

D.3.4.3 INNOVATIVE SOLUTION ON MOBILITY HABITS TO RAISE AWARENESS ON THE EFFECTS OF RECKLESS MOBILITY CHOICES FROM CITIZENS

Document Control Sheet

Project Number:	10249002
Project Acronym	MIMOSA
Project Title	Maritime and Multimodal Sustainable passenger transport solutions and services
Start Date	01/01/2020
End Date	30/06/2023
Duration	42 months
Related Activity:	A.3.4. Enhancing the knowledge of sustainable mobility options
Deliverable Name:	D.3.4.3 Innovative solution on mobility habits. To raise awareness on the effects of reckless mobility choices from citizen.
Type of Deliverable	Study
Language	English
Work Package Title	Increasing the knowledge of passenger transport and of passenger behaviour
Work Package Number	3
Work Package Leader	PP2 (Ca' Foscari University of Venice)
Status	Final version
Author(s)	PP2
Version	1
Due Date of Deliverable	5/2023
Delivery Date	6/2023

Table of contents

1 Introduction	6
1.1 Real-time data and Public Transport	6
1.2 The Impact of Real-Time Data Visualization on Public Transport	7
1.3 Changing Mobility Behaviours	7
1.4 Evolution of Mobility Apps and Augmented Reality in Public Transport	8
Case Study: Google Live View	9
Case Study: Flight Radar 24	11
2. MIMOSA Mobility AR Application Architecture	13
2.1 Modular Layered Architecture	13
Level 1: Real-time data	14
Level 2: Services and Algorithms	15
Level 3: Augmented Reality Front-end	16
2.2 Architecture Implementation	16
Microservices Table	17
Clients API Table	18
Trip Planning and Analytics	19
Transport Data Connector (GTFS)	20
GTFS Realtime Data	21
GTFS API Implementation	23
Vehicle Position Prediction	24
NeTex Connector	24
Polls and Surveys	25
Polls API	25
Polls Dashboard	27

Home	27
Create and edit polls	28
Poll detail	29
Poll statistics	30
User and Activity Tracking	31
Tracking DATA API	31
Tracking Data	32
Interactive map	33
Spatial Cluster Analysis	33
2.3 Augmented Reality Application	47
Goal-Focused User Experience	47
Implementation of the Augmented Reality View	47
User Navigation	49
Travel by Destination	50
Travel by Route	51
Route visualization	52
Live Mode – 2D Map	53
Live Mode – 2D map to 3D AR Transition	54
Live Mode – Augmented Reality View	55
Live Mode – Augmented Reality Directions	56
3 Open Source and Deployment	57
3.1 Re-use of the codebase	57
3.2 Source Code Repository	58
3.3 Future Development	59
4. Pilot Project	61
4.1 Testing Area	61

4.2 Terms of Service and Privacy Policy	61
4.3 Social Media Campaign	61
4.4 Pilot Project statistics and Results	64
5. Conclusions	65
ANNEX 1. Open-Source Libraries for the AR Application	67
Analysis and compliance (MIT License)	68
ANNEX 2. Cloud Service Deployment	70
Bibliography	71

1 Introduction

This Mimosa pilot action (D.3.4.3) is related to the development of an innovative solution on mobility habits aimed to raise awareness of the effects of reckless mobility choices from citizens. As defined in the Application Form, people, appropriately profiled with a preliminary questionnaire, can obtain more detailed and real-time information on public transport. This information can be considered as disruptive as it pushes people to use public transport, reduce pollution, congestion and increase the liveability of our cities, thanks to a behavioural change process.

Considering all these key challenges, this MIMOSA pilot tested in the Emilia-Romagna territory an innovative App developed with an open-source approach in order to be easily replicated in all the other Italy-Croatia and EU territories.

This deliverable explains in detail how the MIMOSA App was built and the key sustainable mobility objectives it tried to reach.

Paragraph 1 introduces the key topics of the pilot action with a focus on the importance of public transport real-time data and it presented other international case studies that inspired the idea of the MIMOSA App.

Paragraph 2 is a technical paragraph explaining the MIMOSA Augment Reality (AR) Application Architecture and how it was possible to develop such an innovative function.

Paragraph 3 focuses on the Open-Source approach used to develop the MIMOSA App. This paragraph has a fundamental role in order to understand the replicability factors of the Mimosa App.

Paragraph 4 presents the key results of the testing pilot conducted in the Emilia-Romagna Region during the Mimosa project.

Finally, the Annex 1 provides all the information related to the GitHub repository where all the code developed during the Mimosa project is available.

1. 1 Real-time data and Public Transport

Public transportation plays a crucial role in creating sustainable and efficient urban mobility systems. As cities continue to grow and face the challenges of congestion, pollution, and limited resources, encouraging the use of public transport becomes imperative.

To achieve this, it is essential to provide passengers with reliable, up-to-date, and easily accessible information about public transport services. The advent of real-time data visualization, facilitated by the widespread use of mobile applications, has revolutionized the way commuters interact with public transport systems.

This report explores the transformative potential of utilizing real-time data visualization, through mobile apps and Augmented Reality, to promote the use of public transport, describing the MIMOSA Mobilità AR project.

By presenting passengers with dynamic and accurate information, this technology empowers them to make informed decisions, experience improved journey planning, and ultimately opt for public transport as their preferred mode of transportation.

1.2 The Impact of Real-Time Data Visualization on Public Transport

Numerous scientific studies have shed light on the positive impact of real-time data visualization in promoting public transport usage. For instance, a past study (Susilo, 2015) investigated the influence of mobile apps providing real-time transit information on travel behavior. The results revealed that access to real-time data significantly increased the likelihood of choosing public transport over private vehicles, leading to a reduction in traffic congestion and greenhouse gas emissions.

Furthermore, another research paper (Cats, 2017) emphasized the importance of accurate and timely information in improving the overall passenger experience and satisfaction. It demonstrated that when commuters have access to real-time data, such as vehicle locations, arrival times, and service disruptions, they exhibit higher levels of trust in the system, reduced perceived waiting times, and increased perceived reliability. This positive experience contributes to a greater inclination to utilize public transport services.

Additionally, a scientific study (Hong, 2016) focused on the role of mobile apps in enhancing the accessibility of public transport. By leveraging real-time data visualization, users could conveniently plan their journeys, track routes, and receive personalized recommendations based on their preferences and real-time conditions. This level of convenience and customization further incentivizes individuals to opt for public transport as a reliable and efficient mode of travel.

1.3 Changing Mobility Behaviours

MIMOSA deliverable D.4.3 is focused on how real-time information about public transport can significantly impact the perception and usage of users, beyond passenger satisfaction and enhanced accessibility, thus realizing a real change in the behaviour in encouraging individuals to choose public transport over private vehicles.

Numerous recent studies and experiments have highlighted this transformation and its positive impact on urban mobility. For instance, a scientific study (Tang, 2020) investigated the influence of real-time transit information on mode choice behavior. The results revealed that access to

accurate and timely data significantly increased the likelihood of individuals selecting public transport options, leading to a shift away from private vehicles.

This finding was further supported by a study by (Schmöcker, 2019), which emphasized the importance of real-time information in reducing travel uncertainty and increasing the attractiveness of public transport. Moreover, a field experiment conducted by (Li, 2018) demonstrated that providing real-time transit information through mobile apps resulted in a substantial increase in public transport ridership. These studies collectively reinforce the notion that real-time data plays a pivotal role in shaping user behavior, making public transport a more viable and appealing choice for commuters.

1.4 Evolution of Mobility Apps and Augmented Reality in Public Transport

Info mobility applications have undergone significant evolution since their inception, driven by advancements in digital technologies and increasing smartphone penetration. Their journey traces back to simple map-based applications, then progressed to real-time data integration, and now has arrived at the incorporation of **Augmented Reality (AR)**.

Augmented Reality (AR) is an innovative technology that overlays virtual elements into the real world, enhancing user perception and interaction with the surroundings. Unlike Virtual Reality (VR), which creates a fully immersive digital environment, AR enhances the real world by adding digital content in real time. AR enables users to seamlessly blend virtual and physical elements, providing a more immersive and interactive experience. In recent years, AR has gained significant popularity and has become a prominent trend in various industries, including gaming, education, healthcare, and marketing. Advancements in hardware, such as smartphones and smart glasses, along with the development of powerful AR software frameworks and tools, have fuelled the growth and adoption of AR technology. Popular AR tools and platforms, such as (Google ARCore, s.d.), (Apple ARKit, s.d.) and lately the introduction of Apple Vision Pro and the (Apple VisionOS, s.d.), have empowered developers to create engaging AR applications and experiences for users. As the technology continues to evolve, we can expect to see more sophisticated AR applications and increased integration into our daily lives, transforming the way we interact with the world around us.

Initially, mobile applications for info mobility mainly included map services and static timetable information. Examples of these services include MapQuest and early versions of Google Maps. Nowadays nearly every local TPL institution has developed its own info mobility app, like Trenitalia, “daAaB” and ROGER for the Emilia Romagna Region, these applications provide basic location information and help users in route planning but lacked a real and user interaction with the surroundings.

As technology advanced, real-time data integration became the standard for info mobility applications. Services like Citymapper and updated versions of Google Maps began incorporating real-time updates on traffic, public transport schedules, route suggestions, and even bike-sharing

availability. This change led to increased user satisfaction, as it allowed for more dynamic and adaptive planning (Watkins, 2011).

Recently, we have seen the introduction of Augmented Reality (AR) into info mobility applications, representing the next frontier in this evolution. AR provides a more intuitive and immersive way of presenting information, making navigation in unfamiliar environments easier. The integration of AR in Google Live View and Apple's Look Around feature are prime examples. These applications overlay digital information onto the user's physical environment, providing real-time, context-specific directions.

AR in info mobility is still in its early stages, but the potential impact is significant. As seen in studies by (Schöning, 2020) (Wang, 2021), AR has the potential to further improve user experience, reduce navigational errors, and even influence mobility behaviors.

The next sections will introduce two important use cases of mobility Augmented Reality apps that revolutionize the way users interact with their surroundings.

Case Study: Google Live View

Google has been a frontrunner in implementing Augmented Reality (AR) for info mobility purposes. One of its notable initiatives in this space is Google Live View. This feature, integrated into Google Maps, uses AR to provide real-time, turn-by-turn walking directions overlaid on the user's camera view.

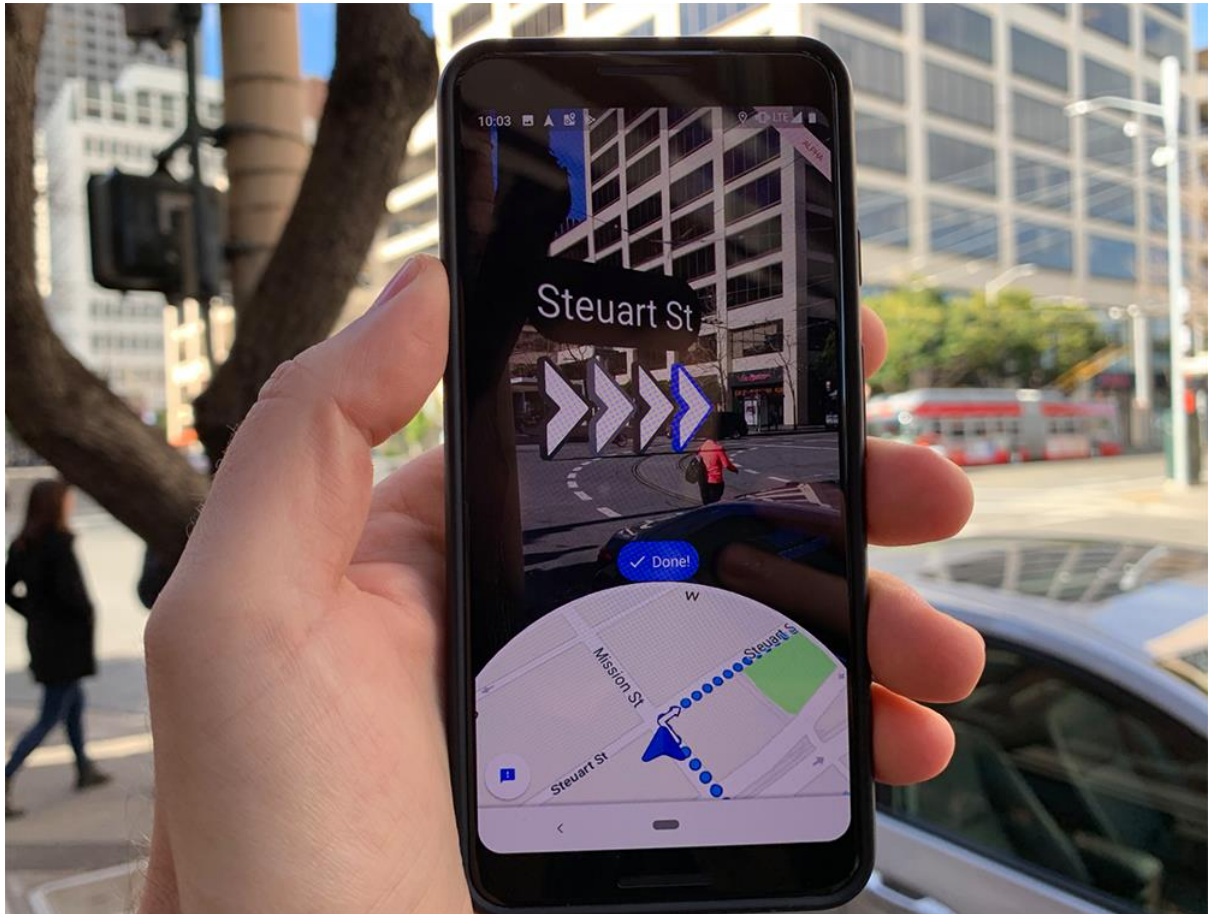


Figure 1 - Google Live View

The aforementioned study (Schöning, 2020) explores the implementation of AR in Google Maps, including the Live View feature. The research highlights how the technology significantly improves navigation by providing users with real-time and contextually relevant information. The study reveals that the use of Google Live View not only enhances the user experience but also reduces instances of disorientation and navigational errors.

AR can assist users in providing spatial knowledge and reducing cognitive load for users during their journey. The results demonstrated a positive effect on user mobility behavior, with a noted increase in confidence while navigating unfamiliar environments.

In addition to these scholarly studies, several anecdotal reports suggest that Google Live View is significantly influencing the way people navigate urban environments. It provides a more intuitive and user-friendly way to navigate, especially in complex or unfamiliar urban spaces. As such, the technology has the potential to change mobility behavior by encouraging users to explore and navigate urban environments independently, without the fear of getting lost.

Google's efforts in AR, particularly with Google Live View, play a pivotal role in info mobility. It provides real-time, context-specific information in a user-friendly format, enhancing the overall user experience and potentially influencing mobility behavior towards more sustainable and efficient practices.

However, the technology used by the Live View requires large amount of client-server interaction and active image-recognition which limits the application of such technology to very update smartphones and locks the usage to Google services.

MIMOSA Mobilità AR application is instead focused on being completely independent from these services, as well as being able to run also on earlier generation smartphones.

Case Study: Flight Radar 24

Another good example of AR mobility App is FlightRadar24, which provides real-time aircraft flight information to users all over the world. The app presents data collected from several sources including radar, ADS-B (Automatic Dependent Surveillance–Broadcast), MLAT (Multilateration), and Flarm. This information enables users to track flights, understand flight paths, view detailed aircraft information, and receive updates on departures, arrivals, and delays.

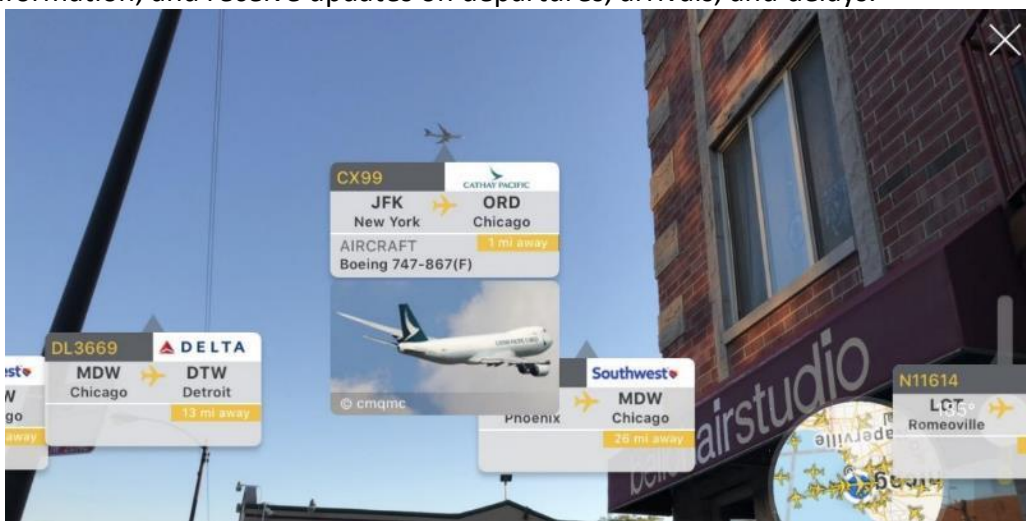


Figure 2 - FlightRadar24

What sets FlightRadar24 apart is its incorporation of augmented reality (AR). Users can point their mobile device towards a visible aircraft in the sky and, thanks to AR, the app overlays real-time, contextual information about the flight onto the live camera view. This information can include flight number, aircraft type, origin, destination, and current speed and altitude.

AR enhances the FlightRadar24 experience by transforming the way users interact with flight data. Instead of solely relying on tabulated data and flight schedules, users can now literally look up into the sky and receive geo-referenced information on flights in real-time. This immersive AR experience adds a layer of engagement that traditional data presentation methods cannot

replicate. It's a more intuitive way for users to interact with flight information and provides them with a spatial and temporal understanding of the aircraft's position. This feature makes FlightRadar24 not just a tool for obtaining flight information, but also an interactive learning platform that brings the world of aviation to the user's fingertips in an engaging and visually compelling manner.

FlightRadar24's augmented reality (AR) feature is a complex integration of real-time flight data, global positioning system (GPS) technology, and smartphone sensors to create an engaging user experience and it is interesting for the MIMOSA project because a very similar approach has been used.

2. MIMOSA Mobility AR Application Architecture

This paragraph introduces the MIMOSA Mobility AR Application and its server architecture designed to enhance the usability and effectiveness of public transport systems.

The application aims to provide users with a seamless and intuitive experience by leveraging real-time data, relative positioning technology, and AR visualization. In this paragraph, we will explore the modular architecture and development choices behind the MIMOSA Mobility AR Application, both in terms of server-side cloud-based services and the front-end mobile AR view. The sections that follow will delve into the architectural components, the adoption of open standards for data integration, the utilization of open-source libraries and modules, as well as the design methodologies employed to ensure scalability, flexibility, and a novel user experience.

2.1 Modular Layered Architecture

The server-side architecture of the MIMOSA project was designed to meet a set of goals centred around modularity, openness, extensibility, and simplicity of deployment. At the heart of the design lies the commitment to a layered, modular architecture that clearly separates the various functions of the system. This design allows for better maintainability, easier testing, and the ability to extend or replace individual components without impacting the whole system.

The architecture embraces open standards, most notably the General Transit Feed Specification (GTFS), which is widely used for public transport data. This choice ensures compatibility with other systems, future-proofs the project, and aligns it with the global trend towards open data in public transportation.

In the spirit of openness and extensibility, the architecture utilizes only open-source modules and libraries. This approach not only keeps the project transparent and trustworthy but also encourages community involvement, allowing anyone to fork and extend the project. It also ensures that the MIMOSA project is not tied to any proprietary technology, making it a more flexible and accessible solution.

Regarding the simplicity of deployment, the architecture is designed to be easily deployed in a variety of environments. Whether it's a cloud-based deployment or an on-premise installation, the MIMOSA project's architecture ensures a streamlined deployment process. This is achieved through a clear installation guide that eases the deployment process of the different open-source modules to various virtual machine instances (detailed in the following sections).

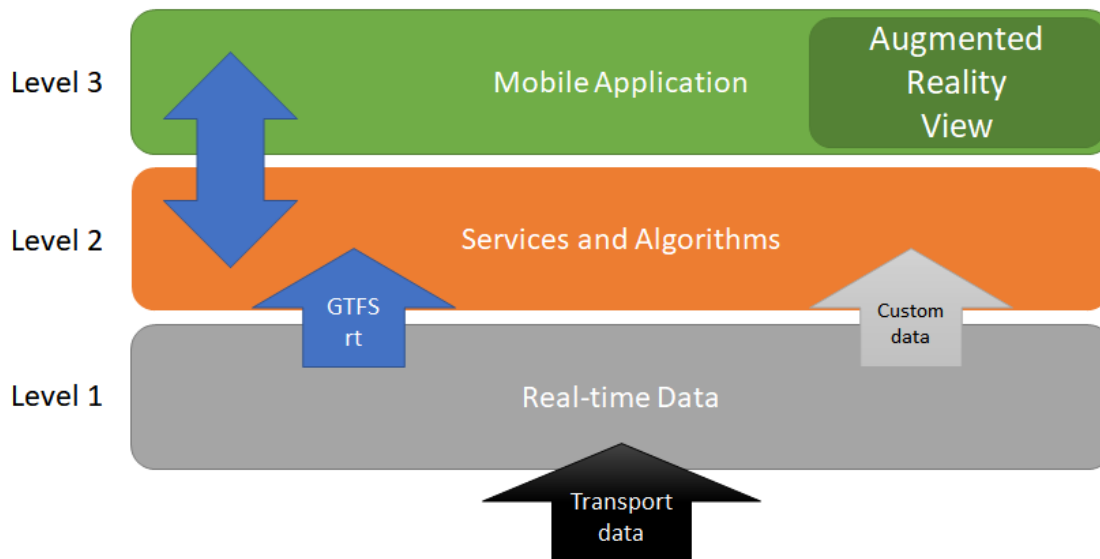


Figure 3 - MIMOSA Mobilità AR architecture

Level 1: Real-time data

An essential aspect of Level 1 is its interplay with existing infrastructures. It leverages the data provided by specific sensors that are already installed onboard public transport vehicles and along their routes. The goal of this layer is to explore how this data can be sampled and processed to yield a higher degree of precision, ultimately translating into a richer and more reliable stream of real-time data. It's essential that this enhanced accuracy is achieved without altering current features and functions in use. In other words, any improvements in data sampling and precision should enhance, not disrupt, the established system.

To further bolster the reliability of the data, Level 1 investigates the possibility of incorporating additional devices and mobile applications equipped with GPS technology into the transport vehicles and, by extension, the drivers' arsenal. It enhances the accuracy of real-time data by processing both older-generation tracking technologies (AVM¹) as well as new GPS-based positioning, solidifying the system's commitment to data reliability and flexibility.

¹ AVM (Automatic Vehicle Monitoring). The Automatic Vehicle Monitoring System is GPS-based vehicle location information system in real time, showing those information on a map. Further, it correlates vehicle information with customer-specific information and streamlines exchange of information between a dispatcher and vehicles.

Level 2: Services and Algorithms

Level 2 of the MIMOSA project's architecture builds upon the solid foundation of reliable data provided by Level 1. This layer's primary function is to apply sophisticated wayfinding and profiling algorithms, all tailored to the project's specific goals and needs. This isn't already about presenting data to users; it's about processing both real-time data and requests coming from the higher levels (Mobile AR Application) that data intelligently to guide them to their destinations in the most efficient and intuitive way possible.

The algorithms at this level consider various metrics to optimize the user's journey. These metrics include travel times, costs, waiting periods, and even events that occur over time. By evaluating and balancing these factors, the algorithms can guide users towards routes that meet their specific needs and preferences. Whether the user prioritizes speed, cost-effectiveness, minimal waiting time, or some other factor, Level 2's algorithms are designed to deliver a customized and optimized navigation experience.

Beyond wayfinding and profiling, Level 2 plays a critical role in managing and structuring the project's data sources. It receives unstructured data and applies advanced filtering and analysis methods to convert this raw data into a structured format to be ingested into wayfinding and AR visualization services. The data processed can be both in GTFS and NeTex formats.

GTFS (along with GTFS-real-time) is a data format and specification that allows public transportation agencies to provide real-time updates and information about their services. It stands for General Transit Feed Specification Realtime and was initially developed by Google in collaboration with various transit agencies. GTFS Realtime provides a standardized way to communicate data such as vehicle locations, arrival and departure times, service alerts, and more in a real-time manner. This data can be consumed by applications and services to provide up-to-date information to passengers, enhancing their travel experience and improving their ability to plan their journeys efficiently.

NeTex, on the other hand, is a CEN Technical Standard for exchanging Public Transport schedules and related data. It defines a standard for exchanging public transport passenger information data in XML format. The functional scope of NeTex is divided into three parts, each covering a functional subset of the CEN Transmodel conceptual model for Public Transport Information. Part 1 describes the fixed Network (stops, routes, lines, etc.); Part 2 is mainly focused on Timetables and Part 3 covers Fare data. All three parts use the same framework of reusable components, versioning mechanism, validity conditions, support to allow the uniquely identification of data elements in a global context.

The NeTex and GTFS formats “should be considered as complementary, covering different stages in the data management process: NeTex is “upstream”, GTFS is “downstream”. NeTex differs from GTFS in that it has a much wider scope, and that it is intended for use in back-office use cases under which data is generated, refined and integrated (requiring the exchange of additional elements used to construct the timetable), rather than just for provisioning journey planning systems (the prime purpose of GTFS)”.

The data structuring function of Level 2 is crucial for the MIMOSA architecture as it also enables the extraction of meaningful insights from the data and make informed decisions about the platform's evolution. It could potentially reveal usage patterns, identify areas for improvement, and even predict future trends, all of which could guide the platform's further development and refinement.

Level 3: Augmented Reality Front-end

Level 3 of the MIMOSA project's architecture addresses the model of user interaction, specifically focusing on the front-end application for Android and iOS mobile devices. This level constitutes the user-facing side of the system, embodying the principle that a system is as good as its ability to serve and satisfy its users.

The design of Level 3 has been shaped by a thorough preliminary study of various existing navigation applications, both global and local. This research highlights diverse approaches in the landscape of navigation apps. For instance, some applications, like Rome2Rio, owe their success to their ability to deliver clear and immediate results with fewer steps than other applications. These insights provide valuable cues for designing the front-end application of the MIMOSA project. User experience (UX) is a central concern at Level 3. The design recognizes that modern users are accustomed to high-speed interaction with data and are increasingly demanding more specific, advanced, and fast features. Hence, the UX design of the application must be flexible and responsive enough to meet these expectations. It needs to deliver clear and concise results swiftly, minimizing the steps and complexity involved in achieving the user's goals.

2.2 Architecture Implementation

After introducing the high-level view of the architecture, this section describes the implementation and deploy details.

MIMOSA project implements the described architecture using a Microservices approach. Each component is responsible for performing a specific logical task. The overall operation of the application is the coordinated sum of all components, for each of which the most appropriate technology was chosen based on the specific performance need.

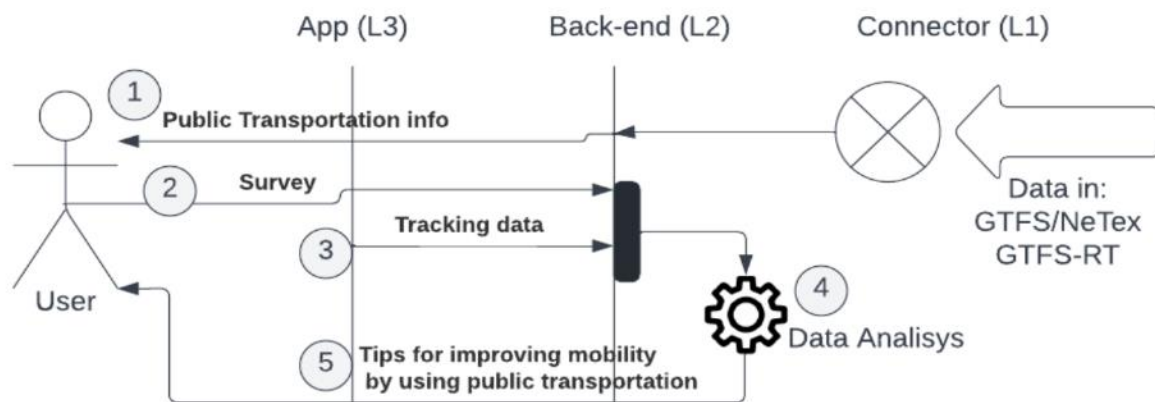


Figure 4 - General User Use Case

Figure 4 shows the general Use Case regarding the interaction of the user with the system. Data coming from the Connector (Level 1) feeds the central intelligence services in Level 2.

Level 2 main focus is providing the user with Public Transportation information (1) (through the AR Mobile app) and also distributes surveys (2) while receiving continuous tracking data from the User device (3). The offline analysis of this data allows Level 2 to provide the User with tips for improving its mobility habits (5).

Microservices Table

The following table describes the API modules developed for the Level 1 and Level 2 services. Different tech stacks and languages have been used depending on the availability of libraries and ease of development (smaller code footprint, faster performance).

In particular, the retrieving tasks regarding data sources not frequently updated (GTFS timetable) have been implemented in Python language, while the real-time data (GTFS real-time) has been processed using the performant Go language and its libraries.

Services directly consumed from the Mobile application have been implemented in NodeJS for the availability of serialization libraries and well-known and robust communication protocols.

Name	Description	Technology Stack
Connector	Retrieves data from an external source and prepares it for use.	Python
GTFS API	Exposes static and real-time GTFS data that are consumed by the application.	Go
POLLS API	API for the creation of user polls, to be distributed through the mobile application.	NodeJS
Tracking DATA API	It receives location data from devices and stores it on a NO-SQL database for analysis purposes.	NodeJS

Clients API Table

Related to the consumption of the Micro-services, here are the client libraries used on the MIMOSA mobile application to retrieve and process server data.

Flutter is the core technology used in order to develop the application cross-platform, working on both iOS and Android mobile operating systems, while Angular is the framework used to interact with the POLLS and Tracking APIS.

Name	Back-end	Technology Stack
App Mobilità AR	GTFS-API, OTP, POOLS-API	Flutter
Back-end client	POOLS-API, Tracking DATA API	Angular

The three APIs with which the AR Mobility App interacts are accessed through a **Proxy**, so the App is unaware of the complexity behind the scene. The Back-end client uses the same Proxy but can interact only with the Polls and Tracking API.

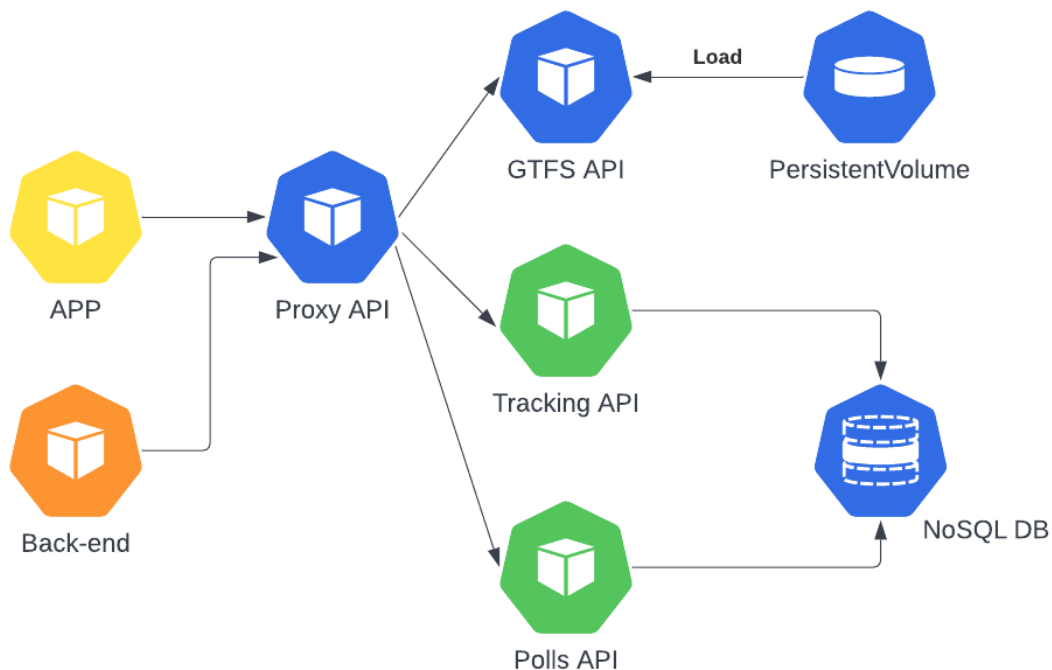


Figure 5 - API Interaction Diagram

Trip Planning and Analytics

The MIMOSA App Mobilità AR application works in conjunction with other open-source applications:

- **Otp (Open Trip Planner)**
- **Matomo**

OpenTripPlanner (OTP) provides route planning and navigation capabilities for the mobile application. OTP, an open-source multimodal trip planner, enables efficient computation of optimal routes by considering various factors such as transportation modes, schedules, and real-time data. By leveraging OTP, the MIMOSA application can offer users accurate and up-to-date route suggestions, facilitating seamless and efficient travel experiences.

Additionally, the architecture integrates **Matomo**, an open-source web analytics platform, to track and analyze the usage of the application. Matomo enables the collection of valuable insights on user interactions, preferences, and behavior, which can be leveraged to optimize the application's features, enhance user satisfaction, and make data-driven decisions for future improvements. The combination of OTP for route planning and Matomo for analytics empowers the MIMOSA

application to provide intelligent and personalized journey recommendations while gathering valuable data for continuous enhancement and refinement.

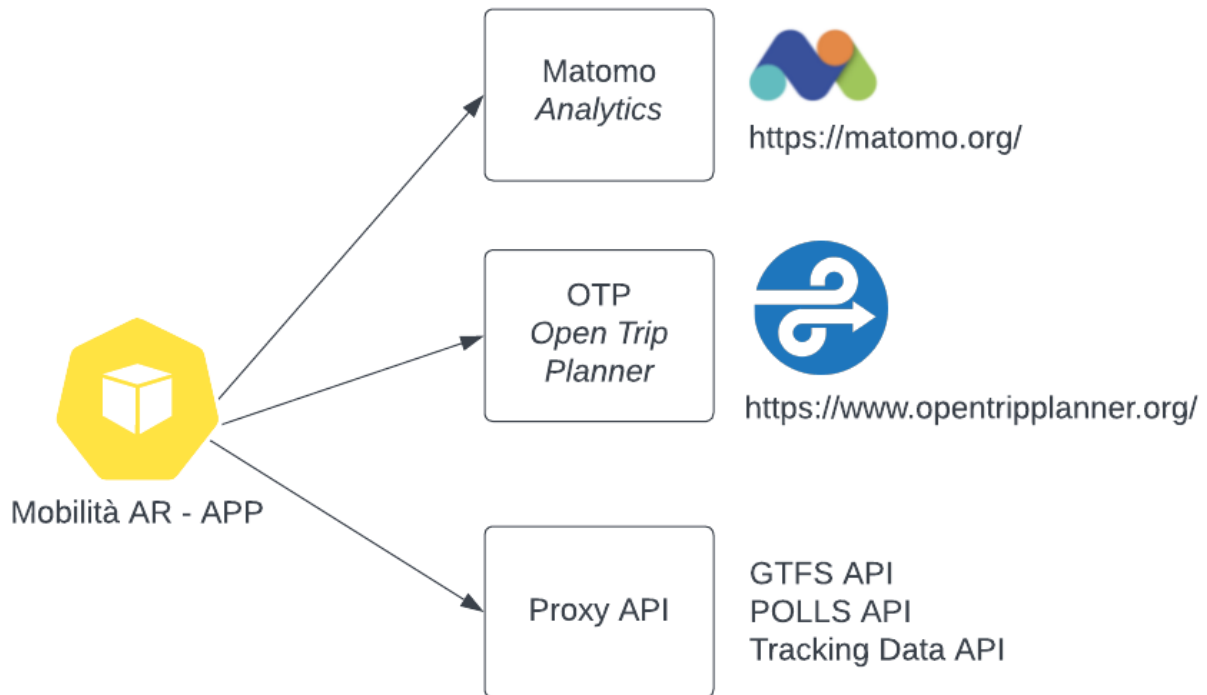


Figure 6 - Open-Source modules and APIs

Transport Data Connector (GTFS)

The connector component retrieves public transport data from a web endpoint provided by Lepida (the public company handling data and services for the Emilia Romagna Region).

The Connector is implemented by a Python script with two different channels:

- the first is for static GTFS data
- the second for real-time data (GTFS-real-time).

The first channel polls the source data once a day, while the second polls it every thirty seconds. Static data undergoes a multi-step ETL process. The first stage combines data from different agencies into a single file. This intermediate file is processed by OTP and GTFS-Loader. GTFS-Loader has been implemented so that the output data is the same as the OTP binary data. This approach ensures the data remains consistent in the application between the two main sections of the application itself: direct bus selection and travel planner.

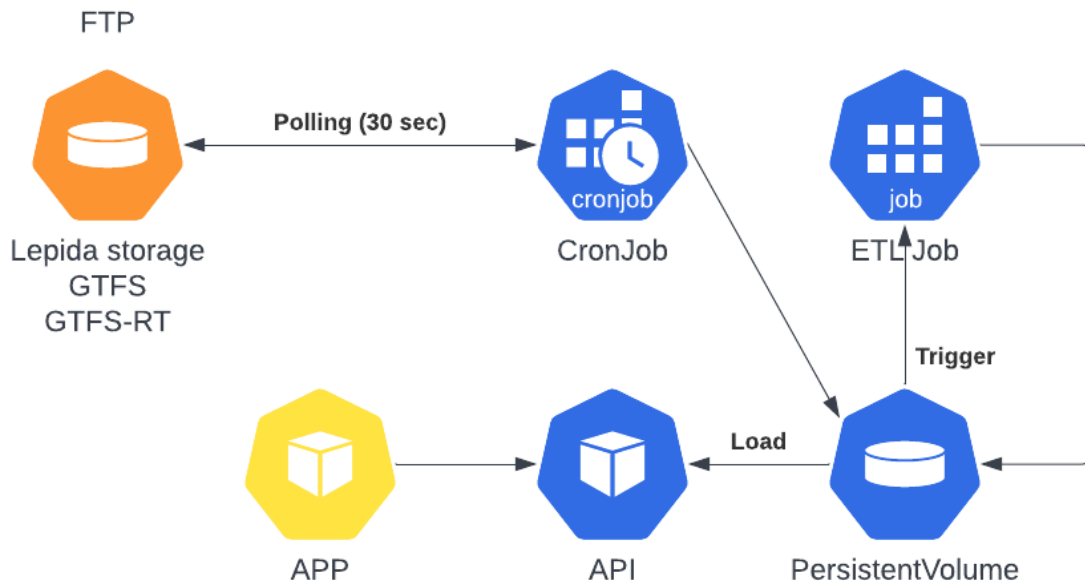


Figure 7 - Connector activities diagram

GTFS data are retrieved from the Lepida archive via HTTP and saved in the Mimosa archive. The CronJob for the real time data is set to make a call every 30 seconds. As soon as the data is saved in the Mimosa storage, an event is triggered that launches the Job ETL. The Job ETL is responsible for aggregating the data from the various companies into a single dataset that is then loaded into memory and used by the GTFS microservice API. Finally, the API is called by the APP to get the data and display the information to the user.

GTFS Realtime Data

Before implementing the algorithms, GTFS Real Time data samples were acquired from the local transport providers (TPER and Start Romagna) in order to be pre-analyzed regarding vehicle count and update frequency:

- Vehicle count can range from 200 to 800 depending on the season
- Position is sampled on-board every 15 seconds and polled by stations, so data can be stale (30-40 seconds)
- Some processing is done by operators in order to reduce GPS errors by projecting the position on the vehicle path.
- Some operators could be able to implement a faster access to the position only, in order to reach 15-20 seconds of latency.

The data is available in the form of both static and real-time General Transit Feed Specification (GTFS). The GTFS format establishes a common standard for public transportation schedules and associated geographic information. GTFS feeds allow public transport companies to publish their data and make it accessible to developers. GTFS Real-time is an extension of GTFS that provides real-time updates on the fleet of vehicles using automatic vehicle location systems.

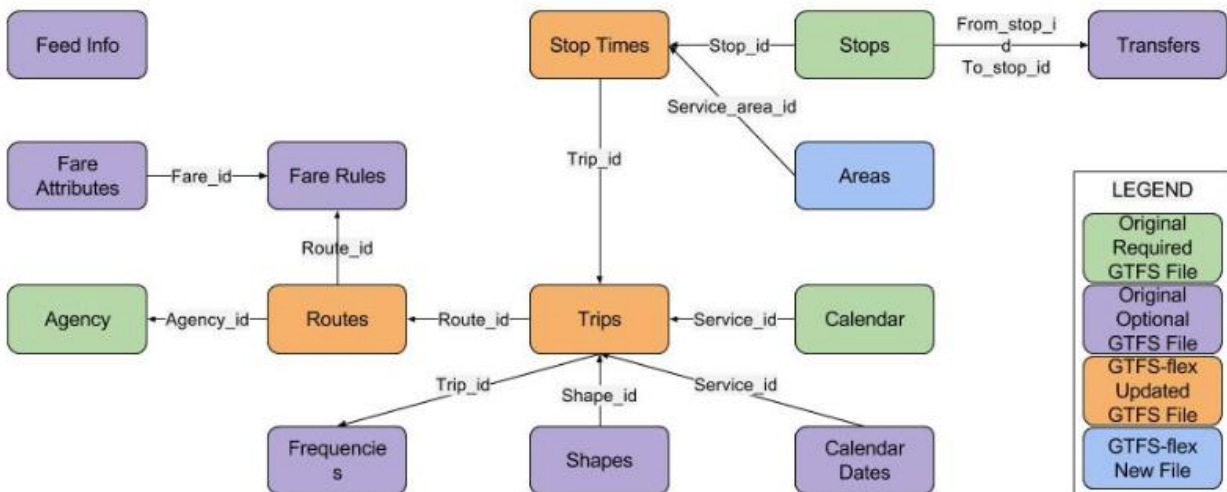


Figure 8 - GTFS entity-relationship diagram

The static data from the folders of various agencies has been consolidated into the following files:

- agency.txt: contains agency information
- stops.txt: contains stop/station information
- routes.txt: contains route information
- trips.txt: contains information about individual trips
- calendar_date.txt: contains information about specific dates
- stops_time.txt: contains arrival and departure time information for each stop on a trip
- shapes.txt: contains information about the route shape

The connections between these files are made possible through unique keys. For example, in the trips.txt file, the 'route_id' key allows linking to the route.txt file, which in turn contains the 'agency_id' key for connection to the agency.txt file. This allows for determining which agency each trip belongs to. To combine data from all agencies and avoid possible duplicates in the keys, which must be unique, the abbreviation of the province of the corresponding city was added to all keys. For example, for the TPER agency responsible for Ferrara and Bologna, 'FE_' and 'BO_' were respectively added to each key.

The real-time data comes from GTFS Realtime Protobuf files and is updated every 30 seconds. The data supports the following types of information:

- Trip updates: delays, cancellations, route changes
- Service alerts: disruptions to a route, unexpected events at a station/stop, or on the entire network
- Vehicle positions: information about vehicles, including location and traffic congestion levels

The real-time data does not have direct unique keys for direct connection with the trips.txt file in static data. However, it includes a 'service_id' key found in the calendar_dates.txt file along with dates. By combining this information, it was possible to associate each real-time data with a specific trips_id and link it to the static information.

GTFS API Implementation

The GTFS Api has been implemented in Go Language to achieve high performance.

All GTFS data are loaded in memory when the Application is started and served without the need to interact with an external Database. The memory is rebuilt with these rules:

- Static GTFS data: only when they are changed by agencies
- Realtime GTFS: every 30 sec

The source is represented by the files processed by the connector. During the startup process, the application loads raw data and builds all models that are stored in memory as arrays and maps (as shown in figure 9) and served by controllers.

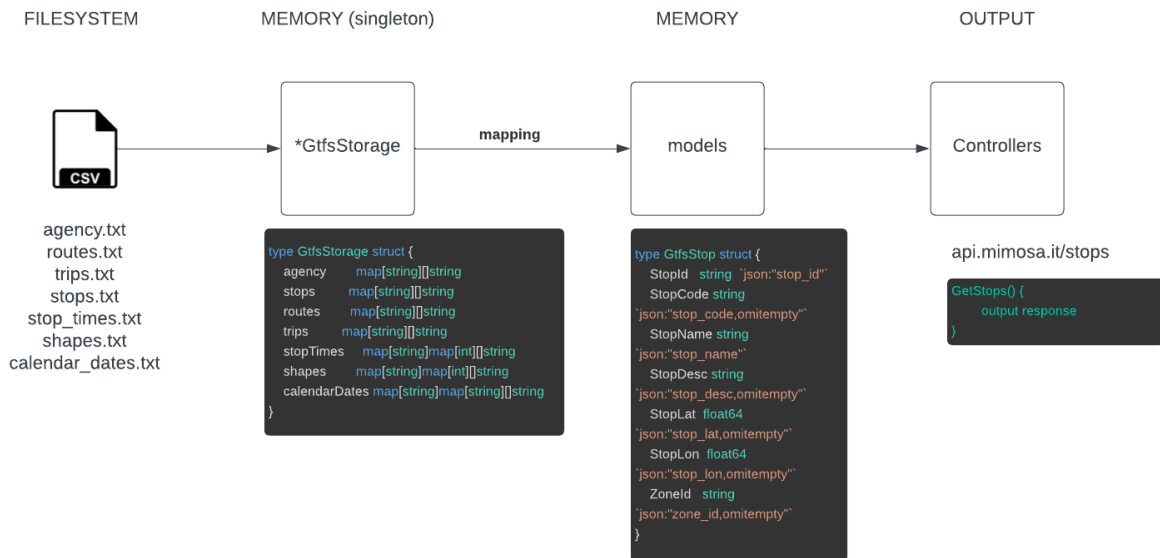


Figure 9 - GTFS Connector data flow

Vehicle Position Prediction

The Vehicle Position Prediction algorithm developed exclusively in the MIMOSA application is a critical component that enables the virtualization of a vehicle's position on the bus route line. This algorithm takes into account the current position, direction, and speed of the vehicle and calculates a projected position.

However, to account for potential delays and ensure conservative positioning, the calculated position is set back by 20% compared to the forecasted position. This adjustment is based on empirical observations made on real cases, where a 20% setback was found to be an effective and reliable approach in compensating for delays and maintaining accurate position estimation.

By incorporating this conservative approach, the MIMOSA application can provide users with a more reliable representation of the vehicle's position along the bus route, enhancing the overall user experience and ensuring the accuracy of real-time information.

This optimization allows for a smooth visualization of the vehicle position in the Augmented Reality experience (Level 3).

NeTex Connector

The MIMOSA application leverages the NeTex connector, which integrates the NeTex specifications implemented in the OTP planner. NeTex format, provides a standardized and interoperable way to exchange public transport data. By utilizing the NeTex connector, MIMOSA

can seamlessly incorporate both GTFS (General Transit Feed Specification) data and NeTex data to feed the trip planning algorithms. This approach ensures that the MIMOSA application is future-proof, as it can adapt to evolving data standards and easily accommodate new data sources. By embracing the flexibility of NeTex and supporting multiple data formats, MIMOSA empowers users with a comprehensive and up-to-date trip planning experience, regardless of the data sources available.

Polls and Surveys

The MIMOSA architecture incorporates a robust "Polls and Surveys" service as part of Level 2. This service consists of both APIs that can be queried by the mobile application and a back-end infrastructure that facilitates the creation and distribution of polls to be filled by users during their interactions with the app, particularly in the pilot project phase. Implementing these Polls and Surveys services provides several advantages for the MIMOSA project.

- Firstly, it enables the collection of valuable usage and statistical data from users in batches, allowing for comprehensive analysis and insights. This data can be gathered at various stages, including the beginning, during the pilot project, and at the end, allowing for a thorough evaluation of user behavior and preferences over time.
- Additionally, the Polls and Surveys service facilitates the collection of user feedback, enabling the project team to gather insights, identify areas of improvement, and make informed decisions based on user preferences.

Polls API

Polls API are used for drafting questionnaires and surveys by the editorial staff, and for recording responses to questions by users. The API are written in node js with Express (a minimalist web framework) and the data are stored in Dynamo-db database.



The endpoints are grouped into 5 groups as shown in the images below:

auth Admin authentication ^

POST	/login Authentication v
POST	/register Authentication v

In auth groups there are *login* and *register* endpoints that are used by the editorial staff to register and subsequently login to the software to manage polls, show polls answers by the users and manage the gamification.

polls Polls requests

GET	<code>/polls/{poll_id}/answers/export</code> export for answers	▼  
POST	<code>/polls</code> Post a poll	▼
GET	<code>/polls</code> List all polls	▼
GET	<code>/polls/{poll_id}</code> Info for a specific poll	▼
PUT	<code>/polls/{poll_id}</code> Update poll	▼
DELETE	<code>/polls/{poll_id}</code> Soft delete poll	▼
PATCH	<code>/polls/{poll_id}/status</code> change poll status	▼

Polls group contains all the requests to manage the polls such as the creations, update and delete, change the status and export the answers of the users.

pollReplies Polls Replies requests

GET	<code>/users/{user_id}/polls/{poll_id}</code> Info for a specific poll replies by user	▼
POST	<code>/users/{user_id}/polls/{poll_id}</code> Post a poll replies	▼
DELETE	<code>/users/{user_id}/polls/{poll_id}</code> Soft delete poll replies	▼

Poll replies group contains end points relative to fruition of the poll by the single user such as show the questions and store the answers.

report Report requests

GET	<code>/polls/{poll_id}/report</code> Report for poll	▼
-----	--	---

Report group has the end point to show a report of all the answers of all the users.

game game requests ^

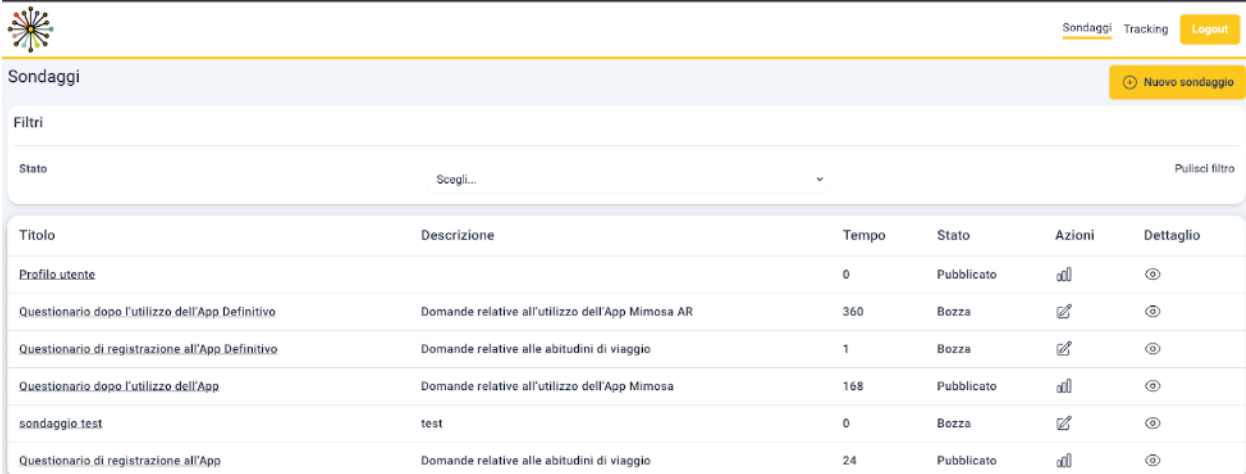
GET	/leaderboard/{user_id}	get leaderboard	▼
POST	/play	Post user points	▼
POST	/user/access	access to app	▼

Game group consists of the end points for fruition of game by the user such as access to game, play and see the leaderboard.

Polls Dashboard

Home

In the home page we find a navigation menu between the dashboard to manage questionnaires and the dashboard to see the tracking data of individual users.



The screenshot shows a dashboard with a navigation menu (Sondaggi, Tracking, Logout) and a table of polls. The table has columns for Titolo, Descrizione, Tempo, Stato, Azioni, and Dettaglio. The table contains six rows of poll data.

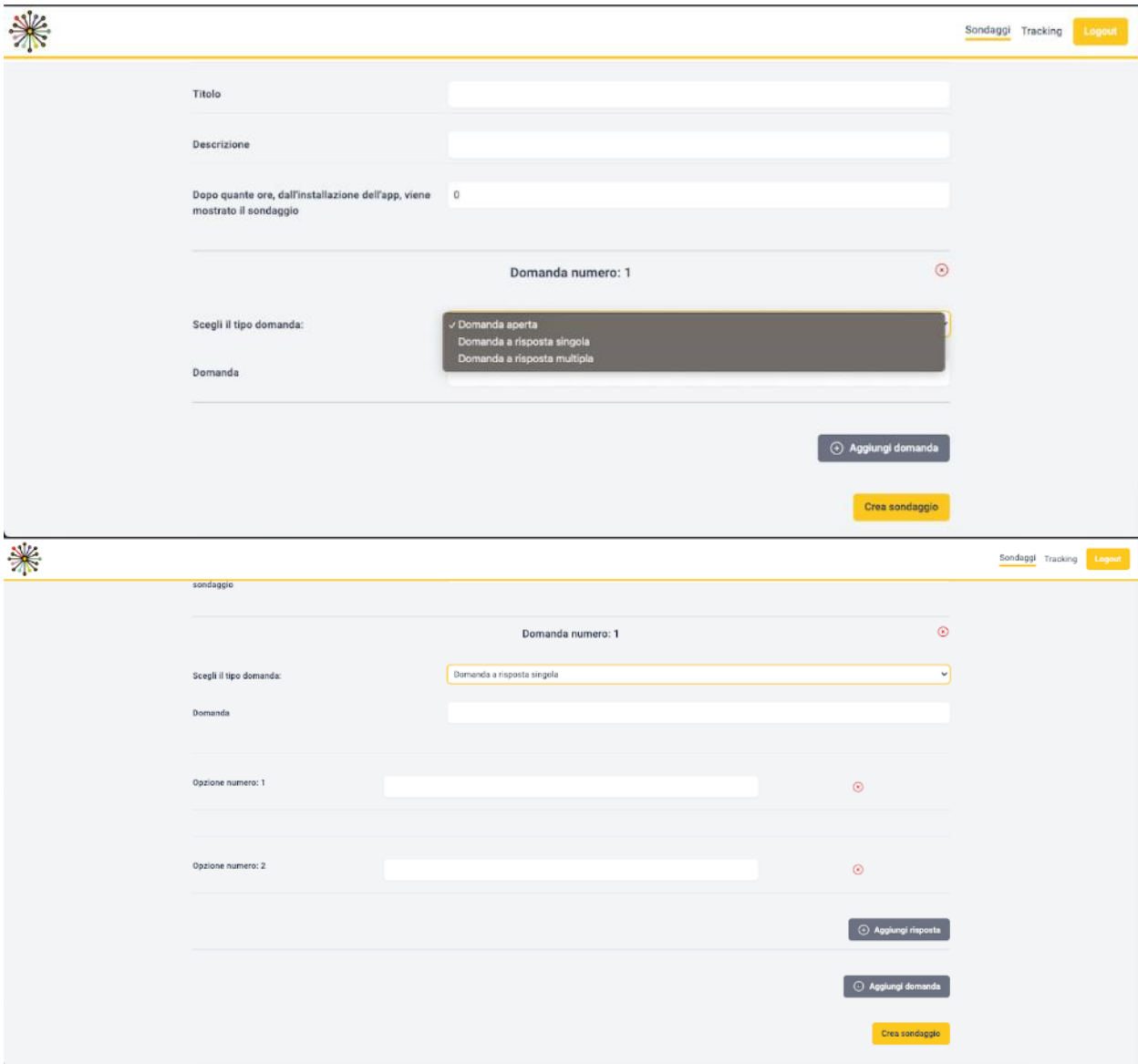
Titolo	Descrizione	Tempo	Stato	Azioni	Dettaglio
Profilo utente		0	Pubblicato		
Questionario dopo l'utilizzo dell'App Definitivo	Domande relative all'utilizzo dell'App Mimosa AR	360	Bozza		
Questionario di registrazione all'App Definitivo	Domande relative alle abitudini di viaggio	1	Bozza		
Questionario dopo l'utilizzo dell'App	Domande relative all'utilizzo dell'App Mimosa	168	Pubblicato		
sondaggio.test	test	0	Bozza		
Questionario di registrazione all'App	Domande relative alle abitudini di viaggio	24	Pubblicato		

Figure 10 - Polls Dashboard Home

There is a table containing a list of polls. The details of the surveys are visible both by clicking the title and the detail button. It is possible to filter the polls according to their type (draft, published or closed). Surveys that have been created but not yet published are of draft type. These

questionnaires can always be modified by the edit button. Published polls can only be modified until there are user responses. Once at least one user has responded to the questionnaire, it becomes unmodifiable, and among the actions we can see the user responses. Closed polls are no longer visible in the app, but their details and user responses can be viewed. Additionally, closed questionnaires can be reopened and republished in the app.

Create and edit polls

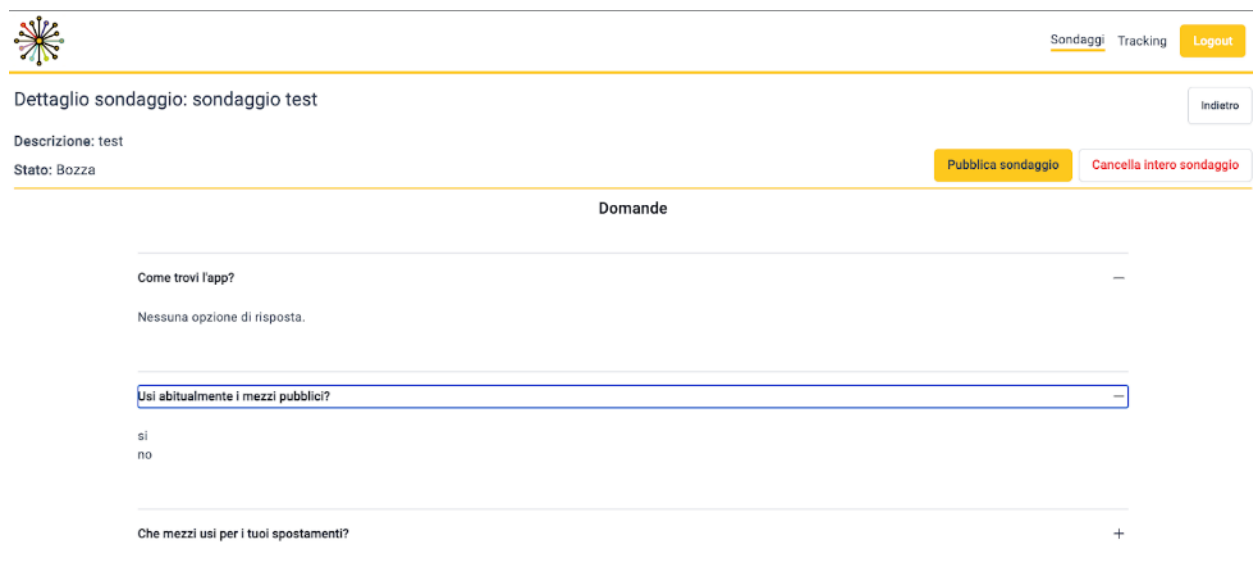


The image displays two screenshots of the 'Create and edit polls' interface in the app. The top screenshot shows the initial form for creating a poll, including fields for Title, Description, and a timer. Below this is a section for adding questions, with a dropdown menu for question types (Open, Single response, Multiple response) and an 'Add question' button. The bottom screenshot shows the editing interface for a specific question, with a dropdown for question type, a question text field, and two option fields. Buttons for 'Add response', 'Add question', and 'Create poll' are visible at the bottom.

Figure 11 - Create and edit polls

The poll creation form requires the input of a title, description, and a time for the poll to appear in a pop-up after the app has been installed for a certain number of hours. Questions can be added using a button. For each question, a type must be chosen between open-ended, single choice, or multiple-choice. In the case of choice-based questions, options must be added. Once completed, pressing the yellow button creates the survey, which will be visible in the home table. The polls can be edited via the same form.

Poll detail



The screenshot shows the 'Poll detail' interface. At the top left is a starburst icon. On the top right, there are links for 'Sondaggi', 'Tracking', and a yellow 'Logout' button. Below this is the title 'Dettaglio sondaggio: sondaggio test' and an 'Indietro' button. The description is 'test' and the status is 'Bozza'. There are two buttons: a yellow 'Pubblica sondaggio' button and a red 'Cancella intero sondaggio' button. The main section is titled 'Domande' and contains two questions:

- Question 1: 'Come trovi l'app?' with a '-' button. Below it, it says 'Nessuna opzione di risposta.'
- Question 2: 'Usi abitualmente i mezzi pubblici?' with a '-' button. Below it, there are two options: 'si' and 'no'.

At the bottom of the question list, there is a '+' button next to the text 'Che mezzi usi per i tuoi spostamenti?'.

Figure 12 - Poll detail

Within the poll detail, there are buttons to publish, close, delete, or reopen the poll. A published poll that has received no responses can be reverted back to draft status. All questions and their possible answer options are visible within the poll details. Upon opening the details, only the question texts are initially visible. Clicking on the + buttons expand the answer options, while clicking on the - buttons collapses the UI widget.

Poll statistics

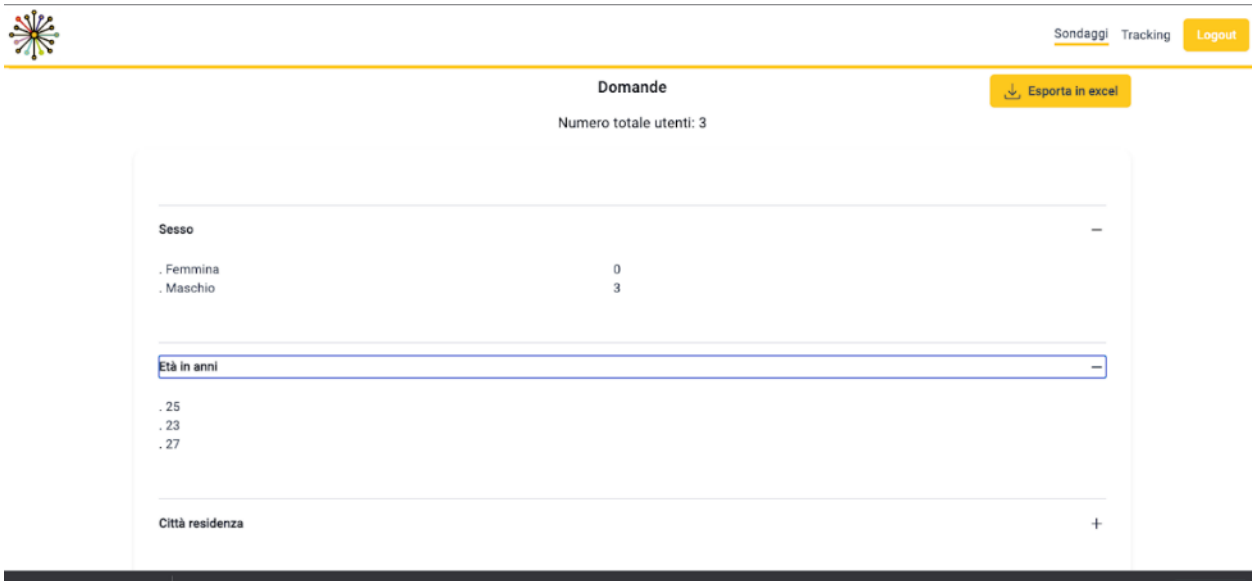


Figure 13 - Poll statistics

On the polls statistics page, it is possible to view all user responses. Initially, only the list of poll questions is visible. Clicking on the + button opens an accordion with all user responses. At the top, the total number of users who completed the questionnaire is displayed. In the case of multiple-choice questions, the number of users who chose each option is shown. For open-ended questions, the list of all user responses is displayed. These responses can be downloaded in an Excel file via a dedicated button.

User and Activity Tracking

The User and Activity Tracking Services implemented within the MIMOSA architecture are essential components that continuously receive data from users' mobile devices to track their position and activities. These services play a crucial role in generating valuable statistics on app usage, enabling the analysis of user behavior and the assessment of changes in behavior resulting from the introduction of new functions, such as Augmented Reality navigation.

By tracking the user's position and activities, the MIMOSA application can provide insightful information about the available public transport options to help users navigate the territory more effectively. This comprehensive tracking of user data allows for a deeper understanding of mobility patterns, preferences, and trends, empowering both the user and the project team with valuable insights that can drive improvements in the app's features and enhance the overall public transport experience.

Tracking DATA API

The MIMOSA AR Mobility App tracks the user's location both when the app is in use and when the app is in the background. Tracking takes place locally on the device and the data is then sent to the server when an Internet connection is available. Data is transmitted in packets. Each packet has the structure below:

```
let schema = {
  'type': 'object',
  'properties': {
    'user_id' : {'type' : 'string' },
    'tracking_posixtime': {'type': 'number'},
    'tracking_data': {
      'type': 'array',
      'items': {
        'properties': {
          'posix_time':{'type': 'number'},
          'lat':{'type': 'number'},
          'lon':{'type': 'number'},
          'speed': {'type': 'number'},
          'heading': {'type': 'number'},
          'activity': {
            'type' : 'string',
            'enum' : ['IN_VEHICLE', 'ON_BICYCLE',
'RUNNING', 'STILL', 'WALKING', 'UNKNOWN'],
          },
        },
      },
    },
    "additionalProperties": false,
```

```

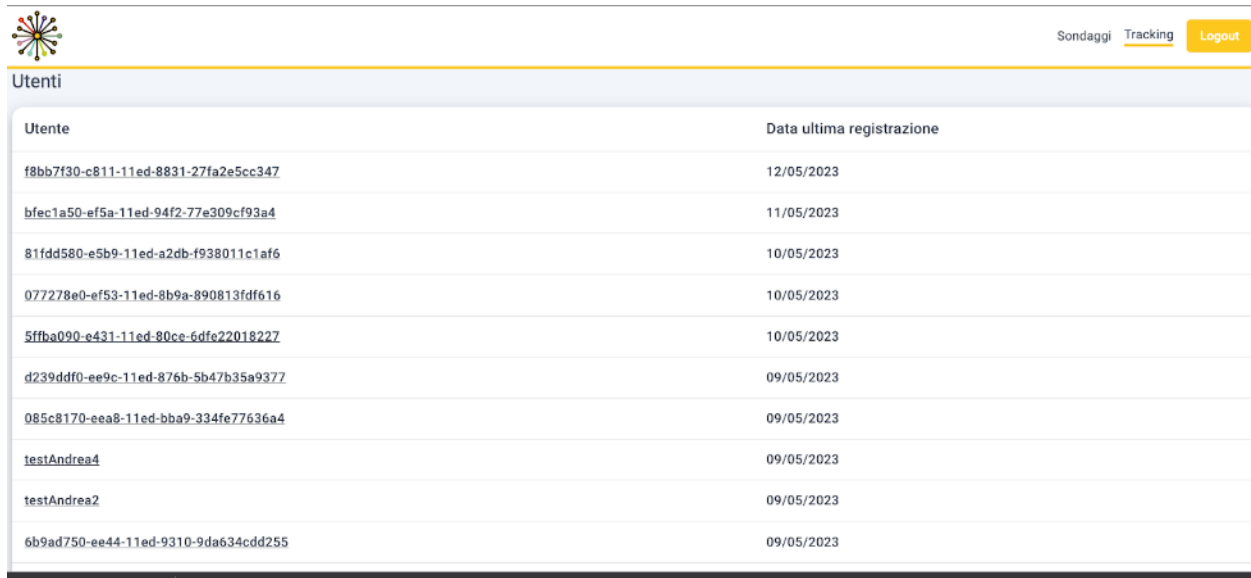
        'required': ["posix_time", "lat", "lon", 'speed',
'heading', 'activity']
    },
},
},
"additionalProperties": false,
"required": ["user_id", "tracking_data"]
}

```

Each record sent to the server has the `user_id` identification code of the single user, `tracking_posixtime` indicates data and time of the record. The posix time is the Unix timestamp, is a way to track time as a running total of seconds. This count starts at the Unix Epoch on January 1st, 1970 at UTC. In addition to these properties, also the `tracking_data` array is required, which consists in at least one record of the user's location.

The track of the user's location contains properties indicating date and time, latitude and longitude of the user position. Other information about movement of the user like speed, direction and type of activity are stored. The possible activities are in vehicle, on bicycle, running, still, walking or unknown. The tracks are stored in the Dynamo-db tracking data table.

Tracking Data



Utente	Data ultima registrazione
f8bb7f30-c811-11ed-8831-27fa2e5cc347	12/05/2023
bfec1a50-ef5a-11ed-94f2-77e309cf93a4	11/05/2023
81fdd580-e5b9-11ed-a2db-f938011c1af6	10/05/2023
077278e0-ef53-11ed-8b9a-890813fdf616	10/05/2023
5ffba090-e431-11ed-80ce-6dfe22018227	10/05/2023
d239ddf0-ee9c-11ed-876b-5b47b35a9377	09/05/2023
085c8170-eea8-11ed-bba9-334fe77636a4	09/05/2023
testAndrea4	09/05/2023
testAndrea2	09/05/2023
6b9ad750-ee44-11ed-9310-9da634cdd255	09/05/2023

Figure 14 - Tracking Data

On the tracking data page, there is a table with all users ordered by the last date of registration of a position. By clicking on the user code, you can view an interactive geographic map with all of their registered positions.

Interactive map

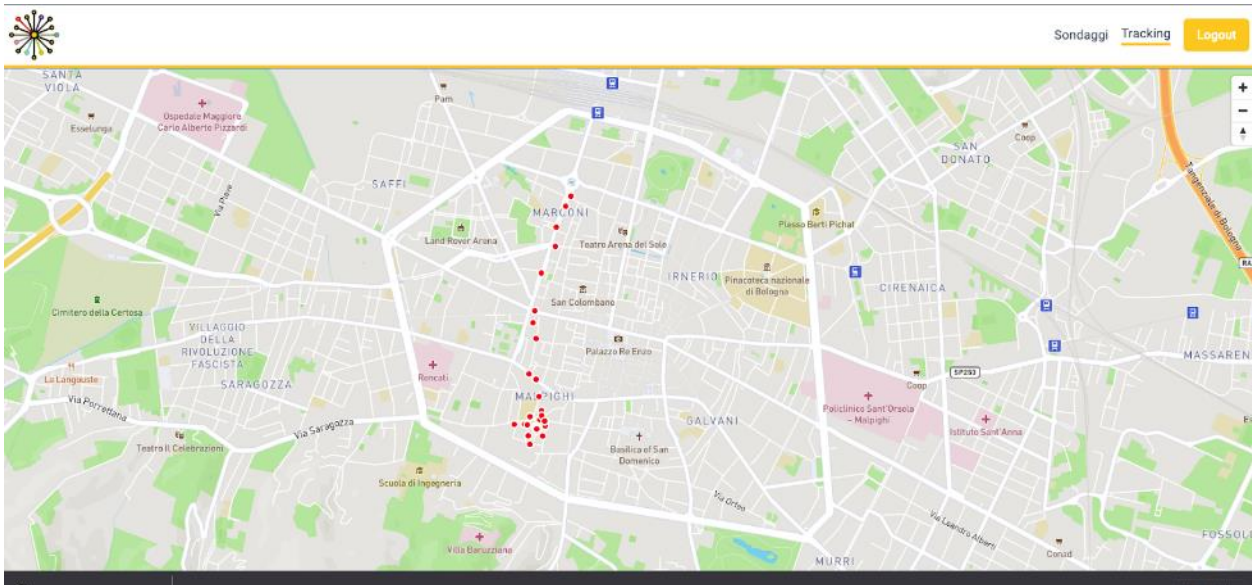


Figure 15 - Interactive Map

Interactive map of a single user.

The red dots show each individual point where a position was registered.

Spatial Cluster Analysis

The tracking data is analysed with a Python script, with the Boto3 libraries to retrieve the data from the Dynamo-db database stored in AWS (Amazon Web Services)

In the first instance, the list of all registered users of the app is retrieved. Then for each user, the tracking data table is read to retrieve their positions and related information (eg: date, speed, direction). The tracking data is managed with the Pandas library.

Tracking Data Table

lon	activity	heading (direction)	lat	speed	date time
11.2497031	WALKING	38.26435852050781	44.6567815	9.541555404663086	2023-03-15 06:51:33
11.1876552	WALKING	259.3337097167969	44.6364324	0.14577257633209229	2023-03-15 10:54:54
11.1876552	STILL	259.3337097167969	44.6364324	0.14577257633209229	2023-03-15 10:55:24
11.1876552	STILL	259.3337097167969	44.6364324	0.14577257633209229	2023-03-15 10:55:54
11.1859525	WALKING	322.7821960449219	44.640117	2.2052884101867676	2023-05-06 15:42:33

A K-means analysis is performed to figure out which are the most frequent coordinates points reached by the user. K-means is an unsupervised clustering algorithm that groups a set of data into K clusters, where K is a value pre-specified by the user.

The algorithm works as follows:

- K initial centroids are randomly chosen from the dataset.
- Each point in the dataset is assigned to the nearest centroid, thus creating K clusters.
- The new centroids of each cluster are calculated as the mean of the points in the cluster.
- The process is repeated from step 2 until there are no more changes in the position of the centroids or a maximum number of iterations is reached.

The objective of the algorithm is to minimise the sum of squared distances between the points in the dataset and their respective centroid.

The k-means algorithm produces two type of graphs:

- The graph with the Elbow method curve is a useful tool for determining the optimal number of clusters to use in the K-means algorithm
- The clustering graph is a useful tool for understanding how the K-means algorithm partitions a set of data into homogeneous clusters.

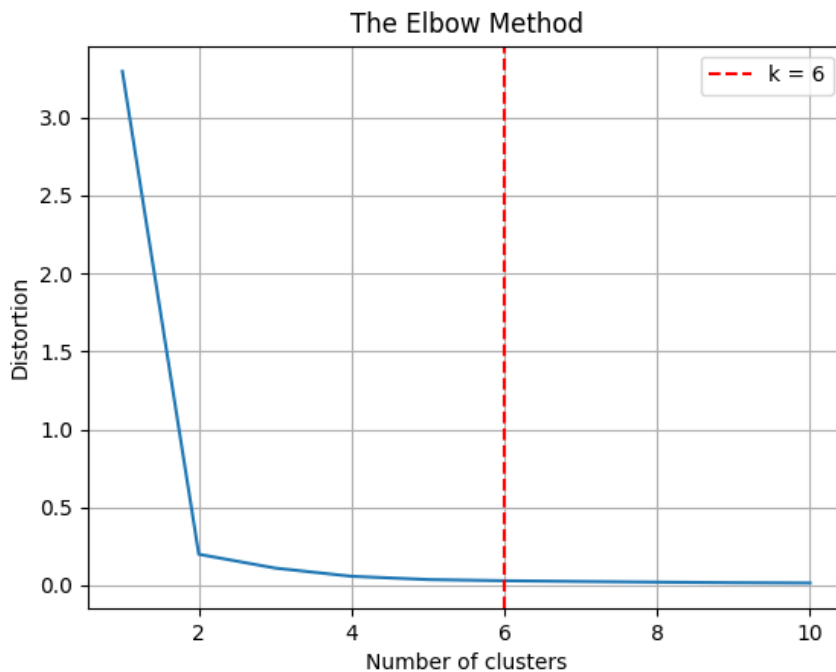


Figure 16 - K-means Elbow Method

Visual representation of the trend of the sum of squared distances within clusters as a function of the number of clusters used in the K-means algorithm.

The graph shows the value of the sum of squared distances for each value of K , where K is the number of clusters being evaluated. The graph's curve typically has an elbow shape, which refers to the point where adding another cluster does not significantly reduce the sum of squared distances.

The inflection point of the curve, where the curve starts to level off, is considered the ideal number of clusters to use in the K-means algorithm.

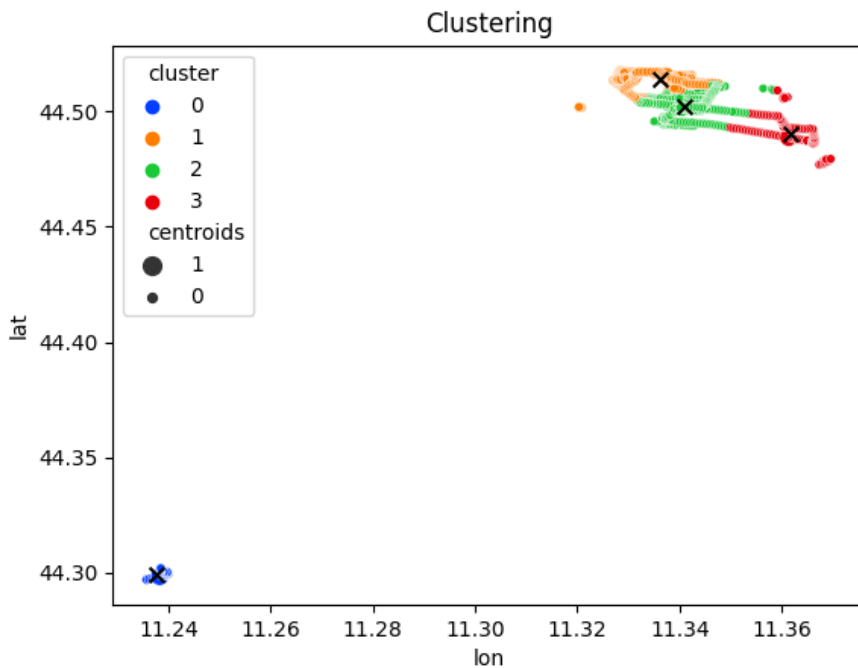


Figure 17 - K-means clustering

A visual representation of the partitioning of a dataset into clusters using the K-means clustering algorithm.

The graph shows each point in the dataset as a point in the Cartesian plane with longitude and latitude on each axis, with a different colour for each cluster assigned by K-means. The X symbol represents the centroids.

Centroids Table

	lat	lon
0	44.50202216869237	11.340997724503724
1	44.2984018641562	11.238340612488933
2	44.51502807163139	11.337322466544427
3	44.488544852955705	11.3613142262717

Table with the coordinates of the cluster centroids.

Example of the output of the k-means analysis: 4 clusters of the most frequent positions are extrapolated for this user.

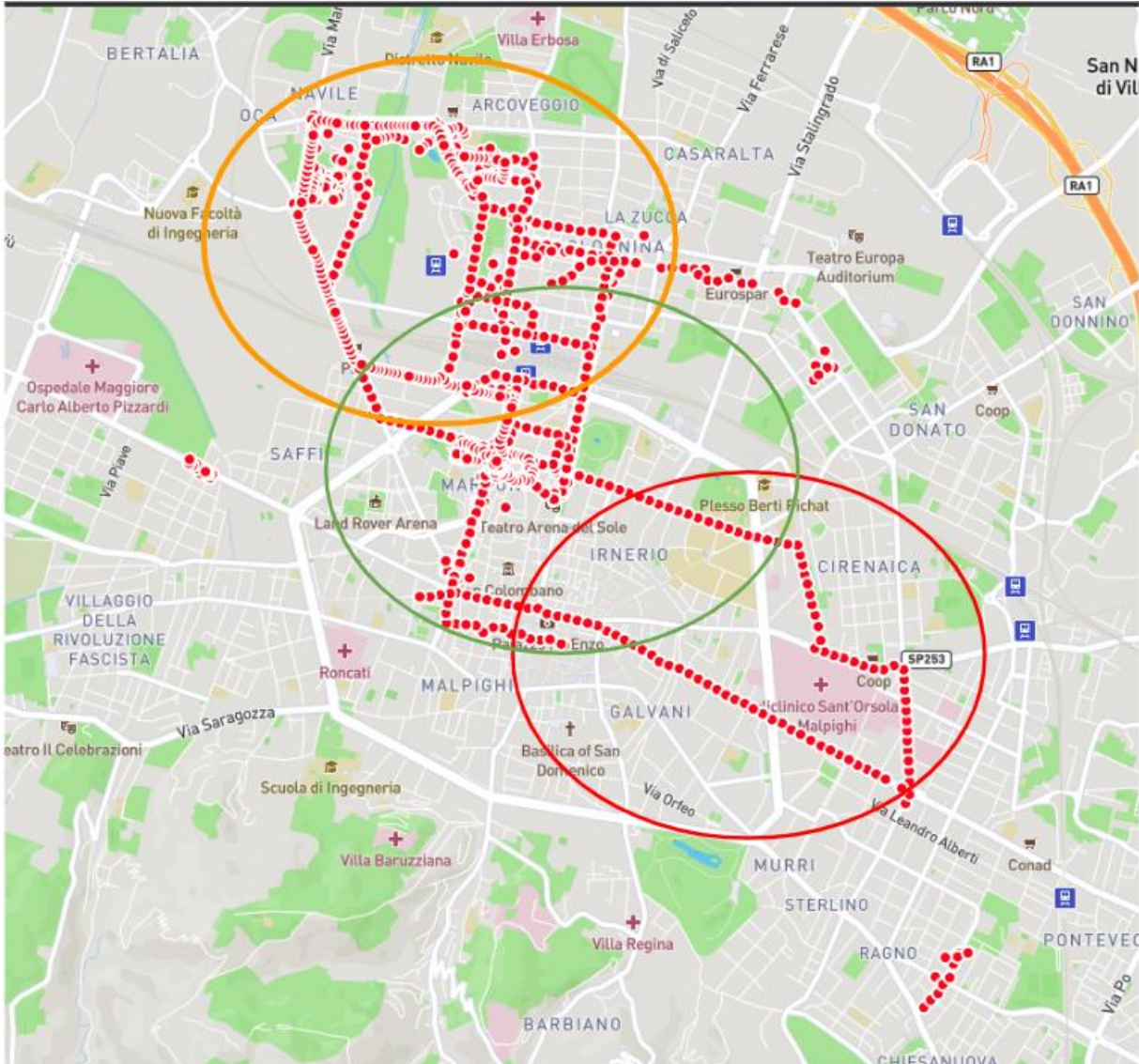


Figure 18 - Zoomed-in clusters

Zoomed Map with all user points in the Bologna city centre, on which the analysis above is based, we can see the three clusters calculated.

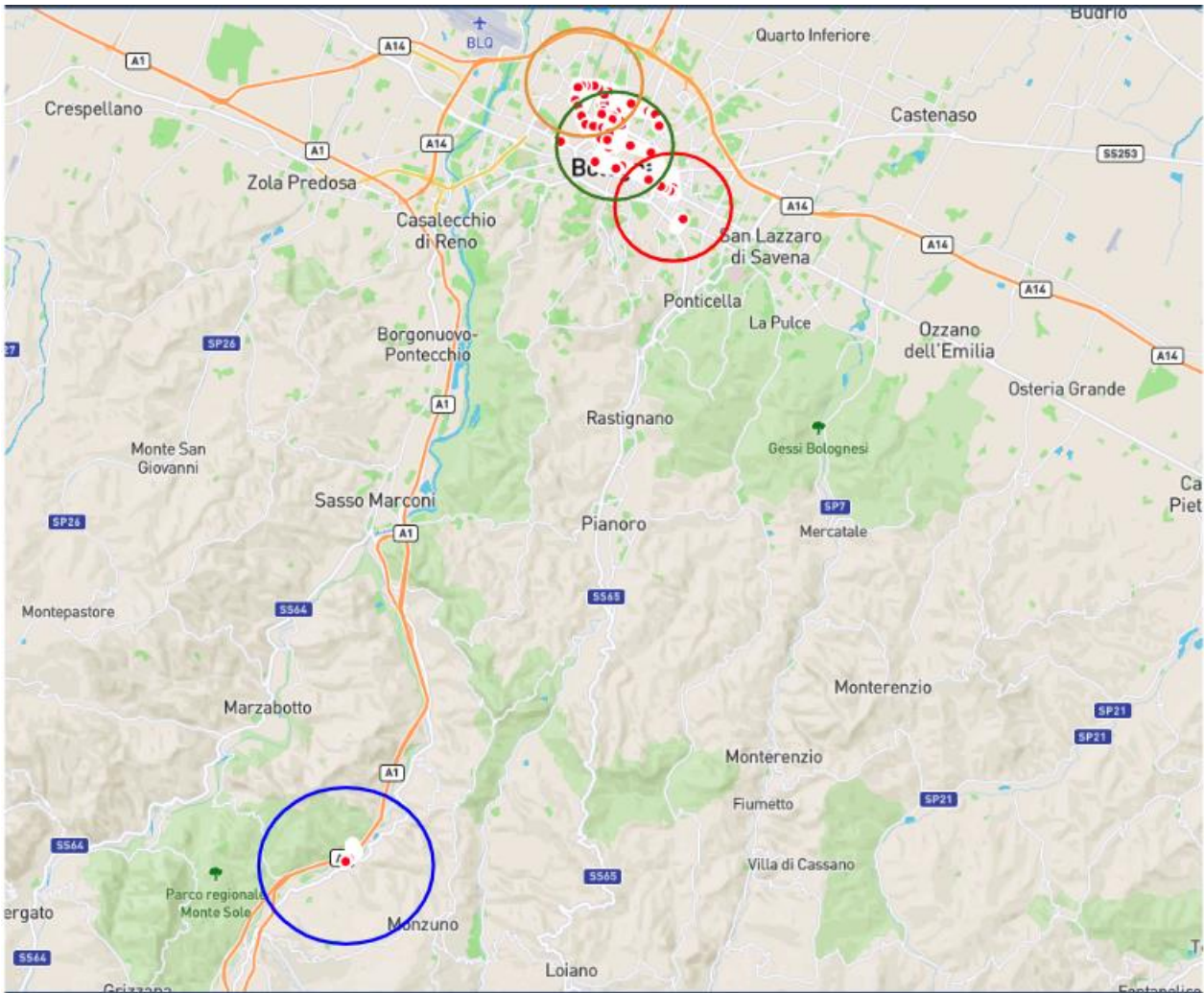


Figure 19 - Zoomed-out clusters

Map with all user points on which the analysis above is based.

From the zoomed image, it can be seen that 3 location clusters were found. A fourth cluster (blue one) was calculated for a set of positions found very far away from the most frequent locations in the city centre. This shows how k-means analysis is able to calculate consistent geospatial clusters, even when the frequency of the positions is lower, but the positions deviate from other clusters already calculated with higher frequencies.

Temporal Segment Analysis

The tracking data was then analysed taking into account the temporal recording of each position. The dates of individual positions were analysed and grouped into 30-minute ranges to identify segments of positions falling within specific time frames.

In the following table, various positions with different dates and times can be observed. The initial recordings belong to the same day but at different times, while the last one was recorded on a different day. It should be noted that this is a small excerpt from a large amount of data on which the analyses were conducted.

The first position refers to 6:51 in the morning, while the subsequent positions were recorded within a time frame ranging from 10:54 to 10:55 on the same day. The last position was recorded at 15:42 in the afternoon of another day. Therefore, we can hypothesise that these positions, although spatially close, are part of different routes that the user took at different times. Upon completion of the analysis, we will be able to obtain a series of user routes that can be compared with cluster analysis.

Segment Tracking Data Color Visualization Table

lon	activity	heading	lat	speed	date
11.2497031	WALKING	38.26435852050781	44.6567815	9.541555404663086	2023-03-15 06:51:33
11.1876552	WALKING	259.3337097167969	44.6364324	0.14577257633209229	2023-03-15 10:54:54
11.1876552	STILL	259.3337097167969	44.6364324	0.14577257633209229	2023-03-15 10:55:24
11.1876552	STILL	259.3337097167969	44.6364324	0.14577257633209229	2023-03-15 10:55:54
11.1859525	WALKING	322.7821960449219	44.640117	2.2052884101867676	2023-05-06 15:42:33

Output of temporary analysis with the positions divided in three different segments of different colours.

The colours here represent a graphical way to convey the output. In practice, when working with a large number of positions, the output generates a table with the reference coordinates of each segment and a progressive numbering based on time, as shown in the following table. This will make it easier to compare this new set of points with the centroids of the previously analysed clusters and study the connection between them to understand the user's trajectory.

Segment Tracking Data Output Table

lon	lat	date	index
11.2497031	44.6567815	2023-03-15 06:51:33	1
11.1876552	44.6364324	2023-03-15 10:54:54	2

Output of temporary analysis with the positions divided in two different segments with progressive indexes.

Temporal data was also used to gather information about users' regular paths. For example, an attempt was made to determine if there was a difference between positions recorded during the week or on weekends. This analysis was conducted by dividing the week into seven parts and further dividing it into two parts using a Python fork. In the first part, the point falls within the first five parts of the week, and in the second part, it falls within the last two parts of the week (weekend).

At the end of the analysis, an additional column is obtained that provides information about which part of the week the points belong to. This information can then be used to analyse user habits.

Segment Tracking Data Weekly Output Table

Index	Date	Latitude	Longitude	Part of week
1	2023-03-15 6:51:33	44,6410887	11,190255	week
2	2023-03-15 10:54:54	44,6364324	11,1876552	week

3	2023-05-06 15:42:33	44,640117	11,1859525	week end
4	2023-05-15 12:52:45	44,4144036	12,1895875	week
5	2023-05-15 13:25:51	44,4925279	11,3367723	week

Output of temporary analysis with the positions and weekly info.

Distances matrix calculation

To search for a connection between cluster centroids and median points of temporal segments, a distance matrix was generated. The distance in kilometers between the coordinates of each cluster centroid and the coordinates of the median points of the temporal segments was calculated. By examining the distance matrix, possible routes that the user takes through points of interest can be hypothesized. At this point in the analysis, we have a set of points of interest, the most probable routes that connect these points, and information about travel times. By providing this data to the OTP (OpenTripPlanner), we can return alternative routes involving public transportation to the user.

Distance Matrix between cluster and segment

	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5	cluster 6
segment 1	577	2331	4197	19378	83340	20422
segment 2	1123	2795	4595	19119	83385	20127
segment 3	896	2681	4554	19516	83634	20536
segment 4	83131	81803	80100	67907	9	68164
segment 5	20371	20163	19621	1767	68435	272

Output of distance analysis between centroids of clusters and median coordinates point of temporal segment.

Merge of spatial and temporal analysis

Each time segment is associated with the nearest spatial centroid. Then, each temporal point, which serves as a reference for a time segment, is assigned the coordinates of its corresponding centroid. As the points are sorted from least recent to most recent, by connecting consecutive centroids from the table, spatial segments representing the user's path are identified. To accomplish this, the points are further divided into groups of 2 hours. Only points within the same 2-hour range are merged to find the actual path to be computed by OTP (OpenTripPlanner).

Segment	Data	Cluster	Latitude	Longitude
1	2023-01-26 16:04:48	6	44.50111268860051	11.339429300816597
2	2023-01-26 16:30:17	1	44.51361473228059	11.3298526552712
3	2023-01-27 09:54:13	3	44.51458220200102	11.339854875672186
4	2023-01-27 16:46:33	6	44.50111268860051	11.339429300816597
5	2023-01-27 17:29:03	6	44.50111268860051	11.339429300816597

6	2023-01-27 17:37:03	6	44.50111268860051	11.339429300816597
7	2023-01-27 18:09:10	6	44.50111268860051	11.339429300816597
8	2023-02-01 15:30:41	6	44.50111268860051	11.339429300816597
9	2023-02-21 20:42:26	6	44.50111268860051	11.339429300816597
10	2023-02-25 09:28:04	4	44.48832698642524	11.361891602586768
11	2023-02-25 09:30:04	2	44.2984018641562	11.238340612488933
12	2023-02-25 10:00:33	2	44.2984018641562	11.238340612488933
13	2023-03-01 11:30:37	6	44.50111268860051	11.339429300816597
14	2023-03-01 13:09:02	6	44.50111268860051	11.339429300816597
15	2023-03-01 13:30:02	3	44.51458220200102	11.339854875672186

Output of merge of spatial cluster analysis and temporal segment analysis.

From the table above, the clusters falling within a two-hour range have been derived and are coloured in grey. These points represent the input for the OTP.

Cluster	Latitude	Longitude
Cluster 6	44.50111268860051	11.339429300816597
Cluster 1	44.51361473228059	11.3298526552712
Cluster 4	44.48832698642524	11.361891602586768
Cluster 2	44.2984018641562	11.238340612488933
Cluster 6	44.50111268860051	11.339429300816597
Cluster 3	44.51458220200102	11.339854875672186

Input for OTP analysis

OTP (Open Trip Planner)

OpenTripPlanner is a group of open-source software applications written in Java, based on OpenStreetMap (OSM) and other standardised data such as GTFS that contain public transportation information. By transitioning to OTP, we can gather user points of interest and their most frequent schedules to provide the user with information on possible public transportation options for their trips.

User Output

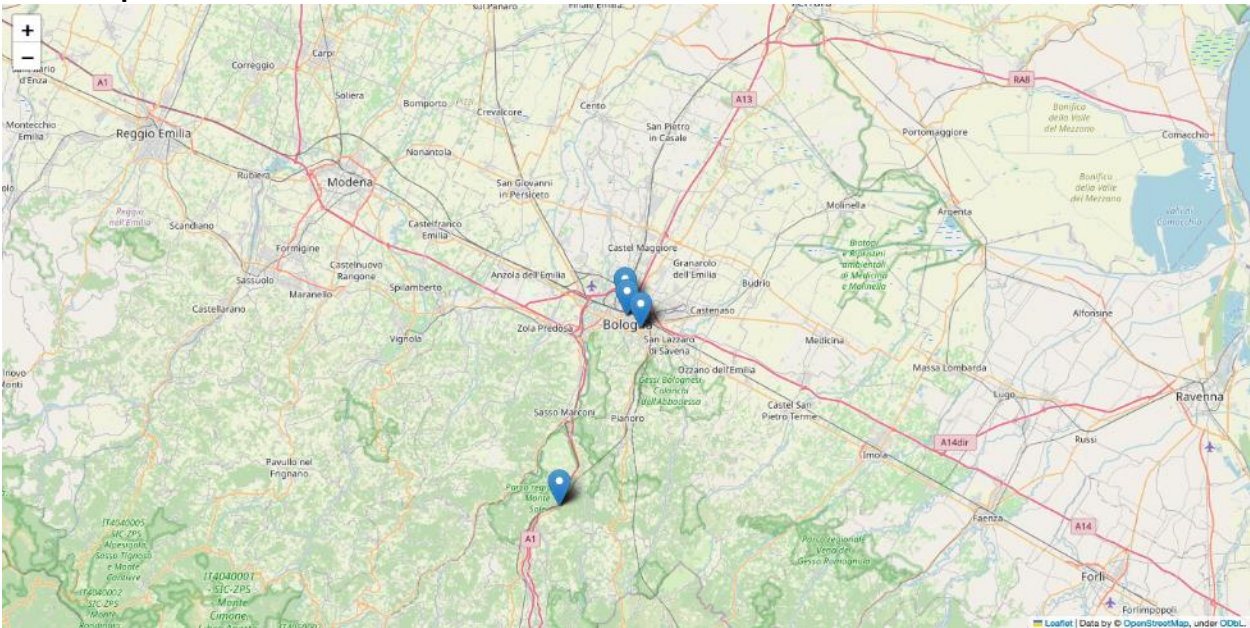


Figure 20 - Final user map with points of interest

The user will be shown a map with markers corresponding to their points of interest.

Querying the OTP with the coordinates of the clusters of interest returns the list of the most advantageous public transport routes in terms of time.

In the images below, we can see the output that is returned to the user.

Final results

In the final step of the analysis, the user points of interest are used as input of OTP components in order to obtain all possible travel solutions by public transportation.

The next figures show an example of the results that will be presented to the user after one month of usage of the App.

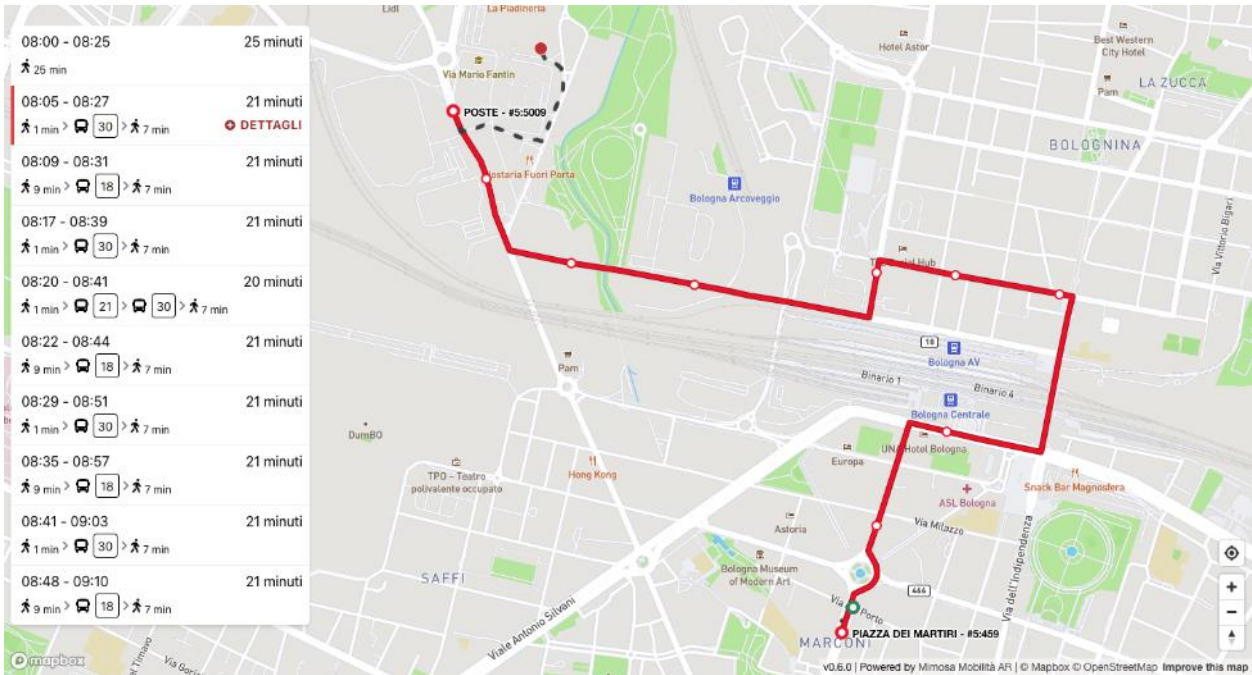


Figure 21 - Minimized walking minutes

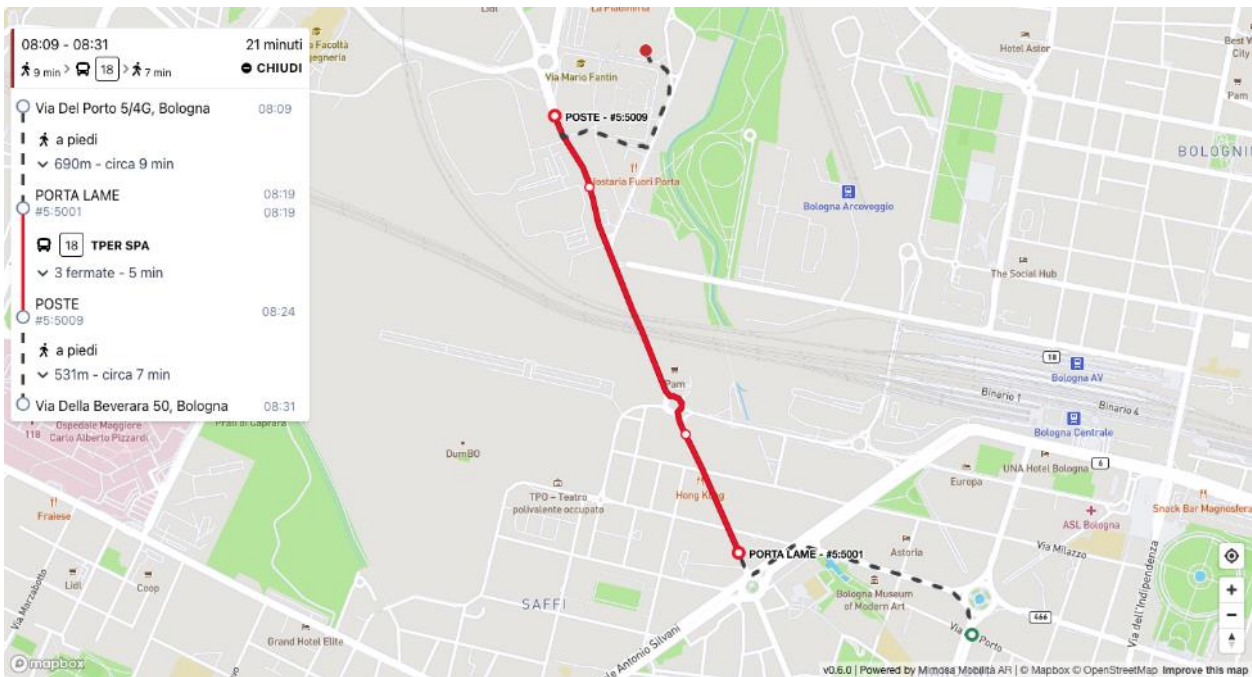


Figure 22 - Bus route less practicable on foot

On the contrary, in the third image, we observe a route that starts with walking and then takes the bus. This allows the user to choose from all possible combinations among their points of interest.

2.3 Augmented Reality Application

This section highlights the UX (user experience) implementation of the MIMOSA Mobilità AR mobile application.

The application builds on the foundation of the APIs previously described and adds a level of smooth and innovative interactivity to the front-end, implementing the Augmented Reality view.

Goal-Focused User Experience

Application UX is focused on user goals using public transport:

- Travel immediately to a destination
- Travel immediately with a specific transport
- Plan for a future journey

And User constraints:

- Arrive on time
- Travel duration
- Cost (support in the API)

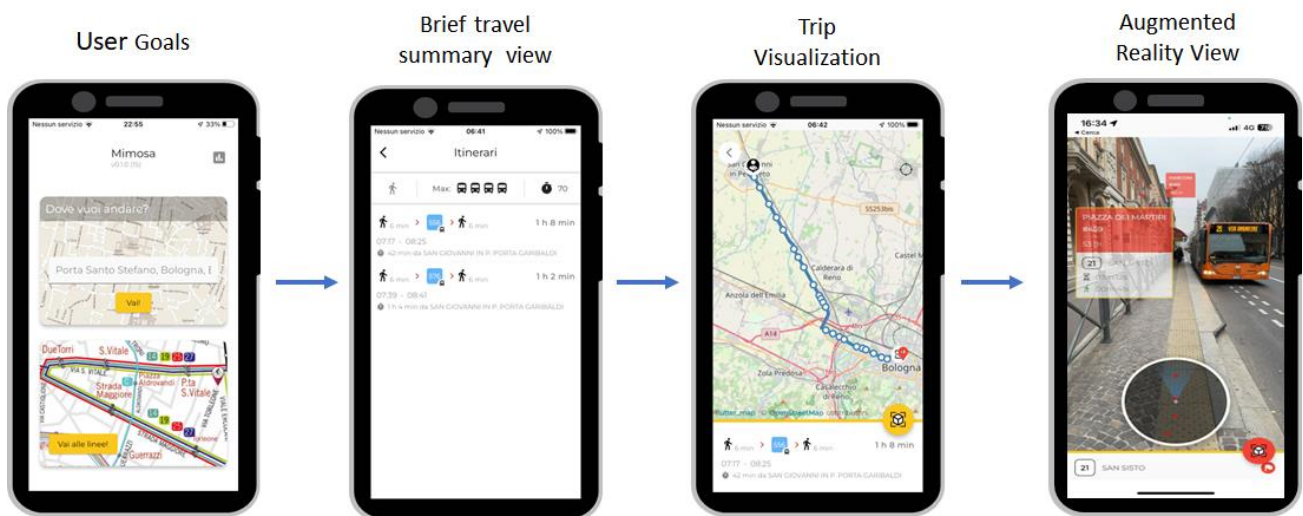


Figure 23 - AR mobile app UX navigation

Implementation of the Augmented Reality View

In order to implement a **locally working** (without the need for continuous server connection) and **image-recognition independent** Augmented Reality view, a blend of sensor fusion techniques is at the core of the implementation.

GPS: The smartphone's built-in GPS provides the precise geographic location of the user. This information allows the app to filter the transport data relevant to the user's location.

Magnetometer: The magnetometer in the smartphone, often used as a digital compass, detects the orientation of the device relative to Earth's magnetic field. When users point their phone towards a bus or a stop on the horizon plane, the magnetometer helps determine the direction the phone is facing.

Gyroscope: The gyroscope detects the orientation of the phone in 3D space. This information is essential in maintaining the correct perspective of the AR overlay as the user moves the phone around.

AR Overlay: With the user's location and orientation data, the app overlays the real-time transport/stop information onto the live camera feed from the user's phone. This creates an AR view where the digital data appears to exist in the physical world.

User Navigation

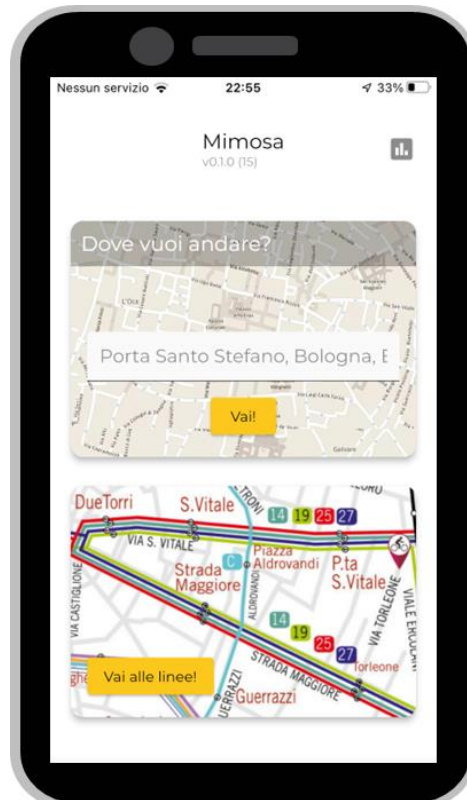


Figure 24 - AR Mobile App - User Navigation

Following the goal-oriented UX design, the User is presented with two main options:

- **Find a route by destination**

In this case, the server's OTP trip-planning service is queried for both the codified position of the User (translating a GPS location to a real address) and the destination, which is matched with the available transport stops nearby.

- **Visualize all routes**

All routes can be visualized by their codified transport number. Once selected, all route destinations will be grouped by direction.

Travel by Destination

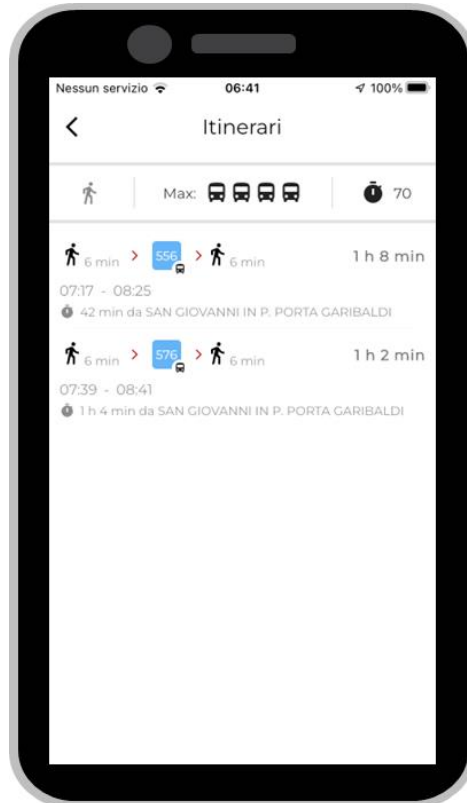


Figure 25 - AR Mobile App - Travel by destination

Routes are reported in a very simplified view.

- The user can quickly filter by the maximum number of changes or time of departure.

Walking time is clearly visualized before and after reaching the departure stop and destination.

Travel by Route

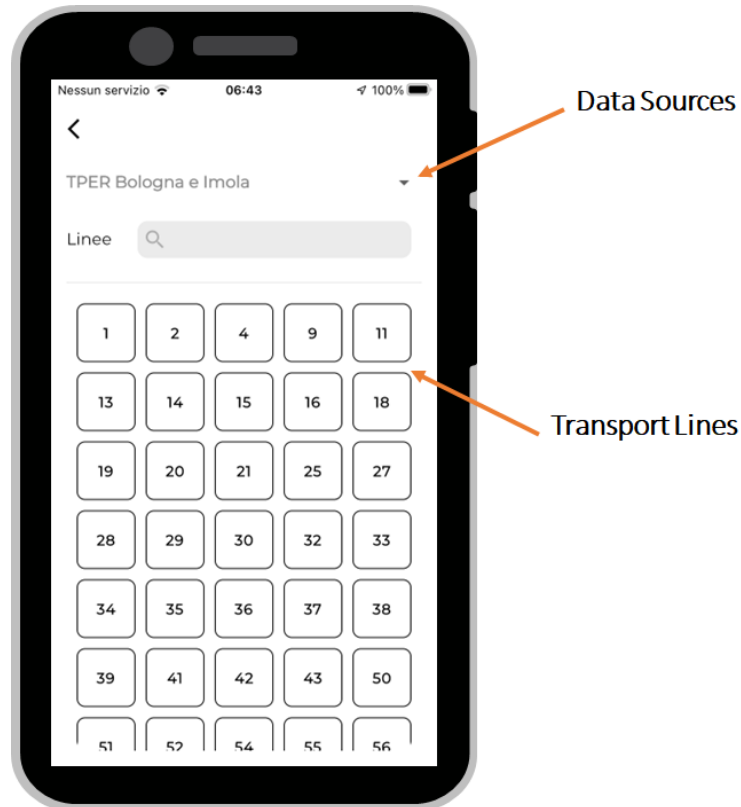


Figure 26 - AR Mobile App - Travel by route

MIMOSA Project implements a selector for different GTFS sources from which all transport routes are extracted.

Transport routes are visualized by their codified number/name for ease of selection, using the common name tags reported by the underlying data source.

Route visualization

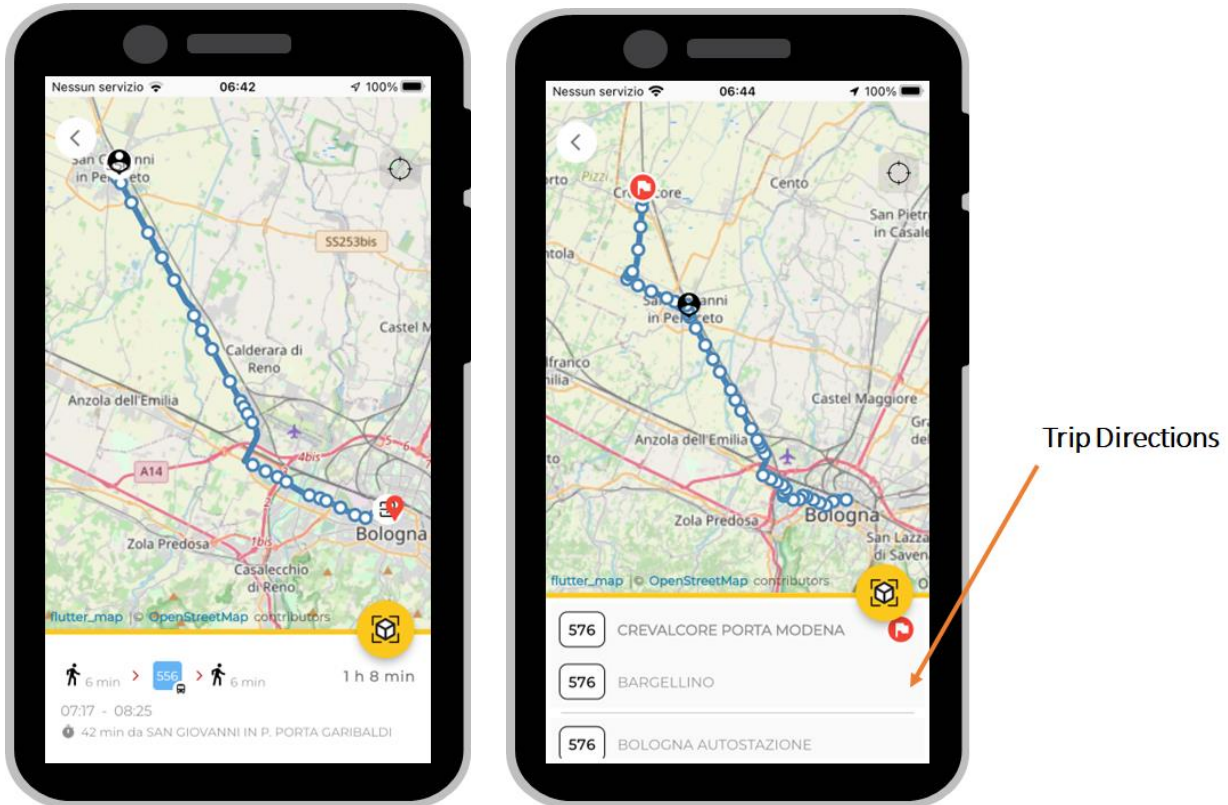


Figure 27 - AR Mobile App - Route visualization

Once a trip is planned, either by destination or route, a 2D map is visualized clearly showing:

- The user position
- the starting / ending transport stop, or the full transport route

In the case of transport routes, the trip directions are merged by destination, this way the user will not have doubts when choosing the direction (ex: side of the road).

The User can also know at a glance if the trip or the route is very distant from its position.

Live Mode – 2D Map

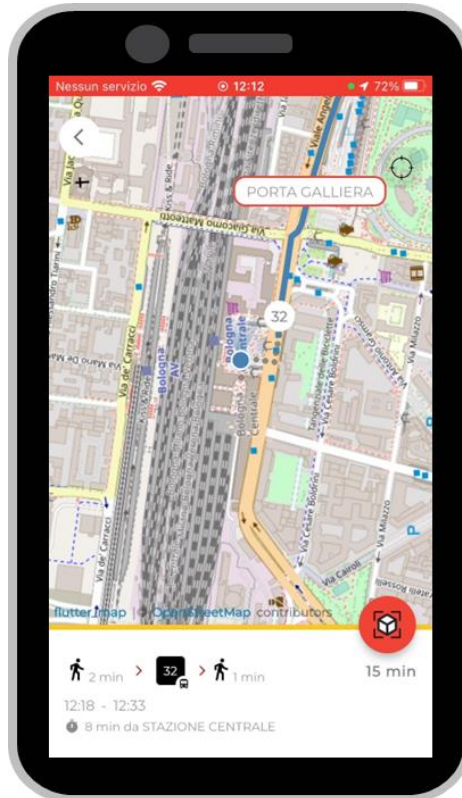


Figure 28 - AR Mobile App - 2D Map

Once a trip is selected and chosen by the User (pressing the AR live button) the application transitions to Live Mode.

Here the map becomes fully real-time:

- The 2D map if the phone is horizontally tilted, like a normal map.
- The map will be oriented like the User (using the compass).
- The positions of the bus stops are visualized and the nearest is highlighted.
- Realtime positions of the bus (virtualized) are shown.

Live Mode – 2D map to 3D AR Transition

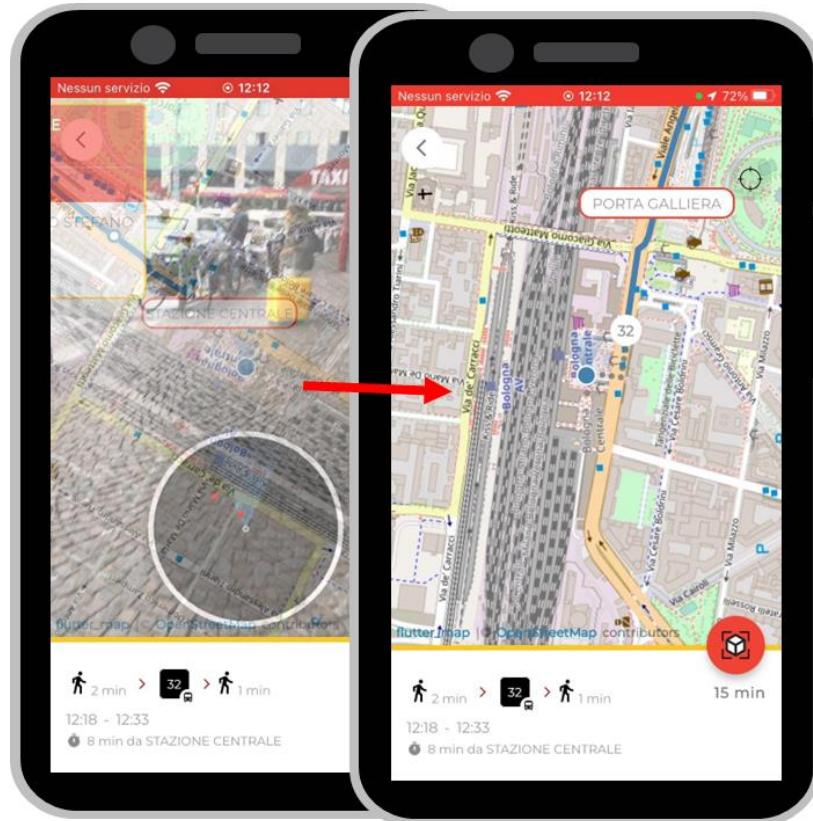


Figure 29 - AR Mobile App - 2D to AR 3D transition

One of the biggest innovations in the MIMOSA Mobilità AR application is the smooth transition between the 2D map view and the 3D AR Wayfinding.

As the user tilts the phone from horizontal to vertical, the two views blend together using alpha blending.

The User can then understand the surroundings and find its way to the nearest bus stop alternating the two views without the use of touch controls.

This mode is at the core of the AR View user experience and proved to be very effective in helping Users to reach their destination in time, also this is a very different and innovative approach compared to other info-mobility apps.

Live Mode – Augmented Reality View

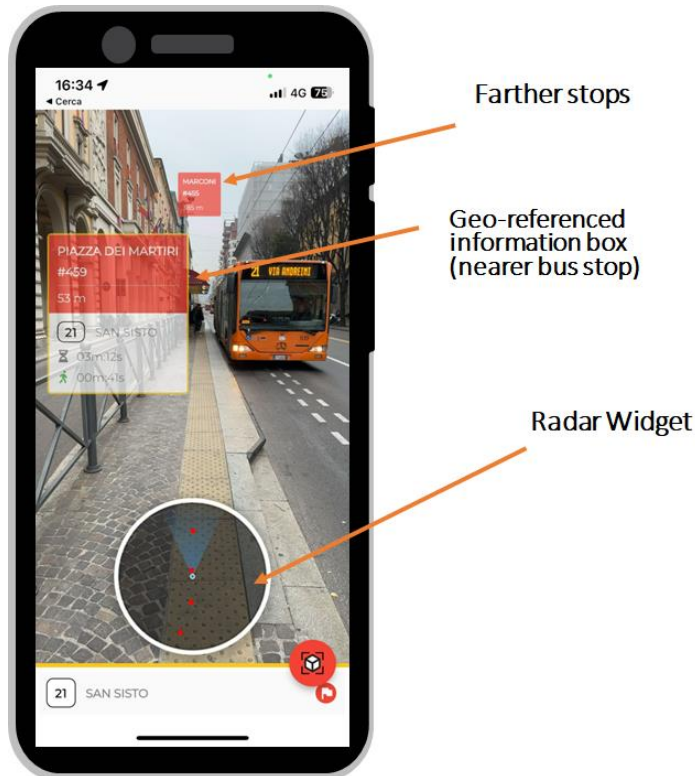


Figure 30 - AR Mobile App - 3D Augmented Reality View

When the user tilts its phone vertically it becomes a real Augmented Reality wayfinding device.

The live-view camera feed is super-imposed with Augmented Reality tags and widgets:

- The stops are shown as info-boxes with information, the bus arriving at the location, its time of arrival and the relative time for the user to reach the stop (this way the User can instantly know he/she is able to arrive in time and get the bus).
- Other, more distant stops are shown as smaller info-boxes, which can be opened by touching them.

This information-showing mode retains the ability of the user to tell what is near and far.

- A **Radar Widget** shows the position of the nearby:
 - Stops
 - Bus in real-time
 - Directions, if enabled

Live Mode – Augmented Reality Directions

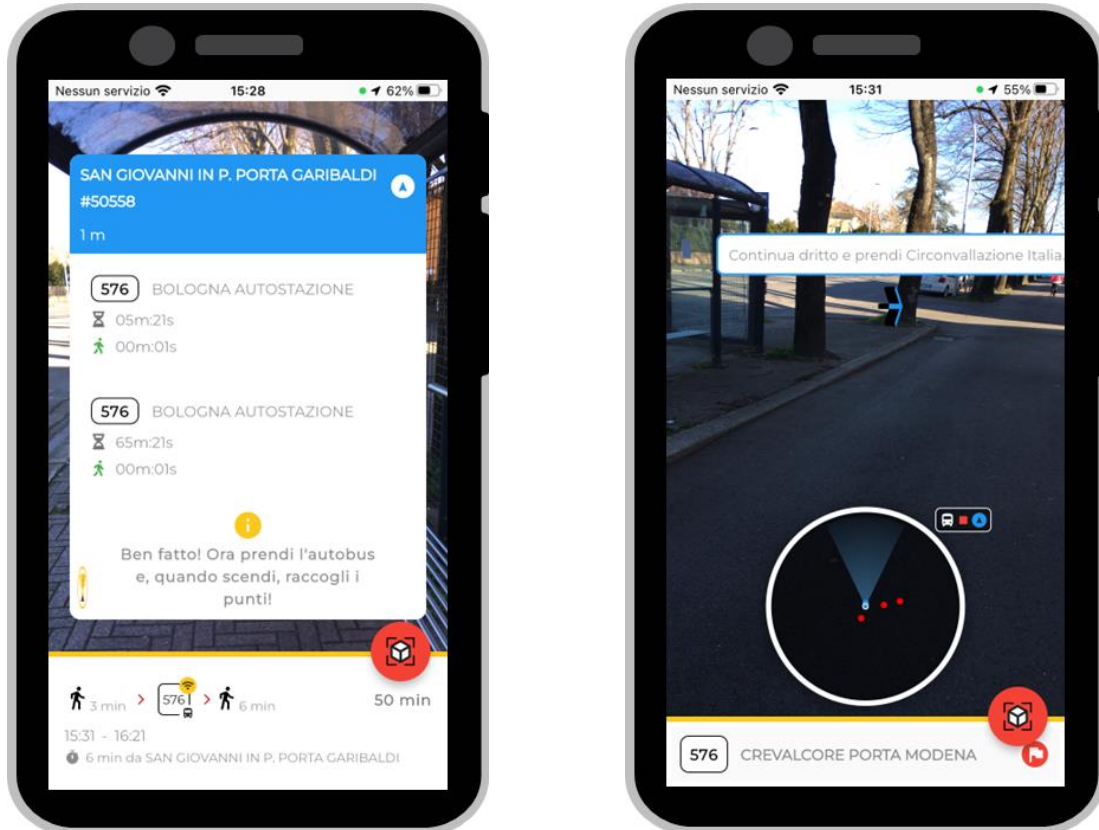


Figure 31 - AR Mobile App - AR Directions

The **Augmented Reality Directions** have been implemented as an extra and experimental feature. Once a bus stop is selected, the User can ask for direction to be generated in order to reach it:

- The 2D view will show also the direction in blue colour, every turn or road change;
- The 3D AR View will show the direction hints as virtual signs and information, the User can then follow the next waypoint using the AR view as a way-finder.

This mode, which is available in very complex applications like Google Live Street, does not need the camera feed to perform extra object recognition, instead uses the same sensor-fusion approach translating GPS points (generated by the OTP module) into virtual waypoints.

3 Open Source and Deployment

This MIMOSA pilot action embraces the philosophy of open-source development. With a strong emphasis on fostering collaboration and innovation, the pilot action aims to create an architecture of code that can be readily utilized by different institutions and organizations.

By leveraging the power of open-source software, MIMOSA seeks to eliminate the dependency on costly external APIs and services, enabling developers to create robust and feature-rich augmented reality mobility apps without the financial burden associated with proprietary solutions.

The underlying principle of open-source development within the MIMOSA project is to promote transparency, flexibility, and community-driven progress. By making the code fully accessible and openly available, the project empowers a wider network of developers, researchers, and stakeholders to contribute, modify, and build upon the existing codebase. This collaborative and inclusive approach enables the collective intelligence of the community to enhance the capabilities of the MIMOSA platform and drive continuous innovation in the field of augmented reality mobility applications.

The decision to prioritize open-source development in MIMOSA brings several distinct advantages to the table.

- It fosters **interoperability and standardization**, allowing for seamless integration and compatibility with diverse software ecosystems and infrastructure. Developers and institutions can freely adapt and customize the codebase to suit their specific needs, ensuring a flexible and tailor-made solution. Moreover, the reliance on open-source code eliminates vendor lock-in and mitigates potential risks associated with proprietary software, providing long-term sustainability and autonomy.
- Another notable benefit is the **democratization of technology**. By removing the financial barriers and dependencies on paid APIs and services, the project empowers a broader range of institutions, including those with limited resources, to develop augmented reality mobility apps that enhance the user experience and promote sustainable transportation practices. This democratization paves the way for greater innovation, collaboration, and knowledge sharing within the public transport sector, ultimately benefiting both developers and end-users.

3.1 Re-use of the codebase

The re-use of the MIMOSA codebase in other public administrations presents a wanted opportunity for the widespread adoption of augmented reality mobility applications without the

need for additional paid APIs and services. The philosophy behind MIMOSA is rooted in creating an architecture of code that can be easily utilized by different institutions, allowing them to develop their own customized augmented reality applications for mobility. By providing the codebase as open source, MIMOSA promotes collaboration, innovation, and cost-effectiveness in the development of similar applications.

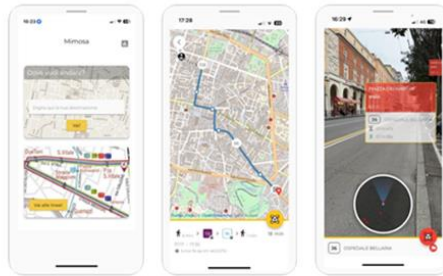
The modular design of the MIMOSA applications is a key factor that enables their re-deployment in various public administrations. The applications have been developed in modules, allowing them to be rebuilt and re-published in local projects. This modularity ensures that the applications can be tailored to meet the specific needs and requirements of different administrations, accommodating variations in infrastructure, policies, and user preferences. With this flexibility, public administrations can leverage the existing MIMOSA codebase as a solid foundation for their own augmented reality mobility applications, saving time and resources on development efforts. Moreover, the modularity of MIMOSA allows for easy extension of the codebase, enabling public administrations to add new functionalities and features as per their unique requirements. By building upon the existing modules, administrations can enhance the capabilities of their applications and address specific mobility challenges in their respective regions. This extensibility ensures that the MIMOSA codebase can evolve and adapt to future needs, providing a sustainable and future-proof solution for public administrations.

Another advantage of the MIMOSA project is the ability to embed **specific modules**, such as the augmented reality (AR) view, into other applications used by public administrations. This integration allows administrations to leverage the power of the AR module within their existing platforms, providing users with an immersive and interactive visualization of real-time mobility data. By integrating the AR module into their applications, administrations can enhance the user experience and deliver valuable information in a more engaging and intuitive manner.

3.2 Source Code Repository

The full source code is published in the GitHub account of ITL, publicly available and accessible.

<https://github.com/ITLBologna/MIMOSA-Mobilita-AR/>



Navigate your City using Augmented Reality

Figure 32 - Banner of the GitHub repository

The repository README contains the installation instructions that any developer/institution must follow to reproduce both the server infrastructure and the mobile application.

The repository LICENSE contains the MIT compliant license under which the repository is published.

3.3 Future Development

Here are some examples of possible implementations and improvements to the current architecture:

Connect and inter-operate with other applications, data sources, and transports

Public administrations can utilize the MIMOSA codebase to establish seamless connections and interoperability with other applications, data sources, and transportation systems. By integrating the MIMOSA modules with existing platforms, administrations can enhance the overall efficiency of their mobility ecosystem. This can enable the exchange of real-time data, such as public transport schedules, traffic updates, and event information, between different applications and transport types (for example, rentable electric vehicles).

Adapt real-time data to newer tracking technologies

As technology advances, new tracking technologies emerge that offer enhanced accuracy and precision in capturing real-time data. Public administrations can leverage the flexibility of the MIMOSA codebase to adapt real-time data from various sources to these newer tracking

technologies. For example, integrating advanced GPS systems or leveraging data from IoT devices and sensors can provide more accurate positioning information for vehicles and users. By incorporating these newer tracking technologies into the MIMOSA architecture, administrations can ensure that the real-time data presented to users is up-to-date and reliable, improving the overall user experience and promoting the use of public transport.

Extend the gamification module to promote city-related activities and projects

The gamification module within the MIMOSA codebase offers a unique opportunity for public administrations to engage citizens in city-related activities and projects. By extending this module, administrations can create interactive challenges, rewards, and incentives that encourage users to explore the city, discover landmarks, and participate in sustainable mobility initiatives. For example, administrations can design gamified features that promote walking or cycling by rewarding users for covering specific distances or visiting certain points of interest. This not only promotes healthier and more sustainable modes of transportation but also enhances user engagement and fosters a sense of community participation in the city's mobility goals.

Extend the AR View module to include new visual data and placeholders

The AR View module in the MIMOSA codebase provides a captivating and immersive way of visualizing real-time data and information. Public administrations can extend this module to include new visual data and placeholders that enhance the AR experience for users. For instance, administrations can integrate additional layers of data, such as air quality indexes, weather conditions, or even historical information about landmarks and cultural sites. By augmenting the user's view with contextual and relevant information, administrations can enrich the user experience, improve navigation accuracy, and promote a deeper understanding and appreciation of the city's surroundings. This expanded functionality of the AR View module can serve as a valuable tool for both residents and tourists, fostering a stronger connection between individuals and their urban environment.

4. Pilot Project

The first release of the MIMOSA Mobilità AR Application was published in **May 2023** only for a limited group of internal testing users.

The final version of the MIMOSA Mobilità AR Application is being tested in a Pilot Project deployed on June 15, 2023, and running until the end of July 2023. The extension after the project conclusion was justified to fully test all the App functions and set the bases for the utilization of the Mimosa solutions also after the project conclusion.

4.1 Testing Area

While the Romagna/Rimini area has been excluded due to ongoing transportation issues caused by recent floods, the chosen areas for the pilot project were **Ferrara, Imola and Bologna**. However, thanks to the availability of GTFS data for the entire Emilia Romagna region, the MIMOSA application will be functional throughout the entire region, providing comprehensive mobility solutions.

4.2 Terms of Service and Privacy Policy

To ensure compliance with legal requirements, significant attention has been dedicated to the Terms of Service and Privacy Policy of the application. Privacy permissions have been carefully addressed, and users will have the option to opt-in for GPS tracking, while also having the ability to opt-out from **passive tracking and profiling**, as well as the **gamification** features.

This approach ensures that users have control over their privacy and data usage within the application.

4.3 Social Media Campaign

To drive widespread adoption and usage of the MIMOSA application, a strategic social media campaign has been designed in collaboration with an advertising agency.

The objective is to create a compelling and impactful presence on social media platforms, engaging the target audience and raising awareness about the benefits of using the MIMOSA App for their mobility needs. Additionally, a contest with prizes has been incorporated to incentivize active participation from users. By rewarding the most active users, the pilot project aims to foster a sense of gamification and motivation among the users, further encouraging them to explore the functionalities and benefits of the MIMOSA application.



**LA TUA CITTÀ,
IN MANO**

**MIMOSA
MOBILITÀ AR**

PLAN | MOVE | WIN

**SCARICA
L'APP**

IMPOSTA LA META
AL VIAGGIO PENSIAMO NOI

MIMOSA MOBILITÀ AR È
DISPONIBILE SU APP
STORE E PLAY STORE

Figure 33 - MIMOSA Mobilità AR marketing campaign Ad



**REALTÀ
AUMENTATA**

 **ITL**
ISTITUTO DEI TRASPORTI
E DELLA LOGISTICA

Con un semplice sguardo attraverso il tuo dispositivo, potrai accedere a informazioni dettagliate sulle tue opzioni di trasporto pubblico

 **Interreg**
Italy - Croatia
MIMOSA


EUROPEAN UNION

SCARICA L'APP 

Figure 34 - MIMOSA Mobilità AR marketing campaign Ad

4.4 Pilot Project statistics and Results

Here is the data referred to the use of the application until June 30th:

Social Media Campaign

The following table shows the social media campaign (Facebook and Instagram) statistics to date:

Impressions	369.779
Interactions (link click)	732
Budget spent	€ 754

The campaign trend is low-performing for a public application and it is expected, as the info-mobility app space is very crowded and users are usually tied to an existing application they already know.

Nonetheless, the statistics are very good considered the low marketing budget and the very small time frame. The campaign, also, was “cold started” without building a community first.

Android



blue: new installations orange: updates

The MIMOSA Mobilità AR Android application was downloaded and installed by 64 users.

Apple (iOS)



The MIMOSA Mobilità AR iOS (iPhone) application was downloaded and installed by 53 users.

Usage Statistics

The following table reports the fine statistics on the real application usage:

Total Users	81
Consent to Gamification	30
Consent to Trip Analysis	28
Consent to Polls	25
Number of Trip searches	140
Number of Trip selection	480
Augmented Reality View uses	305
Number of Games	30
Users using Gamification	3

The statistics show a very good trend in the use of the Augmented Reality View. Users are being held back by the need to give all the privacy and security consents. Other info-mobility applications are less precise and compliant in terms of user privacy, but in MIMOSA Mobilità AR focused on policies of security and anonymity, so the on-boarding process is much more tedious than in commercial apps. The Gamification feature is still not really used, so the end of the Pilot Project must be reached before analysing the results.

5. Conclusions

This Mimosa pilot action (D.3.4.3) is related to the development of an innovative solution on mobility habits change aimed to raise awareness of the effects of reckless mobility choices from citizens. This Mimosa pilot propose an innovative approach based on augmented reality as innovative solutions able to improve the provision of real-time information on public transport to final users. Moreover, this pilot proposes some technical solutions on how it is possible to monitor the mobility behavioural change process. All these solutions can be considered as disruptive as it pushes people to use public transport, reduce pollution, congestion and increase the liveability of our cities.

The Mimosa App was tested in the Emilia-Romagna Region in the first semester of 2023 and in this report the key lessons learnt are summarized. It is important to underline that all the code used in the App is open-source and published online. All the interested stakeholders can use this code, or part of it. In developing similar applications in their territories. Another important aspect is related to the utilization of European data standard as the NeTEx (Network Timetable Exchange – standard CEN/TS 16614). The adoption of this EU standard allows to increase the replicability potential of this Mimosa solution.

The modules related to Augmented Reality and gamification processes are important innovations tested in this Mimosa pilot action as new solutions aimed to improve the access of final users to public transport key information. The results collected in the Mimosa pilot action demonstrate they are effective tools for the activation of mobility behavioural change processes. As defined in the last paragraphs, the ambition of PP3-ITL is to continue to work on the promotion and improvement of this Mimosa App. For these reasons PP3-ITL intends to continue the dialogue

with the key technical and political stakeholders in order to continue also after the project conclusion the valorisation of the key lessons learnt.

ANNEX 1. Open-Source Libraries for the AR Application

Here is the list of all the open-source libraries used for the development of the cross-platform mobile AR Application. Most of these are for the (Flutter, s.d.) ecosystem based on the Dart programming language.

ar_location_view: This library provides a widget that enables augmented reality (AR) location views in Flutter applications. It allows you to place 3D objects and annotations in the real-world location using GPS coordinates.

bitapp_functional_dart: A library that provides functional programming utilities for Dart. It includes common functional programming concepts such as higher-order functions, immutable data structures, and functional composition.

bitapp_http_x: This library is a wrapper around the HTTP client in Dart, providing additional features and convenience methods for making HTTP requests. It aims to simplify the process of making HTTP requests and handling responses.

bitapp_cache: A caching library for Dart that provides an easy way to cache data in your Flutter applications. It supports various caching strategies and allows you to store and retrieve data from different sources, such as memory or disk.

auto_size_text: This library provides a widget that automatically resizes the text to fit within its available space. It's useful when you have limited space and want to ensure that the text is readable and doesn't overflow.

confetti: A Flutter library for adding confetti animations to your applications. It provides various options for customizing the confetti, such as colors, shapes, and explosion patterns. It can be used to add a festive touch to your UI.

cupertino_icons: This library provides the Cupertino icon set, which includes icons commonly used in iOS apps. It allows you to easily use these icons in your Flutter applications, following the Cupertino design guidelines.

enum_flag: A Dart library that allows you to use enum as flags, enabling you to combine multiple enum values into a single value using bitwise operations. It provides utility methods for working with flag enum.

flutter_activity_recognition: This library allows you to recognize the user's current activity (such as walking, running, or driving) using the activity recognition capabilities of the underlying platform (Android or iOS). It provides methods for detecting and monitoring the user's activity.

flutter_inappwebview: A Flutter plugin that provides a WebView widget for embedding web content within your application. It supports various features such as JavaScript execution, navigation control, and cookie management.

flutter_keyboard_visibility: This library allows you to listen to keyboard visibility changes in your Flutter application. It provides a widget that can be used to determine whether the keyboard is currently visible or hidden.

flutter_launcher_icons: A command-line tool that automatically generates the necessary icon files for your Flutter application. It simplifies the process of creating and configuring app icons for different platforms and resolutions.

flutter_local_notifications: This library allows you to display local notifications in your Flutter application. It provides an interface for scheduling and showing notifications, as well as handling user interactions with them.

flutter_map: A Flutter library for displaying interactive maps. It supports various map providers (such as OpenStreetMap and Mapbox) and allows you to customize the map appearance and add markers, polygons, and polylines.

flutter_polyline_points: This library provides utilities for decoding encoded polylines and calculating polyline distances on the map. It's useful when working with routes or directions on a map in your Flutter application.

flutter_randomcolor: A Dart library for generating random colors. It provides methods for generating random colors with different configurations, such as brightness, saturation, and hue.

flutter_timezone: This library allows you to retrieve and work with time zone data in your Flutter application. It provides methods for getting the current time zone, converting between time zones, and retrieving time zone names.

flutter_typeahead: A typeahead (autocomplete) widget for Flutter that provides suggestions as the user types in an input field. It supports custom suggestion builders and allows you to handle user interactions with the suggestions.

font_awesome_flutter: This library provides the Font Awesome icon set, which includes a wide range of icons. It allows you to easily adapt the App to a specific context.

Analysis and compliance (MIT License)

The mobile application uses 47 selected open-source libraries.

- 38 are based on MIT License
- 7 are based on BSD-style license
- 1 is based on Apache License 2.0
- 1 is based on SIL Open Font License

While the server-side modules use these open-source libraries:

- MATOMO Analytics, based on MIT License
- OTP Open Trip Planner, based on Apache License 2.0

According to our analysis, all these licenses are compatible with the **MIT License**, which is the release license of the MIMOSA produced code-base published on the GitHub repository.

The MIT License (MIT License, s.d.) is a permissive open-source license that allows the developer to distribute and modify the code under certain conditions. It grants users the freedom to use, copy, modify, merge, publish, distribute, sublicense, and/or sell the software.

ANNEX 2. Cloud Service Deployment

Here is the technical description of the virtual machine cloud instances deployed for the core MIMOSA services, dimensioned for the Pilot Project, which should handle from 100 to 1000 active users:

Amazon Web Services Instances:

- 5 VM EC2, tu O.S. Ubuntu 22.04 LTS
 - Mimosa Production (r6g.medium, EBS 16Gb)
 - Mimosa Development (r6g.medium, EBS 16Gb)
 - Mimosa OTP (OpenTripPlanner, r6g.medium, EBS 16Gb)
 - Mimosa Analytics (Matomo, t4g.small, EBS 50Gb)
 - Mimosa Pelias (Pelias, c6i.large, EBS 24Gb)

Network environment:

- 1 VPC, 3 IP elas (Prod/Dev/Analycs). Security group with following rules:
 - Prod: TCP/80 and TCP/443 public
 - Dev: TCP/80 and TCP/443 public
 - OTP: TCP/8080 on LAN
 - Analytics: TCP/80 and TCP/443 public
 - Pelias: TCP/4000 on LAN

Bibliography

- Apple ARKit*. (s.d.). Tratto da <https://developer.apple.com/augmented-reality/>
- Apple VisionOS*. (s.d.). Tratto da <https://developer.apple.com/visionos/>
- Cats, O. S. (2017). *Understanding traveler satisfaction and acceptance of real-time public transport information in a multimodal travel setting. Transportation Research Part C: Emerging Technologies, 75*, pp.86-104.
- Flutter*. (s.d.). Tratto da <https://flutter.dev/>
- Google ARCore*. (s.d.). Tratto da <https://developers.google.com/ar>
- Hong, J. Y. (2016). *Smartphone applications that visualize real-time transit data: Effects on users' spatial cognition and orientation. Computers, Environment and Urban Systems, 59*, pp.110-120.
- Li, Q. C. (2018). *Impact of real-time bus arrival information on ridership: Evidence from a field experiment. Transportation Research Part C: Emerging Technologies, 95*, 548-562.
- MIT License*. (s.d.). Tratto da <https://opensource.org/license/mit/>
- Schmöcker, J. D. (2019). *Does real-time information reduce travel uncertainty and increase the attractiveness of public transport? Transportation Research Part C: Emerging Technologies, 101*, 76-94.
- Schöning, J. e. (2020). *Google Maps and Augmented Reality: Improving Navigation in Public Spaces. Journal of Location Based Services, 14(1)*, 1-22.
- Susilo, Y. a. (2015). *Influence of real-time transit information on travel behaviour: The London case study. Transportation Research Part C: Emerging Technologies, 54*, pp.1-18.
- Tang, J. T. (2020). *Investigating the impact of real-time transit information on mode choice behavior. Transportation Research Part C: Emerging Technologies, 112*, 360-378.
- Wang, L. e. (2021). *The Role of Augmented Reality in Infomobility: A Study on Google Live View. Journal of Information and Communication Technology, 20(1)*, 35-57.

Watkins, K. F. (2011). *Where Is My Bus? Impact of mobile real-time information on the perceived and actual wait time of transit riders. Transportation Research Part A: Policy and Practice, 45(8), 839-848.*