

Article

Not peer-reviewed version

A Recommendation Engine for Multimodal Transport Route Planning Using Shared Vehicles of Different Types

[Efthymios Chondrogiannis](#)*, [Leonidas Avdelas](#), [Antonis Litke](#)*, Theodora Varvarigou

Posted Date: 9 February 2026

doi: 10.20944/preprints202602.0595.v1

Keywords: vehicle sharing; multimodal transportation; route planning; recommendation systems; machine learning; smart cities



Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a [Creative Commons CC BY 4.0 license](#), which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

A Recommendation Engine for Multimodal Transport Route Planning Using Shared Vehicles of Different Types

Efthymios Chondrogiannis *, Leonidas Avdelas, Antonis Litke * and Theodora Varvarigou

Institute of Communication and Computer Systems, Ir. Politechniou Str. 9, Zografou Campus, 15773, Athens, Greece

* Correspondence: chondrog@mail.ntua.gr (E.C.); litke@mail.ntua.gr (A.L.)

Highlights

What are the main findings?

- A layered architecture for personalized multimodal environment- and traffic-aware route-planning recommendations built upon existing services, systems, and pre-trained LLMs.
- An advanced route-planning mechanism for heterogeneous station-based and dockless shared vehicles, evaluated using simulated data.

What are the implications of the main findings?

- Facilitates the daily travel of smart city citizens by considering a wide range of transport options, including shared vehicles of different types and conventional modes of transport.
- Contributes to sustainable transformation by promoting the use of environmentally friendly shared vehicles that can be easily integrated with conventional means of transport.

Abstract

Vehicle-sharing platforms are constantly gaining ground in smart cities around the world, reducing the number of traditional fuel-based vehicles on the roads in busy areas and thus contributing to the development of a sustainable environment. On the other hand, the availability of a plethora of shared vehicles of different types across a city increases the need for their seamless combination, so that they are considered part of a unified transportation system within a smart city rather than independent solutions. In this work, we present a novel system that enables authorized users to gain access to shared vehicles of different transport modalities, allowing them to reach their destination without relying on a private car or public transport. For this purpose, we have used existing systems and techniques from different fields, such as recommendation systems, machine learning, and route planning, which provide appropriate multimodal routes while taking into consideration several parameters, including user demographics, the current status of vehicles, environmental conditions, and road traffic congestion. The evaluation of the system using simulated data indicated that it can adequately serve its purpose and highlighted some issues that should be considered in future work.

Keywords: vehicle sharing; multimodal transportation; route planning; recommendation systems; machine learning; smart cities

1. Introduction

Vehicle sharing is constantly gaining ground [1], including micro-vehicles such as e-scooters and e-bikes, as well as larger vehicles such as cars and even sea vessels. Each of these vehicle types has its own advantages and limitations, which are strongly related to user preferences, environmental conditions, traffic levels, and city infrastructure. Hence, their efficient combination and usage by members of a community is rather challenging, given the plethora of issues that need to be addressed. For instance, car driving is directly affected by traffic conditions and the availability of parking

spaces. On the other hand, e-scooters and sea vessels are highly affected by environmental conditions. In any case, these vehicles can improve access to public transport and simultaneously reduce the usage of private vehicles and road traffic, thus contributing to the development of a sustainable environment.

In this work, we present the system developed for sharing vehicles provided by the partners involved in the EcoMobility¹ project among members of a community, with particular focus on identifying the most appropriate route for each user, taking into consideration the unique characteristics of the vehicles, users' profiles, environmental conditions, such as weather and traffic levels. For this purpose, we have used existing systems and services, such as OpenStreetMap [2], as well as pre-trained large language models (LLMs) such as Qwen3 [3], to efficiently process user data, identify feasible routes, and rank them so that the most appropriate options appear at the top of the list. The evaluation of the developed system using simulated data involving different types of vehicles and users in Istanbul indicated that it can adequately serve its purpose, while also highlighting its strengths and areas for further improvement.

The document is organized as follows. In Section 2, we present the state of the art in two distinct but highly relevant fields, namely recommendation systems and route planning, along with the use of machine learning (ML) and deep learning (DL) techniques in these areas. In Section 3, we analytically describe the proposed approach and the developed system, which enables authorized users to identify appropriate routes using multiple means of transport, while taking into account their demographic characteristics, environmental conditions, and relevant events such as traffic jams. An example is presented in Section 4, providing further details on the algorithms implemented in each layer, based on the distribution of several vehicles around Istanbul, and relevant issues are discussed. In Section 5, we present the evaluation of the proposed system using simulated data, and the lessons learned through this process are summarized. The limitations of the developed system and its underlying background mechanism are further discussed in Section 6. Finally, in the last section, we highlight the main contributions of this paper and outline future work.

2. Related Work

2.1. Recommendation Systems and AI Techniques

Recommendation systems have been in the market for more than 30 years. However, the techniques used have significantly changed, taking into consideration the technological advancements occurred during this period.

Recommendation systems can be organized into three broad categories, namely content-based filtering, collaborative filtering, and hybrid systems [4]. Content-based filtering systems rely on the attributes of the items already selected by the user in order to recommend new ones. Collaborative filtering systems are based on evaluation data explicitly or implicitly provided by users to identify their preferences, and they can be further divided into memory-based and model-based recommendation systems. Hybrid recommendation systems combine two or more techniques to improve their performance, usually in order to address the cold-start problem. Burke [5] has presented different ways in which two or more techniques can be combined. The simplest approach is to apply weighting schemes. Alternatively, the system may choose which technique to use each time or combine the recommendations produced independently by each technique, among other options.

The ideal context-aware recommendation system should reliably label each user action with the appropriate context and effectively tailor the system output to the user in the given context. Adomavicius et al. [6] focused on how context can be defined and used in a recommendation system to provide more intelligent and useful recommendations. More precisely, they have focused on specific contextual factors such as time, location, and purchasing purpose, what a recommendation

¹ EcoMobility project, <https://www.ecomobility-project.eu/>

system may know about these factors (i.e., the extent to which this information is available or observable to the system), and how they change over time (i.e., some factors remain stable, whereas others vary over time). Contextual information can be obtained explicitly (i.e., directly from users and/or sensors) or implicitly through the analysis of historical data, and it can be used at various stages of the recommendation process, such as filtering the input data (i.e., contextual pre-filtering) and ranking the results according to user preferences. Therefore, AI/ML techniques [7] can be used to learn from historical data (e.g., static unobservable and/or dynamic observable contextual information) and to leverage the resulting ML models within the recommendation process.

Over the past decades, recommendation systems have used traditional data mining techniques, such as text mining (e.g., TF-IDF), data clustering (e.g., the K-means algorithm [8]), and matrix factorization [9], to analyse the characteristics of users and items, and especially the relationships and patterns among them. In the big data era [10], where the information available about both users and items has increased significantly, neural networks have been widely adopted in recommendation systems, benefiting from improvements in hardware computing power and from recent deep learning advances, such as transformers and self-attention mechanisms [11].

Deep neural networks (DNNs) [12,13] can be used to identify complex patterns among users and items that traditional ML techniques and recommendation systems cannot adequately model or capture. Based on the survey conducted by Zhang et al. [14], standard deep learning (DL) techniques such as multilayer perceptrons (MLPs), convolutional neural networks (CNNs), recurrent neural networks (RNNs), including long short-term memory (LSTM) and gated recurrent unit (GRU) networks, restricted Boltzmann machines (RBMs), and autoencoders (AEs), as well as more contemporary DL architectures such as generative adversarial networks (GANs), transformer-based architectures (using attention mechanisms), deep reinforcement learning (DRL), and graph neural networks (GNNs), have been incorporated into recommendation systems. These techniques have demonstrated tremendous success in both supervised and unsupervised learning tasks. Moreover, in some cases, recommendation systems have utilized more than one of the aforementioned approaches.

Xu et al. [15] studied how multimedia information can improve recommendation systems (e.g., increase performance and mitigate the cold-start problem), particularly through feature extraction from multimedia data and the incorporation of these features at different stages of the recommendation process. Based on their survey, different application scenarios encompass varying types of modality information, leading to diverse datasets with distinct multimodal features. Nevertheless, most datasets include at least three primary modalities, namely interaction, visual, and textual. Within the visual domain, convolutional architectures are widely used for feature extraction, whereas in the text domain, both traditional methods such as TF-IDF and more sophisticated neural network approaches are used. In particular, after the introduction of transformers with self-attention mechanisms, existing pre-trained DNNs, including large language models (LLMs) [16,17], have been widely adopted for this purpose. The stage at which information from different modalities is integrated (e.g., before or after the encoding process), as well as the integration method used (e.g., concatenation or element-wise operations, with or without weighting), can also affect recommendation performance.

2.2. Multimodal Transportation and Route Planning

Route planning aims to identify the optimal path between two locations and is closely related to graph theory. A road network can be represented as a graph, where intersections are modelled as nodes and roads as edges, which can then be searched to determine the shortest path. In road networks, the best route often depends on the departure time in a predictable way [18]. For example, certain roads are consistently congested during rush hours, and certain buses or trains operate with different frequencies throughout the day. Therefore, the duration of each intermediate step can be represented in the graph (e.g., as edge weights), allowing the quickest route to be identified using a shortest-path algorithm such as Dijkstra, Bellman-Ford, or A*, among others. A similar approach can

also be applied to public transit networks. Nevertheless, the exact process depends on the model used to incorporate the temporal dimension into the graph (e.g., time-expanded or time-dependent).

Detecting the appropriate path may be quite complicated, since it is often necessary to take into consideration multiple parameters, such as travel time and monetary cost. In such cases, we can either use an algorithm that supports multiple criteria (e.g., RAPTOR [19]) or compute a linear combination of different costs (often referred to as generalized cost [20]) and handle the problem accordingly as an earliest-arrival problem using a shortest-path algorithm. In case of public transport networks, precomputing transfer patterns can significantly improve the efficiency of routing queries, even in very large and poorly structured networks, as shown by Bast et al. [21]. When several different modes of transport need to be considered, a multimodal graph can be obtained by first constructing individual graphs for each transportation mode and then merging them by introducing appropriate links to enable modal transfers. The complexity of both the graph and the algorithm increases, especially when additional parameters, such as uncertainty and delays, must also be taken into account.

Wu et al. [22] focused on the use of ML techniques for providing personalized recommendations in multimodal transportation. More precisely, they defined the problem and the required data so that supervised ML techniques can be used to learn user behaviour based on historical data, and the resulting ML model can then be applied to select appropriate modes of transport and/or rank them. According to their work, it is necessary to consider the origin and destination (i.e., query data), the responses previously proposed by the system (i.e., candidate data), and the options selected by users, along with user attributes and road traffic conditions, in order to extract appropriate features that serve the intended purpose. However, these data should be further processed before being used by supervised ML techniques.

To detect the most appropriate route(s), it is often necessary to generate all alternatives (i.e., the choice set) and then either select the most appropriate one or determine the order in which they should be presented to the user. In the latter case, Path Size Logit (PSL) route choice models [23] are often used, especially in public transport [24]. These models are based on logit models (i.e., logistic regression) and take into consideration the correlations among the available options (i.e., common parts of the routes). On the other hand, they depend on the structure of the utility functions, which must be specified in advance by the modeller and are often defined as a linear combination of relevant parameters (e.g., duration and cost).

Deep neural networks (DNNs) can capture more complex interactions among variables and have recently been used in route planning. Marra and Corman [25] presented a DNN that takes into account all available options (i.e., their feature vectors), along with additional parameters (e.g., weather conditions), and applies several convolutional layers with multiple filters and a non-linear activation function in order to learn the utility function. The adopted approach and the structure of the DNN architecture ensure that the system provides the same results regardless of the order in which the available options are given (i.e., permutation invariance). On the other hand, the developed DL-route model, like any other DNN, is not easily interpretable.

Hydra [26] is a personalized and context-aware multimodal transportation recommendation system used by Baidu Maps² that leverages existing unimodal route engines to generate feasible routes across different transport modes. For multimodal transportation, a network is constructed (i.e., a graph in which nodes represent locations and links represent travel actions associated with a specific mode), which Hydra then searches to identify feasible paths while taking into account the required number of modal transfers. For each path, several features are computed that are closely related to the origin and destination, the stations and other points of interest (POIs) in the surrounding area, the routes that users are expected to follow, and the city region(s) they traverse. These features, together with the user profile, temporal constraints, and meteorological data, are used

² Baidu Maps, <https://map.baidu.com/>

to compute a score based on a pre-trained ML model which is finally used by Hydra to recommend the appropriate mode.

Deep reinforcement learning (DRL) algorithms have also been used to address a variety of transportation problems [27], including autonomous energy-efficient driving, adaptive traffic signal control, and vehicle route planning. Furthermore, several new DRL techniques have been proposed to solve routing problems based on encoder–decoder architectures with attention mechanisms and graph embedding networks [28]. However, processing data generated through direct user interaction with the system is costly and risky. Therefore, in many studies, DRL has been applied offline to learn an appropriate policy based on historical data, which is subsequently leveraged in the online version of the recommendation system [29].

Multimodal transportation also encompasses the use of micro-vehicles, that is, vehicles with a mass of no more than 350 kg and a design speed of no more than 45 km/h [30]. Human-powered and electrically assisted micro-vehicles, such as bicycles, e-bikes, and e-scooters, can improve access to public transport by addressing, to some extent, the first- and last-mile problem [31]. Apart from privately owned micro-vehicles, such as bicycles, which have been used as a mode of transport for a long time, micro-vehicle sharing systems (e.g., Lime and Bird for e-scooters) are constantly gaining popularity in many urban cities around the world [32]. The micro-vehicles in these systems may be located at predefined stations (i.e., station-based systems), or parked almost everywhere (i.e., dockless systems) within specific zones or areas which complicate the route planning process. Especially in the case of electric vehicles, this process becomes more complicated because the system must also consider battery levels, vehicle range, and charging-station availability [33].

Vehicle sharing is, to some extent, conceptually related to carpooling and ride-sharing. In carpooling or ridesharing, a driver and one or more riders share part of a common trip using the driver's car, offering clear economic and ecological benefits. On the other hand, carpooling complicates the route planning process, since there are no fixed stops and drivers are often willing to make short detours. To address this problem, Huang et al. [34] modelled public transport and carpooling as two independent time-expanded graphs, which were subsequently merged into a single graph while taking into account the fuzziness and flexibility introduced by carpooling. The output of this process is a time-dependent multimodal graph that can be searched to identify potential routes from an origin to a destination using standard route planning algorithms, while also considering the departure or arrival time range, when specified. Deep reinforcement learning (DRL) algorithms have also been applied to freight delivery and, in particular, to on-demand ridesharing problems [27], including order dispatching (i.e., matching rider requests with available vehicles), vehicle repositioning (i.e., proactively repositioning idle vehicles from one zone to another in order to balance vehicle supply and rider demand), and joint decision-making for order dispatching and vehicle repositioning.

3. Methodology

3.1. Approach Followed and System Developed

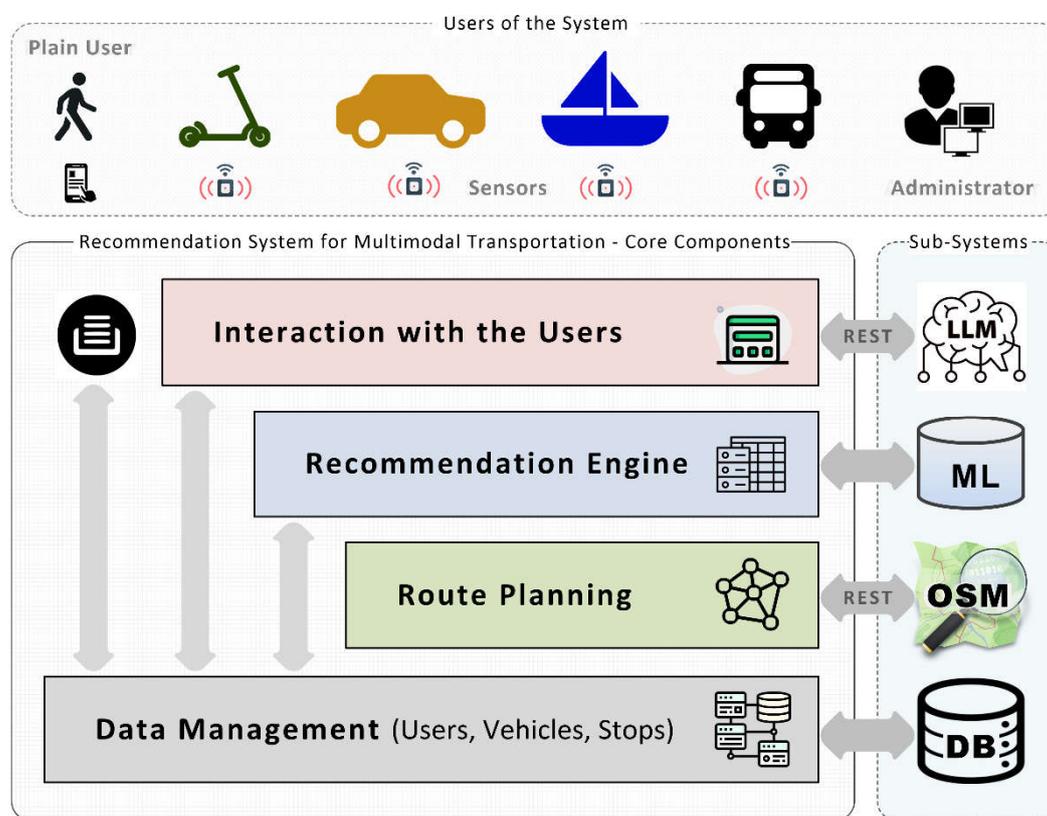
To facilitate the daily mobility of citizens in a smart city, we developed a system that enables them to identify the route that best suits their needs by using multiple transportation modes, each with its own capabilities and characteristics. In this work, we focus on four different means of transport, namely walking, using an e-scooter, traveling by car, and traveling by sea vessel. All vehicles considered in this work are being developed by organizations involved in the EcoMobility project [35], and they are, to a great extent, autonomous and environmentally friendly. These vehicles are intended to be temporarily shared among members of a community under a specific pricing scheme. Nevertheless, the proposed approach can be extended to encompass additional private vehicles and public means of transport, such as cars, bikes, buses, metro, trains, and airplanes.

Citizens can use the developed system to identify the most appropriate means of transport for reaching their final destination, taking into account vehicle availability, environmental conditions,

and traffic levels. To meet the requirements of a real-world scenario, the system distinguishes three different types of users, each with different capabilities and interaction methods. System administrators are responsible for initializing the system, such as registering available vehicles and entering additional information about them, as well as monitoring their overall status (e.g., problem reports). Data collected from sensors installed on vehicles and/or at different locations within a smart city can provide information about the current location of vehicles, as well as weather conditions and traffic levels on roads. Regular users (or simply users) can interact with the system using their smartphones in order to reach their final destinations. Therefore, they can register in the system, receive recommendations regarding the paths or routes they should follow, reserve relevant vehicles in advance (by paying the required amount), and release them at the end of the trip or even earlier.

The architecture of the developed system, along with the involved user types, is presented in Figure 1. The upper layer facilitates the interaction between end users and the system, whereas the bottom layer handles data management. The two intermediate layers detect the available means of transport, identify the route(s) that can be followed using a combination of different transportation modes (route planning), and rank them according to the user's profile and goals or preferences (recommendation engine). This constitutes the most challenging part of the system; therefore, in the remainder of this paper, we focus on the algorithms implemented in these two software components.

Users (either regular users or administrators) interact with the system through the GUI by invoking the corresponding REST APIs. In contrast, sensors communicate with the system via an MQTT³ broker by publishing messages to the appropriate queue, which are then further processed by the system. In both cases, the exchanged data are provided in JSON⁴ format and must comply with a predefined schema. For user convenience, registered users can also submit requests using natural language, and the system can automatically extract the required technical details (e.g., the latitude and longitude of the destination) before interacting with the underlying components of the architecture through the corresponding REST APIs.



³ Message Queuing Telemetry Transport (MQTT) protocol, <https://mqtt.org/>

⁴ JavaScript Object Notation (JSON), <https://www.json.org/json-en.html>

Figure 1. Layered System Architecture and User Roles.

The recommendation engine generates the available options for reaching the final destination in an appropriate order, taking into consideration the characteristics of the users, their origin and destination, as well as any additional parameters specified. From a technical perspective, users must provide their unique IDs, the latitude and longitude of their origin and destination, and an additional object describing their preferences (e.g., searching for trips with minimum total duration) and any restrictions that eligible paths should satisfy (e.g., avoiding the use of a sea vessel). However, these additional parameters are optional. The system relies on data stored in the data source to retrieve further information about users (e.g., demographic characteristics) and vehicles (e.g., capabilities and status), as well as their current locations on Google Maps⁵. Moreover, it takes into account current weather conditions and road traffic levels, ensuring that the routes proposed by the system are feasible and satisfy the specified constraints. More information about the data management and the data structures used is available in Appendix A.

For efficient processing of the input data, the system is based on widely used models, services, and software components, enabling it to take advantage of existing knowledge available on the web. In our work, the Qwen3⁶ LLM was used to transform user requests expressed in natural language into JSON format, so that they can be further processed by the underlying software components. In addition, the OpenStreetMap⁷ API was used to identify the route that should be followed to move from one location to another using a particular mode of transport (e.g., on foot or by car). Regarding data storage, SQLite⁸ was used in combination with an appropriate Object-Relational Mapping (ORM) library. Furthermore, a ML model was developed based on historical data and subsequently used to rank the suggested paths before presenting them to the end user.

Before proceeding to the description of the algorithms, we describe the assumptions made for the different means of transport, and in particular for the vehicles supported by the system. First, we assume that a user can walk from one location to another under any conditions, provided that a valid path exists between them and the distance does not exceed a predefined threshold. All e-scooters are considered identical and may be available almost everywhere. Therefore, a user can rent an e-scooter and leave it at a convenient location (e.g., close to an intermediate or final destination), provided that weather conditions are suitable (i.e., no rainfall) and the travel distance does not exceed a second predefined threshold. For car travel, and considering the difficulty of finding a free on-street parking space, especially near busy areas, we assume that cars should be parked in predefined parking areas that guarantee the availability of free parking spaces. Thus, cars can be used for traveling between different parking locations or areas independently of weather conditions, provided that the travel distance does not exceed another predefined threshold. Nevertheless, drivers may leave the vehicle at a different location when this is feasible. Car trips are also strongly affected by traffic congestion. Regarding sea vessels, we assume that travel between ports is possible provided that there are no strong winds and that a valid route exists between the corresponding ports. Furthermore, we assume that vehicles cannot be carried using another vehicle. For example, users cannot take an e-scooter with them when traveling by car or sea vessel. Consequently, a vehicle must be released before switching to another mode of transport.

3.2. Route Planning

This software component identifies the feasible routes that users can follow to reach their destination and computes several parameters for each alternative, so that the system can subsequently use this information to select the most appropriate one. This is a rather complex process,

⁵ Google Maps, <https://www.google.com/maps>

⁶ Qwen3, <https://lmstudio.ai/models/qwen3>

⁷ OpenStreetMap, <https://www.openstreetmap.org/>

⁸ SQLite, <https://sqlite.org/>

given the large number of available options and the multiple parameters that must be considered. To address this challenge, we apply a novel approach in which feasible routes are progressively detected and refined. The process begins with an abstract representation of the entities within a given area and the links among them, enabling the system to identify a set of candidate routes. These candidates are then further processed and filtered to determine the concrete elements of each route and ensure that all relevant constraints are satisfied.

The proposed approach consists of three consecutive steps, which are illustrated in Figure 2. First, the system identifies the functional vehicles registered in the system that are not currently in use by any other user (available vehicles), the existing car parking areas and sea-vessel ports (stops), the current weather conditions, and the road segments affected by traffic congestion. Next, it examines the current locations of cars and sea vessels in order to determine the stop to which they belong (note that a car may also be parked outside the predefined car-parking locations), so that the system is aware of vehicle availability at the corresponding locations. The scooters are clustered using DBSCAN [36], and a representative scooter located near the center of each cluster is subsequently selected (referred to as an e-scooter representative or e-scooter area).

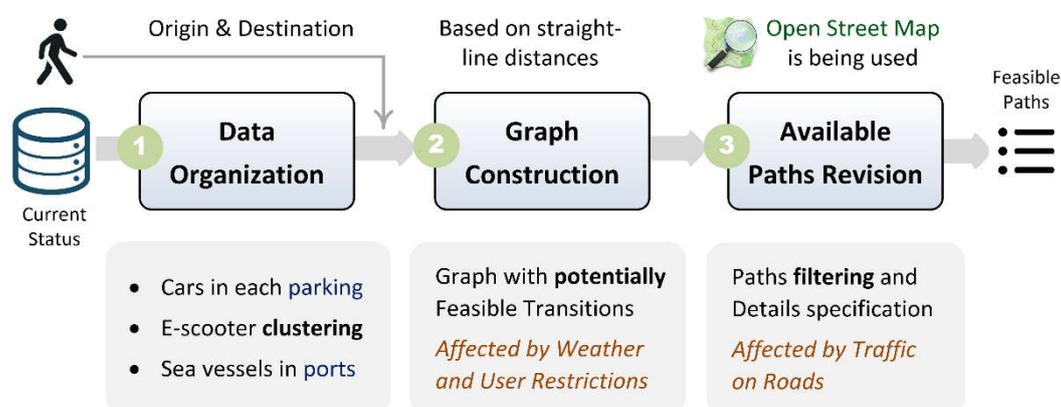


Figure 2. The three main steps of the route planning algorithm.

Then, the system constructs a graph that represents all available means of transport, along with links among the relevant entities, so that it can be used to identify candidate routes for reaching the destination. This graph includes the user's origin and destination, as well as intermediate stops where the user may change the mode of transport, including parking areas, ports, and e-scooter areas. The links between the nodes of the graph are defined based on the characteristics of each node. More precisely, travel from one parking area to another is allowed provided that a car is available at the origin parking area and the distance is within the allowable driving limits. The same assumption applies to sea vessels. In addition, walking between two nodes is permitted if the distance is within a predefined walking threshold. For e-scooters, since they can be left almost anywhere, they can be used to travel to any other node, provided that the travel distance does not exceed a predefined threshold. During the construction of the graph, straight-line distances are used to quickly obtain a rough estimation of the potential travel distance and filter infeasible links. However, the thresholds used for filtering links are slightly smaller than those specified during system configuration, since the actual distance between them (if it exists) is expected to be shorter. Finally, for each link included in the graph, we record the associated mode of transport and the corresponding straight-line distance.

It should be noted in advance that the construction of this graph is highly affected by environmental conditions and user restrictions or preferences. More precisely, e-scooters cannot be used in cases of heavy rain, and sea vessels cannot be used in the presence of strong winds in the corresponding areas. In addition, users may explicitly specify that one or more means of transport should be avoided; therefore, the corresponding links are not introduced into the graph during the aforementioned process. Using this graph, the system can identify all feasible paths that can potentially be followed to reach the destination. However, each candidate path must be further

examined to determine the actual route to be followed (if it exists) and to estimate the real distance and expected duration of each step.

For each path, we initially compute a pattern that consists of the sequence of transportation modes it includes, so that it can be used to quickly reject inappropriate alternatives before proceeding to the next steps, which are more complex and time-consuming. For example, we reject paths in which the same mode of transport is used in two consecutive steps without justification, such as walking to a parking area and then walking again to another location instead of switching to a car. Next, we examine the intermediate steps of each path in order to determine the concrete route to be followed for the specified transportation modes, as well as the expected distance, duration, and cost, among other parameters. During this process, the OpenStreetMap (OSM) service is used to ensure that the transition between two points is feasible using the selected mode of transport and to retrieve route details, along with estimations of the expected travel distance and duration, which are further evaluated based on the specified thresholds. In the case of e-scooters, if the travel distance exceeds the specified threshold, we additionally examine whether the remaining distance can be covered on foot. Regarding cost, in our work, it depends only on the selected mode of transport; however, more complex pricing policies could also be used.

For each of the remaining paths, we compute several features, such as the total distance, duration, and cost. In addition, we compute the number of edges, the set of distinct transportation modes used, and the total distance and duration covered by each mode separately. In case a car or a sea vessel is used, the total duration is slightly increased to account for the additional time required to pick up and return the vehicle at the corresponding locations. In contrast, e-scooter usage is relatively straightforward and does not introduce a significant delay. Traffic conditions are also considered. In particular, when car travel is involved, we examine whether a route passes through high-traffic areas and, if so, we increase the estimated travel duration accordingly.

At the end of the aforementioned process, we obtain a list of feasible paths, each associated with multiple parameters, including distance, duration, and cost. These alternatives can then be presented to the end user, either in full or as a subset, often ranked in ascending order of total duration. However, this ranking is not always optimal, since users may have different goals and preferences. Therefore, in order to increase the likelihood that the recommendations are useful and selected, routes must be ranked by taking multiple parameters into account, as described in the next section.

3.3. Recommendation Engine

The aim of the recommendation engine is to refine the feasible routes generated by the route planning component and present them to the end user in an appropriate order, so that the most suitable routes appear at the top of the list. To achieve this, the system offers users two main options. First, it allows users to specify their preferences regarding the parameter(s) of particular importance (e.g., prioritizing total duration) as well as any restrictions (e.g., avoiding the use of a scooter), so that this information can be used to filter and subsequently rank the available routes. The specified restrictions are Boolean expressions that define conditions the routes must satisfy (e.g., the route should not include a scooter and/or the total walking distance should be below a threshold) and are therefore used to filter the routes identified in the previous step. The preferred parameter(s) are expressed as an ordered list of one or more route features, which are then used to sort the remaining eligible routes before they are presented to the user. For example, an adult user may specify that, at a given moment, they are interested in the route with the minimum total duration, without necessarily requiring the use of a car or a sea vessel.

It should be noted in advance that the above parameters are optional and may be specified by users each time they interact with the system. This approach may be daunting for some users, since they are required to provide not only their origin and destination, but also additional information at every request. Moreover, this mechanism does not always allow the system to fully capture user intent, as it focuses on individual parameters and constraints rather than the user's needs as a whole and the complex dependencies among the computed features. For this reason, a second option is

provided, allowing users to avoid specifying additional information each time. Under this approach, the system learns from historical data in order to automatically determine the order in which feasible options should be presented. Nevertheless, user preferences remain important for two main reasons. First, they are particularly useful during the initial phase of system deployment, when limited historical data are available. Second, they are also relevant when users seek routes that differ from their typical behavior. For example, a senior citizen may generally avoid using e-scooters due to safety concerns. However, in a specific situation, the same user may decide to use an e-scooter if necessary in order to reach their destination on time.

Provided that an adequate amount of historical data is available, the recommendation engine can automatically rank feasible routes by taking into account each user's profile and their past decisions. To support this process, each time a user interacts with the system, we record both the recommendations generated by the system and the route selected by the user, along with additional parameters collected upon completion of the trip. This information can then be used to model user behaviour. The developed ML model is used by the recommendation engine to rank the eligible routes identified in the previous step. The input to the ML model consists of the user profile and a subset of the features computed for each route. The output is a value in the range [0, 1], indicating the suitability of each route. This value is subsequently used to order the routes, so that the most appropriate options appear at the top of the list. The development of the ML model is almost completely automated so that it can be periodically updated, if necessary, to capture changes in user behaviour.

For the development of the ML model, a dataset was constructed containing information from five different sources: (a) temporal data regarding the date and time the system was used, (b) user demographic characteristics specified during registration, (c) route features automatically computed by the system, (d) routes selected by users before starting their trip, and (e) additional route features automatically computed by the system upon trip completion. An additional column was included in the dataset to indicate whether each route was selected or not. The dataset was split into two subsets, namely training and validation sets, which serve different purposes. In this work, a relatively simple ML model was employed, namely a Random Forest classifier [37], due to the limited amount of available data and computational resources. Nevertheless, more advanced ML models can be incorporated without substantially modifying the overall process. The proposed model predicts not only the class label (i.e., whether a route is selected or not), but also the corresponding probability of selection. This probability is particularly useful, since higher values indicate more suitable routes, and it can therefore be used to rank the available alternatives.

3.4. LLM-enabled User Interface

User interaction with the system is supported through a user interface that enables authorized users to specify the origin and destination (mandatory), along with additional optional parameters when needed, and to view the suggested routes ranked in an appropriate order. Users can then select the most suitable option, reserve the corresponding vehicles, and release them upon disembarkation. The data provided by users through the web interface are expressed in JSON format so that they can be further processed by the system's REST⁹ services.

To facilitate user interaction with the system, a pre-trained LLM was also incorporated. More precisely, the system enables users to express their requests in natural language and automatically converts them into a valid JSON message that can be processed by the underlying components. To achieve this, we first developed a JSON schema¹⁰ that defines the structure of both mandatory and optional parameters, along with the possible values for each field (e.g., route features and constraints), as illustrated in Figure 3. Using prompt engineering, we then provided this schema to the LLM together with a set of instructions and a small number of examples [38], enabling the model

⁹ REST, <https://restfulapi.net/>

¹⁰ JSON schema, <https://json-schema.org/>

to automatically convert user input into a valid JSON message. Finally, the system can automatically determine the user's origin, provided that this information is available on the device (e.g., computer or mobile phone) on which the user interface is deployed.

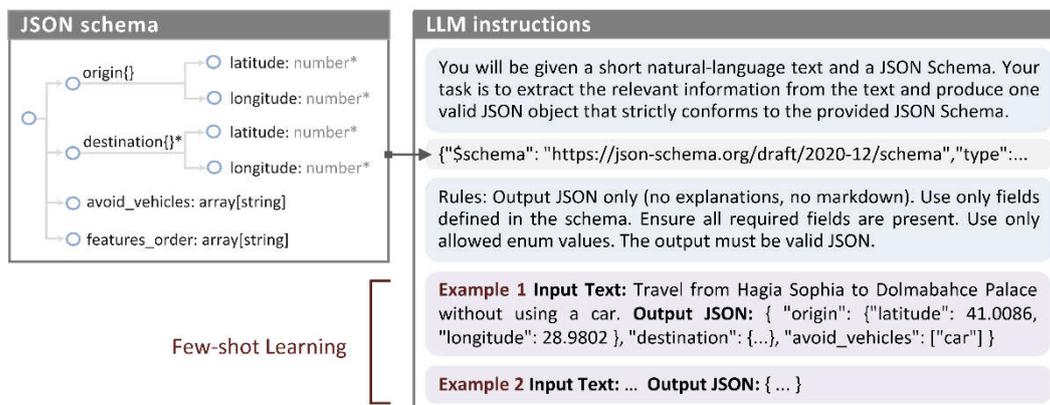


Figure 3. JSON Schema and LLM Instructions (Few-Shot Learning).

As can be observed in the above figure, the JSON schema used is slightly different from the one required for expressing user requests. First, we excluded the “user_id” parameter, since it can be automatically assigned by the system. In addition, we made the origin field optional, as it may not always be explicitly specified by the user. In such cases, the user's current location, which can be automatically detected, is used instead. Finally, user preferences were expressed in a more straightforward manner to ensure that they can be correctly interpreted by the LLM, even when using a model with limited size and capabilities, such as the Qwen3-4B LLM adopted in this work.

4. Example of Usage and Discussion

In this section, we focus on the multimodal transportation algorithm and illustrate its operation through an example. In this scenario, we assume that a set of vehicles is distributed across Istanbul and can be used to reach a given destination. More precisely, three cars are parked in two out of the three available parking areas, eight e-scooters are distributed around well-known Istanbul attractions (i.e., the Blue Mosque, Topkapi Palace, and Galata Tower), and two sea vessels are anchored at the port near the Galata Bridge (i.e., one out of the three predefined ports), as shown in Figure 4(a). We further assume that the user is located near Hagia Sophia (origin) and intends to visit the Istanbul Modern Art Museum (destination). In addition, we assume favourable weather conditions and no significant traffic congestion.

With respect to system configuration, the maximum walking distance per step is set to 1 km, while e-scooters and cars are limited to 4 km and 15 km, respectively. Sea vessels can travel between any of the predefined ports.

4.1. Route Planning Algorithm Demonstration

Based on the algorithm described in Section 3.2, we first identify the vehicles available at each stop. We then cluster the available e-scooters according to their locations on Google Maps and construct a directed graph consisting of the relevant entities and the links among them. These links correspond to straight-line distances that satisfy the system constraints and represent potentially feasible transitions. As a result, the graph contains 11 nodes, namely the origin and destination, the three representative scooters selected after the clustering process, the three car parking areas, and the three ports, as well as 31 edges, as shown in Figure 4(b). As illustrated in this figure, the user can travel from the origin (start location) to e-scooter 2 and parking area 1, but not to e-scooter 1, since it is too far away based on the maximum walking distance defined in the system configuration. In addition, the graph allows a transition from parking area 2 to parking area 1, but not in the opposite

direction, since no car is available in parking area 1 at that time. A similar constraint applies to transitions among ports. Finally, e-scooters are connected to a larger number of nodes, including more distant ones, since they can generally cover several kilometres, as defined by the maximum scooter distance in the system configuration.

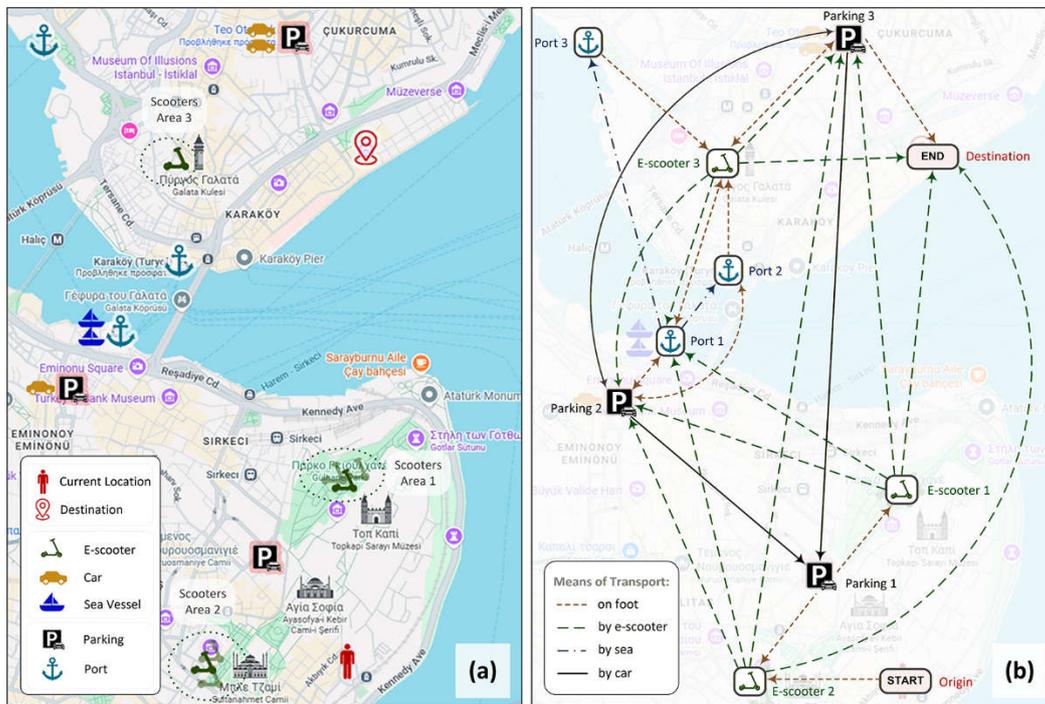


Figure 4. Map of a selected area in Istanbul showing (a) the current locations of the POIs and (b) the constructed multimodal transportation graph.

Based on the constructed graph, the system identified 54 potential paths that could be followed to reach the destination. However, not all of them are feasible. More precisely, the system quickly rejected most of them (in this example, 46 out of 54) based on the sequence of entities involved in each path. For instance, there is no need to walk from the initial position to parking area 1 and then continue walking towards e-scooter 2. Similarly, there is no need to travel by car from parking area 2 to parking area 3 and then use a car again to travel to parking area 1. For the remaining eight paths, OpenStreetMap was used to compute the actual routes and the corresponding travel distances required when moving between consecutive nodes using different transportation modes. This process further reduced the candidate paths to five, as shown in Table 1, and ensured that the suggested routes satisfy the specified constraints.

Table 1. Suggested paths along with a small subset of the computed features.

ID	Path	Total Distance (m)	Total Duration (s)
1	START → Scooter-2 → Stop (before reaching our destination) → END	5302.4	1737.3
2	START → Scooter-2 → Car in Parking-2 → Parking-1 → Scooter-1 → END	9631.0	2778.6
3	START → Scooter-2 → Car in Parking-2 → Parking-3 → Scooter-3 → END	10298.2	2949.4
4	START → Scooter-2 → Car in Parking-3 → Parking-1 → Scooter-1 → END	12592.1	3160.4
5	START → Scooter-2 → Sea Vessel in Port-1 → Port-2 → Scooter-3 → END	7047.6	2357.4

It should be noted that traveling from the current location of e-scooter 2 to the destination using the scooter for the entire route was not feasible, since the required distance exceeded the maximum distance that can be covered using an e-scooter. Therefore, the system suggested using the scooter for as long as possible and then continuing on foot (path no. 1), given that the destination was within walking distance from that point, as shown in Figure 5(a,b). Alternative options could also be considered (e.g., leaving the scooter earlier and continuing on foot), but they were deliberately ignored in order to reduce the complexity of the algorithm. As shown in Table 1, the fastest way to reach the destination is to use e-scooter 2 (path no. 1). This route is also illustrated in Figure 5(a,b) using thicker lines.

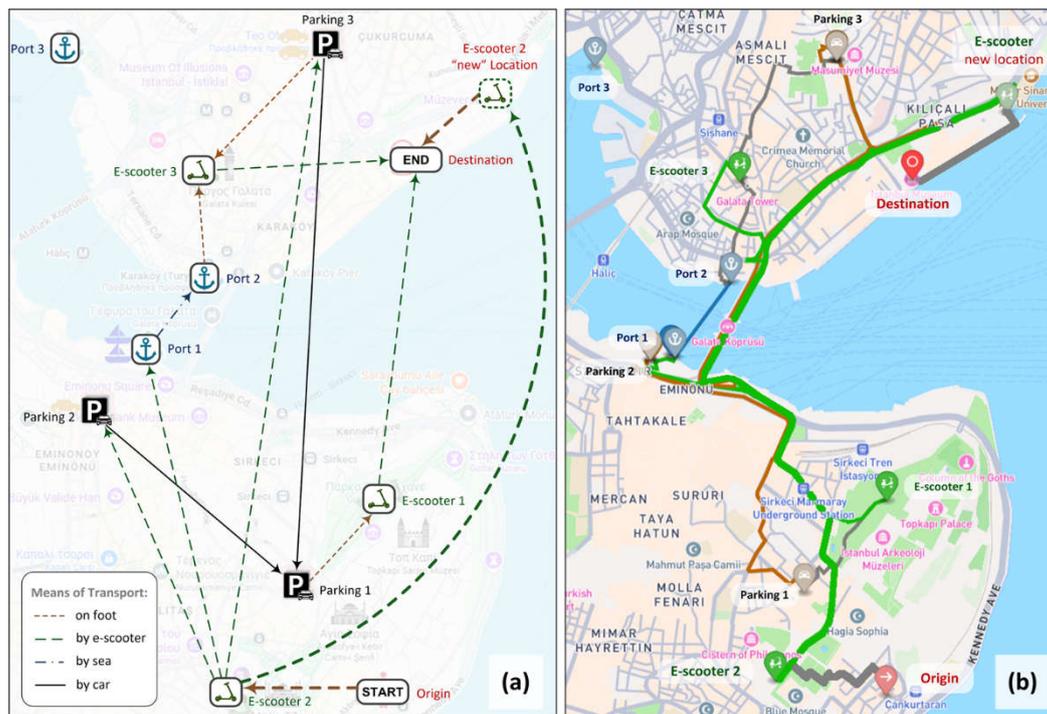


Figure 5. Graph demonstrating the five feasible options: (a) using links and (b) with suggested Google Maps routes. This figure prepared using geojson.io¹¹.

In the above example, we assumed good weather conditions and, therefore, all available means of transport could be used. In the presence of strong winds, sea vessels cannot be used; however, this does not affect the system's recommendations in the specific scenario considered, since the top-ranked route will remain the same. In the case of rainfall, e-scooters cannot be used (according to the assumptions adopted during the system design), and consequently no feasible path exists, since the remaining means of transport are located too far away. We also assumed normal traffic conditions; therefore, the suggested routes involving car usage are not affected. In the case of a traffic jam in a specific area (e.g., near the Galata Bridge), the system identifies the routes passing through this area and significantly increases their estimated travel duration. However, it should be noted that, in our example, we only examine the route returned by OpenStreetMap and whether it passes through the high-traffic area. Ideally, all possible alternative car routes between two locations should be considered. Nevertheless, this was deliberately omitted in this work, since our focus is on multimodal transportation rather than optimizing routing for a specific mode of transport.

4.2. Recommendations Based on User's Preferences

¹¹ geojson.io website, <https://geojson.io/>

Regarding the routes proposed by the system, it returns, by default, the quickest route among those identified in the previous step. However, as described in Section 3.3, the system can be instructed to operate differently either by explicitly specifying user preferences (as illustrated in the following example) or by automatically learning user behaviour from historical data. For instance, if a user (e.g., a child) is not willing or not allowed to use a car and prefers to walk as little as possible, the system suggests using e-scooter 2 to reach port 1, then taking a sea vessel to port 2, and finally using e-scooter 3 to reach the destination (i.e., path no. 5 in Table 1). In contrast, if a user (e.g., a senior citizen) wishes to avoid using e-scooters, the system cannot identify a feasible path, as previously mentioned.

It should be noted that some user-specified restrictions, namely those indicating that a particular mode of transport should be avoided, are passed to the route planning algorithm to reduce the number of entities introduced into the graph and, consequently, accelerate the overall process. In contrast, user-defined constraints that require the computation of additional features in order to support decision making (e.g., limiting e-scooter usage to at most two times), as well as the features used to rank the remaining routes, are handled by the recommendation engine. The use of the ML model for ranking suggested routes presumes the availability of historical data and is therefore addressed in Section 5.3, which examines system performance using simulated data.

4.3. JSON-Schema Driven Data Extraction

Users can submit the above requests to the system using natural language. More precisely, they can write one or more sentences expressing their goals, and the system can automatically identify the corresponding parameters and encode them according to the provided JSON schema so that they can be further processed by the recommendation engine. For example, when submitting the request “I would like to travel from Hagia Sophia to Galata Tower without using a car”, the system correctly identifies the locations (latitude, longitude) of both the origin and destination, along with the specified restriction, even when a few grammatical or lexical errors are deliberately introduced for testing purposes. Similarly, when submitting the request “I’d like to go to Topkapi Palace as quick as possible”, the system correctly identifies the location (latitude, longitude) of the destination and infers that “total duration” should be used to rank the suggested routes, even though this preference is not explicitly stated. However, in this case, the origin is not specified, as expected.

In our experiments, we observed that in some cases the detected locations were inaccurate (e.g., when requesting Istanbul Modern Art Museum). Moreover, in certain cases, additional information provided by the user could not be mapped to the properties defined in the JSON schema. For instance, when specifying that only a subset of transportation modes should be used (e.g., only a car or a sea vessel), the system did not correctly infer that the remaining modes should be avoided (e.g., e-scooters). The aforementioned problems are strongly associated with the limited size of the locally deployed pre-trained LLM. However, several of these issues still persist when a larger, online-accessible LLM is used. This clearly indicates that the output produced by the LLM should be presented to users in a user-friendly manner, allowing them to review and modify it, if necessary, before it is passed to the remaining components of the system.

5. Study System Behaviour Using Simulated Data

5.1. Simulated Data Generation

To test the behaviour of the system over time, we developed an additional software program that initializes the system with a configurable number of users and vehicles within a specific area of Istanbul. The program then generates multiple user requests over an extended period and collects the data produced by the system for further analysis. To create a realistic scenario, we randomly positioned the aforementioned entities around real locations that may be encountered in practice. For this purpose, we identified popular attractions in Istanbul, widely visited hotels, as well as several

parking areas and ports using ChatGPT¹² (v.5.2). This model has been trained on a massive amount of publicly available data and is therefore capable of detecting not only the appropriate points of our interest (POIs), but also their popularity and their locations on Google Maps (i.e., latitude and longitude). For completeness, the POIs used in our experiments are available in Appendix B.

Initially, the program generated a predefined number of users and randomly distributed them around the selected POIs. Male and female users were assumed to be almost equally represented in Istanbul. With respect to age, according to publicly available statistics¹³, approximately 20% of Istanbul citizens are children, 72% are adults, and 8% are senior (or elderly) citizens. Nevertheless, whether the generated data follow these proportions was configurable within the system. Furthermore, the program generated a predefined number of vehicles (i.e., cars, e-scooters, and sea vessels) and randomly distributed them within the selected area of Istanbul. More precisely, cars were randomly assigned to a predefined set of parking locations, and sea vessels were distributed across the predefined ports. For e-scooters, since they can be available almost anywhere, their initial locations were sampled around known POIs by adding random spatial noise. The generated locations were then filtered by retaining only those located close to road segments, using the OpenStreetMap API.

In our experiments, we tacitly assumed that all vehicles of the same type are identical. For simplicity, we deliberately ignored the battery and/or fuel level of vehicles. Moreover, we assumed that cars and sea vessels can be released only at specific locations (e.g., predefined stops such as parking areas and ports) without additional cost, in the sense that this cost is included in the rental fee. We also assumed that the cost of using each vehicle type is predefined (as part of the system configuration) and is independent of the travelled distance. With respect to environmental conditions, we assumed that the weather in the area of interest is favourable (i.e., a sunny and windless day) and that there is no significant traffic congestion. Nevertheless, these parameters can be easily modified when needed.

To test the behaviour of the system over a given period of time, the program periodically generated a few user requests at fixed intervals (i.e., once per minute) and simulated the corresponding transitions, as described by the algorithm presented in Figure 6. At each iteration, we randomly selected a subset of available users (i.e., users who were not already traveling) and initiated new trips. More precisely, for each selected user, we randomly assigned a destination among the available POIs and then used the system to obtain route recommendations for reaching that destination. To ensure that at least one vehicle would be required, destinations were selected sufficiently far away so that they could not be reached on foot. Subsequently, we selected the first route proposed by the system (if any), taking into account the custom preferences described below, and reserved the corresponding vehicles in order to perform the trip. In this work, we assumed that users followed the suggested routes using the reserved vehicles and released them immediately when they were no longer needed. Nevertheless, in real-world settings, this behaviour may not always occur. With respect to the recommended routes, we assumed that user choices were influenced by their demographic characteristics. More precisely, children were assumed to avoid using cars, whereas senior citizens were assumed to avoid using e-scooters. In all cases, users were assumed to prefer the fastest route among the available feasible alternatives.

¹² ChatGPT, <https://chatgpt.com/>

¹³ Istanbul City population, <https://www.citypopulation.de/en/turkey/istanbulcity/>

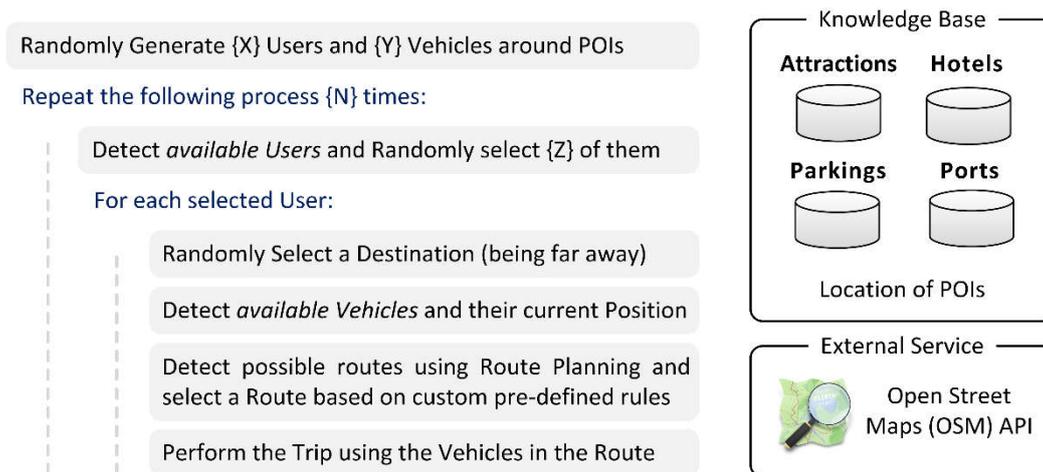


Figure 6. Algorithm for simulating data generation in a multimodal transportation system.

Each time a user initiates a trip based on a selected route, both the user and the corresponding vehicles become unavailable (i.e., reserved), so that they cannot be assigned to another user. The user becomes available again at the new location only after the trip is completed. In contrast, the reserved vehicles may become available earlier, since the user releases each of them during the trip when switching to another mode of transport. To simulate this behaviour, we computed the expected trip duration as well as the expected release time for each vehicle. These values were temporarily stored in two separate dictionaries, which maintain the list of users and vehicles to be released at the end of each minute, together with their updated locations. Therefore, at the end of each minute, we examined these dictionaries to identify the users and vehicles that should become available, updated their status accordingly, and replaced their current location with the new one.

5.2. Data Analysis and Discussion

Following the aforementioned process, we recorded the data produced over a specific period of time using a predefined number of users and vehicles. According to the conducted experiment and the selected configuration (duration: 3 hours, users: 80, and vehicles: 90), the database contains information about 169 trips performed by 57 distinct users within this time interval. Figure 7 presents the initial locations of both users and vehicles, as well as their locations at the end of the simulation. As can be observed, the number of users (and vehicles) shown on the right side of the figure is lower, since some users were still traveling when the simulation ended and were therefore unavailable at that time (the same applies to the corresponding vehicles). In addition, several users (approximately 25%) were not involved in the simulation due to imposed restrictions (e.g., senior citizens avoiding e-scooters), which prevented them from identifying an eligible route (e.g., because they were located too far from the remaining available vehicles), as reflected in the statistics reported in Table 2.

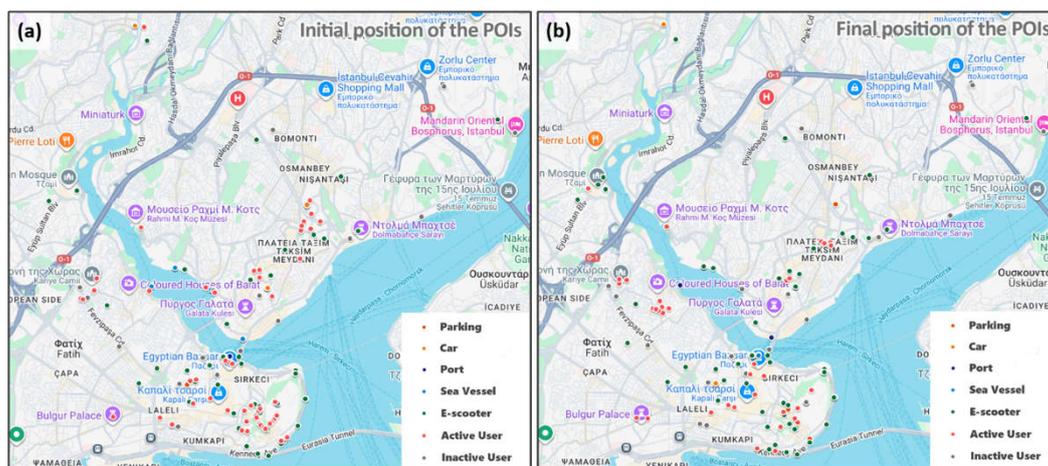


Figure 7. Locations of users and vehicles in the study area, (a) before and (b) after the simulation process was completed.

Table 2 provides summary statistics regarding the number of different types of entities at the beginning and at the end of the simulation process, further supporting the above observations. For instance, when the simulation ended, 25 users (57 – 32) were still traveling, and 27 vehicles (90 – 63) were reserved for these trips. Note that some vehicles used earlier during these trips had already been released during intermediate steps. For this reason, the number of unavailable users is slightly higher than the number of reserved vehicles, despite the fact that more than one means of transport may be used during their trip.

Table 2. Simulation Process Statistics.

Parameter	Before Simulation Started	After Simulation Completed
Number of Available Users	80	55
Number of Available “Active” Users	57	32
Number of Available “Inactive” Users	23	23
Number of Available Vehicles	90	63
Number of Available Cars	30	26
Number of Available E-scooters	50	29
Number of Available Sea Vessels	10	8

Further analysis of the collected data indicated that the system requires a considerable amount of time to identify an appropriate route when at least one feasible route exists (mean: 2.09 seconds, standard deviation: 5.29 s). Nevertheless, it returns very quickly when no route is available (mean: 0.03 seconds, standard deviation: 0.19 s). As a result, the overall average response time is below one second (mean: 0.58 seconds, standard deviation: 1.95 s). Considering the demographic characteristics of the users, we observed that the users who remained inactive were mostly senior citizens (i.e., 20 users out of 23), who were unable to find a car within walking distance, since senior citizens were assumed to avoid using e-scooters, and the available cars were parked far away. Regarding the vehicle usage patterns, in many cases, using only one transportation mode (in addition to walking) was sufficient. However, the system can also propose routes that combine multiple modes of transport, such as using an e-scooter followed by a car or a sea vessel, or vice versa, as shown in Table 3.

Table 3. Top ten vehicle combinations detected and their frequency of usage.

ID	Pattern	Count
1	FOOT-ESCOOTER-FOOT	44

2	FOOT-CAR-FOOT	36
3	FOOT-ESCOOTER	17
4	FOOT-ESCOOTER-SEAVESSEL-FOOT-ESCOOTER	14
5	FOOT-ESCOOTER-CAR-FOOT	10
6	FOOT-CAR-FOOT-SEAVESSEL-FOOT	9
7	FOOT-ESCOOTER-SEAVESSEL-FOOT-ESCOOTER-FOOT	7
8	FOOT-ESCOOTER-CAR-FOOT-ESCOOTER	7
9	FOOT-CAR-FOOT-ESCOOTER-FOOT	6
10	FOOT-CAR-FOOT-ESCOOTER	5

It should be noted that we repeated the above process multiple times under different configurations (i.e., varying the number of vehicles) in order to further evaluate system behaviour under different conditions. More precisely, we examined how the total execution time is affected by the number of available vehicles, including cars, e-scooters, and sea vessels. First, we repeated the simulation three times while increasing only the number of cars and sea vessels by 20 in total. We then repeated the simulation three additional times while increasing only the number of e-scooters by 20. The results are presented in Figure 8.

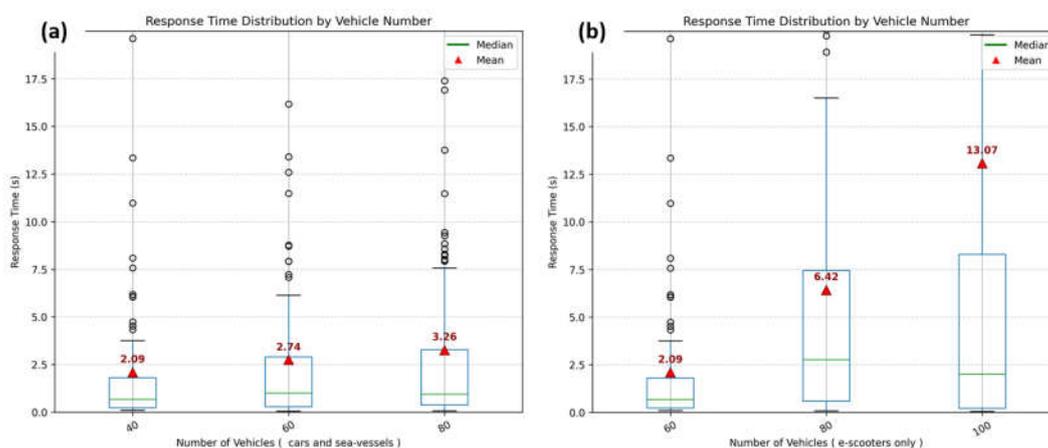


Figure 8. Response time distribution (when at least one feasible route is available) as the number of vehicles increases: (a) cars and sea vessels increase by 20 (total) per step, and (b) e-scooters increase by 20 per step.

As expected, increasing the number of cars and sea vessels does not significantly affect the system response time. This occurs because the complexity of the constructed graph does not increase, since the graph structure (i.e., its nodes) is driven by the predefined car parking areas and sea-vessel ports rather than by the number of vehicles located at these stops. On the other hand, having more vehicles at these locations increases the probability that at least one vehicle is available and, consequently, increases the number of feasible links introduced into the graph, which may slightly increase the number of candidate paths. In contrast, when the number of e-scooters increases, the response time increases significantly, approximately in a linear manner with a relatively steep slope. This is because e-scooters can be located almost anywhere, thus increasing the number of nodes and edges introduced into the graph and, consequently, the overall complexity of the algorithm. Nevertheless, this behaviour is strongly affected by the clustering process and, in particular, by the clustering radius used in DBSCAN, which is part of its configuration. The smaller the clustering radius, the larger the number of clusters detected, and therefore the larger the number of nodes and edges introduced into the graph.

5.3. ML Development and Evaluation

Following the previous process, we collected a considerable amount of data from multiple hypothetical use cases under different configurations. During these experiments, we tacitly assumed that users explicitly specify their preferences, which were then taken into account by the system when generating route recommendations, subject to the given constraints. However, in real-world settings, users will not typically provide such preferences. Therefore, the system should be able to learn user behaviour automatically. As described in Section 3.3, a ML model can be developed for this purpose and can be used to rank the routes identified by the system, so that the most suitable options appear at the top of the list. In this section, we present the evaluation of the developed ML model using the simulated data generated through the aforementioned process. The goal is to automatically rank the suggested routes in an appropriate order, without requiring users to explicitly specify their preferences.

The development of the ML model was based on a dataset generated from five different simulation runs, in which the number of registered vehicles varied. This dataset contains 6,315 records, each described by 21 features, including a label indicating whether the suggested route was selected by the user. It should be noted that the demographic characteristics of users involved in the simulation were almost uniformly distributed. Nevertheless, the dataset is imbalanced, since in each interaction only one out of (at most) ten suggested routes was selected. To address this issue, we applied oversampling by duplicating the records corresponding to the selected routes, which represent the minority class. Alternatively, more advanced techniques could be applied, such as SMOTE [39]. With respect to the input features, we examined their discriminative power using the mutual information score, as well as the correlations among them. This analysis indicated that the biological sex of users was not informative and could therefore be removed. The same applied to system configuration parameters. For the remaining features, we removed those representing the number of vehicles of each type included in a route, since they were strongly correlated with the distance covered using the corresponding transportation modes. As a result, the final dataset used for ML model development is balanced and contains 10,740 records, each represented by 12 numeric features.

This dataset was further split into three distinct subsets, namely training (70%), validation (10%), and testing (20%), which were used during the ML model development process. In our work, we employed a Random Forest classifier, which includes several hyperparameters, such as the number of estimators and the maximum depth of the generated trees that must be specified before training. To select appropriate values, we examined different hyperparameter combinations using grid search and chose the configuration that maximized accuracy on the validation set. We then evaluated the resulting model using the testing set. The results indicate that the model can accurately distinguish selected from non-selected routes, as shown in Figure 9.

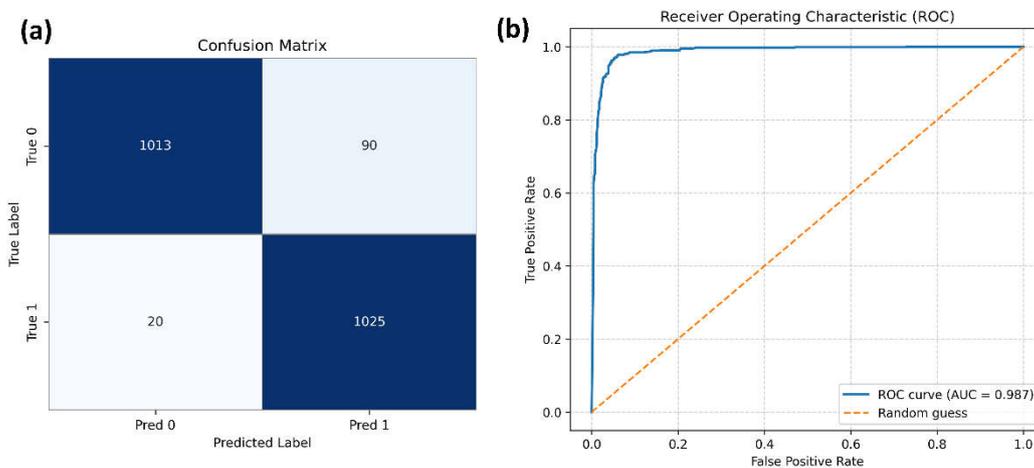


Figure 9. Evaluation of the developed ML model: (a) confusion matrix and (b) ROC curve.

Since the developed ML model was primarily designed to rank the suggested routes, we further examined the extent to which this objective can be achieved based on the predicted selection probabilities. For this purpose, we executed the simulation process again using slightly different configuration parameters and compared the recommendations produced by the system using the existing custom rules with those produced by the ML model. More precisely, we examined whether the top-ranked route in the two lists was the same, namely the route selected based on user preferences versus the route ranked highest by the ML model. When the two routes did not match, we recorded the position at which the ML model ranked the route that was actually selected by the user. This analysis indicated that a match occurred in almost 70% of the cases (i.e., 109 out of 157). For the remaining cases, further analysis of the ranking positions showed that the selected routes appeared within the first four positions, with a mean rank of 2.53 and a standard deviation of 1.76. These results indicate that the ML model can adequately serve its purpose in the absence of explicitly specified user preferences. Nevertheless, its performance can be further improved. For this purpose, the automatically generated suggestions should be evaluated collectively rather than individually.

6. Further Discussion and Future Directions

6.1. Strengths and Limitations of the System

The developed system enables users to identify the most appropriate routes to their destinations by combining multiple shared vehicles of different types. These vehicles may be station-based (e.g., sea vessels), dockless (e.g., e-scooters), or support a hybrid operational model, such as cars that can be parked either in predefined parking locations or in designated free-parking areas. The system further accounts for various parameters related to the vehicles and users, as well as weather conditions and traffic levels, so that it can propose the most appropriate routes for each situation. However, the proposed system has certain limitations regarding the supported vehicles and the underlying mechanisms, which should be addressed in future work.

In this work, we assumed that vehicles such as cars and e-scooters have sufficient battery or fuel capacity to complete a trip and, if necessary, can automatically return to the nearest charging station. As a result, users can complete a trip (i.e., cover the maximum specified distance) without requiring intermediate charging. Alternatively, charging-station availability could be explicitly incorporated into the route-planning algorithm to support longer trips [33]. In addition, we assumed single-user vehicle usage. However, certain vehicle types can accommodate multiple passengers (e.g., cars and small sea vessels), potentially reducing travel cost. This setting increases the complexity of route planning, especially for cars, since pick-up and drop-off locations are not fixed. Therefore, carpooling [34] and ride-sharing [40] approaches should be considered in future extensions. Private cars and public transport, which have been deliberately ignored in this work, should be taken into consideration in future extensions of the system. Nevertheless, given the adopted approach, which supports both station-based and dockless vehicles, their integration into the system is relatively straightforward.

The proposed system can provide personalized services to each user individually, rather than processing requests coming from multiple users (e.g., within a short time period) collectively. The latter approach could enable more informed decisions regarding potential traffic congestion, for example by avoiding the recommendation of cars to many independent users traveling to the same destination simultaneously [22]. Moreover, integrating the proposed system with existing infrastructure and/or private vehicles (e.g., via sensors or camera-based monitoring) could enhance its understanding of user demand and traffic conditions, thereby enabling improved, environmentally friendly routing decisions and increasing user satisfaction.

The suggestions provided by the developed routing system are based on the current status of the vehicles (e.g., availability and location), as well as prevailing weather and/or traffic conditions. Consequently, the system does not take into account nearby social events, such as concerts and sports games, which may significantly increase road traffic. Moreover, sudden changes in weather

conditions, which may also affect a trip (especially when “vulnerable” vehicles are used at a later stage of a trip), are ignored. The system could provide improved suggestions by forecasting these factors (e.g., predicting weather conditions for the next hour) and by considering relevant events that may directly affect them. To support this functionality, the integration of external services and registries would be necessary.

Regarding the underlying mechanism used to identify eligible routes, we observed that it requires considerable time. Accelerating this process is feasible by maintaining the graph in memory, reducing the number of edges (and hence paths) introduced during the initial steps of route planning, deploying and using external services locally, and executing independent time-consuming processes in parallel (e.g., route detection using OpenStreetMap). Furthermore, e-scooters could be organized into predefined regions (as in Hydra [26]) to keep the number of clusters stable. Alternatively, e-scooters could be initially grouped into coarse clusters (e.g., using a larger DBSCAN radius or k-means) and then refined into smaller ones only when the corresponding coarse clusters are involved in feasible routes, enabling faster and more targeted vehicle selection. Response time can be further improved by precomputing potential paths between distinct areas of a city and frequently used means of transport (as in work [21]), thereby limiting the number of graph nodes and, consequently, the number of edges that must be considered when a new routing request arrives.

For ranking feasible routes, we developed a Random Forest classifier trained on historical data. The performance of the ML model could be further improved by employing DL techniques that directly optimize route ranking [41], rather than independently predicting route eligibility for each user. In this work, the model was trained using data collected from the entire user population rather than from individual users. This approach enables the system to provide meaningful recommendations even for new users who have just entered the system, provided that their behaviour is similar to that of other users within the same group. Alternatively, training a separate ML model for each user could also be considered, assuming that sufficient user-specific data are available. In such cases, system recommendations could be derived by combining both approaches.

6.2. Sustainable Transportation and Urban Infrastructure

There is also a growing need for the development of environmentally sustainable transportation solutions [42]. In this work, we focused on vehicles developed within the EcoMobility project, which are environmentally sustainable by design, as they rely on electric power rather than fossil fuels. Consequently, we did not incorporate additional sustainability-related factors such as energy consumption, carbon dioxide emissions, and noise pollution. These factors should be taken into account, particularly when conventional vehicles are integrated into the system (e.g., public transportation and private vehicles), since most of them still depend on fuel-powered engines.

Road traffic congestion is a major problem in urban areas and is often exacerbated by the limited availability of free parking spaces. This can increase trip duration and cost, as well as energy consumption and carbon dioxide emissions [43]. In our work, we adopted a simplified approach for modelling road traffic by primarily focusing on roads experiencing high congestion (i.e., traffic jams). Although this assumption enables efficient route computation and supports basic traffic-aware recommendations, it does not capture more fine-grained traffic dynamics, such as time-dependent variations, traffic flow patterns, or congestion propagation across the road network. Therefore, in future work, more detailed and realistic traffic models [44] should be incorporated to further improve the accuracy and robustness of the proposed route-planning mechanism. Regarding parking, we focused on paid parking areas where users are more likely to find an available parking space at a predictable cost. Alternatively, free parking areas could also be considered, while accounting for the probability of finding an available parking spot and the uncertainty regarding its exact location.

Multimodal transportation is also relevant to Park and Ride (P&R) systems. A P&R system [45] provides a set of facilities distributed throughout the urban environment of a city in order to establish connection points, or modal interchanges, between private vehicle users, such as cars, and more sustainable modes of transport, such as public transport. These systems are particularly useful for

users living in areas outside the urban perimeter, where there are few or no direct connections to public transport, who travel daily to a central business district with high traffic and limited parking availability. The facilities included in a P&R system are important, and several parameters must be taken into consideration (e.g., their location and cost) in order for the system to succeed. On the other hand, the complexity of the corresponding recommendation systems is relatively limited, compared to general multimodal transportation systems, since they focus on specific transport modes for which transitions take place at predefined locations (e.g., metro or train stations).

The infrastructures and services needed for micro-vehicles are also important [32]. In the case of private micro-vehicles, they usually need to be stored at both ends of a trip, and therefore the availability of parking areas near public transport stations and other points of interest (e.g., shopping centers) is essential. In the case of shared micro-vehicles, their availability at the aforementioned locations, along with designated areas to temporarily park them, is particularly important. Furthermore, the existence of a safe, comfortable, and continuous network of dedicated and protected lanes for micro-vehicles is critical for the successful use of both private and shared micro-vehicles by city residents for their daily transportation needs. For electric vehicles, charging station locations are essential for enabling longer trips. Additional factors should also be considered, including the technologies used by the relevant software platforms (e.g., real-time data), the condition and status of micro-vehicles, and the pricing scheme.

7. Conclusion

In this work, we presented a novel system that enables authorized users to identify the most appropriate route for traveling from an origin (e.g., the current location) to a destination (e.g., a point of interest) using multiple transportation modes. The system takes into account the current status of shared vehicles, user demographic characteristics, environmental conditions, and relevant events such as traffic congestion. The developed system leverages existing map services to identify feasible routes at a given time and context and subsequently ranks them according to user preferences, which can either be explicitly specified or inferred from historical data. The evaluation of the proposed system indicated that it can adequately serve its purpose by enabling users to combine different transportation modes based on their needs. In particular, the developed system can support both station-based and dockless shared vehicles, enabling users to identify routes that best suit their needs, which may be specified either explicitly or implicitly (based on historical data), while taking into account environmental conditions and traffic levels. The underlying mechanisms could be further improved, for example by adopting more efficient hierarchical data processing in order to reduce execution time, and by incorporating more advanced DNN architectures to improve the accuracy of the suggested routes.

In this work, we focused on the vehicles being developed by the partners directly involved in the EcoMobility project, namely cars, e-scooters, and sea vessels, which can be shared among members of a community. The proposed approach and the developed system can be extended to support additional transportation modes, including private and public vehicles, with minor modifications. Our work mainly focuses on identifying the most appropriate combination of transport modalities rather than on precisely estimating trip start and end times or designing the billing mechanisms required for such services. These aspects are particularly important when shared vehicles are integrated with public transport (e.g., buses and trains), which typically operate according to fixed timetables. With respect to payment, additional systems would be required to support a unified and user-friendly billing process that covers all involved transportation services, including those operated by external providers (e.g., public transport), which may follow different pricing policies and payment infrastructures.

Author Contributions: The work presented in this document was carried out by Dr. Efthymios Chondrogiannis, who conducted the relevant study, implemented the core components of the system (route planning and recommendation engine), described the approach followed and background mechanism being used and

performed the experimental evaluation. Leonidas Avdelas developed the data management module as well as the web application used for user interaction. The manuscript was reviewed by Prof. Antonis Litke and Prof. Theodora Varvarigou. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the European Commission under the Horizon Europe project EcoMobility, grant number 101112306.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A: Data Management

The data management component ensures the persistent storage of the data generated and collected throughout the two main phases of the system. During system initialization, it enables system administrators to register new vehicles, such as e-scooters, cars, and sea vessels, along with additional information related to them, such as car parking locations and sea-vessel ports. It also supports the registration of regular users, including their demographic characteristics. During the operational phase, when citizens of a smart city use the system for their daily transportation needs, the system enables different entities (e.g., sensors and regular users) to update vehicle status information (e.g., current location and availability), environmental conditions (e.g., weather conditions and high-traffic areas), as well as other relevant data generated during system usage. In particular, the system provides the necessary functionality to record the recommendations generated by the system, including the computed features for each alternative, as well as the routes ultimately selected by users. This layer also enables registered users to reserve or release a vehicle upon embarkation or disembarkation, respectively.

The data structures used are presented in Figure A1. The system records information about users, vehicles, stops (e.g., car parking areas), environmental conditions (e.g., weather and road traffic), as well as data generated through user interaction with the system (e.g., recommendations produced, routes selected, and actions performed). In our work, only a limited number of parameters were recorded for each of the aforementioned entities. With respect to user demographic characteristics, we recorded biological sex and the corresponding age group. For the latter, we adopted a simple and coarse classification scheme that groups users into three categories, namely children, adults, and senior citizens, which was sufficient for the needs of our experiments. We also followed a simplified approach for modelling weather conditions and road traffic. More precisely, we recorded whether it was raining or not, and whether strong winds were present at a given time and location. For road traffic, we recorded the locations of traffic jams, assuming that routes passing through these areas (i.e., areas around the corresponding points) would experience significant delays.

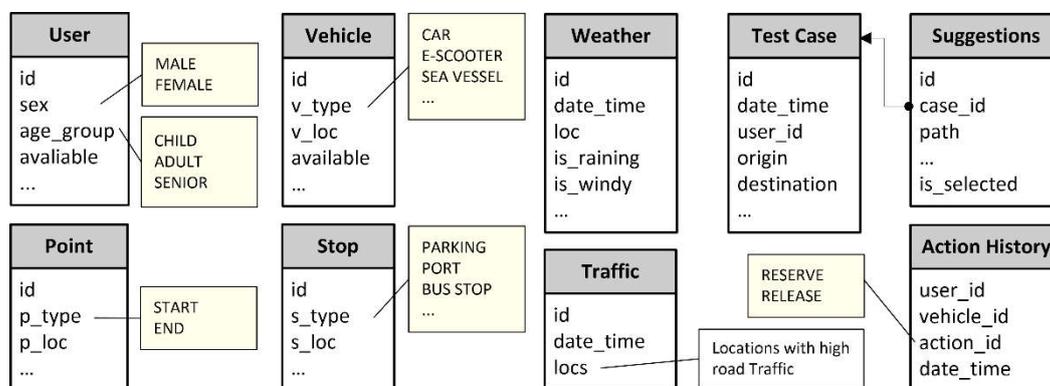


Figure A1. Data structures of the main entities of the system.

Appendix B: Points of Interest and System Libraries

For our experiments, we generated simulated data for testing purposes by considering highly visited attractions in Istanbul, widely known hotels, as well as several car parking areas and sea-

vessel ports. Providing an exhaustive list of these locations is beyond the scope of this document. Nevertheless, we provide a visual representation of the selected points of interest (POIs) in Figure A2 so that readers can understand the covered area, since users and vehicles were randomly distributed around these points.



Figure A2. Points of interest used in the simulation process.

The system was developed in Python using several libraries, including OpenRouteService¹⁴ (ORS) for integration with the OpenStreetMap API, SQLAlchemy¹⁵ for data storage in an SQLite¹⁶ database, and geojson for representing data in GeoJSON¹⁷ format, which is supported by various publicly available visualization tools. For the development of the ML model, we used the scikit-learn library, which provides implementations of multiple ML algorithms, along with NumPy, pandas, and matplotlib, among others. In addition, the system functionality was exposed to relevant entities through REST services implemented using the FastAPI¹⁸ library.

References

1. Liao, F., & Correia, G. (2022). Electric carsharing and micromobility: A literature review on their usage pattern, demand, and potential impacts. *International Journal of Sustainable Transportation*, 16(3), 269-286.
2. Vargas-Munoz, J. E., Srivastava, S., Tuia, D., & Falcao, A. X. (2020). OpenStreetMap: Challenges and opportunities in machine learning and remote sensing. *IEEE Geoscience and Remote Sensing Magazine*, 9(1), 184-199.
3. Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., ... & Qiu, Z. (2025). Qwen3 technical report. arXiv preprint arXiv:2505.09388.

¹⁴ Openrouteservice, <https://openrouteservice.org/>

¹⁵ SQLAlchemy, <https://www.sqlalchemy.org/>

¹⁶ SQLite, <https://sqlite.org/>

¹⁷ GeoJSON, <https://geojson.org/>

¹⁸ FastAPI, <https://fastapi.tiangolo.com/>

4. Ko, H., Lee, S., Park, Y., & Choi, A. (2022). A survey of recommendation systems: recommendation models, techniques, and application fields. *Electronics*, 11(1), 141.
5. Burke, R. (2007). Hybrid web recommender systems. *The adaptive web: methods and strategies of web personalization*, 377-408.
6. Adomavicius, G., & Tuzhilin, A. (2010). Context-aware recommender systems. In *Recommender systems handbook* (pp. 217-253). Boston, MA: Springer US.
7. Zhang, Q., Lu, J., & Jin, Y. (2021). Artificial intelligence in recommender systems. *Complex & intelligent systems*, 7(1), 439-457.
8. Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern recognition letters*, 31(8), 651-666.
9. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), 30-37.
10. McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D. J., & Barton, D. (2012). Big data. The management revolution. *Harvard Bus Rev*, 90(10), 61-67.
11. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
12. Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2). Cambridge: MIT press.
13. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, 521(7553), 436-444.
14. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, 52(1), 1-38.
15. Xu, J., Chen, Z., Yang, S., Li, J., Wang, W., Hu, X., ... & Ngai, E. (2025). A Survey on Multimodal Recommender Systems: Recent Advances and Future Directions. *arXiv preprint arXiv:2502.15711*.
16. Kim, S., Kang, H., Choi, S., Kim, D., Yang, M., & Park, C. (2024, August). Large language models meet collaborative filtering: An efficient all-round llm-based recommender system. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 1395-1406).
17. Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., ... & Wen, J. R. (2023). A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).
18. Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., ... & Werneck, R. F. (2016). Route planning in transportation networks. In *Algorithm engineering: Selected results and surveys* (pp. 19-80). Cham: Springer International Publishing.
19. Delling, D., Pajor, T., & Werneck, R. F. (2015). Round-based public transit routing. *Transportation Science*, 49(3), 591-604.
20. Koopmans, C., Groot, W., Warffemius, P., Annema, J. A., & Hoogendoorn-Lanser, S. (2013). Measuring generalised transport costs as an indicator of accessibility changes over time. *Transport Policy*, 29, 154-159.
21. Bast, H., Carlsson, E., Eigenwillig, A., Geisberger, R., Harrelson, C., Raychev, V., & Viger, F. (2010, September). Fast routing in very large public transportation networks using transfer patterns. In *European Symposium on Algorithms* (pp. 290-301). Berlin, Heidelberg: Springer Berlin Heidelberg.
22. Wu, F., Lyu, C., & Liu, Y. (2022). A personalized recommendation system for multi-modal transportation systems. *Multimodal transportation*, 1(2), 100016.
23. Duncan, L. C., Watling, D. P., Connors, R. D., Rasmussen, T. K., & Nielsen, O. A. (2020). Path Size Logit route choice models: Issues with current models, a new internally consistent approach, and parameter estimation on a large-scale network with GPS data. *Transportation Research Part B: Methodological*, 135, 1-40.
24. Nielsen, O. A., Eltved, M., Anderson, M. K., & Prato, C. G. (2021). Relevance of detailed transfer attributes in large-scale multimodal route choice models for metropolitan public transport passengers. *Transportation Research Part A: Policy and Practice*, 147, 76-92.
25. Marra, A. D., & Corman, F. (2025). Modelling route choice in public transport with deep learning. *Transportation*, 1-31.
26. Liu, H., Tong, Y., Zhang, P., Lu, X., Duan, J., & Xiong, H. (2019, July). Hydra: A personalized and context-aware multi-modal transportation recommendation system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2314-2324).

27. Farazi, N. P., Zou, B., Ahamed, T., & Barua, L. (2021). Deep reinforcement learning in transportation research: A review. *Transportation research interdisciplinary perspectives*, 11, 100425.
28. Goyal, P., & Ferrara, E. (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151, 78-94.
29. Rossiiev, O. D., Shapovalova, N. N., Rybalchenko, O. H., & Striuk, A. M. (2025). A comprehensive survey on reinforcement learning-based recommender systems: State-of-the-art, challenges, and future perspectives. In *CEUR Workshop Proceedings* (pp. 428-440).
30. Møller, T. H., Simlett, J., & Mugnier, E. (2020). *Micromobility: moving cities into a sustainable future. Safe Micromobility. Report by the International Transport Forum OECD/ITF.*
31. Grosshuesch, K. (2019). Solving the first mile/last mile problem: Electric scooters and dockless bicycles are positioned to provide relief to commuters struggling with a daily commute. *Wm. & Mary Envtl. L. & Pol'y Rev.*, 44, 847.
32. Oeschger, G., Carroll, P., & Caulfield, B. (2020). Micromobility and public transport integration: The current state of knowledge. *Transportation Research Part D: Transport and Environment*, 89, 102628.
33. Schneider, M., Stenger, A., & Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation science*, 48(4), 500-520.
34. Huang, H., Bucher, D., Kissling, J., Weibel, R., & Raubal, M. (2018). Multimodal route planning with public transport and carpooling. *IEEE Transactions on Intelligent Transportation Systems*, 20(9), 3513-3525.
35. Bongiovanni, L., Armengaud, E., Jover, S. F., Yalcin, I. K., Rossini, E., Mamei, M., ... & Litke, A. (2025, April). Smart City Pilots: Advancing Sustainable Mobility and Urban Innovation. In *2025 Smart Systems Integration Conference and Exhibition (SSI)* (pp. 1-6). IEEE.
36. Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems (TODS)*, 42(3), 1-21.
37. Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5-32.
38. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33, 1877-1901.
39. Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16, 321-357.
40. Agatz, N., Erera, A., Savelsbergh, M., & Wang, X. (2012). Optimization for dynamic ride-sharing: A review. *European Journal of Operational Research*, 223(2), 295-303.
41. Varun, B., Raghuvaran, K., Chandrakanth Rao, M., Hemanth Kumar, G., & Sanjay Ramdas, B. (2025). A Review of Machine Learning Ranking Systems: Methods, Applications, and Challenges. *J Contemp Edu Theo Artific Intel: JCETAI-119.*
42. Shah, K. J., Pan, S. Y., Lee, I., Kim, H., You, Z., Zheng, J. M., & Chiang, P. C. (2021). Green transportation for sustainability: Review of current barriers, strategies, and innovative technologies. *Journal of Cleaner Production*, 326, 129392.
43. Assemi, B., Baker, D., & Paz, A. (2020). Searching for on-street parking: An empirical investigation of the factors influencing cruise time. *Transport Policy*, 97, 186-196.
44. Alghamdi, T., Mostafi, S., Abdelkader, G., & Elgazzar, K. (2022). A comparative study on traffic modeling techniques for predicting and simulating traffic behavior. *Future Internet*, 14(10), 294.
45. Ortega, J., Tóth, J., & Péter, T. (2021). Planning a park and ride system: A literature review. *Future Transportation*, 1(1), 82-98.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.