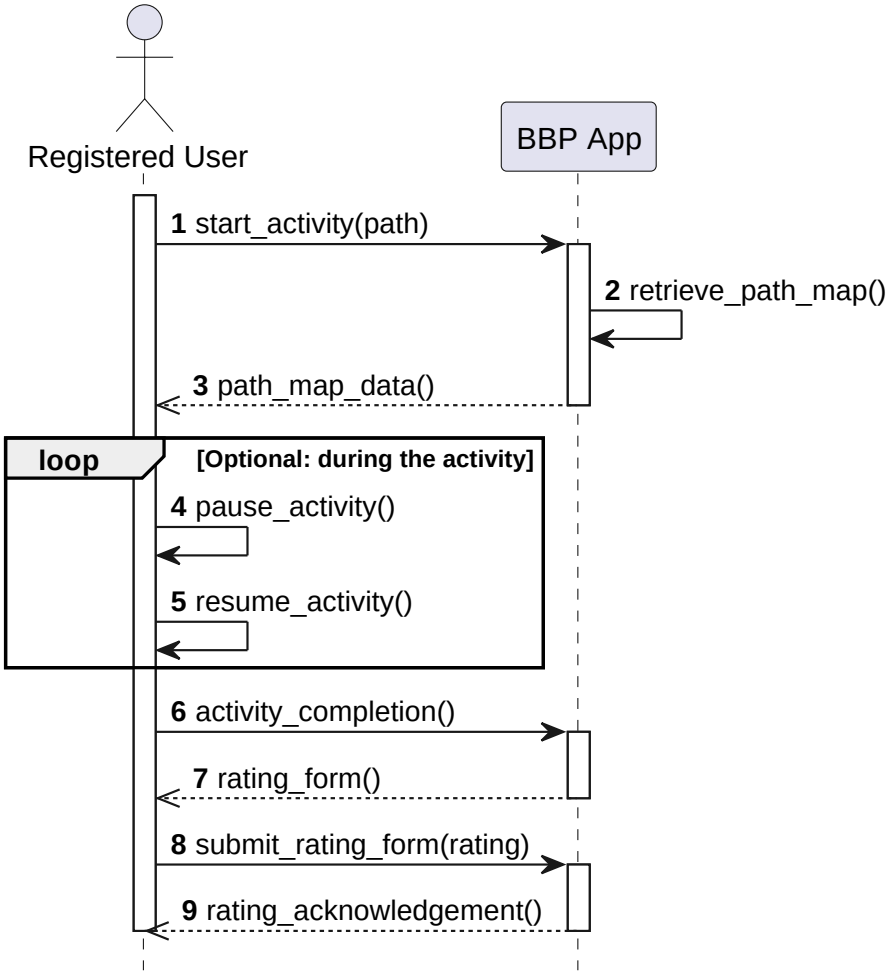
[UC7] Route planning without route match



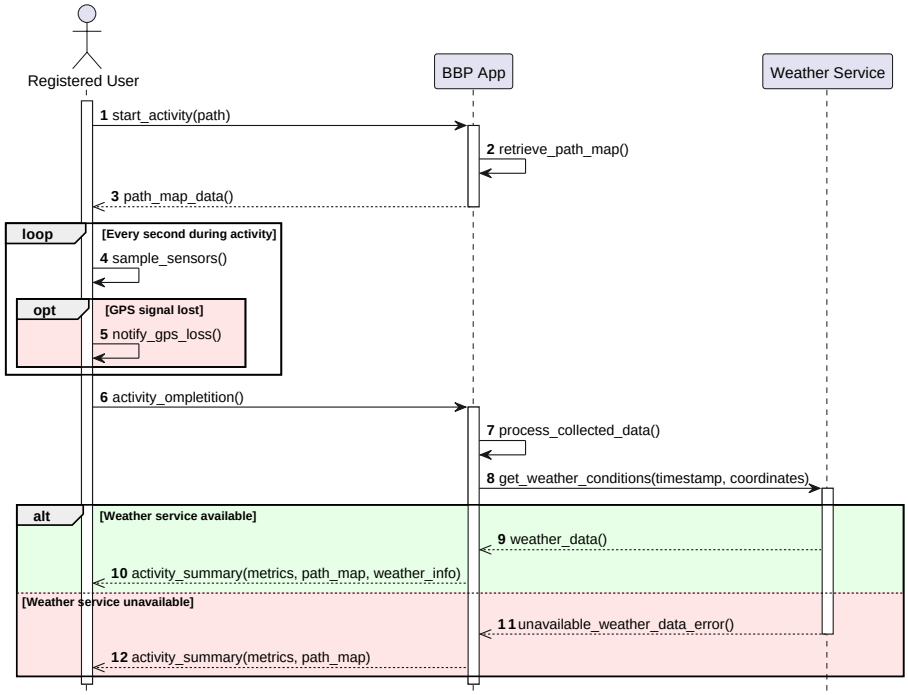
[UC8] Trip activity for unregistered user



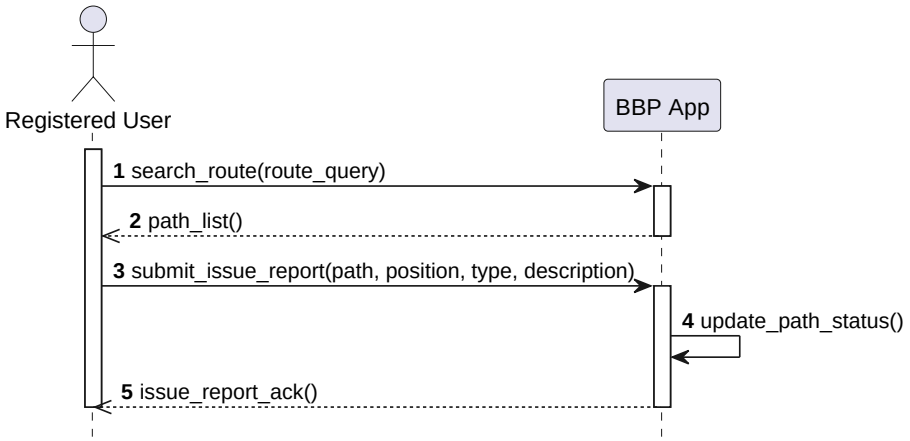
[UC9] **Trip activity for registered user**



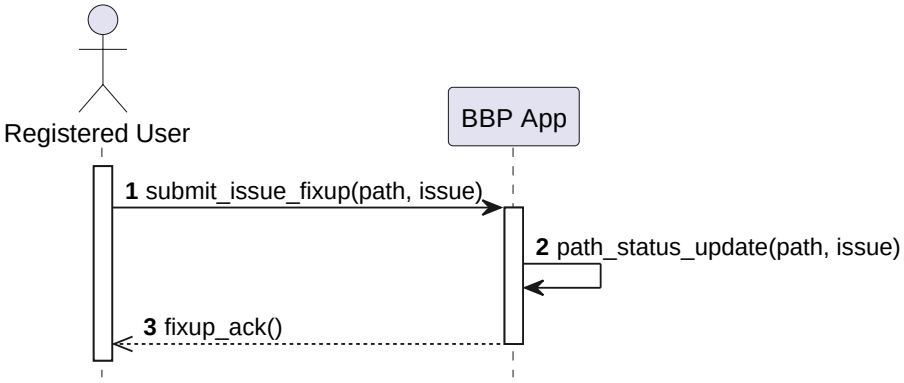
[UC10] Automatic trip monitoring



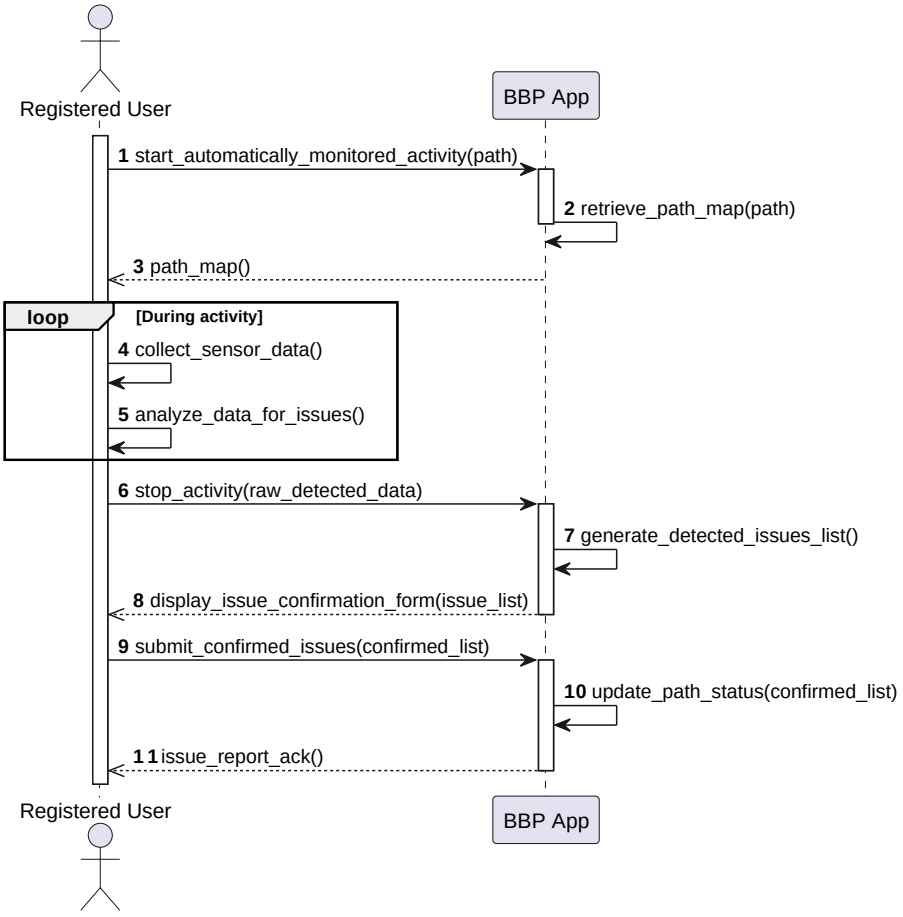
[UC11] Manual route problem report



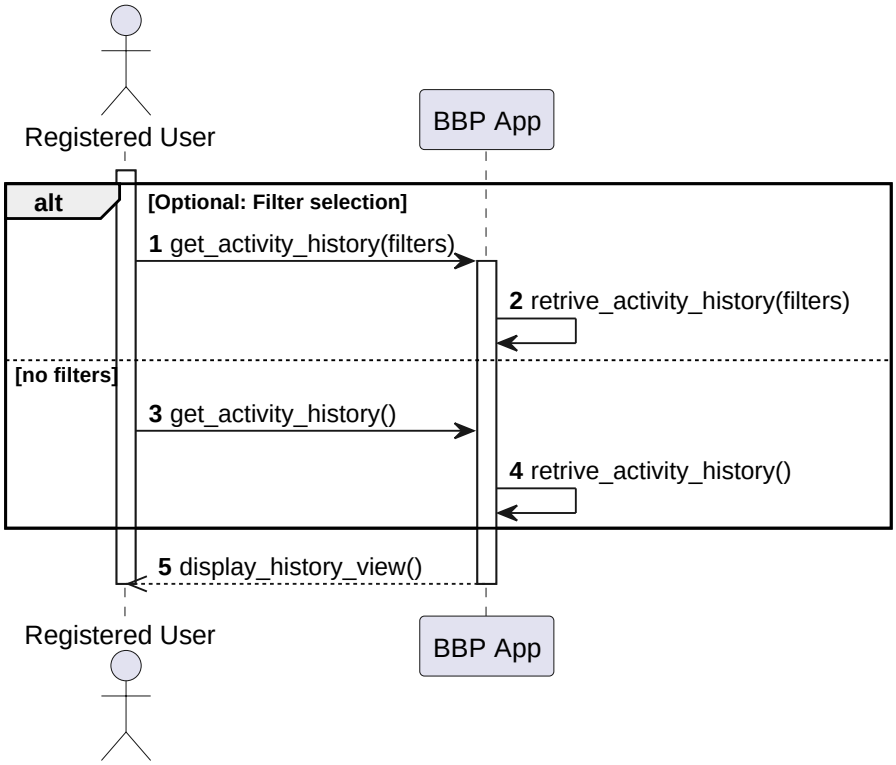
[UC12] Route problem fixup report



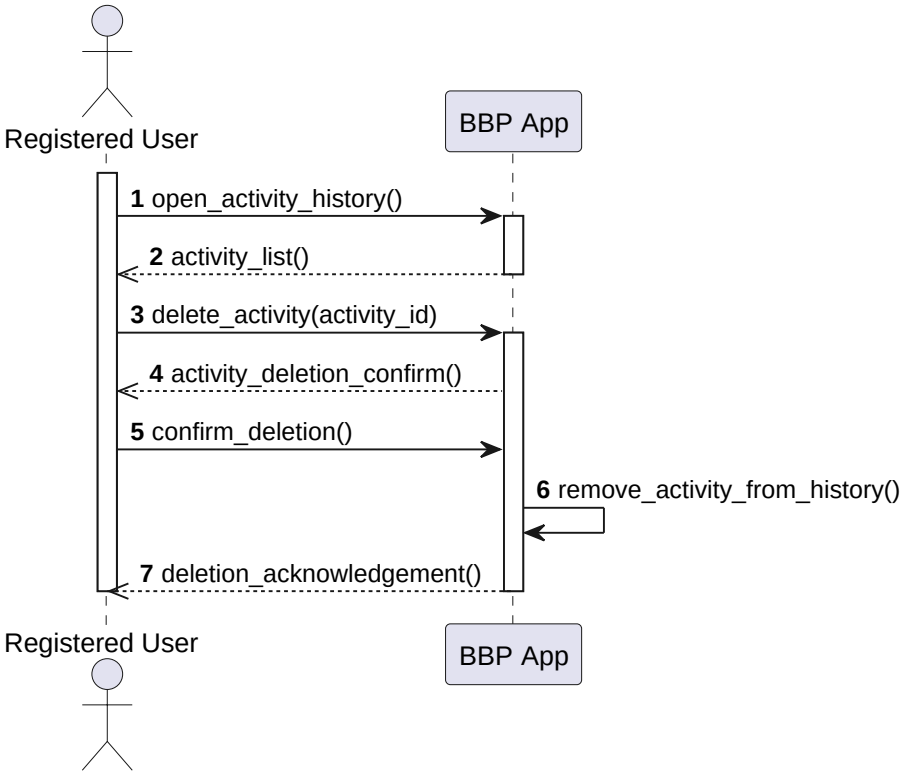
[UC13] Automatic route problem detection and report



[UC14] **User's activity history consultation**



[UC15] User activity deletion



3.2.4 Requirement Mapping

This section maps the Goals identified in Section 1 to the Functional Requirements and Domain Assumptions. This mapping demonstrates that the set of requirements, supported by the assumptions, is sufficient to satisfy the system goals ($R \wedge D \models G$).

G1: A registered user wants to track their personal cycling activities and related performance statistics.	
Requirements	**Domain Assumptions**
[**R1**] The system shall allow any user to create an account. [**R2**] The system shall allow registered user to log in using their credentials. [**R3**] the system shall allow registred user to reset their account password [**R4**] The system shall allow registered user to update their personal profile information. [**R5**] The system shall allow registered user to delete their account. [**R6**] The system shall allow registered user to start the recording of a new trip. [**R7**] The system shall allow registered user to pause and resume the recording of an active trip. [**R8**] During the recording, the system shall track the user's position and its performance statistics. [**R9**] Upon completion of a trip, the system shall automatically retrieve weather data from an external service, if available, and associate it with the saved trip. [**R22**] The system shall allow registered user to view the list of its past trips. [**R23**] The system shall allow registered user to view the details of a specific past trip, including the route on the map, statistics, and weather data (if they exist). [**R24**] The system shall allow registered users to delete a specific trip from their history. [**R25**] The system shall allow the user to search a specific trip in its history. [**R26**] The system shall allow the user to filter the view of its history.	**D1 - Hardware Equipment:** It is assumed that the user's mobile device is equipped with functioning and calibrated hardware, specifically: GPS receiver, accelerometer, and gyroscope. **D5 - GPS Signal Availability:** It is assumed that, for most of the duration of an outdoor trip, satellite coverage is sufficient to ensure useful location accuracy.

Table 16: Requirements Mapping for Goal G1

G2: A registered user wants to contribute to the community inventory by sharing reliable information on the condition of the trails (e.g. quality, obstacles, potholes).	
Requirements	**Domain Assumptions**
[**R2**] The system shall allow registered user to log in using their credentials. [**R5**] The system shall allow registered user to start the recording of a new trip. [**R7**] The system shall allow registered user to save a trip. [**R10**] The system shall allow registered user to insert manual reports regarding the status of a path. [**R11**] The system shall allow registered user to insert a personal rating of a path. [**R12**] The system shall allow registered user to insert manual reports regarding problems on the path. [**R13**] The system shall allow registered user to enable automatic detection for a trip. [**R14**] When automatic detection is active, the system shall analyze data from the device's sensors to detect potential anomalies. [**R15**] The system shall present the list of automatically detected anomalies to the registered user at the end of the recorded trip for review. [**R16**] The system shall allow the registered user to confirm or discard a detected anomaly.	**D1 - Hardware Equipment:** It is assumed that the user's mobile device is equipped with functioning and calibrated hardware, specifically: GPS receiver, accelerometer, and gyroscope. **D3 - Cooperative Behavior:** It is assumed that the majority of registered users act cooperatively and with good intent to contribute to the community, validating the data truthfully. **D6 - Distinguishable Movement Patterns:** It is assumed that the physical characteristics of cycling are sufficiently distinct from those of other modes of transport or walking in order to allow classification algorithms to operate with an acceptable level of accuracy.

Table 17: Requirement Mapping for Goal G2

G3: Any user (registered or not) wants to find and view the best cycling route between an origin and a destination, based on up-to-date and relevant data.	
Requirements	**Domain Assumptions**
[R17] The system shall allow any user to search for cycling paths between starting point and a destination. **[R18]** The system shall compute and visualize one or more valid routes between the specified points on a map. **[R19]** The system shall calculate a Path Score for each route. **[R20]** The system shall display confirmed obstacles on the map with visual markers. **[R21]** The system shall allow the user to filter the search on Path properties.	**D4 - Accuracy of Basemaps:** It is assumed that third-party mapping services provide a correct topological representation of reality, that is if a road exists on the map then it's assumed that physically exists and that it'is drivable safely by bycicles (unless otherwise reported on BBP). **D1 - Hardware Equipment:** It is assumed that the user's mobile device is equipped with functioning and calibrated hardware.

Table 18: Requirement Mapping for Goal G3

G4: The cycling association aims to provide the community with a tool to create, consult, and maintain a reliable and centralized inventory of cycling routes.	
Requirements	**Domain Assumptions**
[R1] The system shall allow any user to create an account. **[R10]** The system shall allow registered user to insert manual reports regarding the status of a path. **[R12]** The system shall allow registered user to insert manual reports regarding problems on the path. **[R16]** The system shall allow the registered user to confirm or discard a detected anomaly. **[R19]** The system shall calculate a Path Score for each route.	**D2 - Accuracy of user registration data:** It is assumed that the information entered by users during registration phase is correct and truthful. **D3 - Cooperative Behavior:** It is assumed that the majority of registered users act cooperatively and with good intent to contribute to the community.

Table 19: Requirement Mapping for Goal G4

3.3 Performance requirements

Given the nature of BBP as a mobile application that also operates in active mobility contexts, performance is critical not only for the user experience, but also for the security and reliability of the collected data.

- **Interface Responsiveness:** The system must ensure immediate response times for critical interactions during cycling, with a latency of less than 200 ms, to avoid dangerous distractions for the user.
- **Real-Time Data Processing:** During "Automatic Detection" mode, the local algorithm on the device must process sensor data in real time.
- **Routing:** The route search functionality must return results, complete with *Path Score*, within 3 seconds for requests in a standard urban environment (10 km radius), ensuring smooth planning.
- **Backend Scalability:** The system must be able to handle simultaneous load peaks (e.g., weekends or cycling events), scaling horizontally to support thousands of simultaneous trip uploads without data loss.

3.4 Design Constraints

3.4.1 Standards Compliance

The BBP system adheres to rigorous international standards to ensure interoperability, security, and regulatory compliance.

Standard	Description
GDPR (EU 2016/679)	The system manages sensitive geolocation and user profiling data. All processing must comply with the General Data Protection Regulation, guaranteeing the right to be forgotten and data minimization.
WGS 84	Geodetic reference standard for the GPS system. All stored and exchanged coordinates must comply with this standard to ensure compatibility with global maps.
ISO/IEC 27001	Standard for information security management, applied to protect the backend infrastructure and user data from unauthorized access.

Table 20: Compliance standards adopted by BBP

3.4.2 Hardware Limitations

The mobile application must operate in a resource-constrained environment, typical of mobile devices during extended outdoor use.

- **Power Consumption:** The automatic detection algorithm (GPS + Sensors) must be optimized to consume no more than 10-15% battery power per hour of use on an average device, ensuring the user does not run out of battery power while traveling.
- **Required Sensors:** Full use of the app is contingent on the physical presence of a calibrated accelerometer and gyroscope. Older or low-end devices without these sensors will only be able to use the app in limited mode (without automatic detection).
- **Intermittent Connectivity:** The design must include an "offline-first" mode for data recording. Upload to the server must occur asynchronously when the connection is stable, handling any timeouts without losing local data.

3.4.3 Any Other Constraint

- **GPS Accuracy:** The accuracy of obstacle detection is limited by the accuracy of the device's civilian GPS. The system must include clustering or manual correction mechanisms to handle inherent hardware inaccuracy.

- **Operating System:** The application must be compatible with Android and iOS versions released in the last 3 years to ensure access to the latest APIs for efficient background sensor management.
- **Local Data Processing:** To ensure responsiveness and minimize mobile data usage, the raw processing of high-frequency sensor data must be performed locally on the user's device. The system is constrained to transmit only the identified "candidate anomalies" to the server, rather than the continuous raw data stream.

3.5 Software system attributes

3.5.1 Reliability

The system must provide robust, scalable storage capable of handling high-volume data ingestion generated by frequent user activity sampling. To ensure data integrity and fault tolerance, replication strategies must be implemented across multiple nodes to prevent data loss. The system should employ a differentiated consistency approach: strong consistency must be enforced for sensitive user data to guarantee correctness, like account credentials, while eventual consistency models are acceptable for non-critical data such as path status updates and user monitoring activities.

3.5.2 Availability

The system must maintain high availability with minimal downtime to ensure continuous service delivery. Core functionalities requiring guaranteed uptime include path planning, activity recording, and real-time user guidance—these services are mission-critical and should target 99.9% availability. To accommodate fluctuating demand patterns, with expected peak traffic periods—such as holidays, weekends, and seasonal recreational periods, urge the need for auto-scaling capabilities to dynamically adjust computational resources and maintain performance under variable load conditions. Since data about path and user have a strict correlation with geographical location,the system should implement a geo-distributed architecture with regional data partitionin, minimizing latency throught data locality.

3.5.3 Security

The system processes sensitive user credentials, including personal email addresses and passwords, necessitating robust security mechanisms. To ensure data confidentiality, the system must implement end-to-end encryption for stored data, while all client-server communications must be secured via security protocols.

3.5.4 Maintainability

The system architecture must prioritize modularity and loose coupling between components to enable independent updates and minimize maintenance overhead. The development process must follow a modular design approach from inception, ensuring clear interface definitions, dependency management, facilitate isolated testing and enable parallel development workflows.

3.5.5 Portability

The system must achieve cross-platform compatibility across a diverse range of devices and operating systems. To meet this requirement, the technology stack should prioritize platform-independent frameworks and languages that support multiple execution environments without significant code modifications. Therefore, the core system logic must be abstracted from platform-specific dependencies to facilitate seamless deployment across various user devices.

4 Formal analysis using Alloy

In this section, we verify the consistency of the BBP system specifications using **Alloy 6**. We focused our modeling efforts on four critical aspects of the system:

1. **User Hierarchy and Permissions:** Ensuring a strict separation between Users and Registered Users who can contribute data.
2. **Activity lifecycle:** Veryfing that activities follow the desired path during their evolution.
3. **Data Governance Lifecycle:** Verifying that sensor data (Reports) follows a strict temporal evolution.

The model leverages Alloy 6's temporal logic capabilities to verify the dynamic behavior of the system over time.

4.1 Domain and Users

First, we define the physical domain and the user hierarchy. We explicitly model `RegisteredUser` as a specialization of the generic `User`. This structural constraint ensures that only registered users possess an email and the capability to be authors.

Listing 1: BBP Static Structure and Hierarchies

```
module BBP_System

// --- GENERAL DOMAIN ENTITIES ---
sig GeoPoint {}

sig Path {
    starting_point: one GeoPoint,
    ending_point: one GeoPoint
}

fact starting_point_diff_ending_point {
    all p: Path | p.starting_point != p.ending_point
}

sig DateTime {}

sig Ride {
    trip_path: one Path,
    starting_datetime: one DateTime,
    ending_datetime: one DateTime
}

fact starting_datetime_not_ending_datetime {
```

```
24          all t: Trip | t.starting_datetime != t.
                ending_datetime
25      }
26
27      // "User" represents any user of the system.
28      sig User {
29          user_trips: set Ride
30      }
31
32      // A trip belongs to exactly one user.
33      fact no_shared_trips {
34          all t: Trip | one user_trips.t
35      }
36
37      sig Email {}
38
39      // "RegisteredUser" is the only actor allowed to have an
            account and contribute.
40      sig RegisteredUser extends User {
41          email_address: one Email
42      }
43
44      // Constraint: One email per account.
45      fact single_mail_single_account {
46          all e: Email | one email_address.e
47      }
48
49      // enum listing all Activity states (see 2.1.3, Activity
            Lifecycle)
50      enum ActivityStatus {
51          START,
52          ROUTE_DEF,      // Route definition
53          ON,             // Ogoning
54          MON_ON,         // Monitored ongoing
55          PAUSE,          // Paused
56          WAIT_CONF,      // Waiting detections confirmation
57          WAIT_RAT,       // Waiting rating
58          WEAT_ENR,       // Waether enrichment
59          COMPLETED       // completed
60      }
61
62      sig Activity extends Ride {
63          var act_status : one ActivityStatus
64      }
65
66      // --- activity constraints
67      fact activity_only_to_registered_user {
68          all a : Activity | user_trips.a in RegisteredUser
69      }
70
```

```
71      fact ride_to_normal_users {
72          all r: Ride | r in Ride-Activity implies user_trips.
                r in User-RegisteredUser
73      }
```

4.2 Activity Lifecycle

In thie part we introduction the portion of the model to represent the evolution of an **Activity**.

Listing 2: Dynamic Logic and State Transitions of Activities

```
1
2       // --- ACTIVITY LIFECYCLE ---
3       fact init_activity_state {
4           all a : Activity | a.act_status = ROUTE_DEF
5       }
6
7       pred do_nothing_activity {
8           act_status' = act_status
9       }
10
11      pred start_activity [a : Activity] {
12              a.act_status =  ROUTE_DEF
13              all ac : Activity-a | ac.act_status' = ac.
                    act_status
14              a.act_status' = ON
15      }
16
17      pred start_monitored_activity [a : Activity] {
18              a.act_status =  ROUTE_DEF
19              all ac : Activity-a | ac.act_status' = ac.
                    act_status
20          a.act_status' = MON_ON
21      }
22
23      pred stop_activity [a : Activity] {
24              a.act_status in ON+MON_ON
25              all ac : Activity-a | ac.act_status' = ac.
                    act_status
26              a.act_status' = PAUSE
27      }
28
29      pred  resume_activity [a : Activity] {
30                  a.act_status = PAUSE
31                  all ac : Activity-a | ac.act_status' = ac.
                        act_status
32              a.act_status' in ON+MON_ON
33      }
34
```

```
35      // the two following facts are needed to keep the
            activity consistent with the choice of normal or
            monitored after PAUSE state
36      fact resume_activity_constraint_1 {
37          all a : Activity | always (
38              a.act_status = ON implies historically (a.
                    act_status != MON_ON)
39          )
40      }
41      fact resume_activity_constraint_2 {
42          all a : Activity | always (
43              a.act_status = MON_ON implies historically (a.
                    act_status != ON)
44          )
45      }
46
47      pred terminate_monitored_activity [a : Activity] {
48              a.act_status = MON_ON
49              all ac : Activity-a | ac.act_status' = ac.
                    act_status
50              a.act_status' = WAIT_CONF
51      }
52
53      pred rate_activity [a : Activity] {
54              a.act_status in ON+WAIT_CONF
55              all ac : Activity-a | ac.act_status' = ac.
                    act_status
56              a.act_status' = WAIT_RAT
57      }
58
59      pred get_weather_info [a : Activity] {
60              a.act_status = WAIT_RAT
61              all ac : Activity-a | ac.act_status' = ac.
                    act_status
62              a.act_status' = WEAT_ENR
63      }
64
65      pred complete_activity [a : Activity] {
66              a.act_status in WEAT_ENR+WAIT_RAT
67              all ac : Activity-a | ac.act_status' = ac.
                    act_status
68              a.act_status' = COMPLETED
69      }
70
71      // ensures that once an activity reaches the COMPLETED
            state, it will remain forever in it
72      fact final_activity_state_stability  {
73          all a : Activity | always (a.act_status = COMPLETED
                implies always a.act_status = COMPLETED)
74      }
```

```
75
76      fact valid_traces {
77          always (
78              do_nothing_activity or
79              ( some a : Activity | start_activity[a] ) or
80              ( some a : Activity | start_monitored_activity[a
                    ] ) or
81              ( some a : Activity | stop_activity[a] ) or
82              ( some a : Activity | resume_activity[a] ) or
83              ( some a : Activity |
                    terminate_monitored_activity[a] ) or
84              ( some a : Activity | rate_activity[a] ) or
85              ( some a : Activity | get_weather_info[a] ) or
86              ( some a : Activity | complete_activity[a] )
87          )
88      }
```

4.3 Data Lifecycle

Here we model the **Data Governance** process. We introduce the `Report` signature with a mutable field `status`. The `Stability` fact ensures that once a report is validated, it cannot revert to a pending state, preserving the integrity of the public inventory.

Listing 3: Dynamic Logic and State Transitions for Reports

```
1       // --- DATA GOVERNANCE ---
2       enum ReportStatus { Pending, Confirmed, Discarded }
3
4       sig Report {
5           // SECURITY: Only Registered Users can be authors.
6           author: one RegisteredUser,
7           refersTo: one Path,
8           // DYNAMIC: The status changes over time.
9           var report_status: one ReportStatus
10      }
11
12      // Initial State: All reports start as Pending.
13      fact Init {
14          all r: Report | r.report_status = Pending
15      }
16
17      // Stability: Terminal states are final.
18      fact Stability {
19          always (all r: Report | r.report_status = Confirmed
                implies
20                  always r.report_status = Confirmed)
21          always (all r: Report | r.report_status = Discarded
                implies
```

```
                always r.report_status = Discarded)
    }

    pred confirmReport [r: Report] {
        r.report_status = Pending
        r.report_status' = Confirmed
        all r2: Report - r | r2.report_status' = r2.
            report_status
    }

    pred discardReport [r: Report] {
        r.report_status = Pending
        r.report_status' = Discarded
        all r2: Report - r | r2.report_status' = r2.
            report_status
    }

    pred doNothing { report_status' = report_status }

    fact Traces {
        always (
            doNothing or
            (some r: Report | confirmReport[r]) or
            (some r: Report | discardReport[r])
        )
    }
```

4.4 Verification Predicates

To validate the model, we defined specific predicates that force the solver to search for instances satisfying our structural and temporal requirements.

The first predicate validates the structural separation of roles. It compels the solver to find a world containing both a simple Guest (with activity but no privileges) and a Registered User (with full privileges).

Listing 4: Predicate: User Role Distinction

```
    pred showUserDistinction {
        // We look for a user who is NOT registered but has
            trip activity
        some u: User {
            u not in RegisteredUser
            some u.user_trips
        }

        // We look for a registered user with associated
            trips and reports
        some ru: RegisteredUser {
            some ru.user_trips
```

```
            some r: Report | r.author = ru
        }
    }

    run showUserDistinction for 5
```

The second predicate validates the temporal logic. It ensures that the model allows a Report to evolve from its initial state to a finalized state, confirming the reachability of the "Confirmed" status.

Listing 5: Predicate: Activity lifecycle

```
    pred showActivityLifecycle {
        // just to make the model instance more readable
        no User-RegisteredUser
        no Activity-Ride
        ------------------------------
        one Path
        one Activity
        one RegisteredUser
        eventually ( some a : Activity | a.act_status =
            COMPLETED )
    }

    run showActivityLifecycle
```

Listing 6: Predicate: Activity lifecycle with pause

```
    pred showActivityLifecycleWithPause {
        // just to make the model instance more readable
        no User-RegisteredUser
        no Activity-Ride
        ------------------------------
        one Path
        one Activity
        one RegisteredUser
        eventually ( some a : Activity | a.act_status =
            COMPLETED )
        not (some a : Activity | a.act_status = COMPLETED)
            until (some a : Activity | a.act_status = PAUSE)
    }

    run showActivityLifecycleWithPause
```

Listing 7: Predicate: Activity lifecycle with weather enrichment

```
    pred showActivityLifecycleWithWeatherEnrichment {
        // just to make the model instance more readable
        no User-RegisteredUser
        no Activity-Ride
```

```
        ------------------------------
        one Path
        one Activity
        one RegisteredUser
        eventually ( some a : Activity | a.act_status =
            COMPLETED )
        not (some a : Activity | a.act_status = COMPLETED)
            until (some a : Activity | a.act_status =
            WEAT_ENR)
    }

    run showActivityLifecycleWithWeatherEnrichment
```

4.5 Verification Results

We executed the model to verify two key scenarios. The solver successfully generated consistent instances for both, proving the validity of the specifications.

4.5.1 Scenario 1: User Distinction

The first run highlights the structural difference between a generic `User` and a `RegisteredUser`. As shown in Figure 9, the system correctly isolates the capabilities: only the registered user is associated with an email and content authorship.

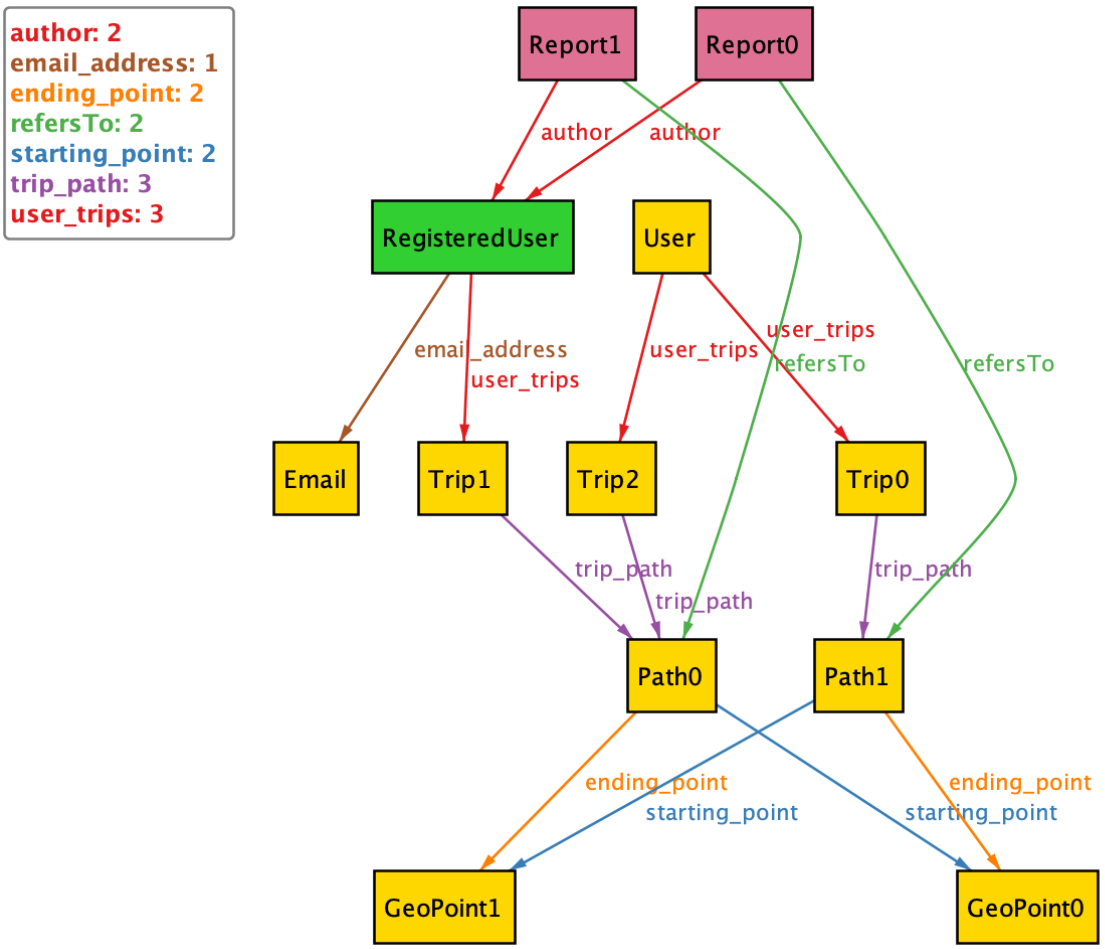


Figure 9: Alloy World generation showing the distinction between a Generic User and a Registered User.

4.5.2 Scenario 2: Activity Lifecycle

The second round of run explores the evolution of an Activity.

- Figures 10,11,12 show the evolution of an activity from the beginnning to its completition, without retriving weather data (the figures shown the transiton only throught the `Ogoing` state, however the evolution is the same also by passing for the `Monitored ongoing` state).
- Figures 13, 14 show the case in which an activity is paused and then resumed by the user.
- Figures 15, 16 show the case in which weather data are available for the

terminated activity.

To avoid repeating the same steps, for the last two point only relevant changes in the evolution are shown.

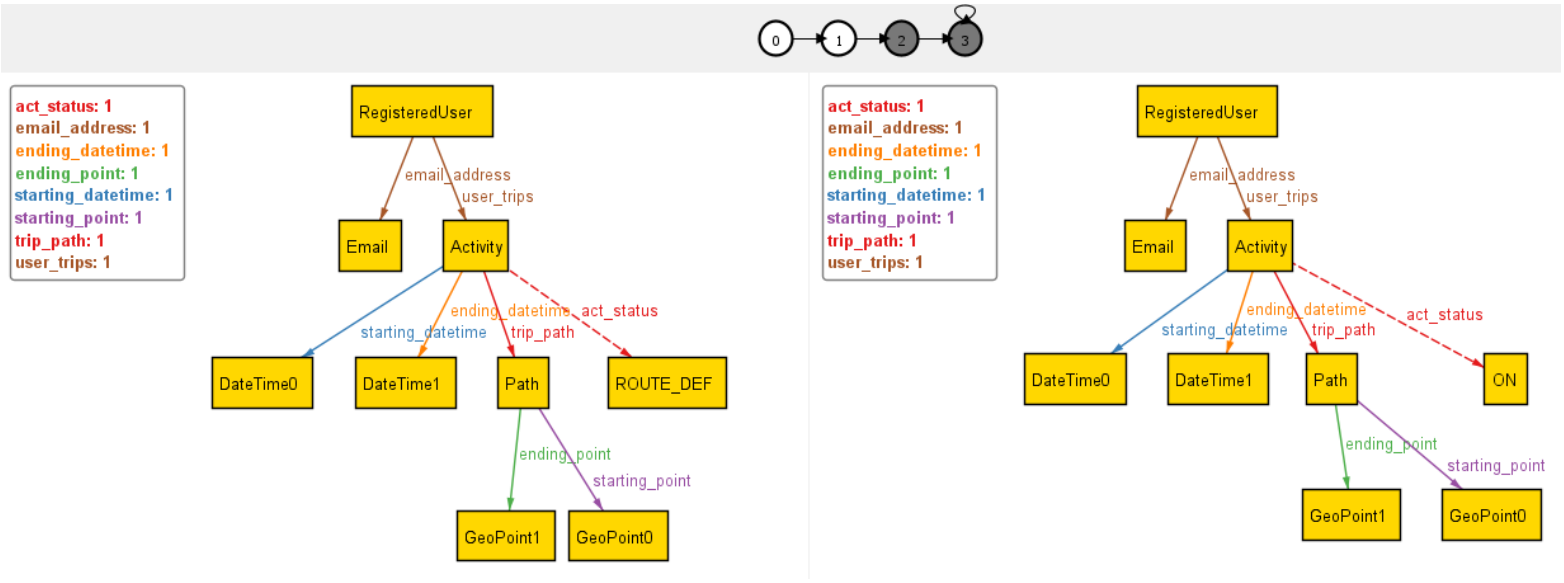


Figure 10: Step T0, T1 - The Activity starts in the `Route definition(ROUTE_DEF)` state. Once the user starts it, the activity transitions in the `Ongoing(ON)`.

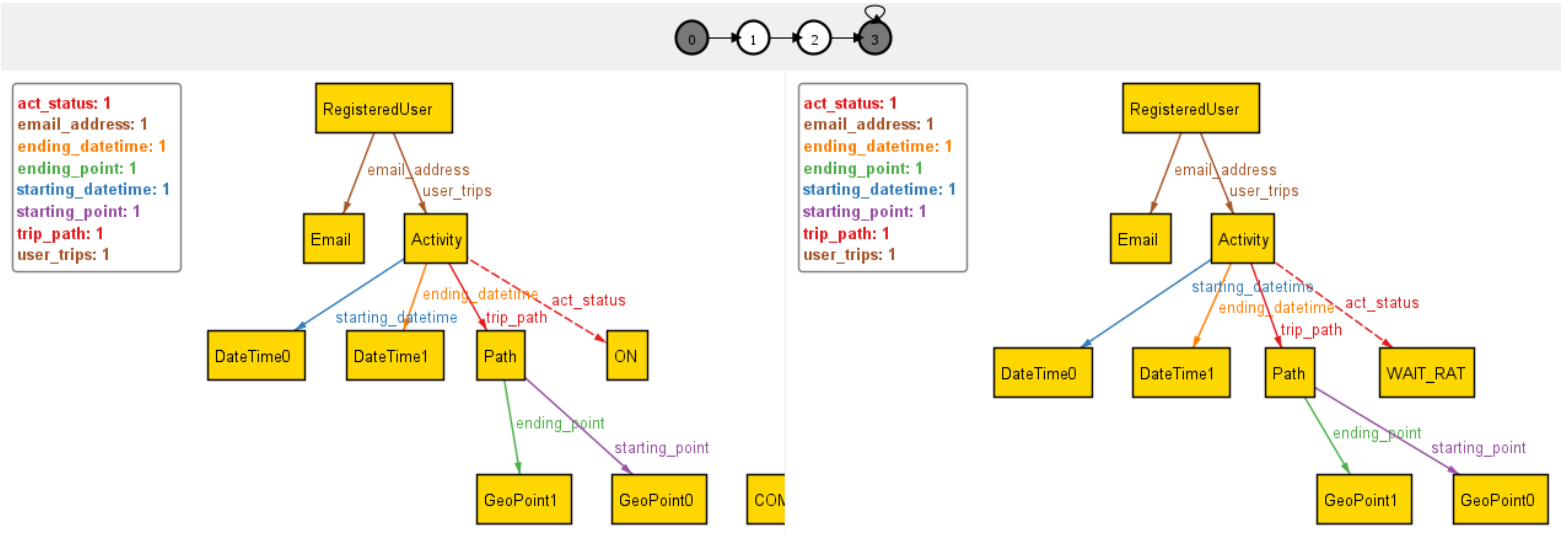


Figure 11: Step T1, T2 - The Activity is in the `Ongoing(ON)` state. Once stopped, it transitions to `Waiting rating(WAITING_RAT)` in which is waiting for the user to rate the path.

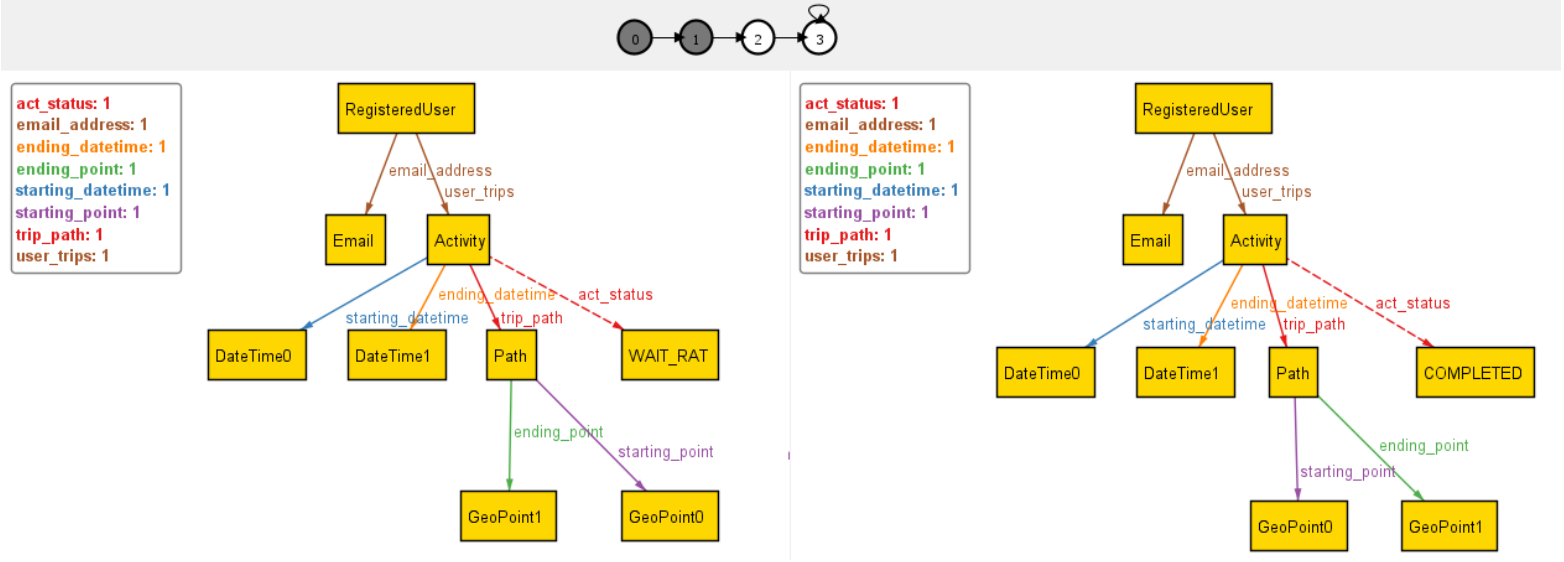


Figure 12: Step T2, T3 - The Activity is in the `Waiting rating(WAITING_RAT)` state. After receving the rating, the Activity transtions in the `COMPLETED` state and will remain in it.

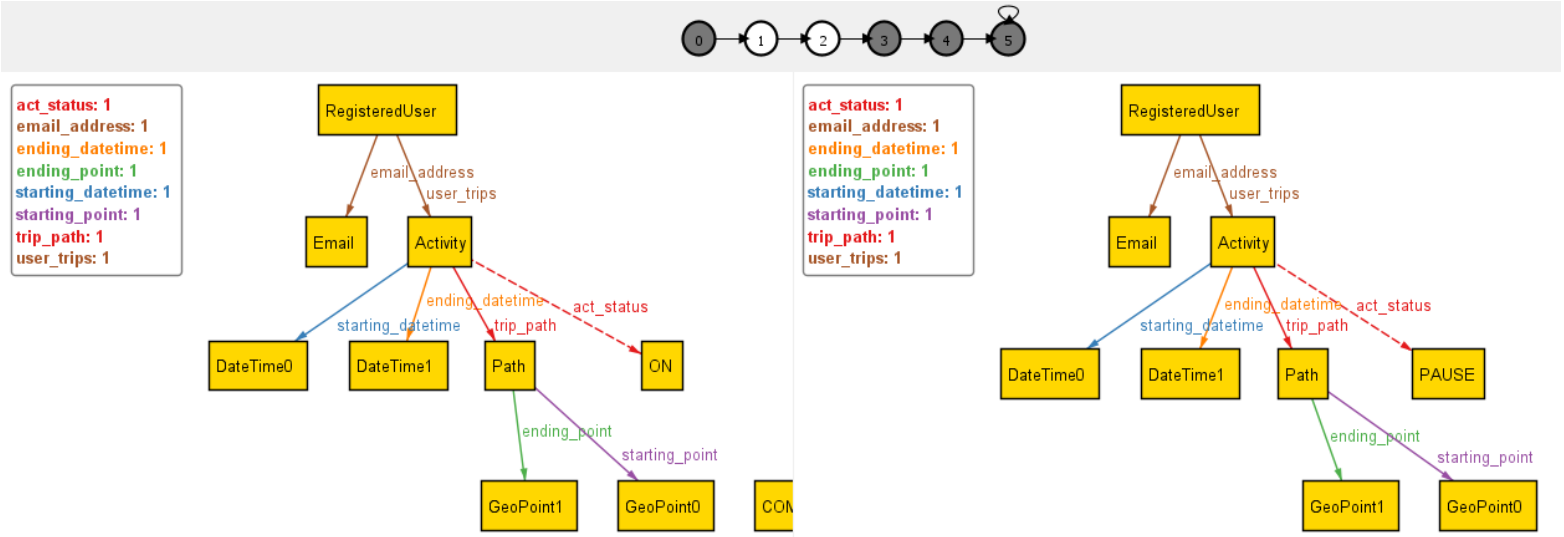


Figure 13: Step T1, T2 - The Activity is in the `Ongoing(ON)` state. The user might stop it, therefore it transitions in the `PAUSE` state.

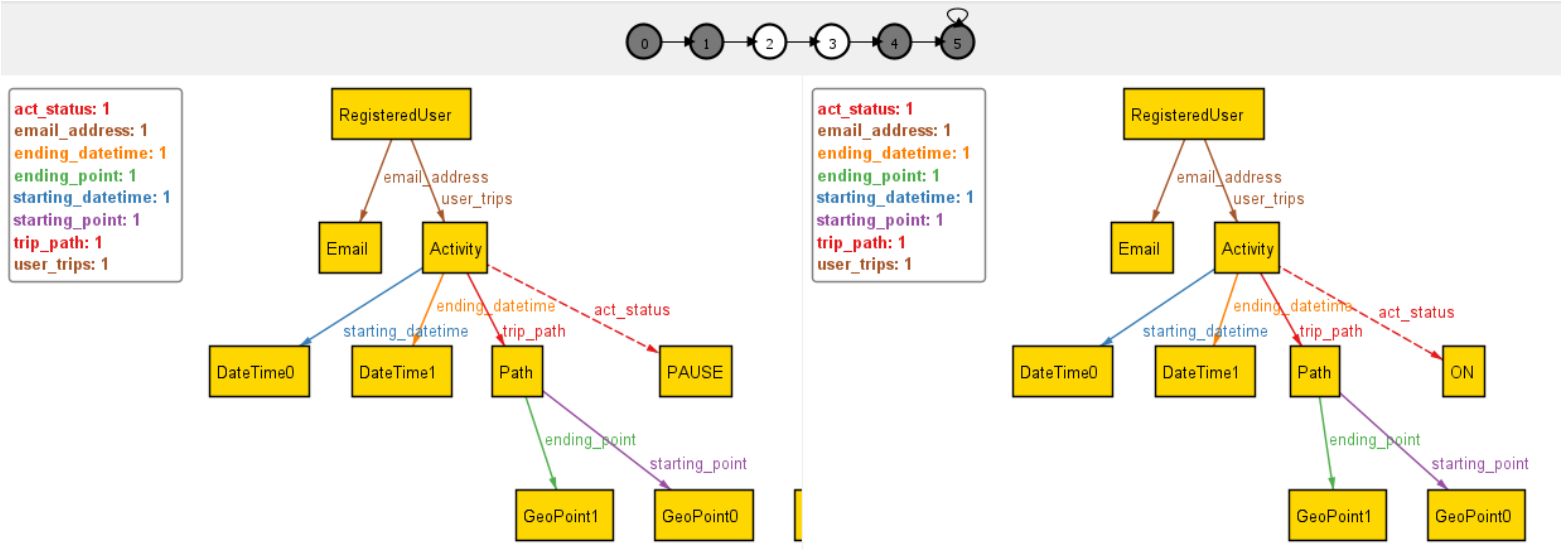


Figure 14: Step T2, T3 - The Activity is in the `Waiting rating(PAUSE)` state. Once the user resumes the activity, it transitions back in the `Ongoing(ON)` state (the model avoid to transition in the wrong previou state e.g. `MON_ON`).

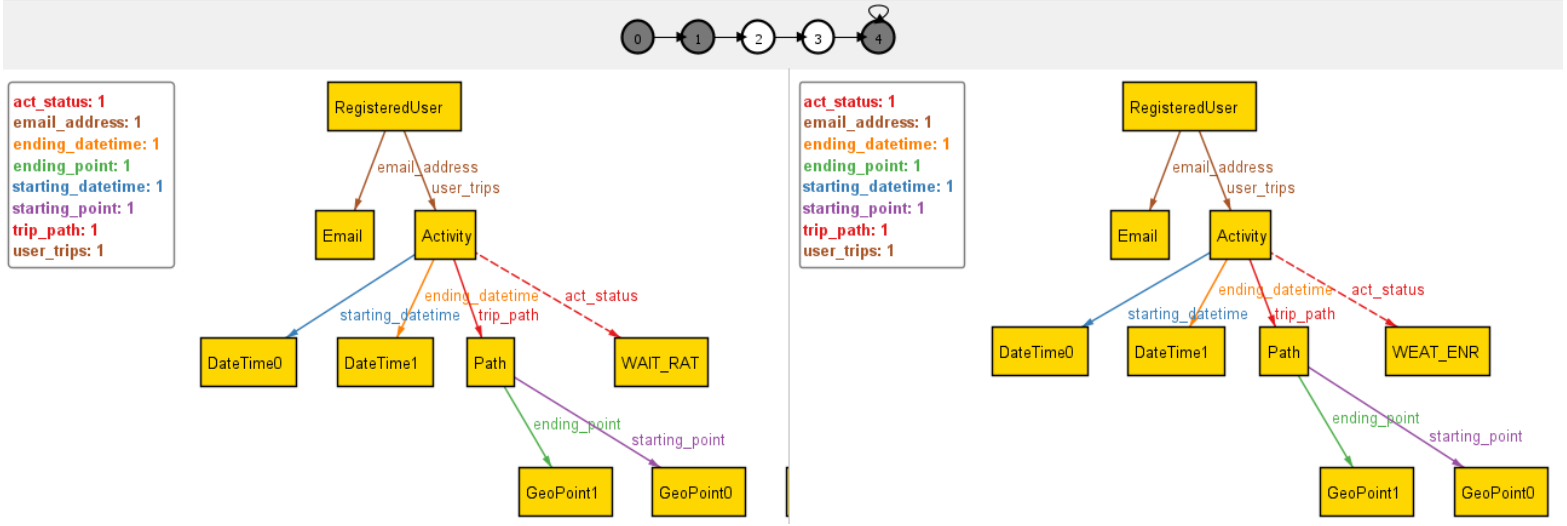


Figure 15: Step T2, T3 - The Activity is in the `Waiting rating(WAITING_RAT)` state. Now the weather data during the activity is available, so it transition in the `Weather enrichment(WEAT_ENR)` state.

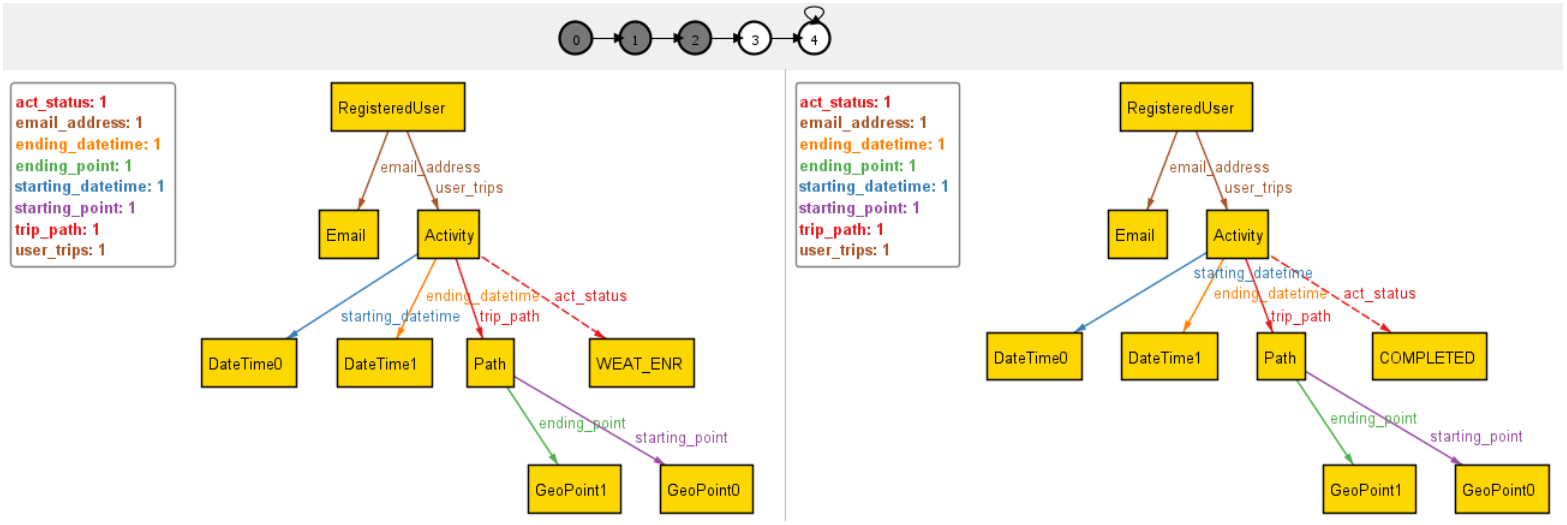


Figure 16: Step T3, T4 - The Activity is in the `Weather enrichment(WEAT_ENR)` state. Once weather data about the activity are correctly loaded, The activity transtions in the `COMPLETED` state.

4.5.3 Scenario 3: Data Lifecycle Evolution

The third run verifies the temporal evolution of a Report. Figure 17 illustrates the transition satisfying the Data Governance requirements defined in Section 3, depicting the evolution of the system state over time:

- **Step T0 (Pending):** The initial state where the Report is created. It is associated with a Path and an Author, but the status is `Pending`, meaning it is not yet public.
- **Step T1 (Confirmed):** The subsequent state where the Registered User validates the report. The status transitions to `Confirmed`, preserving the data integrity defined in the Stability fact.

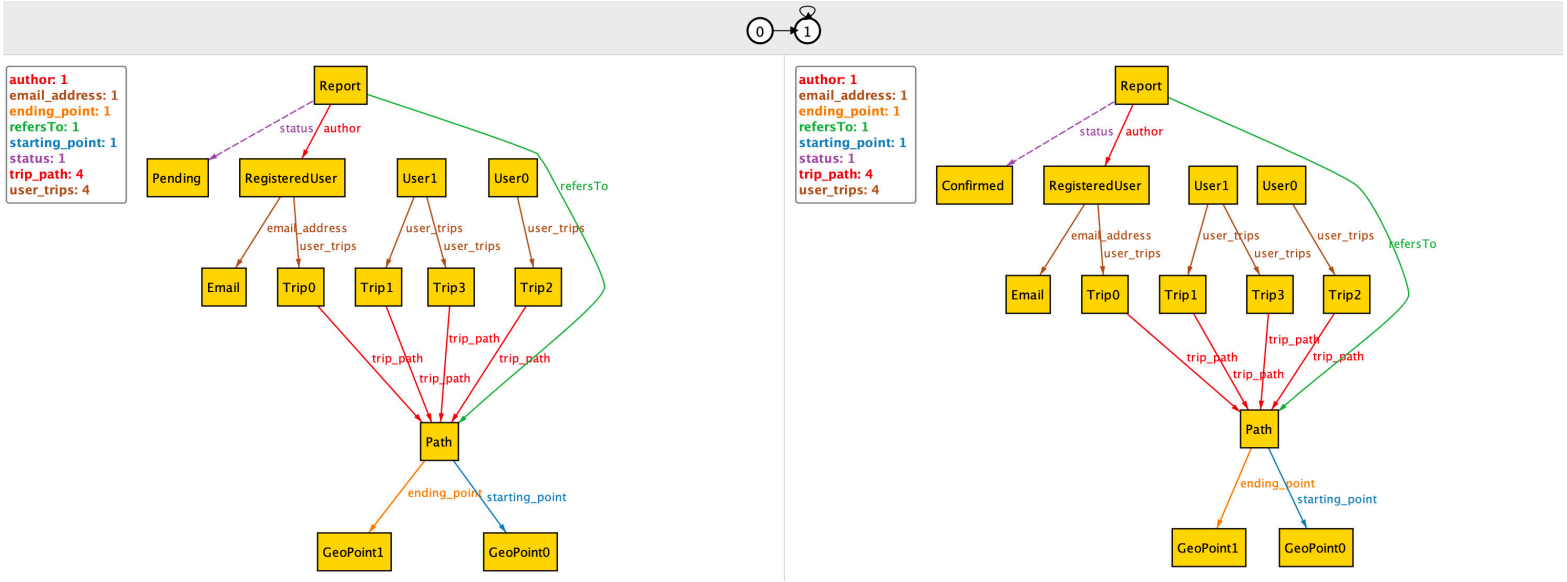


Figure 17: Temporal evolution of a Report: transition from 'Pending' (T0) to 'Confirmed' (T1) status.

5 Effort Spent

Student	Section 1	Section 2	Section 3	Section 4
Guglielmi Leonardo	2h	7h	6h	0h
Lo Conte Francesco	4h	8h	10h	4h

Table 21: Effort spent by each team member per section

6 References