software AG

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbeit
FACULTY OF
ELECTRICAL ENGINEERING AND
INFORMATION TECHNOLOGY

Master Thesis

Planning charge events for driving schedule of electric vehicles

A thesis submitted for the degree of

Master of Science (M.Sc.)

to Darmstadt University of Applied Sciences
Department of Electrical Engineering and Information Technology

February 28, 2022

submitted by

Rahul Roy JANNAPURA SHIVAKUMAR

Matr. Nr. 766526

First supervisor	:	Prof. Dr. Christian Weiner
Second supervisor	:	Prof. Dr.-Ing. Markus Haid
Industrial supervisor	:	Dipl. Math. Christian Winter

Declaration

I hereby declare that this thesis is a presentation of my original research work and that no other sources were used other than what is cited.

I furthermore declare that wherever contributions of others are involved, these contributions are indicated, clearly acknowledged and due reference is given to the author and source.

I also certify that all content without reference or citation contained in this thesis is original work.

I acknowledge that any misappropriation of the previous declarations can be considered a case of academic fraud.

Darmstadt, February 28, 2022

Rahul Roy JANNAPURA SHIVAKUMAR

Confidentiality

This master thesis contains confidential data and information of Software AG. Publications and/or reproductions are not permitted – even in extracts – without the prior express written consent of Software AG. This work may only be made available to the proofreaders, the examination board and authorized members of Darmstadt University of Applied Sciences.

Acknowledgements

I want to wholeheartedly thank my industrial supervisor Dipl. Math. Christian Winter, for the opportunity to conduct my Master Thesis at Software AG. His inspiration and insights helped me stay on the right course during this entire duration. Furthermore, I thank all my colleagues at Software AG for their ideas and the learning environment created at the workplace.

I sincerely thank Prof. Dr. Christian Weiner and Prof. Dr.-Ing. Markus Haid for being my supervisors at Hochschule Darmstadt and their constant support, without which the successful completion of this thesis was not possible. A special thanks to Prof. Dr. Christian Weiner for reviewing my work constantly.

I finally thank my family and friends who extended their moral support and help I needed during challenging moments.

Abstract

It is believed that electric vehicle (EV)s are the future of vehicular mobility. A well-known aspect of EVs is that they have an indispensable role in tackling climate change with advantageous levels of carbon emissions and pollution. Nevertheless, that future seems to be a long time coming since few facets of EV-based mobility pose challenges for larger market uptake. In particular, the concerns about their cost, limited range and poor coverage area of the recharging infrastructures discourage people from purchasing them. Hence, to tackle these aspects and aid EV customers with charging operations, there is a need for higher investments in EV technologies and better planning in building associated infrastructures. However, Information and Communication Technologies (ICT)-based solutions, in the form of web/mobile applications, also play a significant role in the transition to electric mobility.

Charging an EV must be aligned with its daily usage. Ideally, recharging happens while the vehicle is not in use, either at home or a remote destination. However, sometimes charge stops are also required during long trips. This thesis aims to demonstrate a prototype of the use case titled “Intelligent route and charging management” proposed by the research project “Daten Tanken” from its spectrum of application areas, see section 1.4. The vision of the use case is to support EV users to plan their journeys. In parallel, the data from the users, the EVs or the charging infrastructure accelerate interwoven smart-service missions of the project.

A platform/application is desired from a user’s perspective, taking relevant calendar entries as input and generating and displaying the ideal charging schedule and accompanying routing information. A route planner application is proposed to support EV mobility by predicting range and charging stops, ultimately generating travel itineraries. The application connects to the locally deployed charging station inventory (to retrieve the location information of charging stations) and to Google Maps services (for routing and geolocation information). The user calendar data and infrastructure data form the basis whereas, the planning algorithm regulates route and charging management. A simulation of distinct EV scenarios validates the efficacy of the proposed work based on the results.

Contents

List of Figures

List of Tables

Acronyms

2-D two-dimensional.

AC alternating current.

API Application Programming Interface.

BFS breadth-first search.

CPMS Charge Point Management System.

CPO charging point operator.

CRUD create, read, write, delete.

DC direct current.

EM electric mobility.

eMSP e-Mobility service provider.

EV electric vehicle.

EVSE Electric Vehicle Supply Equipment.

GPS Global Positioning System.

GUI Graphical User Interface.

HTTP Hypertext Transfer Protocol.

HTTPS Hypertext Transfer Protocol Secure.

ICE Internal Combustion Engine.

ICS Internet Calendaring and Scheduling Core Object Specification.

ICT Information and Communication Technologies.

IDE Integrated Development Environment.

IoE Internet of Energy.

ISO International Organization for Standardization.

JDK Java Development Kit.

JRE Java Run-time Environment.

JSON JavaScript Object Notation.

Li-ion Lithium-ion.

MongoDB Mongo Database.

NoSQL Not only Structured Query Language.

OCM Open Charge Map.

OCPI Open Charge Point Interface.

OCPP Open Charge Point Protocol.

POI Point of Interest.

SOA service-oriented architecture.

SoC state of charge.

URL Uniform Resource Locator.

XML Extensible Markup Language.

1 Introduction

1.1 Motivation

Over the recent years, electric mobility has gained commendable interest and investments, thanks to more intense competition between vehicle manufacturers and an increase in improved EV models in the market. Moreover, the critical aspect supporting the wide use of EVs is the deployment of charging infrastructure, which is gradually improving, thanks to public and private investments. However, the lack of intelligent services available for EV users leads to a consequentially poor acceptance rate, as described by the United States National Energy Technology Laboratory [1].

Some companions using EVs provided feedback describing insistent challenges during their vehicle usage. They anticipate clever electric mobility (EM)-specific solutions in the form of intelligent services. For instance, imagine a user charging his/her electric vehicle at a charging station and at the same time receiving valuable information about the condition of the battery or a plan of when and where one has to return to a charging station according to his/her appointments. This is precisely one of the primary motivations for this master thesis; to develop an intelligent solution on top of existing navigational systems that expect minimal user involvement and generate more user-oriented results in the form of calendar events that contain charge schedules and suitable driving plans.

As of now several initiatives worldwide have invested in upbringing applications that demand excessive manual data and generate similar results. The rationale of this thesis is to overcome the challenges above in the form of ICT-based applications that transform how EV customers interact with their vehicles and the components that power them. They are a vital source of information, control and motivation for EV customers.

1.2 Objective

Primary objectives intended to accomplish in this thesis are:

- Study of existing approaches proposed to assist EV users, such as showing information about charging opportunities along the travel path, route planner applications, EV charging and reservation mechanisms.
- Read and write calendar events that contain relevant schedule information.

- Investigate existing Application Programming Interface (API) services that provide charge point Point of Interest (POI)s and real-time navigational information.
- Setting up a local inventory containing charge point infrastructure details obeying existing EV mobility communication standards.
- High-level analysis of EV characteristics to estimate the charging profile of individual charge sessions.
- Evaluation of existing algorithms for traversing directed graphs in route planning services.
- Design of optimal route planner algorithm to compute minimal-distance-minimal-charge stop path.
- Finally, implement an optimal route planner application that generates travel itineraries as calendar events.

1.3 Structure of the Thesis

This report is structured as follows: Chapter 2 outlines the problem description, beginning with a couple of EV user feedbacks. In addition, this chapter details research questions gathered from the initial study. Chapter 3 describes the technical fundamentals of various technologies used to build the proposed application. Additionally, this chapter reviews related works on EV-related services and applications. Chapter 4 covers the design and architecture of the proposed system. Chapter 5 covers the implementation of the system and, in particular, an integrated route planner algorithm, allowing the complete planning of travel itineraries. Chapter 6 presents the evaluation of the route planner service on multiple simulated EV scenarios. Chapter 7 concludes the report and discusses future work.

1.4 Project Background

As in many metropolitan areas in Germany, the limit values for air pollutants have exceeded in the city of Dresden in recent years. To comply with this, the city of Dresden has been following the “Green City Plan Dresden” since 2018. It deals with measures for the digitization of transport systems and their networking with local public transport and the electrification of transport and urban logistics. The “Daten Tanken” project [6] funded by the Federal Ministry for Economic Affairs and Energy is a cornerstone of the Green City Plan Dresden.

The project aims to build a high-performance, grid-compatible, public and non-public charging infrastructure for electric vehicles in Dresden. In addition, the project should ascertain the economically feasible operation of the infrastructure through the conception, implementation and evaluation of suitable data-based services. The unique thing about it is to create added value by charging electric vehicles. For this reason, services are being developed that are particularly useful for users of electric cars and provide fleet operators, network operators and urban planning agencies a benefit based on charge-related data. The data can come from the EV users, the vehicles or the charging infrastructure.

2 Problem Description

The EV market uptake depends on customer satisfaction, which affects the customer's willingness to buy and promote an electric vehicle of choice [33]. Currently, specific parameters such as an EV's range, charging time, availability of charging infrastructure and cost of ownership seem inferior compared to Internal Combustion Engine (ICE) vehicles, particularly when these parameters are viewed separately. However, considering the bright side of EV mobility, there resides a great potential for improvements through intelligent solutions to meet customer expectations. The first part of the chapter describes some of the experiences of a couple of known users who own electric cars, followed by a discussion of the research questions that emerged while addressing the thesis' goals.

2.1 Specific User Experiences

Prof. Dr. Christian Weiner, Fachbereich EIT - Hochschule Darmstadt, often faces certain challenges when planning journeys in his Volkswagen e-Golf (typical range: 200 kilometres approx.). First, the dashboard application that comes with the vehicle offers minimal information regarding charging stations in the area/travel path. He adds that there is no advanced reservation option even when the application informs about available charging stations. Consequently, one is not aware of the status of the charging stations until they arrive at the location, and they are more likely to find either a non-functional charging station or another user already occupying it. As a result, an occupied charging station adds up to a considerable amount of waiting time. Additionally, Prof. Weiner states that the payment process took a long time after each recharge operation. He suggests that an intelligent/automated payment process could reduce added waiting times for successful payments, allowing the onward journey to resume.

Mr Vishwanath Balekai, a digital strategy expert at Merck KGaA, Darmstadt, owns a Tesla Model 3 and resides in Seeheim. He believes that electric mobility might constitute a viable vehicle solution only for urban mobility where adequate charging stations are available. Although Tesla cars are equipped with flagship features in their vehicles, Mr Balekai faces difficulties planning his long-haul holiday trips with his family due to the poor availability of charging stations along the travel path. Moreover, industrial and commercial activity in the Seeheim-Jugenheim region is close to nonexistent. As a result, it is hard to find a fair number of public charging stations for quick charging. Fuelling his electric car with a home charging setup would take an overnight's time. Besides, Mr Balekai prefers cost-free charge points found in the parking lot of nearby supermarkets for quick charging, sadly often occupied with regular customers' vehicles. He strongly demands clever route planner solutions with certainty in charging station reservations to plan more long-haul trips with ease.

2.2 Research Questions

The stated challenges and objectives reveal the need for a holistic and generic reference architecture to design the proposed route planner solution. Well, within the scope of the thesis, the most relevant actors are considered: EV, EV users, Electric Vehicle Supply Equipment (EVSE)s and external services. Furthermore, the proposed system design involves various embedded factors that define the optimality of the route planner solution. Some of the questions that emerged while deriving the thesis' goals were answered by existing literature that discusses EV-mobility solutions. The main research questions addressed by this thesis are as follows:

- What is the best approach to creating a generic architectural design that collaborates with various actors in the EV ecosystem, ensures the integration of proposed and new features, and facilitates building a system of interest for application?

  Route planning and calendar integration are among the proposed features, whereas the reservation of charge stations would be an appropriate and immediate addition. Chapter 7 describes possible prospective features for the application.

- What main factors influence the optimality of the routes calculated by the proposed route planner solution?

- How to find optimal routes algorithmically?

  Specifically, choosing the practical techniques to obtain a distance/time-optimal path determines a favourable route planner solution. In doing so, a few questions arose as follows:

- How can a trip include minimal charge stops possible?

- How to minimize travel distance under the need for visiting charging stations along the route?

- What techniques are available to deal with the route request, formulated as a graph traversal problem?

- What external services provide geolocation and real-time routing information for precise travel distance/time calculation?
- How to obtain the overall shortest time of travel itinerary?
- How to find the minimal combined time on the road and charging time?
- What factors affect the charging time of an EV battery? How to minimize the effect of adverse factors?

2.2.1 Research questions addressed by literature

This subsection outlines a few related research questions about battery charging that have been addressed in the literature as follows:

- What effects do intrinsic and external variables have on charging from an empty battery to a full battery?
- Can charging be considered a non-linear process?

Section 3.2 narrates relevant explanations to the above-outlined research questions.

3 Technical Fundamentals and Related Works

An overview of various technologies and concepts used to build the proposed application is presented in this chapter. Furthermore, this chapter discusses related works on electric vehicle mobility services and applications.

3.1 Communication Protocols

Smart charging at scale requires a uniform communication architecture to allow interactions between different system levels. Open Charge Point Interface (Open Charge Point Interface (OCPI)) and Open Charge Point Protocol (Open Charge Point Protocol (OCPP)) standards promote a uniform way of data exchange between different players in the EV market. The application scheme presented in this thesis utilizes communication standards offered by OCPI. On the other hand, OCPP allows interoperability among different charging equipment, software systems, and charging networks. However, the services of OCPP have not been utilized in the scope of this thesis since the proposed system addresses communication between the user application and the local database and not among actual stakeholders.

3.1.1 Open Charge Point Interface (OCPI)

Charging Point Operators (charging point operator (CPO)) operate and manage a network of charging points, whereas e-Mobility Service Providers (e-Mobility service provider (eMSP)) give EV customers access to charging services [19]. The OCPI supports connections between eMSPs and CPOs who manage charging points via Charge Point Management System (CPMS). "The Charge Point Management System (CPMS) is responsible for keeping track of the status of the charging points, communicating with external parties about their locations and charging sessions, and providing an interface for aggregators to control the charging speed of the charging stations" [18]. In addition, the 'Locations' module [22] offered by OCPI realizes the data structure of the charge points in the form of location objects that reside in the proposed CPO back-end system (local database server), see section 4.1.

3.2 EV charging profile

“In many designs and studies related to EV battery chargers, the EV battery loads are considered as a static load, and the realistic system behaviour of the batteries during the charging process has been ignored” [32]. Certain external variables such as State-of-Charge, type of current, environmental factors, degradation and so on influence the charging time of a battery. Mies et al. [36] analyse EVs’ charging profiles by studying public AC charging points with real-world data.

3.2.1 Charging phases

The pace of energy transfer during charging is dependent on the Voltage (V) and the current, expressed in Ampere (A). A lower current with the same Voltage decreases the energy transfer rate, resulting in a lower charging speed of batteries. The charging of a Lithium-ion (Li-ion) battery consists of three phases [36]. The first phase, called ‘pre-charge’, occurs when charging is initiated, during which the current is kept low while the Voltage steadily increases. Figure 3.1 [39] illustrates that most of the charging occurs in the subsequent two phases. The second phase, called the ‘constant current’ phase, starts when the State-of-Charge hits 10%. During this phase, the current is kept constant at a high level until the Voltage reaches a specified level. In the final phase, a constant Voltage is maintained, while the current is exponentially decaying, during which the battery is charged with trickle current, thereby reducing the speed of charging [32].

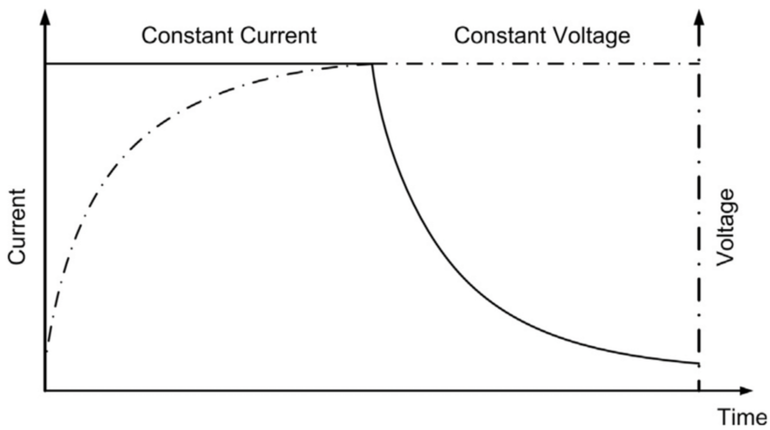


Figure 3.1: The charging profile of a Lithium-ion battery reprinted from [39]

3.2.2 Charging methods

Charging an EV involves supplying direct current (DC) to the battery assembly. As electricity distribution systems deliver alternating current (AC) power, a converter/on-board charger makes direct current (DC) power available to the batteries. In addition to current, Voltage also affects charging speed. Single-phase (230V) and three-phase (400V) charging methods fall under alternating current (AC) charging. Three-phase charging promotes charging speeds three times faster than single-phase charging. Helmus et al. [36] examine the difference between the two methods of energy transfer.

In addition, there is a physical limit to the size of the on-board charging device. As a result, the amount of power they can deliver to the battery is relatively low, so charging is typically slower. Besides, a DC fast charger bypasses the on-board charging device, supplying power directly to the EV's battery. The DC charger is external to the car, so it isn't constrained in size or cost, which means that charging is typically much faster.

3.2.3 Power ratings

Various connector types correspond with different charging requirements and vehicle compatibility that EV users should know. Type 1 and Type 2 connectors are the most commonly used AC sockets - typically for household charging. CHAdeMo and Type 2 CCS (Combined Charging System) are the most widely used DC fast charging connectors [17]. Table 3.1 shortly details power ratings corresponding to a few connector types.

Connector type	Type 1	Type 2	CHAdeMo	Type 2 CCS
Charging method	AC	AC	DC	AC / DC
Capacity	up to 7.4kW	up to 43kW	up to 50kW	up to 170kW
Voltage	230V	230V / 400V	500V	450V
Current rating	up to 32A	up to 63A	up to 125A	up to 125A

Table 3.1: Power ratings of charging methods

One of the leading electric vehicle manufacturers, KIA [17], details the above rating values based on their vehicle specifications. However, actual power consumption depends on the vehicle conditions and battery size/level.

3.3 Path finder conceptions

3.3.1 Haversine formula

The earth is spherical but enormous, and the circumference is roughly about 40,000 kilometers. Flat-earth formulas for calculating the distance between two points may seem sufficient for short distances but pose noticeable errors for more considerable geodesic distances [35]. One well-known trigonometric function, the haversine formula, accurately calculates the shortest distance between two points on the sphere's outer surface using the latitude and longitude of the points [34].

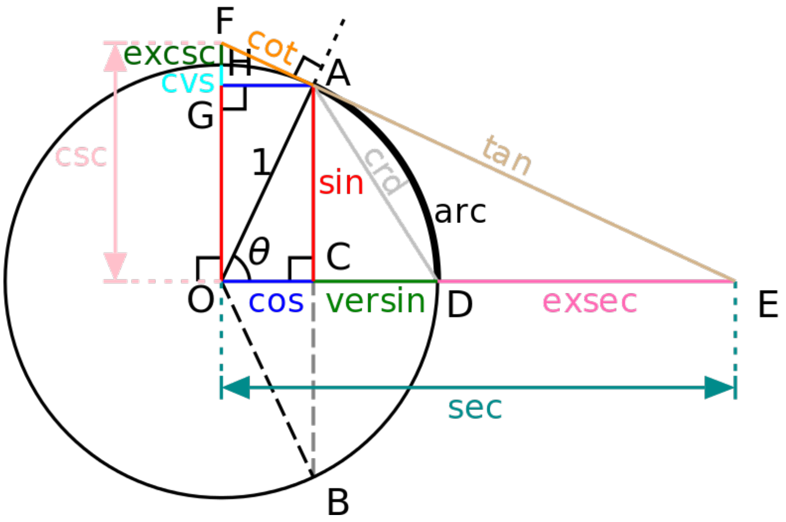


Figure 3.2: Illustration of the geometrical spherical triangle with trigonometric functions [34].

'Ha' in haversine stands for "half versed sine", where

$$\text{haversin}(\theta) = \frac{\text{versin}(\theta)}{2} = \sin^2\frac{\theta}{2}$$

The following equation computes the shortest distance d between points A and B with ϕ and λ representing latitude and longitude respectively, and R the earth's radius (mean radius = 6,378 kilometres). The angles are in radians.

$$a = \sin^2\left(\frac{\phi B - \phi A}{2}\right) + \cos\phi A * \cos\phi B * \sin^2\left(\frac{\lambda B - \lambda A}{2}\right)$$

$$d = 2 * R * \arcsin(\sqrt{a})$$

3.3.2 Coordinate geometry of a circle on the surface of a sphere

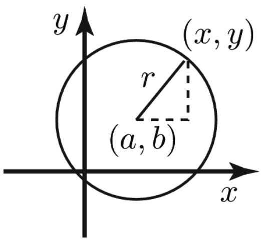


Figure 3.3: Representation of a circle in the xy-plane [5].

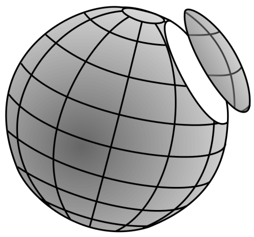


Figure 3.4: Representation of a circle on a sphere [24].

For a given circle in a two-dimensional (2-D) plane, as depicted in figure 3.3, the radius r is the straight line distance between centre (a, b) and point (x, y). However, by representing a circle on the outer surface of a sphere, as shown in figure 3.4, the distance calculation between two points uses the Haversine formula instead of the Pythagorean theorem [23]. This approach generates accurate results compared to straight-line results from the 2-D plane applied to spherical coordinates.

Let r be the radius of a circle on a sphere and d be the distance from the circle's centre to the given point, calculated using the Haversine formula. Three cases arise in determining whether a given point is inside, on or outside the circle: inside, for $d < r$; on, for $d = r$; outside, for $d > r$. Section 4.1 realizes this logic in drawing the reachability graph.

3.3.3 SSSP - single source shortest path algorithms

In graph traversing, SSSP algorithms solve the problem of finding the shortest path from a source node to all other nodes inside the graph. The nodes denote geographical location in the proposed application. Two main algorithms that fall under this definition are breadth-first search (BFS) and Dijkstra's algorithms. The application implements these two approaches in its route finder algorithm to obtain the optimal path. Said Sryheni [38] broadly elaborates the working mechanism of these algorithms.

Breadth-first search vs. Dijkstra's algorithm

The breadth-first search (BFS) approach deals with unweighted graphs to reduce the number of visited edges. This idea expands to weighted graphs, where all edges have the same cost. This way, the algorithm calculates potentially multiple shortest paths with an equal number of nodes. Figure 3.5 shows a directed graph with unit cost edges with node 0 as the source and node 5 as the destination. The BFS algorithm yields three shortest paths with exactly three edges on each path.

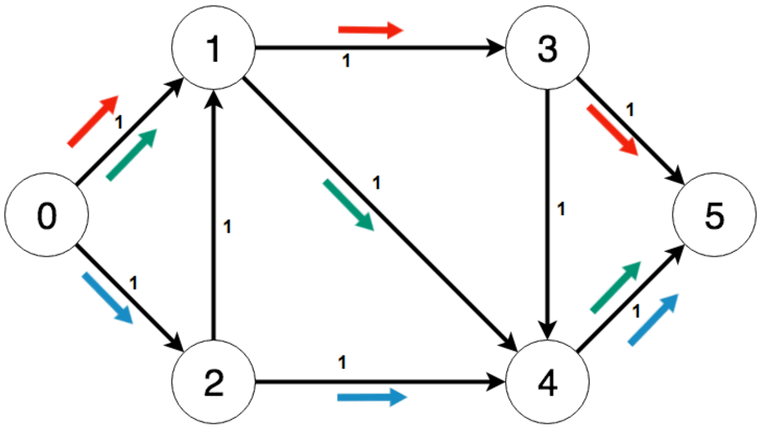


Figure 3.5: Directed graph of unit cost edges with possible shortest paths

Dijkstra's algorithm deals with weighted graphs with edges connecting to nodes having different weights/costs. The idea is to obtain the shortest path with the lowest cost. In contrast to the BFS approach, Dijkstra's algorithm disregards the total number of nodes visited to get to the destination. Figure 3.6 below shows a directed graph similar to the previous one, but with distinct positive costs on the edges. Dijkstra algorithm on this graph yields only one optimal shortest path with four edges (contrasting three edges using BFS), costing five units to traverse from the source node 0 to the destination node 5.

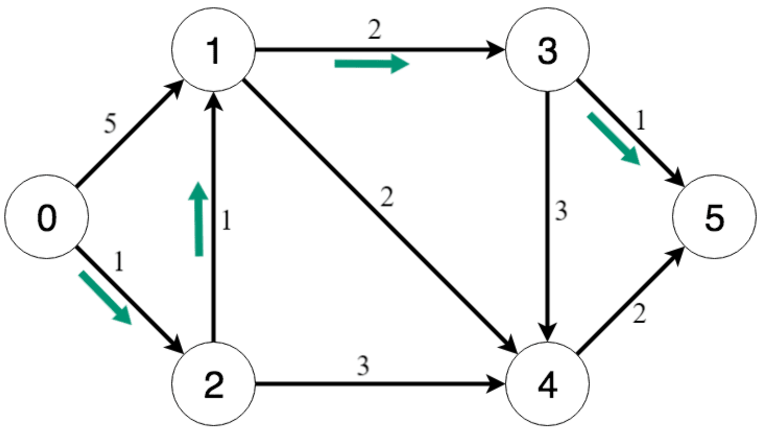


Figure 3.6: Directed graph of varying cost edges with optimal shortest path

3.4 Resource storage

3.4.1 JSON

JavaScript Object Notation (JSON) is a compact and readable text format for structuring data. JSON stands for JavaScript Object Notation and is a text format that is completely language independent. Moreover, the format is a more lightweight alternative to Extensible Markup Language (XML). The structural description is detailed in [25]. JSON files containing EV customer details and the vehicle features feed the application as input arguments.

3.4.2 MongoDB

Mongo Database (MongoDB) classifies itself as a Not only Structured Query Language (NoSQL) database program that stores and operates on flexible JSON-like documents, allowing for changes in the data structure over time. This database program ensures that the document model maps to the objects of the application code, making it easier to work with data [21]. The Java driver for MongoDB [20] connects the application and the database and accomplishes real-time create, read, write, delete (CRUD) operations.

3.5 External Resources

3.5.1 Open Charge Map

Open Charge Map is a non-commercial, non-profit electric vehicle data service hosted and supported by a community of businesses, charities, developers and interested parties worldwide. The API service from Open Charge Map fetches POIs, a set of charge point locations, in JSON format. The schema involves usage type, connection types, operator, and some necessary information helpful for application development [13].

3.5.2 Google Maps web services

Google Maps services offer reliable, accurate and up-to-date navigational information, and hence, has a vast majority of consumers. The Google Maps Platform provides a collection of APIs to integrate Google Maps services into web applications and mobile apps. Among several offerings, the proposed route planner application uses Directions and Distance Matrix API services (for routing directions and real-time traffic data) and services of the Geocoding API (for precise geographic coordinates). A licensed API key authorises every API request as part of authentication requirements. JSON formatted responses form the output of a valid request [10].

Geocoding API

An Hypertext Transfer Protocol Secure (HTTPS) interface accesses the Geocoding API to obtain geographic coordinates for given static addresses. Upon successful execution, the “results” element from the JSON response contains an array of geocoded address information and geometry information [8]. Section 5.2 illustrates the implementation of this service.

Distance Matrix API

The Distance Matrix API service provides travel distance and time for a matrix of origins and destinations. The API returns information regarding the recommended route between start and end points, as calculated by the Google Maps API, and consists of rows containing duration and distance values for each pair [15].

Directions API

The API returns information on the most efficient route for a given origin, destination and series of waypoints. Optimised travel time is crucial, but the API may also consider other factors such as distance, number of turns, and many more when deciding which route is the most efficient [9].

3.6 Development Environment and Tools

The working software of the thesis consists of a web application entirely developed in Java language, able to perform services mentioned in the proposed architecture and actively interact with the database. Eclipse as an Integrated Development Environment (IDE) with Java 8, Java Development Kit (JDK) and the compatible Java Run-time Environment (JRE) version form a development toolset. As a build tool, Apache Maven does the job of dependency management; essential libraries retrieved from public repositories aid efficient coding. Finally, MongoDB Community version driver for Java establishes the connection to the database and the application.

3.7 Related works

In response to the growing importance of smartphones and tablets in modern life, several web and mobile applications have been designed to assist EV drivers. With the purchase of an EV, the vehicle manufacturers tend to supply primary driver-assist applications, which they term vehicle enhancements. Usually, deployed applications behave like advanced dashboards of an EV where parameters affecting the performance of the electric motor (e.g. battery state of charge and health) are displayed.

The following subsections mention worldwide initiatives in facilitating charging station mapping services, route planning and charging operations.

3.7.1 Web mapping of charging stations

Charging point finders in the form of web applications display information about EVSE infrastructure existing in a given area (e.g. location, operator info, power profile, availability, etc.). Notably, Open Charge Map [14], GoingElectric [4], ChargeMap [3] allow finding charging stations in their embedded web maps. Open Charge Map and GoingElectric also offer APIs to interact with the charging station directory remotely [7, 13].

3.7.2 Route planner applications

This category includes several publications addressing route planning approaches [28, 29, 30]. In general, selecting the optimal path in EV-related scenarios is an instance of the shortest path problem with additional constraints. Bedogni et al. [29] details that the solution's optimality returned by a route planner refers to various domains. Such domains include spatial domain: the path with the shortest distance toward the destination; time domain: the path consuming less time, considering both travel and charging times; cost/energy domain: the path with minimal cost, including recharging operations, and energy consumption. Chale-Gongora et al. [30] particularly address internal and external parameters affecting EV energy consumption.

Dijkstra's algorithm [38] as referred earlier in section 3.3.3, solves single source shortest path problems for a graph with non-negative edges (negative edges occur in the cost/energy domain due to regenerative braking mechanisms, e.g. deceleration in downhill). However, the route planner proposed in this thesis doesn't consider negative edges between the source and the destination. Artmeiter et al. [26] aim to find the shortest path adapting Bellman-Ford algorithm [2] allowing negative edges.

Though all previous solutions may assist in finding optimal paths, in the case of battery depletion, the planned routes need to incorporate charging stops in between. Basmadjian et al. [27] focus on finding the optimal charging station concerning its proximity (in comparison to the planned trip) and corresponding availability in a given region.

3.7.3 EV charging and reservation applications

This section discusses some existing scientific contributions related to the charging of EVs and charging point reservation along the travel path.

Mies et al. [36] thoroughly analyse the realistic system behaviour of the batteries during the recharging process. Charging profiles of EV batteries depend on charging infrastructures, EV characteristics, and other external factors. The conclusive model Mies et al. [36] presents the influence of these variables on charging speed and total charging time. As mentioned in the previous subsection, along with finding optimal charging stations in a given region, Basmadjian et al. [27] also propose a generic reference architecture for reserving EV charging stations that consider relevant stakeholders involved in electric mobility.

Additionally, Orcioni et al. [37] present the advanced reservation mechanism as an extension to the OCPP standard deployed in a mobile application to assist EV drivers. Although the reservation feature doesn't fall within the scope of this thesis, the above literature [37] addresses battery monitoring, range prediction and smart charging mechanisms that are fairly realised in this thesis.

4 Service-Oriented Architecture and System Design

This chapter describes a generic structure of the system proposition in the form of service-oriented architecture (SOA) and the design modules necessary and valuable for its implementation.

4.1 Three-tier Software Architecture

The section discusses the Service-Oriented Architecture to support the design and development of the proposed application system presented in this thesis. The architecture illustrates the key requirements of the target scenario. First, it allows data collaboration among various actors of the EV ecosystem (e.g. CPOs as EVSE providers, EV customers). Second, it favours service-level expansion, i.e. the option to deploy new software services and to integrate them with existing ones through data sharing facilities. Finally, it supports the end users' device heterogeneity by separating back-end computation and data visualization facilities. In order to meet these requirements, a three-tier software architecture, as shown in figure 4.1, is proposed. Bedogni et al. [29] presents a similar approach in drawing out an architecture based on results of the Internet of Energy (IoE) European project.

Data Tier

A database plays the role of a knowledge base that stores the information produced within the EV ecosystem and that are required by the proposed system. MongoDB as a database manager enables document-oriented storage of information corresponding to JSON-based data structure representation since it is commonly used in EV charging protocols (i.e. OCPI). Furthermore, data interoperability ensures concepts and relationships among various actors of the EV ecosystem, such as EV customers concepts (e.g. schedules, registration data, EV model), EVSE concepts (e.g. locations, power profiles), and EV-EVSE relations (e.g. reservation actions). In real-world scenarios, each EVSE provider (i.e. CPO) is provided with its own data management platform, characterised by specific data representation formats [2]. Therefore, the database serves as a data collector and collaboration repository of the EV scenario.

Specific parsers (JSON parser as in the scope of this thesis) enable the data translation process from the JSON source format to the required Java object structural representation operated within system implementation.

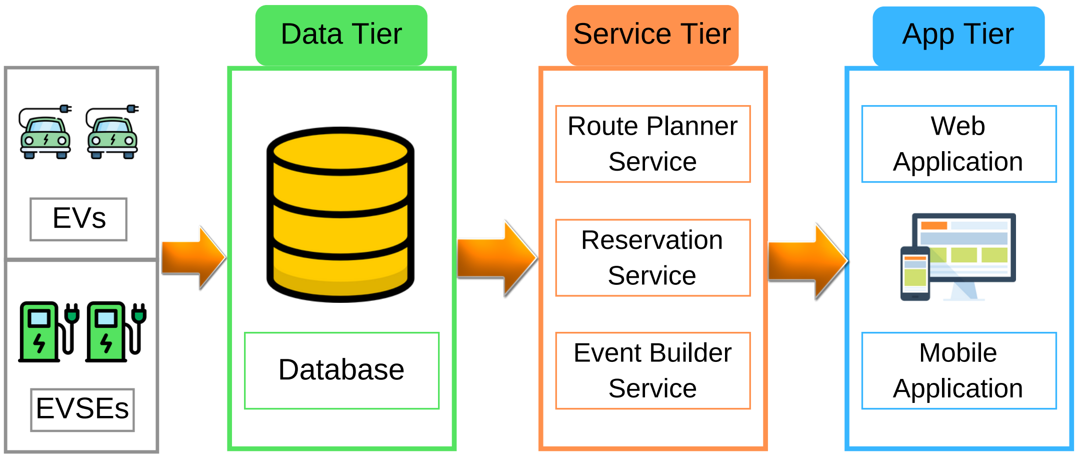


Figure 4.1: Service-oriented architecture of the proposed system

Service Tier

This tier provides utility services to EV customers. Each service accomplishes associated functionalities with a separate task to retrieve/insert/manage data contained in the database. Moreover, each service facilitates an interface for communication with the end-user applications. For example, route planning, calendar event building and reservation services offer required functionality in the project scope. Additionally, the architecture facilitates synchronisation and communication among different services with the local database. Finally, as mentioned before, the tier enables new services deployment and integration with existing ones in the future scope.

App Tier

This tier includes web and mobile applications for EV customers. However, the front-end application design is out of scope of this thesis; the architecture addresses the end-user's device heterogeneity by separating the back-end and the front-end functionalities. Each application performs remote requests to the available offerings from the service tier; well within the project scope is to fetch charging schedule and accompanying route planning information in the form of calendar events. As mentioned in section 3.7, several mobile travel assistance applications address similar approaches aiding electric mobility.

4.2 System Design

Based on the SOA illustrated above, this section describes a system of interest designed to build a route planning application for EVs. Figure 4.2 depicts various system agents, and the following paragraphs brief out their involvement. The system designed for route planning and associated services consists of a central system that runs thoroughly developed algorithms organised into distinct modules, the database storing EVSE data, the EVSEs where the recharge takes place, and the user who feeds calendar and vehicle information. Chapter 5 illustrates more detailed implementation aspects of the designed system of interest.

EV parameters

Each EV model is associated with a list of parameters (e.g. make, current/initial battery state of charge (SoC), range, connector type, etc.) fed to the central system for charge estimation operation integrated within the route planner service.

Event details

An Internet Calendaring and Scheduling Core Object Specification (ICS)/iCalendar format calendar file containing user schedules (event start & end time, location, etc.) feeds necessary parameters for the path and time calculation modules, which aid in preparing travel itineraries.

Open Charge Map API

As mentioned in section 3.5.1, the Open Charge Map API provides a set of charge point location objects that form EVSE data for the system. However, a secure request to the CPOs should yield EVSE data in real-world scenarios.

Google Maps API

The computation modules of the central system utilize Google Maps APIs services mentioned in section 3.5.2, retrieving information regarding path and geolocations.

Database

The database stores information about the EVSE infrastructure, and it is populated by querying external data sources such as Open Charge Map (OCM) data,

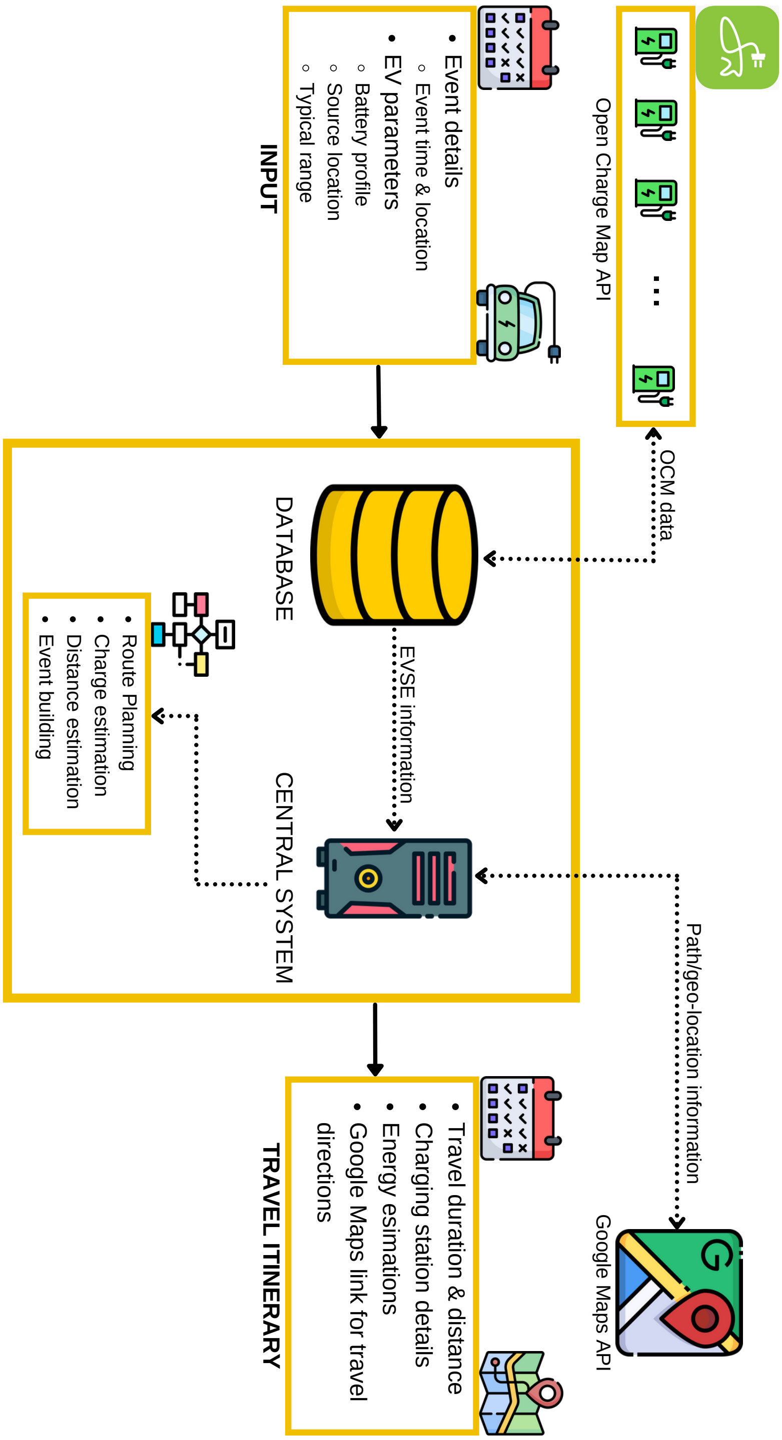


Figure 4.2: System of Interest

particularly in this thesis. The 'Locations' module offered by OCPI realizes the data structure of charge point location objects, and figure 4.3 depicts associated attributes. Table A.1 provides a detailed object description for the same.

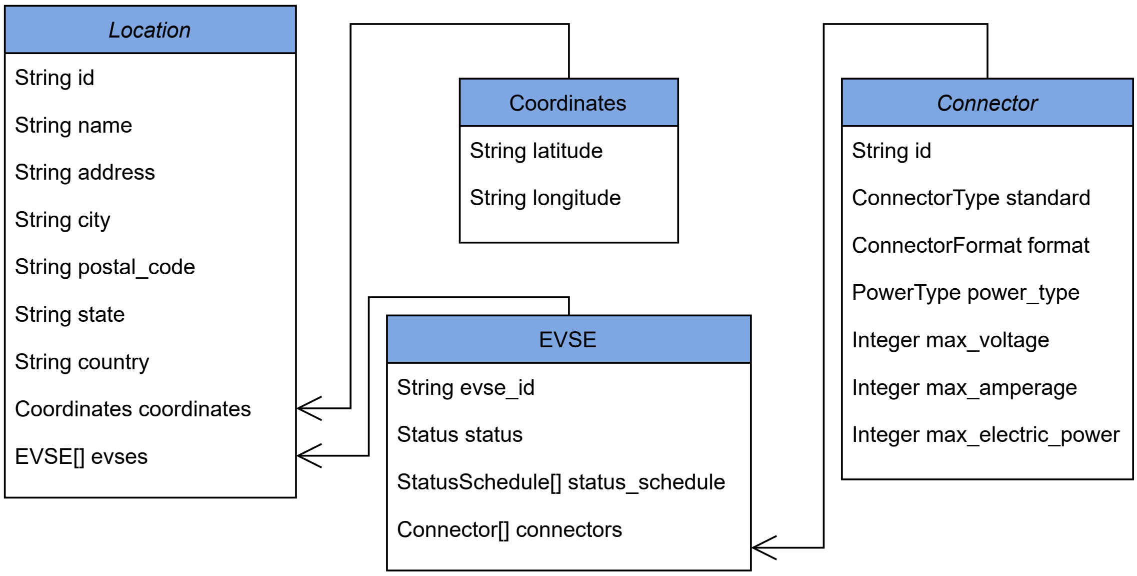


Figure 4.3: Location object attributes

Central system

The central system coordinates a set of algorithms and computation modules, thereby creating a travel itinerary in an iCalendar format calendar file. The system consists of a Java web application with inter-dependent libraries that handles multiple operations such as fetching OCM data from external resources into the database, optimal path and charge estimation, event building, etc. Moreover, the system is open to future developments with new features as independent services.

Travel itinerary

A calendar event summarises a travel itinerary with intermediate charge stop details and precise estimation of travelling times and charging times. A link to the Google Maps web application provides driving directions from source to destination with possible intermediate charge stops as waypoints.

5 Implementation

The implementation phase is carried out in a three-fold approach.

- Foremost, setting up the database with OCM data consisting of charge point objects. Also, creation of manual input parameters and calendar files.

- Second, the route selection is conceived as a shortest path problem, and algorithms are derived to compute the minimal-charge-stop-minimal-distance-travel path, also considering the recharging operations.

- Finally, design and implement a web application generating charging information and accompanying routing information, integrating real-time data from Google Maps services, EVSE data, range prediction and route discovery algorithms. Further, the creation of calendar events embedding relevant itinerary information.

5.1 Database Setup

5.1.1 MongoDB server setup with Compass GUI

This section explains setting up a local MongoDB server and the Graphical User Interface (GUI) on the Windows platform. The official website [11] provides installation instructions for different operating systems.

Step 1: Download the latest MongoDB Community version from the official MongoDB website.

Step 2: Once the executable is downloaded, run and follow the prompts to complete the installation process. Starting from version 4.0, MongoDB is configured as a Windows Service during installation.

Step 3: Upon successful installation, the MongoDB service should be up and running. Locate the MongoDB service from the Services console for its status.

Step 4: Download the latest Compass GUI for MongoDB installer file from the official website. Run the installer and follow the prompts. Once installed, the MongoDB Shell (mongosh) within the GUI aids in testing queries and operations directly with the database.

5.1.2 Retrieval of OCM data into database

The MongoDB database stores POI data from the OCM registry of charging locations. An API request with a registered API key as one of the Uniform Resource Locator (URL) parameters yields charging location information within a geographic boundary (country/region) in JSON format. A base URL is shown below and the associated query parameters is describes in table 5.1.

API Base URL [13]:

`https://api.openchargemap.io/v3/poi/<var>parameters</var>`

Parameter	**Datatype**	**Description**
countrycode	string	2-character International Organization for Standardization (ISO) country code to filter one specific country
maxresults	integer	limit on the maximum number of results returned
camelcase	boolean	set to `true` to get property names in camelCase format
key	string	a licensed API key for authentication

Table 5.1: Open Charge Map API query parameters

OCM - API documentation [13] describes detailed usage of this API and associated schema. Appendix A.2 documents a request/response sample of the OCM API call. The parsing logic aided by MongoDB driver for Java restructures the schema obtained from the JSON response to the new data structure collection obeying the OCPI 'Locations' module, as depicted in figure 4.3. Furthermore, it inserts charge point objects as documents in JSON format by omitting redundant attributes from OCM data and includes necessary data that are sufficient to the scope of the thesis. Appendix A.1 documents a restructured schema of charge point documents.

5.2 Input Parameters

When issuing a route request to the route planner, the system requires a few input parameters from the user. For instance, a user-defined calendar file with event details and a JSON file with EV parameters constitute input data. Firstly, an iCalendar file consists of a couple of attributes described in table 5.2.

Attribute	Description
$T_{\text{start}}^{\text{appointment}}$	event/appointment start time. The itinerary to be planned is supposed to have an end time before $T_{\text{start}}^{\text{appointment}}$. A flexible threshold time (t minutes) is subtracted to obtain $T_{\text{end}}^{\text{travel}}$, i.e. $T_{\text{end}}^{\text{travel}} = T_{\text{start}}^{\text{appointment}} - t$
d	the destination address. Google Maps Geocoding API query returns the corresponding coordinates of the given address string (refer to section 3.2).

Table 5.2: Input attributes from iCalendar file

The below API query returns a JSON response for a given address and an API key as query parameters described in table 5.3.

API Base URL [8]:

```
https://maps.googleapis.com/maps/api/geocode/json?<var>parameters</var>
```

Parameter	Datatype	Description
address	string	string containing the human-readable formatted address of the location, often equivalent to the postal address
key	string	licensed Google Maps API key

Table 5.3: Google Maps Geocoding API query parameters

Each EV model associates a few parameters used by the energy computation module integrated within the route planner. These are categorised as the user battery profile of the vehicle in use. A JSON input file consists of source coordinates and relevant vehicle parameters described in table 5.4.

Attribute	Description
s	source coordinates. It may either be the current location or one of the saved addresses (corresponding coordinates fetched from Geocoding API if required) of the user.
connectorType	the socket or plug standard of the charging point connector. This value helps in querying suitable charge point objects from the database.
typicalRange	maximum driving range of the vehicle in use, i.e. maximum range per full recharge
$\mathrm{SoC}_{\mathrm{ini}}$	the initial state of charge at the source expressed in battery capacity percentage
$\mathrm{SoC}_{\mathrm{dest}}$	the desired state of charge after arriving at the destination expressed in battery capacity percentage

Table 5.4: Input attributes from user defined JSON file

5.3 Integrated Route Planner Module

Based on the system design illustrated in section 4.2, the route planner algorithm consists of number of core modules executed sequentially, as shown in figure 5.1.

Each module's implementation steps are outlined in the following subsections. Besides, the dashed box in blue, enclosed with shaded processes, as shown in figure 5.1, indicates the possibility of integrating a reservation feature for charging slots. While this thesis does not address the reservation mechanism, it can be regarded as future work.

The reachability graph $G(V, E)$ is integral to the route planner algorithm. Beforehand, there is a possibility to avoid constructing the graph if the vehicle has sufficient energy to travel towards the destination without making a charge stop. The approach is to find an optimal shortest path denoted as $P^{s,d}_{\mathrm{optimal}}$, from the source to destination, with the least number of charge stops (saving charging time) and the path with a shorter distance (saving travel time).

5.3.1 Direct path computation

The direct path edge (e.g. blue dotted arrow as shown in the figure 5.2) from the source to the destination denotes that there is a possibility to travel without intermediate charge stops. The weight of the direct path edge is not the geodesic distance (refer to section 3.2), but actual road path distance computed by the Google Maps Distance Matrix API (applies to remaining edges depicted in the figure). However, this edge is built with the condition that $\mathrm{SoC}_{\mathrm{ini}}$ at the source is feasible enough to drive the EV successfully without any charge stops, while guaranteeing that the remaining charge at the destination is sufficient for the desired range at the destination. Besides, the route planner does not consider the SoC at the destination. Hence, $P^{s,d}_{\text{direct}}$, if feasible, will be the optimal path $P^{s,d}_{\text{optimal}}$.

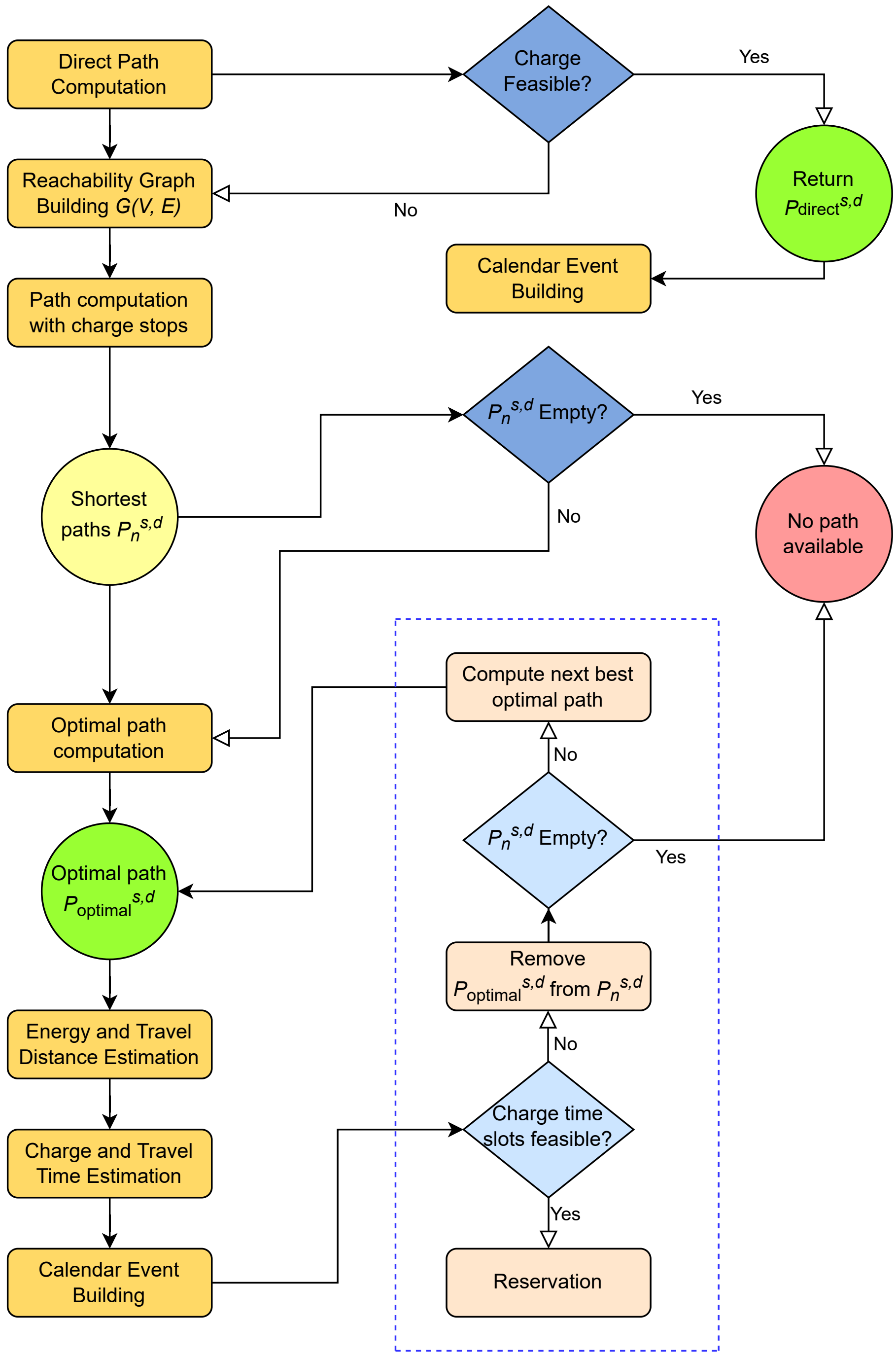


Figure 5.1: The route planner flowchart

5.3.2 Reachability graph building

The EV mobility problem is modelled as a set of vertices in a directed graph with a set of connected edges angling from the source towards the destination. Bedogni et al. [28] presents a similar approach in building a route planner application considering a deterministic energy consumption model based on physical equations. However, this thesis contemplates only a few primary attributes at its prototype level. Figure 5.2 shows a reachability graph model with source, destination and feasible charging stations connected with edges. The straight line edges depicted in the figure is an abstraction, and the actual model uses precise road distance instead of straight-linear geodesic distance.

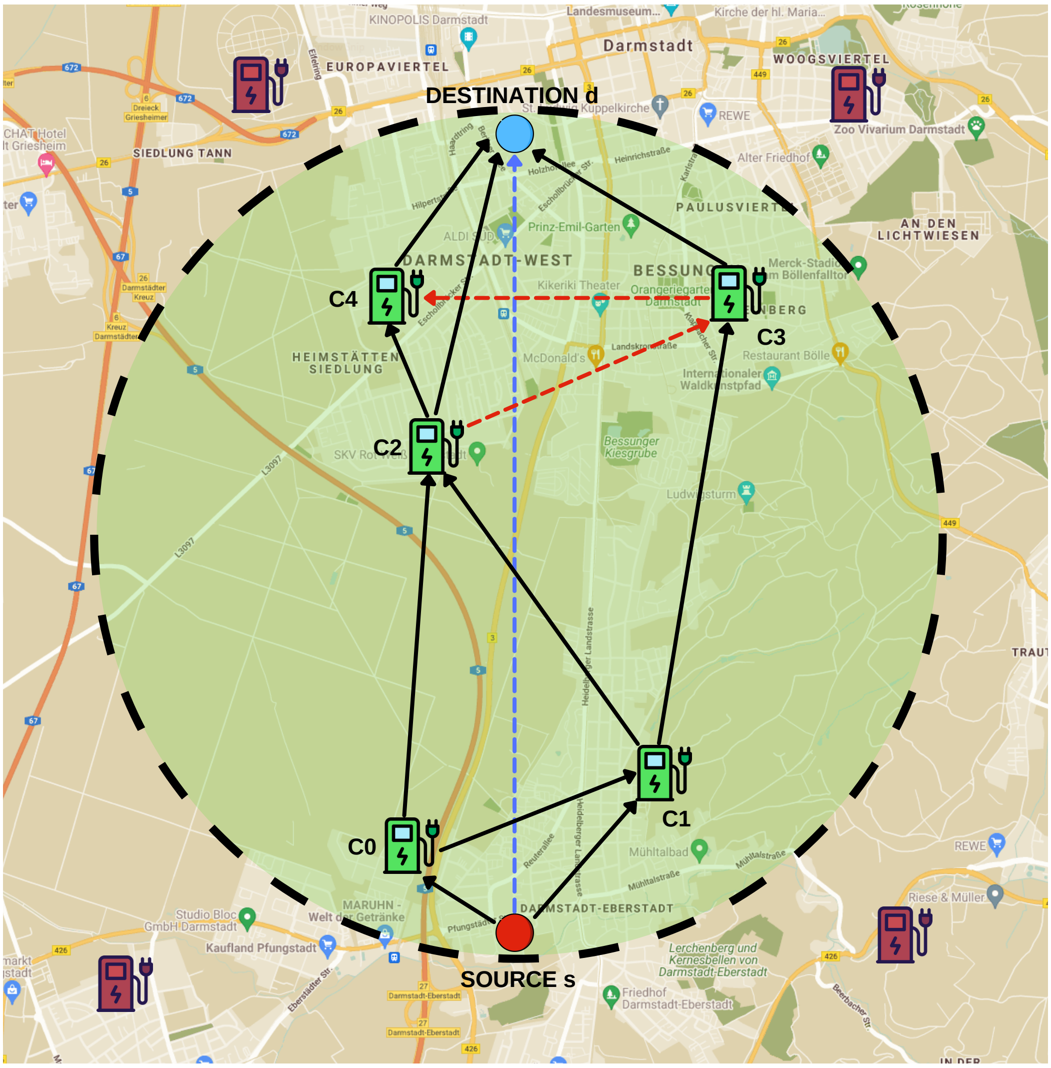


Figure 5.2: Reachability Graph $G(V, E)$

The following defines the construction of the Reachability Graph $G(V, E)$:

- The set of vertices $V = \{s\} \cup \{d\} \cup C$ includes the source (s) and the destination (d) vertices, and all the EVSEs (charge points) $C = \{c_0, c_1, \ldots c_m\}$ located in the circular area between the source and the destination as depicted in figure 5.2.

- The haversine distance (refer to section 3.3.1) $D_H(s, d)$ gives the precise diameter of the circular region depicted as a dashed circle in figure 5.2. Set C consists of EVSEs located within the circular region, i.e. the haversine distance from the centre of the circular region to each $c_i \in C$ is less than or equal to the radius of the circular region.

- Each vertex $v_i \in V$ has a set of adjacent vertices, and the connection is established through edges between each other after computing their associated weights. The weight factor also helps in avoiding redundant edges in the graph (explained in subsequent paragraphs).

- The weight of an edge $e(v_i, v_j)$ between two vertices, $v_i \in V$ and $v_j \in V$, denoted as $w(e(v_i, v_j))$, is defined by the real-time travel distance $D(v_i, v_j)$ returned by the Google Maps Distance Matrix API service (refer to section 3.5.2). The service returns the distance value of the best path out of available route options, often the shortest distant path distance.

- The weight computation for each edge translates to performing Hypertext Transfer Protocol (HTTP) requests to Google Maps API to obtain precise road travel distance value. This indicates that the number of HTTP requests affects the algorithm's time responsiveness. Moreover, requests are billed under the pay-as-you-go pricing model defined by Google Maps service. So, it is desired to restrict redundant API requests wherever feasible. Moreover, caching of API responses is adopted such that redundant HTTP requests to find the distance between the same vertex pairs are evaded for a given route request.

- Range computation after recharge at the charging station offers prerequisite data in building the reachability graph. Based on the charging behaviour of Li-ion batteries (refer to section 3.2), an optimal SoC target is fixed at 80% at each charging station except the final one . However, the target value varies based on battery health and specifications. This yields a travel range for a vehicle after each recharge.

- Besides, a couple of instances arise in computing the target at the possible final charging station as follows:
  - Depending on the level of charge required after arriving at the destination, the battery may recharge to a certain level. Therefore, it is not necessary to fulfil the 80% recharging requirement. This illustration saves overall travel time.
  - The battery may recharge more than 80% and up to its total capacity, thereby avoiding additional charging stops.

  Algorithm 1 shows the operation of the range predictor that computes vehicle range depending on the location of charge stations along the travel path.

Algorithm 1 Range prediction

```
function GETRANGE(tvr, dr, currentVertex, adjacentVertex, s, d)
    epr ← 80% of tvr
    ASSERT(currentVertex ≠ s)    ▷ Range at source is based on initial state of
charge; hence, range prediction is not done here
    ASSERT(currentVertex ≠ d)    ▷ Charging at destination is not performed
    if adjacentVertex = d then
        return tvr − dr
    else
        return epr
    end if
end function
```

- Bedogni et al. [28] presents building all possible edges between vertices beforehand and later pruning the redundant edges, adding additional steps. An improved approach is followed in building an edge set with the help of range predictor logic described in algorithm 1. The edge set is thus defined as $E = \{e(v_i, v_j) \mid v_i, v_j \in V\}$ portrayed in algorithm 2.
- The algorithm eventually constructs the reachability graph with optimal edges between vertices. The successful edges built are shown in black arrows connecting different vertices. Notably, the logic omits those edges for which $D(v_i, v_j) > D(v_i, d)$, which implies skipping the connection to adjacent vertices which are farther than the destination (e.g. red dotted arrows shown in figure 5.2).

Algorithm 2 Connected Vertices Graph

```
function createConnectedVerticesGraph(SoC_ini, SoC_dest, V, s, d)
    tvr ← typical vehicle range of an EV in kilometres
    ivr ← SoC_ini ∗ tvr                               ▷ initial vehicle range in kilometres
    dr ← SoC_dest ∗ tvr     ▷ desired vehicle range in kilometres at after arriving at destination
    Sort V in descending order of their direct road distance to destination
    for v_i ∈ V do
        if v_i ≠ d then
            dist_direct^(v_i,d) ← D(v_i, d)                ▷ direct road distance from v_i to d
            V′ ← V \ {v_i}                                  ▷ Remove v_i from V
            for v_j ∈ V′ do
                dist_direct^(v_i,v_j) ← D(v_i, v_j)     ▷ road distance from v_i to adjacent v_j
                if dist_direct^(v_i,v_j) ≤ dist_direct^(v_i,d) then
                    if v_i = s AND dist_direct^(v_i,v_j) ≤ ivr then
                        E ← E ∪ {e(v_i, v_j)}
                        w(e(v_i, v_j)) ← dist_direct^(v_i,v_j)
                    else if v_i ≠ s AND dist_direct^(v_i,v_j) ≤ getRange(tvr, dr, v_i, v_j, s, d) then
                        E ← E ∪ {e(v_i, v_j)}
                        w(e(v_i, v_j)) ← dist_direct^(v_i,v_j)
                    end if
                else
                              ▷ skip building an edge with a vertex farther than the destination
                end if
            end for
        end if
    end for
end function
```

5.3.3 Path computation with charge stops

This process is executed when the direct path from source to destination is not feasible; consequently, a reachability graph is built as described in the previous section. It implies that charge stops are necessary to gain sufficient charge in a travel path.

The initial approach with Dijkstra's algorithm yields the shortest path considering the travel distance. However, the method produces the shortest path with potentially more than minimally required charge stops.

To minimise the number of charging stops, the BFS approach is utilised, which yields all n possible paths $P_n^{s,d}$ with an equal and minimal number of intermediate charge stops. $P_{\text{optimal}}^{s,d} \in P_n^{s,d}$ will be the optimal path with the shorter distance among the paths with minimal number of charge stops.

Algorithm 3 Compute path with charge stops

function COMPUTEOPTIMALPATHWITHCHARGESTOPS($G(V, E)$, s, d)
  $P_n^{s,d} \leftarrow$ BFS(G(V, E),s,d) ▷ the algorithm BFS returns n paths from source s to destination d with equal and minimal number of edges
  if $P_n^{s,d}$ is empty **then**
    return $P_{\text{empty}}^{s,d}$ ▷ No shortest path available
  else
    Select $P_{\text{optimal}}^{s,d}$ from $P_n^{s,d}$ with minimal travel distance
    return $P_{\text{optimal}}^{s,d}$
  end if
end function

5.3.4 Energy consumption and itinerary time estimation

The optimal path is defined as $P_{\text{optimal}}^{s,d} = \Big(e(s, c_0), e(c_0, c_1), \ldots, e(c_m, d)\Big)$ with $\{c_0, c_1, \ldots, c_m\} \in C_{\text{optimal}} \subset C$. Each sub-path is termed as a journey leg (interpreted as an edge $e(v_i, v_j) \in P_{\text{optimal}}^{s,d}$) and is enriched with a few attributes that aid in computing energy consumption and time estimation. Each leg within a journey has a source (actual journey source or an EVSE) and a destination (actual journey destination or an EVSE). Upon reaching the destination of each leg, the EV will charge its battery at that charging station. Charging operations at the final leg do not occur since the actual destination is not a charging station. The attributes of each leg with source v_i and destination v_j with $e(v_i, v_j) \in P_{\text{optimal}}^{s,d}$, are defined in table 5.5.

At each $c_i \in C_{\text{optimal}}$, a charge estimation is computed so that sufficient energy is charged to reach the next charge stop $c_{i+1} \in C_{\text{optimal}}$ or possibly the final destination d if $c_i = c_m$ is the final charge stop. At c_m the desired range after the journey is also considered. Further, algorithm 4 portrays a straightforward approach consisting of physical calculations of energy variables. The logic is implemented to estimate energy consumption and duration for recharging at each charge station and each leg's travel distance and duration.

Attribute	**Description**
source	the source vertex of the journey leg
destination	the destination vertex of the journey leg
travelDistance	travel distance in kilometres of this leg (weight $w(e(v_i, v_j))$ obtained from Google Maps Distance Matrix API call)
travelDuration	travel duration in minutes of this leg (obtained from Google Maps Directions API call)
availableRangeAtDestinationBefore Recharge	available range of the EV in kilometres corresponding to the estimated battery SoC before recharge operation
availableRangeAtDestinationAfter Recharge	desired range of the EV in kilometres after recharge operation
energyNeeded	energy to be charged computed for the EV in kWh to start the next journey leg
chargeDuration	charge duration computed against calculated energy and the battery intake estimated for the vehicle
$T^{\text{leg}}_{\text{start}}$	start time of journey leg
$T^{\text{leg}}_{\text{end}}$	end time of journey leg

Table 5.5: Attributes of a journey leg

Moreover, `connector` and `batteryProfile` attributes feed the algorithm as input parameters. `connector` attribute from the 'Location' object structure (refer to figure 4.3) gives associated attributes of the charge station connector, whereas table 5.6 describes attributes of `batteryProfile`.

With the computed charge durations for each journey leg, the estimation of charging time slots for each charge operation at the charging station is carried out. However, the reservation of charge points is not addressed in the scope of the thesis; the computed time slots aids in planning precise charging schedules and reservation operations in future work. An adjustable overhead time (t' minutes) is assumed at each charge station during which basic tasks such as parking an EV, connecting/disconnecting charge cables, user credential validation, etc. are performed.

Attribute	Description
connectorType	described in section 5.2
typicalRange	described in section 5.2
$\mathrm{SoC}_{\mathrm{ini}}$	described in section 5.2
averageVehicleEfficiency	average injected amount of energy in kWh needed for a charge in order to cover the desired number of kilometers (for instance, 15kWh/100 kilometres). This value varies with the EV model.
avgChargingEfficiency	percentage of power drawn from the charging station connector by the vehicle battery
onBoardChargerPowerLimitation	charging speed is limited by vehicle's on-board charger in case of AC power source at charging station. Notably, charging strategy for protecting batteries life span is addressed by the battery management system (both in AC and DC cases).
powerFactor	"Power factor of an AC power system is defined as the ratio of the real power absorbed by the load to the apparent power flowing in the circuit, and is a dimensionless number in the closed interval of -1 to 1" [16]. "EV charging station has a power factor of 1 unless load management is available. This is because EV's require a large amount of electricity for potentially hours at a time" [12]. The thesis considers power factor of 1 during energy computation.

Table 5.6: Attributes of EV battery profile

In addition, total itinerary duration is considered beforehand in estimating the charging time slots. Thus the aggregate sum of travel and charging durations of each leg including overhead time yields the total itinerary duration of the optimal shortest path. Besides, the travel itinerary is scheduled so that the journey ends before $T_{\mathrm{start}}^{\mathrm{appointment}}$ (an adjustable threshold time t minutes is subtracted from $T_{\mathrm{start}}^{\mathrm{appointment}}$, refer to table 5.2). Algorithm 5 represents the estimation of charging time slots of each journey leg.

Algorithm 4 Energy and time estimation

function COMPUTEENERGYANDTIME(batteryProfile, connector, journeyLeg)
  powerFactor $\leftarrow 1$
  if connector.power_type = DC **then**
    socketOutPut $\leftarrow \frac{\text{connector.max\_amperage}*\text{connector.max\_voltage}}{1000}$
  else
    if connector.power_type = AC_1_PHASE **then**
      socketOutPut $\leftarrow \frac{\text{powerFactor} * \text{connector.max\_amperage} * \text{connector.max\_voltage}}{1000}$
    else if connector.power_type = AC_3_PHASE **then**
      socketOutPut $\leftarrow \frac{\sqrt{3} * \text{powerFactor} * \text{connector.max\_amperage} * \text{connector.max\_voltage}}{1000}$
    end if
  end if
  if connector.power_type $\neq$ DC **then**
    if socketOutPut > batteryProfile.onBoardChargerPowerLimitation **then**
      socketOutPut $\leftarrow$ batteryProfile.onBoardChargerPowerLimitation
    end if
  end if
  batteryIntake $\leftarrow$ batteryProfile.avgChargingEfficiency $*$ socketOutPut
  ASSERT(destination $\neq d$) ▷ Recharge operation is not done at final destination; desired range after the final destination is already considered at the previous stop at $c_m \in C$
  if destination $\neq d$ **then**
    currentRange $\leftarrow$ journeyLeg.availableRangeAtDestinationBeforeRecharge
    desiredRange $\leftarrow$ journeyLeg.availableRangeAtDestinationAfterRecharge
    rangeNeeded $\leftarrow$ desiredRange $-$ currentRange
    journeyLeg.energyNeeded $\leftarrow \frac{\text{batteryProfile.averageVehicleEfficiency}}{100} * \text{rangeNeeded}$
    journeyLeg.chargeDuration $\leftarrow \frac{\text{journeyLeg.energyNeeded}}{\text{batteryIntake}} * 60$
  end if
end function

5.3.5 Event building

Once the route planner algorithm successfully executes computation modules, the ultimate task is building a new calendar event with charge schedules and associated itinerary information. Referring to the iCalendar file structure (refer to section 3.3), a simple calendar builder approach is implemented to write new events that include vital properties such as title, summary, description, start and end times and alert trigger details. In addition, the event description is embedded into plain HTML content.

Algorithm 5 Total itinerary duration and charging time slots estimation

function CALCULATEITINIRARYDURATION($T_{\text{start}}^{\text{appointment}}$, t, t')
  $J \leftarrow \{j_0, j_1, j_2, ..., j_n\}$ ▷ Set of journey legs in the optimal path
  totalTravelDuration ← 0
  totalChargingDuration ← 0
  totalOverhead ← 0
  for $j_i \in J$ **do**
    COMPUTEENERGYANDTIME(batteryProfile, connector, $\mathrm{j_i}$)
    totalTravelDuration ← totalTravelDuration + j_i.travelDuration
    if destination $\neq d$ **then**
      totalChargingDuration ← totalChargingDuration + j_i.chargeDuration
      totalOverhead ← totalOverhead + t'
    else
      ▷ skip; charging at actual journey destination is not performed
    end if
  end for
  totalItineraryDuration ← totalTravelDuration + totalChargingDuration + totalOverhead
  for $j_i \in J$ **do**
    if destination $\neq d$ **then**
      if source $= s$ **then** ▷ actual journey source
        $T_{\text{start}}^{\text{travel}} \leftarrow T_{\text{start}}^{\text{appointment}}$ − totalItineraryDuration − t ▷ start time of a new travel itinerary
        j_i.chargeStartTime ← $T_{\text{start}}^{\text{travel}}$ + j_i.travelDuration ▷ first leg of the journey
      else
        $j_i.T_{\text{start}}^{\text{leg}} \leftarrow j_{i-1}$.chargeEndTime ▷ start time of journey leg
        j_i.chargeStartTime ← $j_i.T_{\text{start}}^{\text{leg}}$ + j_i.travelDuration
      end if
      j_i.chargeEndTime ← j_i.chargeStartTime + j_i.chargeDuration + t'
    else
      ▷ skip: end of the journey after arriving at actual journey destination
    end if
  end for
end function

6 Test and Verification

A few test scenarios are illustrated in this chapter involving a given set of input parameters fed to the implemented application, and the optimality of results are verified. Furthermore, the impact of HTTP requests to Google Maps services on the system performance is evaluated along with the optimality of the route planner.

6.1 Test Scenarios

An iCalendar file with upcoming event details and a user-defined JSON file containing relevant vehicle and source location information form the input parameters for this scenario. Since the charging station POIs (refer to section 3.5.1) fetched from Open Charge Map constitute many charge points, the proposed reachability graph consequently contains several nodes (refer to section 5.3.2).

Therefore, the test scenarios in use case 1 and use case 2 consider a single simulated EV with an unrealistically low vehicle range in order to force the algorithm to plan possible intermediate charging stops for shorter journeys. Furthermore, use case 3 considers a significantly more extended trip to assess the impact of several API requests on the system performance. Finally, use case 4 deals with a scenario that portrays multiple causes of the unavailability of suitable charge points along the travel path.

6.1.1 Use case 1

This scenario is a straightforward route request. Figure 6.1 shows a scheduled appointment created using Microsoft Outlook that denotes an input calendar event. The relevant attributes such as the upcoming event start/end time and location information make route planner input parameters. The corresponding plaintext calendar file is portrayed in appendix A.7.1. Further, the JSON data provides source location coordinates, battery's $\mathrm{SoC}_{\mathrm{ini}}$ and the typical driving range of the EV, as shown in listing 6.1.

As described in subsection 5.3.1, the initial step of the route planner algorithm is to determine whether the direct route is feasible if the vehicle has sufficient charge. Based on real-time traffic data provided by Google Maps services, the direct path distance of 19.8km falls within the vehicle's available driving range (25km). Hence, the requirement of intermediate charging stop(s) is irrelevant in this scenario.

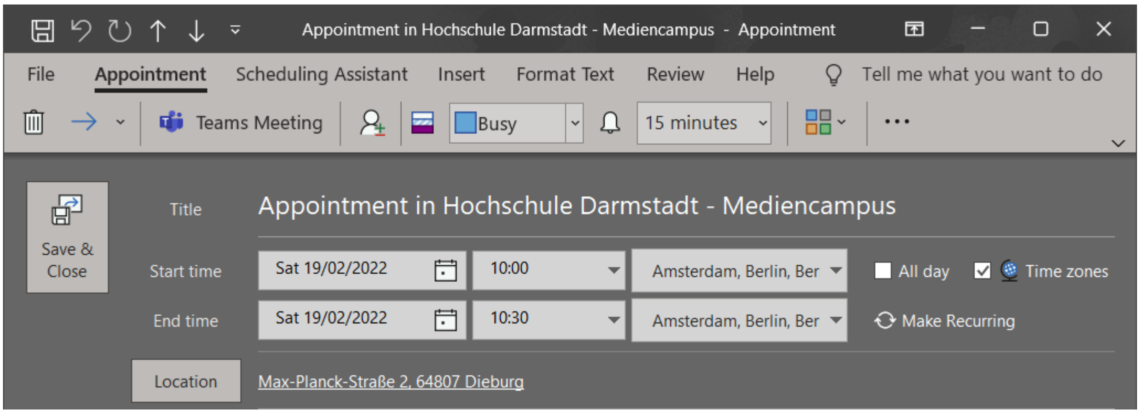


Figure 6.1: Outlook appointment snapshot – Use case 1

Consequently, a new calendar event is created with direct path directions, ensuring the trip ends well within the time of the upcoming appointment (minus t minutes; minus 15 minutes in this case). Figure 6.2 shows the direct path itinerary description of the newly created calendar event. In addition, it provides a link to Google Maps Directions that embeds the source and destination addresses, and when clicked, the Google Maps web client delivers directions to the location requested, as shown in figure 6.3. The plain-text iCalendar file of the new event is presented in appendix A.7.2.

```
{
        "id": "DT-UC1-CUST-001",
        "owner_name": "Software AG Employee",
        "source_address": {
                "name": "Home",
                "address": "Kurzer Weg 3, 64285 Darmstadt",
                "coordinates": {
                        "latitude": "49.8561992",
                        "longitude": "8.6454676"
                }
        },
        "model_specs":{
                "make": "Smart",
                "model": "Smart EQ forfour"
        },
        "battery_profile":{
                "connector_type" : "IEC 62196-2 Type 2",
                "typical_range_in_km": "30",
                "soc_ini": "100%"
        },
        "soc_dest": "30%"
}
```

Listing 6.1: User-defined JSON input file for use case 1

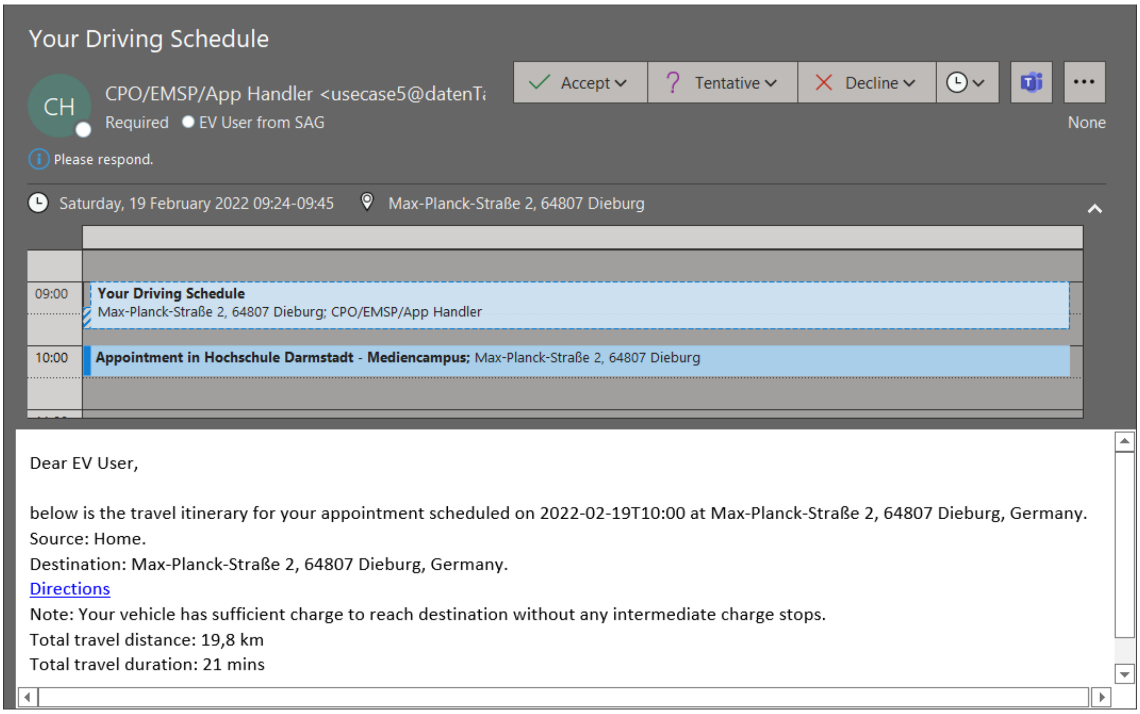


Figure 6.2: Driving schedule snapshot – Use case 1

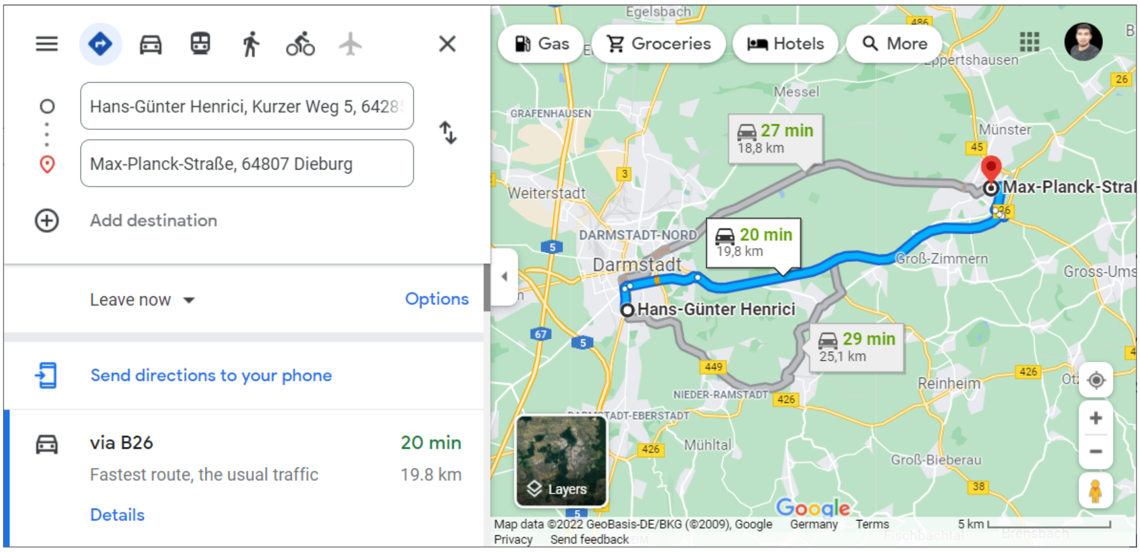


Figure 6.3: Directions from Google Maps – Use case 1

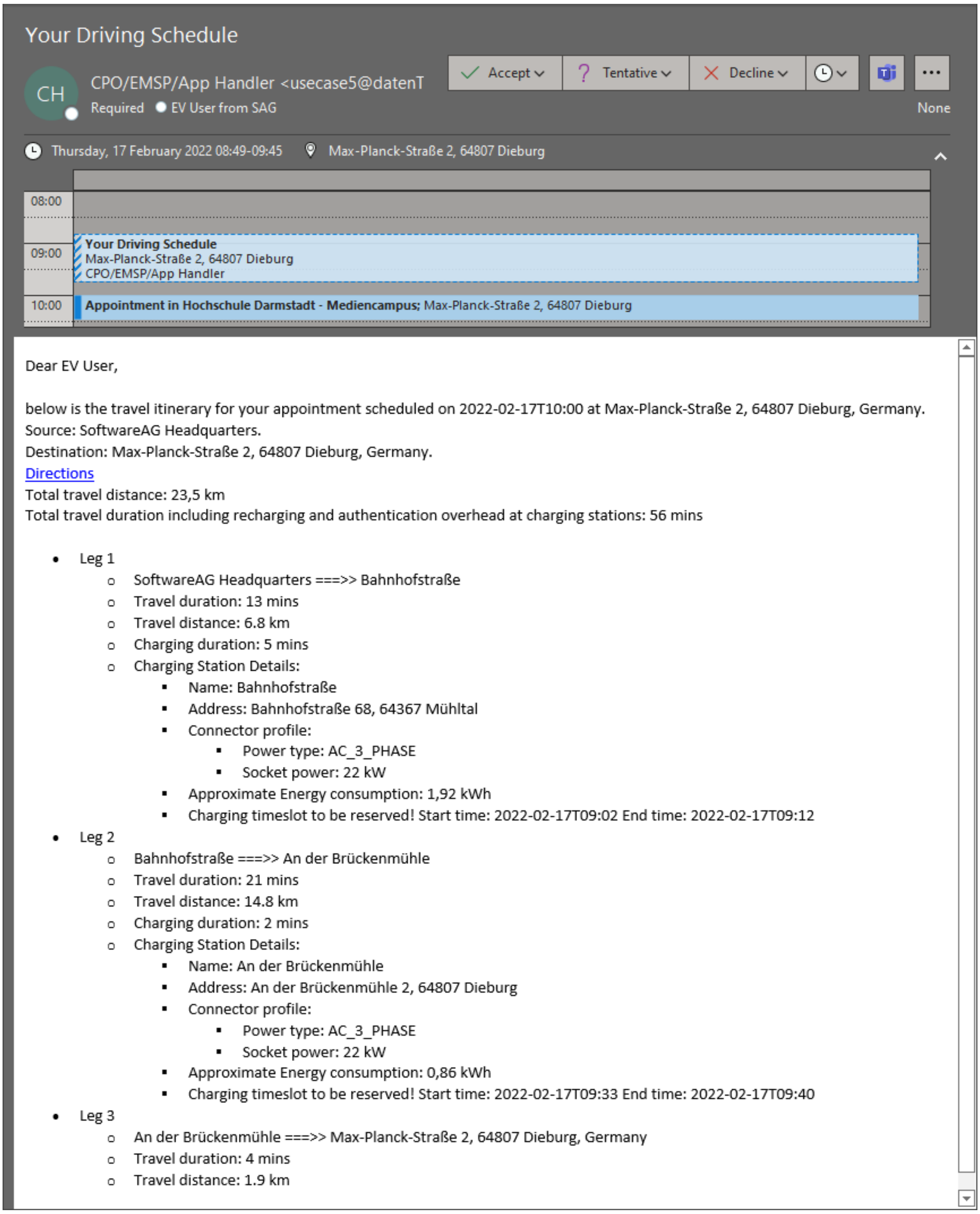
6.1.2 Use case 2

As mentioned earlier, the vehicle's driving range is deliberately fixed to unrealistic low values. In this use case we will use a typical range of 20km and an $\mathrm{SoC}_{\mathrm{ini}}$ of 50%. As a result, the available driving range would be 10km. The direct path is impractical in this case since the battery's $\mathrm{SoC}_{\mathrm{ini}}$ is insufficient to drive the vehicle to its destination without intermediate charging stops. Consequently, the algorithm created a reachability graph $G(V, E)$.

Figure 6.4: Outlook appointment snapshot – Use case 2

```
{
        "id": "DT-UC2-CUST-001",
        "owner_name": "Software AG Employee",
        "source_address": {
                "name": "SoftwareAG Headquarters",
                "address": "Uhlandstraße 12, 64297 Darmstadt",
                "coordinates": {
                        "latitude": "49.81435569634197",
                        "longitude": "8.637055512872708"
                }
        },
        "model_specs":{
                "make": "Volkswagen",
                "model": "e-Golf"
        },
        "battery_profile":{
                "connector_type" : "IEC 62196-2 Type 2",
                "typical_range_in_km": "20",
                "soc_ini": "50%"
        },
        "soc_dest": "30%"
}
```

Listing 6.2: User-defined JSON input file for use case 2



Your Driving Schedule

CPO/EMSP/App Handler <usecase5@datenT

Required EV User from SAG

Accept Tentative Decline

None

Thursday, 17 February 2022 08:49-09:45 Max-Planck-Straße 2, 64807 Dieburg

08:00

09:00 **Your Driving Schedule** Max-Planck-Straße 2, 64807 Dieburg CPO/EMSP/App Handler

10:00 **Appointment in Hochschule Darmstadt - Mediencampus;** Max-Planck-Straße 2, 64807 Dieburg

Dear EV User,

below is the travel itinerary for your appointment scheduled on 2022-02-17T10:00 at Max-Planck-Straße 2, 64807 Dieburg, Germany.
Source: SoftwareAG Headquarters.
Destination: Max-Planck-Straße 2, 64807 Dieburg, Germany.
Directions
Total travel distance: 23,5 km
Total travel duration including recharging and authentication overhead at charging stations: 56 mins

- Leg 1
  - SoftwareAG Headquarters ===>> Bahnhofstraße
  - Travel duration: 13 mins
  - Travel distance: 6.8 km
  - Charging duration: 5 mins
  - Charging Station Details:
    - Name: Bahnhofstraße
    - Address: Bahnhofstraße 68, 64367 Mühltal
    - Connector profile:
      - Power type: AC_3_PHASE
      - Socket power: 22 kW
    - Approximate Energy consumption: 1,92 kWh
    - Charging timeslot to be reserved! Start time: 2022-02-17T09:02 End time: 2022-02-17T09:12
- Leg 2
  - Bahnhofstraße ===>> An der Brückenmühle
  - Travel duration: 21 mins
  - Travel distance: 14.8 km
  - Charging duration: 2 mins
  - Charging Station Details:
    - Name: An der Brückenmühle
    - Address: An der Brückenmühle 2, 64807 Dieburg
    - Connector profile:
      - Power type: AC_3_PHASE
      - Socket power: 22 kW
    - Approximate Energy consumption: 0,86 kWh
    - Charging timeslot to be reserved! Start time: 2022-02-17T09:33 End time: 2022-02-17T09:40
- Leg 3
  - An der Brückenmühle ===>> Max-Planck-Straße 2, 64807 Dieburg, Germany
  - Travel duration: 4 mins
  - Travel distance: 1.9 km

Figure 6.5: Driving schedule snapshot – Use case 2

Similar to use case 1, figure 6.4 shows the upcoming appointment details, and the corresponding plain-text calendar file is presented in appendix A.8.1. Further, listing 6.2 provides JSON data with relevant input parameters that feed the route planner algorithm. The algorithm computed an optimal path out of several shortest paths constituting two intermediate charging stops. The associated metrics derived during computations are addressed in section 6.2.

Lastly, figure 6.5 displays the newly created calendar event with itinerary travel information such as energy and time estimates, and details of the charging stations along the computed travel path. Additionally, the event provides a link to Google Maps web client with embedded source, destination, and charging station locations, as shown in figure 6.6. The plain-text iCalendar file of the new event is presented in appendix A.8.2.

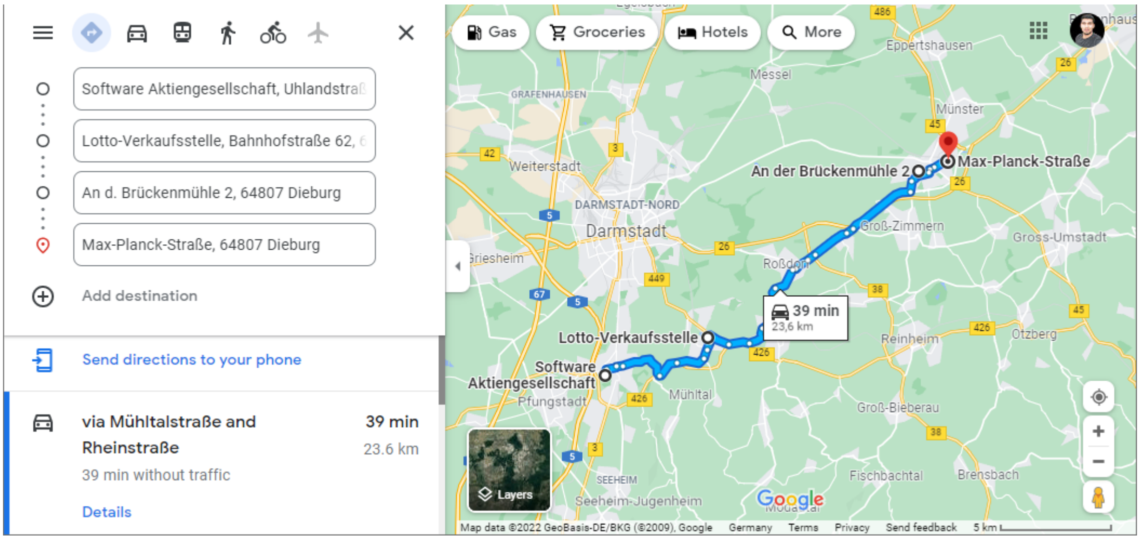


Figure 6.6: Directions from Google Maps – Use case 2

6.1.3 Use case 3

A much longer route request is presented in this scenario. Thus, a practical use case with a vehicle's typical driving range of 200km is considered. The input calendar file, as shown in figure 6.7 and the JSON data, as shown in 6.3, supply parameters with the route request.

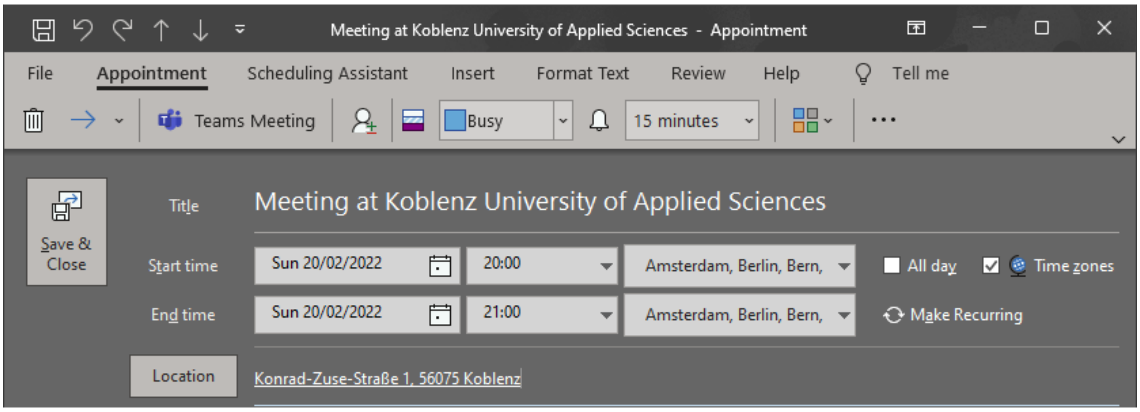


Figure 6.7: Outlook appointment snapshot – Use case 3


```
{
        "id": "DT-UC3-CUST-001",
        "owner_name": "Software AG Employee",
        "source_address": {
                "name": "SoftwareAG Headquarters",
                "address": "Uhlandstraße 12, 64297 Darmstadt",
                "coordinates": {
                        "latitude": "49.81435569634197",
                        "longitude": "8.637055512872708"
                }
        },
        "model_specs":{
                "make": "Volkswagen",
                "model": "e-Golf"
        },
        "battery_profile":{
                "connector_type" : "IEC 62196-2 Type 2",
                "typical_range_in_km": "200",
                "soc_ini": "50%"
        },
        "soc_dest": "30%"
}
```


Listing 6.3: User-defined JSON input file for use case 3

Similar to use case 2, the direct path is not feasible here as well. Therefore, the reachability graph $G(V, E)$ consists of a significantly higher number of EVSE nodes since the region covered by the graph is much larger (roughly 116km in diameter).

In $G(V, E)$, the edge weights are set based on the road travel distances between the nodes according to Google Maps (refer to section 5.3.2). Thus, a significantly higher number of HTTP API requests were performed. As a result, system performance is largely impacted by the latency of these requests. Clever techniques such as response caching and multi-dimensional querying (refer to section 3.5.2) minimised redundant API requests. The associated metrics derived in this scenario are discussed in section 6.2. An equivalent approach is utilised in use case 2 as well. However, the latency factor is significantly lesser because of fewer EVSE nodes.

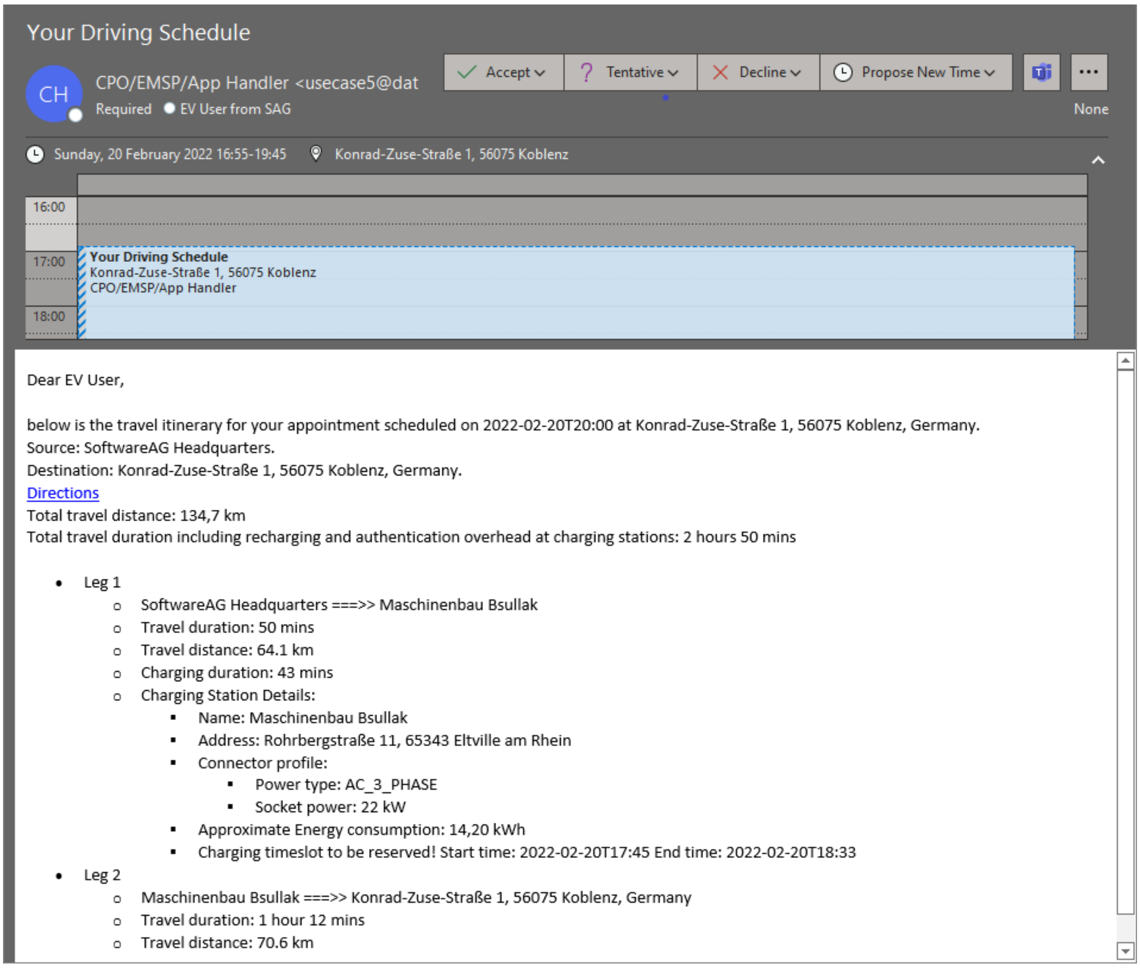


Figure 6.8: Driving schedule snapshot – Use case 3

Eventually, the algorithm determined the best path out of several shortest paths comprising one intermediate charging stop. Once the optimal path is derived, similar to the results of use case 2, energy and time estimations and a link to Google Maps web client are detailed in a new calendar event, as shown in figure 6.8. Figure 6.9 depicts directions for the optimal path with one intermediate charging stop.

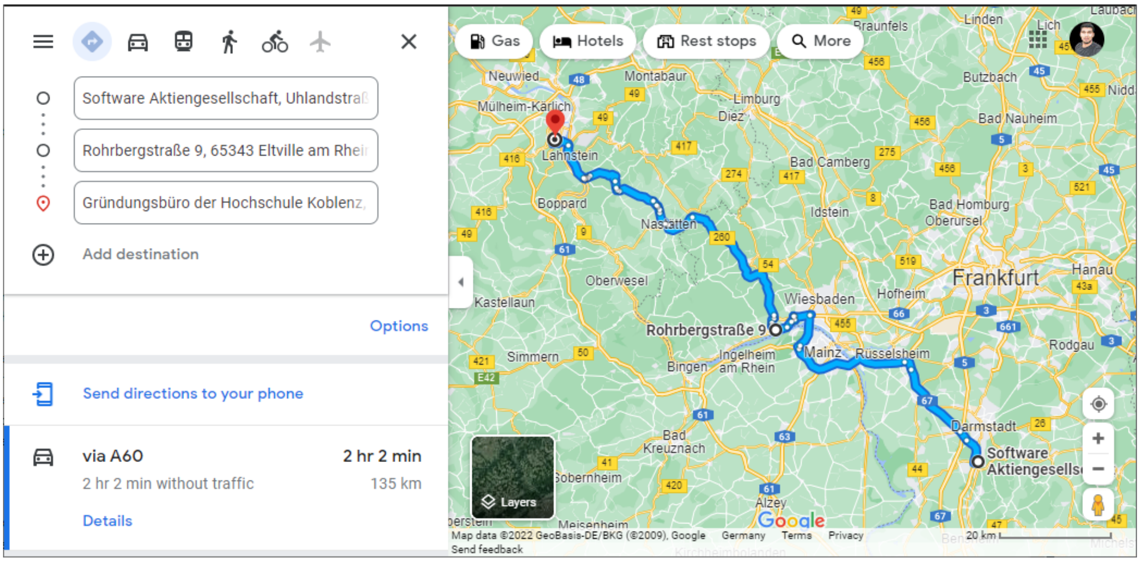


Figure 6.9: Directions from Google Maps – Use case 3

6.1.4 Use case 4

A typical scenario where the route planner algorithm fails to produce an optimal path is illustrated by this use case. Similar to use case 2, it considers a short journey with unrealistic input parameters. Based on the address of the destination retrieved from the input calendar event, as shown in figure 6.10, the direct travel path from the source results in a distance of 19.7km. However, the direct path is not feasible with the available initial driving range of 6km (SoC_{ini}: 30%, typical driving range: 20km) derived from the JSON input data, as shown in listing 6.4.

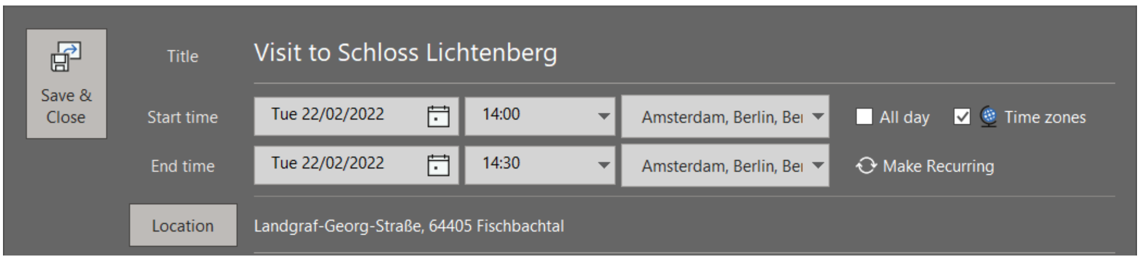


Figure 6.10: Outlook appointment snapshot – Use case 4

Specifically, in this case, the nearest charging station (10km away from the source; other stations are farther) cannot not be reached with the available SoC of the battery. Hence, neither the edges from source to potential charging station nodes were built nor the reachability graph. Therefore, the scenario yields no travel path.


```
{
        "id": "DT-UC4-CUST-005",
        "owner_name": "Software AG Employee",
        "source_address": {
                "name": "SoftwareAG Headquarters",
                "address": "Uhlandstraße 12, 64297 Darmstadt",
                "coordinates": {
                        "latitude": "49.81435569634197",
                        "longitude": "8.637055512872708"
                }
        },
        "model_specs":{
                "make": "Nissan",
                "model": "Leaf"
        },
        "battery_profile":{
                "connector_type" : "IEC 62196-2 Type 2",
                "typical_range_in_km": "20",
                "soc_ini": "30%"
        },
        "soc_dest": "30%"
}
```


Listing 6.4: User-defined JSON input file for use case 4

Notably, a similar scenario might arise where the algorithm fails to build a connected reachability graph. This is when the region covered by the reachability graph is poorly equipped with fewer charging stations, and weighted edges cannot be built among the nodes due to the inconsistent positioning of the charging stations.

6.2 Scalability Test

The system's scalability and performance is primarily influenced by the time responsiveness of the route planner service, which is the time elapsed between submitting a route request and generating a calendar event with suitable travel itinerary.

From a reachability graph constructed by the route planner algorithm, the position of nodes (source, destination, EVSEs) is assumed to be static (due to fixed geographical coordinates). However, the edge weights between connected nodes can vary since they are the function of travel distance, which is retrieved from the Google Maps services based on real-time traffic conditions.

Moreover, the complexity of the algorithm depends on the number of EVSEs present in the reachability graph. However, since the availability of the EVSEs is not uniform across geographical regions (urban regions are evidently equipped with a comparatively large number of EVSEs than remote areas), it cannot precisely be presumed that response time increases with the area covered by the reachability graph. Nevertheless, the response time is mainly affected by the latency of the HTTP requests to the Google Maps services.

Table 6.1 denotes relevant metrics derived while computing optimal paths for use cases 2 and 3. The intelligent usage of API calls determines the effectiveness of the route planner algorithm through caching and multi-dimensional querying, thereby avoiding redundant requests. With this approach, as in use case 2, the possibility of performing 1275 individual HTTP requests was reduced to 75 calls. Similarly, as in use case 3, it was reduced to 1126 from 25651 individual calls, consequently reducing high latency.

Metric	**Use case 2**	**Use case 3**
Diameter of $\boldsymbol{G(V,E)}$ in kilometres	19.4	116
Total charging stations in $\boldsymbol{G(V,E)}$	49	225
Number of Geocoding API calls	1	1
Number of Directions API calls	1	1
Possible number of individual Distance Matrix API calls	1275	25651
Number of cached Distance Matrix API calls	50	226
Number of Distance Matrix API calls through multi-dimensional querying	75	1126
Feasible paths computed by route planner algorithm	165(2 charging stops)	202(1 charging stop)
Time taken to construct $\boldsymbol{G(V,E)}$	4.63 seconds	71.76 seconds
Total elapsed time in executing a route request	6.44 seconds	73.65 seconds

Table 6.1: Metrics derived from use cases 2 and 3

For a constructed reachability graph $G(V, E)$, let n be the total number of vertices in V. Total number of possible HTTP requests when queried individually is given by

$$\sum_{k=1}^{n-1} n - 1 - k + 1$$

Moreover, the Distance Matrix API request has a maximum dimension limit [15]. Therefore, a request cannot contain more than 25 'origins' or more than 25 'destinations'. Based on the results returned by API call, each result corresponding to a origin-destination pair is termed an 'element'. So, requests are billed at a price-per-element model in a multi-dimensional API call. Thus, the number of origins times the number of destinations defines the total number of elements returned by the API request. Hence, multi-dimensional querying enabled a significant reduction of requests. In addition, redundant elements were desirably avoided by utilising cached responses within the system.

Upon constructing the reachability graph, the algorithm's ability to generate an optimal path $P^{s,d}_{\text{optimal}}$ from number a of feasible paths with equal and minimal charging stops determines its efficiency. Use case 2 has 165 unique paths with two charging stops, whereas use case 3 at 202 unique paths with only one charging stop. Besides, the number of charging stops depends on initial SoC and the typical driving range of an EV. In both scenarios, the algorithm computed an optimal (minimal-charge-stop-minimal-distance) path as depicted in their results.

The optimal path computation is followed by energy and time estimations, ultimately creating a travel itinerary in the form of a new calendar event. Table 6.1 displays the time elapsed during system execution. According to the results, the time elapsed during system execution excluding the time required for reachability graph construction can be considered acceptable for both use cases 2 and 3 (provided the route request is submitted only once per execution). Hence, it can be concluded that the response time is mainly impacted by the latency of numerous HTTP requests and the algorithm is efficient in computing an optimal path in acceptable duration.

7 Conclusion and Future Work

This chapter intends to provide a conclusion to the thesis report and an overview of the future work.

7.1 Conclusion

Planning travel itineraries with suitable charge stops for electric vehicle driving schedules in the form of calendar events is demonstrated successfully in this thesis using an integrated route planner service.

The initial study looking at electric mobility challenges revealed that EV users are concerned about vehicle costs, limited range, and inconsistent coverage of charging infrastructure; thereby requiring users to carefully plan their itineraries taking into consideration of possible yet uncertain charging opportunities along the travel path. In addition, the thesis narrated feedback provided by a couple of EV users with similar challenges.

Accordingly, the thesis discussed the need for intelligent solutions based on ICT-based applications to overcome such EV mobility challenges. Multiple research questions were formulated while addressing the goals of the thesis. In order to realize the core objectives of the thesis, a service-oriented architecture was introduced to design and implement a novel system of interest constituting relevant agents and their functionalities.

A local MongoDB database manager was successfully set up for the purpose of storing and retrieving EVSE data by the route planner service. EVSE data with relevant attributes were retrieved from the external resource, while Google Maps API services provided precise routing and geolocation information.

Existing user calendar files contained information about appointments, while newly created calendar events containing travel itinerary details were generated as system output.

Further, an algorithmic route planner service, comprising range prediction and optimal route finder modules was developed successfully. The reachability graph formed the basis for the route planner algorithm that included a set of source, destination and feasible EVSEs.

In addition, the developed application was tested with distinct use case scenarios, provided with user-defined input parameters. A direct path from the source to the destination was feasible with use case 1, which did not require the core logic of the route planner algorithm. However, use cases 2 and 3 demonstrated the functionality of the route planner yielding an optimal travel path with suitable charge stop(s). The route planner ultimately generated travel itineraries for respective test scenarios in the form of calendar events comprising relevant information such as travel/charging duration, energy consumption estimations and travel directions. Besides, use case 4 dealt with a scenario where the application fails to generate an optimal path due to a lack of suitable charge points along the travel path.

Scalability and the effectiveness of the algorithm were evaluated through simulated test scenarios. The test results and corresponding metrics revealed that the response time was affected by the complexity of the reachability graph, and thus by the number of HTTP requests. However, the system proved to be approvingly effective in computing the optimal path for a given route request.

Having obtained the results of simulated EV scenarios, it can be concluded that the developed route planner application plays a crucial role in assisting EV users with suitable travel itineraries in the form of calendar events. Especially, it proved to be a convenient solution in both mixed urban/suburban and remote environments where the coverage provided by the charging infrastructure might not be uniform.

7.2 Future Work

The system is designed based on a service-oriented architecture as described in chapter 4. The architecture facilitates the integration of new services and solutions with the existing work presented in this thesis. Having said that, the immediate extension to the route planner service is the integration of an advanced reservation mechanism for the calculated charging time slots at respective EVSEs. The dashed box in blue, containing shaded processes, as shown in figure 5.1, indicates the possibility of integrating a reservation feature for charging slots.

Moreover, the theoretical framework of the application was developed on a prototype level. The route planner algorithm can be extended in order to determine an improved optimal path considering better trade-offs in terms of travel/charging duration based on practical energy/time consumption models (i.e. considering physical parameters of EV and path, and several intrinsic and external factors impacting battery characteristics). Moreover, the route planner service may implement a mobile (Android/iOS) client application with a user-friendly GUI with embedded routing and navigation capabilities.

A Appendix

A.1 Location Object Description

The Location object notation laid out by OCPI describes the location and its properties where a group of EVSEs that belong together are installed. Typically, the Location object is the exact location of the group of EVSEs, but it can also be the entrance of a parking garage that contains these EVSEs. The object structure is limited to a few properties corresponding to the scope of the thesis. Further information on additional properties is detailed in [22].

Property	Datatype	Card.	Description
id	string	1	uniquely identifies the location within the CPOs platform (and suboperator platforms; presently, within the locally deployed database server). This field can never be changed, modified or renamed.
name	string	1	display name of the location
address	string	1	street/block name and house number if available
city	string	1	city/town
postal_code	string	1	postal/zip code of the location, may only be set to null when the location has no postal code: in some countries charging locations at highways don't have postal codes.
state	string	1	state or province of the location
country	string	1	ISO code for the country of this location
coordinates	Coordinates	1	coordinates of this location
evses	EVSE	*	list of EVSEs that belong to this location

Table A.1: Location Object Description

A.1.1 Coordinates Object Description

The Coordinates object defines the Geo-location attributes of the charge point location. **Note**: "Five decimal places is seen as a minimum for Global Positioning System (GPS) coordinates of the Charge Point as this gives approximately 1 meter precision. More is always better. Seven decimal places gives approximately 1cm precision." [22]

Property	**Datatype**	**Card.**	**Description**
latitude	string	1	Latitude of the point in decimal degree. Example: 50.770774. Decimal separator: "." Regex: -?[0-9]{1,2}.[0-9]{5,7}
longitude	string	1	Longitude of the point in decimal degree. Example: -126.104965. Decimal separator: "." Regex: -?[0-9]{1,3}.[0-9]{5,7}

Table A.2: Coordinates Object Description

A.1.2 EVSE Object Description

The EVSE object describes the part that controls the power supply to a single EV in a single session. An EVSE object has a list of connectors which cannot be used simultaneously: only one connector per EVSE at a time.

Property	**Datatype**	**Card.**	**Description**
evse_id	string	1	uniquely identifies an EVSE
status	Status	1	indicates the current status of the EVSE
status_schedule	StatusSchedule	*	indicates the planned status update of the EVSE
connectors	Connector	*	list of available connectors on the EVSE

Table A.3: EVSE Object Description

A.1.3 Status Enum Description

The status of the connector denotes the availability of the EVSEs and hence, these enumeration values are utilized in filtering the charge point objects in order to feed into route planner algorithm.

Value	Description
AVAILABLE	the EVSE/Connector is able to start a new charging session
RESERVED	the EVSE/Connector is reserved for a particular EV driver and is unavailable for other drivers
UNKNOWN	no status information available; inoperative or out of order. More values are detailed in [22]

Table A.4: Status Enum Description

A.1.4 StatusSchedule Object Description

This type is used to schedule status periods in the future. The values can be used for trip planning purposes. The estimated charge session time values are updated if the reservation feature is enabled. However, this feature is to be addressed in the future work. This field is directly linked to the EVSE's status field itself. When the status actually changes, the CPO must push latest update to the EVSE's status field.

Property	Datatype	Card.	Description
period_begin	LocalDateTime	1	begin of the scheduled period
period_begin	LocalDateTime	1	end of the scheduled period, if known; sometimes may be empty or null if the status is UNKNOWN
status	Status	1	status value during scheduled period

Table A.5: StatusSchedule Object Description

A.1.5 Connector Object Description

A Connector is the socket or cable and plug available for the EV to use. A single EVSE may provide multiple Connectors but only one of them can be in use at the same time. These attributes are used for energy and time estimations of each charge sessions.

Property	**Datatype**	**Card.**	**Description**
standard	ConnectorType	1	the standard of the installed connector
format	ConnectorFormat	1	format (socket/cable) of the installed connector
power_type	PowerType	1	connector output power type (AC/DC)
max_voltage	int	1	maximum voltage of the connector, in volt [V]. For example: DC Chargers might vary the voltage during charging depending on the battery State of Charge.
max_amperage	int	1	maximum amperage of the connector, in ampere [A].
max_electric_power	int	1	maximum electric power that can be delivered by this connector, in Watts (W).

Table A.6: Connector Object Description

A.2 OCM API Sample

Sample request URL:

```
https://api.openchargemap.io/v3/poi/?countrycode=DE&camelcase=true&
   maxresults=1&key=<licensed key string>
```

Sample response:


```
{
    "dataProvider": {
        "websiteURL": "http://openchargemap.org",
        "comments": null,
        "dataProviderStatusType": {
            "isProviderEnabled": true,
            "id": 1,
            "title": "Manual Data Entry"
        },
        "isRestrictedEdit": false,
        "isOpenDataLicensed": true,
        "isApprovedImport": true,
        "license": "Licensed under Creative Commons Attribution-
           ShareAlike 4.0 International (CC BY-SA 4.0)",
        "dateLastImported": null,
        "id": 1,
        "title": "Open Charge Map Contributors"
    },
    "operatorInfo": {
        "websiteURL": "http://www.chargeit-mobility.com/",
        "comments": null,
        "phonePrimaryContact": "+49 9321 268 -0700",
        "phoneSecondaryContact": null,
        "isPrivateIndividual": false,
        "addressInfo": null,
        "bookingURL": null,
        "contactEmail": "info@chargeit-mobility.com",
        "faultReportEmail": null,
        "isRestrictedEdit": false,
        "id": 101,
        "title": "ChargeIT mobility"
    },
    "usageType": {
        "isPayAtLocation": false,
        "isMembershipRequired": true,
        "isAccessKeyRequired": true,
        "id": 4,
        "title": "Public - Membership Required"
    },
    "statusType": {
        "isOperational": true,
        "isUserSelectable": true,
```

```
        "id": 50,
        "title": "Operational"
    },
    "submissionStatus": {
        "isLive": true,
        "id": 200,
        "title": "Submission Published"
    },
    "userComments": null,
    "percentageSimilarity": null,
    "mediaItems": null,
    "isRecentlyVerified": true,
    "dateLastVerified": "2022-02-09T13:03:00Z",
    "id": 191415,
    "uuid": "0F293EEA-4FCC-4777-A7E5-661FF44032C0",
    "parentChargePointID": null,
    "dataProviderID": 1,
    "dataProvidersReference": null,
    "operatorID": 101,
    "operatorsReference": null,
    "usageTypeID": 4,
    "usageCost": null,
    "addressInfo": {
        "id": 191774,
        "title": "Regensburger Straße",
        "addressLine1": "Regensburger Straße",
        "addressLine2": null,
        "town": "Nittendorf",
        "stateOrProvince": "Free State of Bavaria",
        "postcode": "93152",
        "countryID": 87,
        "country": {
            "isoCode": "DE",
            "continentCode": "EU",
            "id": 87,
            "title": "Germany"
        },
        "latitude": 49.026213267779724,
        "longitude": 11.957210013615907,
        "contactTelephone1": null,
        "contactTelephone2": null,
        "contactEmail": null,
        "accessComments": null,
        "relatedURL": null,
        "distance": null,
        "distanceUnit": 0
    },
    "connections": [
        {
            "id": 310798,
```

```
            "connectionTypeID": 25,
            "connectionType": {
                "formalName": "IEC 62196-2 Type 2",
                "isDiscontinued": false,
                "isObsolete": false,
                "id": 25,
                "title": "Type 2 (Socket Only)"
            },
            "reference": null,
            "statusTypeID": 50,
            "statusType": {
                "isOperational": true,
                "isUserSelectable": true,
                "id": 50,
                "title": "Operational"
            },
            "levelID": 2,
            "level": {
                "comments": "Over 2 kW, usually non-domestic socket
                   type",
                "isFastChargeCapable": false,
                "id": 2,
                "title": "Level 2 : Medium (Over 2kW)"
            },
            "amps": 32,
            "voltage": 400,
            "powerKW": 22.0,
            "currentTypeID": 20,
            "currentType": {
                "description": "Alternating Current - Three Phase",
                "id": 20,
                "title": "AC (Three-Phase)"
            },
            "quantity": 2,
            "comments": null
        }
    ],
    "numberOfPoints": 2,
    "generalComments": null,
    "datePlanned": null,
    "dateLastConfirmed": null,
    "statusTypeID": 50,
    "dateLastStatusUpdate": "2022-02-09T13:03:00Z",
    "metadataValues": null,
    "dataQualityLevel": 1,
    "dateCreated": "2022-02-09T07:44:00Z",
    "submissionStatusTypeID": 200
}
```

A.3 Charge Point Schema Sample


```
{
    "id":"B0481F37-C65B-4B5C-B04A-91F52A2AD042",
    "name":"Markant Tankstelle",
    "address":"Stiftsallee 36",
    "city":"Minden",
    "postal_code":"32425",
    "state":"Nordrhein-Westfalen",
    "country":"Germany",
    "coordinates":{
        "latitude":"52.34806030173826",
        "longitude":"8.883787743680159"
    },
    "operator":{
        "website":"http://www.essent.nl",
        "name":"RWE Mobility/Essent"
    },
    "evses":[
        {
            "evse_id":"189548",
            "connectors":[
                {
                    "standard":"IEC 62196-2 Type 2",
                    "max_amperage":32,
                    "format":"CABLE",
                    "max_voltage":400,
                    "id":"306617",
                    "max_electric_power":22000,
                    "power_type":"AC_3_PHASE"
                }
            ],
            "status":"AVAILABLE",
            "status_schedule":[
            ]
        }
    ],
}
```

A.4 Geocoding API Sample

The below sample request yield accurate coordinate values for a given address string: Schöfferstraße 3, 64295 Darmstadt.

Sample request URL:

```
https://maps.googleapis.com/maps/api/geocode/json?&address=Schöfferstra
   ße 3, 64295 Darmstadt&key=<licensed key string>
```

Sample response:


```
{
    "results": [
        {
            "address_components": [
                {
                    "long_name": "3",
                    "short_name": "3",
                    "types": [
                        "street_number"
                    ]
                },
                {
                    "long_name": "Schöfferstraße",
                    "short_name": "Schöfferstraße",
                    "types": [
                        "route"
                    ]
                },
                {
                    "long_name": "Darmstadt-West",
                    "short_name": "Darmstadt-West",
                    "types": [
                        "political",
                        "sublocality",
                        "sublocality_level_1"
                    ]
                },
                {
                    "long_name": "Darmstadt",
                    "short_name": "Darmstadt",
                    "types": [
                        "locality",
                        "political"
                    ]
                },
                {
                    "long_name": "Kreisfreie Stadt Darmstadt",
                    "short_name": "Kreisfreie Stadt Darmstadt",
                    "types": [
```

```
                "administrative_area_level_3",
                "political"
            ]
        },
        {
            "long_name": "Darmstadt",
            "short_name": "DA",
            "types": [
                "administrative_area_level_2",
                "political"
            ]
        },
        {
            "long_name": "Hessen",
            "short_name": "HE",
            "types": [
                "administrative_area_level_1",
                "political"
            ]
        },
        {
            "long_name": "Germany",
            "short_name": "DE",
            "types": [
                "country",
                "political"
            ]
        },
        {
            "long_name": "64295",
            "short_name": "64295",
            "types": [
                "postal_code"
            ]
        }
    ],
    "formatted_address": "Schöfferstraße 3, 64295 Darmstadt,
       Germany",
    "geometry": {
        "location": {
            "lat": 49.867354,
            "lng": 8.638264699999999
        },
        "location_type": "ROOFTOP",
        "viewport": {
            "northeast": {
                "lat": 49.86870298029149,
                "lng": 8.6396136802915
            },
            "southwest": {
```

```
                        "lat": 49.8660050197085,
                        "lng": 8.636915719708497
                    }
                }
            },
            "place_id": "ChIJ-y5FIK5xvUcRApc-sSjCpDs",
            "plus_code": {
                "compound_code": "VJ8Q+W8 Darmstadt, Germany",
                "global_code": "8FXCVJ8Q+W8"
            },
            "types": [
                "street_address"
            ]
        }
    ],
    "status": "OK"
}
```


A.5 Directions API Sample

The below sample request yield accurate travel distance and duration values from source to intermediate stops (waypoints) and waypoints to destination. The corresponding coordinates attributes are fed to the URL request.

- Source coordinates: 49.81435569634197,8.637055512872708 (Uhlandstraße 12, 64297 Darmstadt, Germany)
- Intermediate stop coordinates: 49.8384401432898,8.6561322322628 (Auf d. Marienhöhe 8, 64297 Darmstadt, Germany)
- Destination coordinates: 49.874294322646286,8.630791981514493 (Poststraße 4-6, 64293 Darmstadt, Germany)

Sample request URL:

```
https://maps.googleapis.com/maps/api/directions/json?origin
    =49.81435569634197,8.637055512872708&destination
    =49.874294322646286,8.630791981514493&waypoints={{
    wayPointsParameter}}49.8384401432898,8.6561322322628&key=<licensed
    key string>
```

Sample response:


```
{
    "geocoded_waypoints": [
        {
            "geocoder_status": "OK",
            "place_id": "ChIJX7H8bAR7vUcRp2ziE363gQI",
```

```
            "types": [
                "establishment",
                "point_of_interest"
            ]
        },
        {
            "geocoder_status": "OK",
            "place_id": "ChIJq5fwXut6vUcRDyaiZQeekj0",
            "types": [
                "establishment",
                "point_of_interest"
            ]
        },
        {
            "geocoder_status": "OK",
            "place_id": "ChIJQ_okd4VwvUcRxkpe1tBkkLE",
            "types": [
                "establishment",
                "point_of_interest"
            ]
        }
    ],
    "routes": [
        {
            "bounds": {
                "northeast": {
                    "lat": 49.8750199,
                    "lng": 8.656457899999999
                },
                "southwest": {
                    "lat": 49.814826,
                    "lng": 8.630831199999999
                }
            },
            "copyrights": "Map data 2022 GeoBasis-DE/BKG (2009)",
            "legs": [
                {
                    "distance": {
                        "text": "3.7 km",
                        "value": 3686
                    },
                    "duration": {
                        "text": "8 mins",
                        "value": 508
                    },
                    "end_address": "Auf d. Marienhöhe 8, 64297
                       Darmstadt, Germany",
                    "end_location": {
                        "lat": 49.83844029999999,
                        "lng": 8.656264199999999
```

```
        },
        "start_address": "Uhlandstraße 12, 64297 Darmstadt,
            Germany",
        "start_location": {
            "lat": 49.8149046,
            "lng": 8.637051999999999
        },
        "steps": [
                                    /* Steps to drive */
                                    {},
                                    ...
        ],
        "traffic_speed_entry": [],
        "via_waypoint": []
    },
    {
        "distance": {
            "text": "6.0 km",
            "value": 5953
        },
        "duration": {
            "text": "15 mins",
            "value": 923
        },
        "end_address": "Poststraße 4-6, 64293 Darmstadt,
           Germany",
        "end_location": {
            "lat": 49.8739649,
            "lng": 8.630831199999999
        },
        "start_address": "Auf d. Marienhöhe 8, 64297
           Darmstadt, Germany",
        "start_location": {
            "lat": 49.83844029999999,
            "lng": 8.656264199999999
        },
        "steps": [
                                    /* Steps to drive */
                                    {},
                                    ...
        ],
                            "traffic_speed_entry": [],
        "via_waypoint": []
    }
],
"overview_polyline": {
    "points": "..."
},
"summary": "Reuterallee and Heidelberger Landstrasse",
"warnings": [],
```

```
            "waypoint_order": [
                0
            ]
        }
    ],
    "status": "OK"
}
```


A.6 Distance Matrix API Sample

The below sample request yield accurate travel distance and duration values from single/multiple source(s) to single/multiple destination(s) in a single URL request. The corresponding source(s) and destination(s) coordinates are fed to the URL.

- Source coordinates: 49.81435569634197,8.637055512872708 (Uhlandstraße 12, 64297 Darmstadt, Germany)
- Destination coordinates:
  - 49.802319205278636,8.598859153442401 (Kirchstraße 16, 64319 Pfungstadt, Germany)
  - 49.874294322646286,8.630791981514493 (Poststraße 4-6, 64293 Darmstadt, Germany)

Sample request URL:

```
https://maps.googleapis.com/maps/api/distancematrix/json?origins
   =49.81435569634197,8.637055512872708&destinations
   =49.802319205278636,8.598859153442401|49.874294322646286,8.630791981514493&
   key=<licensed key string>
```

Sample response:


```
{
    "destination_addresses": [
        "Kirchstraße 16, 64319 Pfungstadt, Germany",
        "Poststraße 4-6, 64293 Darmstadt, Germany"
    ],
    "origin_addresses": [
        "Uhlandstraße 12, 64297 Darmstadt, Germany"
    ],
    "rows": [
        {
            "elements": [
                {
                    "distance": {
                        "text": "3.6 km",
                        "value": 3644
```

```
                    },
                    "duration": {
                        "text": "9 mins",
                        "value": 552
                    },
                    "status": "OK"
                },
                {
                    "distance": {
                        "text": "8.2 km",
                        "value": 8239
                    },
                    "duration": {
                        "text": "14 mins",
                        "value": 825
                    },
                    "status": "OK"
                }
            ]
        }
    ],
    "status": "OK"
}
```


A.7 Use case 1 - iCalendar I/O files

A.7.1 Input iCalendar plaintext


```
BEGIN:VCALENDAR
PRODID:-//Microsoft Corporation//Outlook 16.0 MIMEDIR//EN
VERSION:2.0
METHOD:PUBLISH
X-MS-OLK-FORCEINSPECTOROPEN:TRUE
BEGIN:VTIMEZONE
TZID:W. Europe Standard Time
BEGIN:STANDARD
DTSTART:16011028T030000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZOFFSETFROM:+0200
TZOFFSETTO:+0100
END:STANDARD
BEGIN:DAYLIGHT
DTSTART:16010325T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=3
TZOFFSETFROM:+0100
TZOFFSETTO:+0200
END:DAYLIGHT
END:VTIMEZONE
```

```
BEGIN:VEVENT
CLASS:PUBLIC
CREATED:20220218T160421Z
DESCRIPTION: \n
DTEND;TZID="W. Europe Standard Time":20220219T103000
DTSTAMP:20220216T112556Z
DTSTART;TZID="W. Europe Standard Time":20220219T100000
LAST-MODIFIED:20220218T160421Z
LOCATION:Max-Planck-Straße 2\, 64807 Dieburg
PRIORITY:5
SEQUENCE:0
SUMMARY;LANGUAGE=en-de:Appointment in Hochschule Darmstadt -
    Mediencampus
TRANSP:OPAQUE
UID:040000008200
    E00074C5B7101A82E00800000000B033B4AE2F23D801000000000000000
         0100000009095085365 6F4D4BA9DBF996315E1AE6
X-MICROSOFT-CDO-BUSYSTATUS:BUSY
X-MICROSOFT-CDO-IMPORTANCE:1
X-MICROSOFT-DISALLOW-COUNTER:FALSE
X-MS-OLK-AUTOFILLLOCATION:FALSE
X-MS-OLK-CONFTYPE:0
BEGIN:VALARM
TRIGGER:-PT15M
ACTION:DISPLAY
DESCRIPTION:Reminder
END:VALARM
END:VEVENT
END:VCALENDAR
```

A.7.2 New itinerary iCalendar plaintext

```
BEGIN:VCALENDAR
PRODID:-//Microsoft Corporation//Outlook 16.0 MIMEDIR//EN
VERSION:2.0
METHOD:REQUEST
X-MS-OLK-FORCEINSPECTOROPEN:TRUE
BEGIN:VTIMEZONE
TZID:W. Europe Standard Time
BEGIN:STANDARD
DTSTART:16011028T030000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZOFFSETFROM:+0200
TZOFFSETTO:+0100
END:STANDARD
BEGIN:DAYLIGHT
DTSTART:16010325T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=3
TZOFFSETFROM:+0100
```

```
TZOFFSETTO:+0200
END:DAYLIGHT
END:VTIMEZONE
BEGIN:VEVENT
ATTENDEE;CN="EV User from SAG";RSVP=TRUE:mailto:evuser123@sag.de
CLASS:PUBLIC
CREATED:20220218T160229Z
DTEND;TZID="W. Europe Standard Time":20220219T094500
DTSTAMP:20220218T092200Z
DTSTART;TZID="W. Europe Standard Time":20220219T092400
LAST-MODIFIED:20220218T160229Z
LOCATION:Max-Planck-Straße 2\, 64807 Dieburg
ORGANIZER;CN="CPO/EMSP/App Handler":mailto:usecase5@datenTanken.de
PRIORITY:5
SEQUENCE:0
SUMMARY;LANGUAGE=en-de:Your Driving Schedule
TRANSP:TRANSPARENT
UID:cdf79b07-56f1-40fa-98ab-4187f245a005
X-ALT-DESC;FMTTYPE=text/html:<html><body><p>Dear EV User\,<br><br>below
    is
        the travel itinerary for your appointment scheduled on
            2022-02-19T10:00 at
         Max-Planck-Straße 2\, 64807 Dieburg\, Germany.<br>Source: Home
             .<br>Desti
        nation: Max-Planck-Straße 2\, 64807 Dieburg\, Germany.<br><a
            href=https:/
        /www.google.com/maps/dir
            /49.8561992\,8.6454676/49.90162\,8.85516>Direction
        s</a><br>Note: Your vehicle has sufficient charge to reach
            destination wit
        hout any intermediate charge stops.<br>Total travel distance:
            19\,8 km<br>
        Total travel duration: 21 mins<br></p><div></div></body></html>
X-MICROSOFT-CDO-BUSYSTATUS:WORKINGELSEWHERE
X-MICROSOFT-CDO-IMPORTANCE:1
X-MICROSOFT-CDO-INTENDEDSTATUS:WORKINGELSEWHERE
X-MICROSOFT-DISALLOW-COUNTER:FALSE
X-MS-OLK-CONFTYPE:0
BEGIN:VALARM
TRIGGER:-PT15M
ACTION:DISPLAY
DESCRIPTION:Reminder
END:VALARM
END:VEVENT
END:VCALENDAR
```

A.8 Use case 2 - iCalendar I/O files

A.8.1 Input iCalendar plaintext

```
BEGIN:VCALENDAR
PRODID:-//Microsoft Corporation//Outlook 16.0 MIMEDIR//EN
VERSION:2.0
METHOD:PUBLISH
X-MS-OLK-FORCEINSPECTOROPEN:TRUE
BEGIN:VTIMEZONE
TZID:W. Europe Standard Time
BEGIN:STANDARD
DTSTART:16011028T030000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZOFFSETFROM:+0200
TZOFFSETTO:+0100
END:STANDARD
BEGIN:DAYLIGHT
DTSTART:16010325T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=3
TZOFFSETFROM:+0100
TZOFFSETTO:+0200
END:DAYLIGHT
END:VTIMEZONE
BEGIN:VEVENT
CLASS:PUBLIC
CREATED:20220216T112556Z
DESCRIPTION: \n
DTEND;TZID="W. Europe Standard Time":20220217T103000
DTSTAMP:20220216T112556Z
DTSTART;TZID="W. Europe Standard Time":20220217T100000
LAST-MODIFIED:20220216T112556Z
LOCATION:Max-Planck-Straße 2, 64807 Dieburg
PRIORITY:5
SEQUENCE:0
SUMMARY;LANGUAGE=en-de:Appointment in Hochschule Darmstadt -
   Mediencampus
TRANSP:OPAQUE
UID:040000008200
   E00074C5B7101A82E00800000000B033B4AE2F23D801000000000000000
        0100000009095085365 6F4D4BA9DBF996315E1AE6
X-MICROSOFT-CDO-BUSYSTATUS:BUSY
X-MICROSOFT-CDO-IMPORTANCE:1
X-MICROSOFT-DISALLOW-COUNTER:FALSE
X-MS-OLK-AUTOFILLLOCATION:FALSE
X-MS-OLK-CONFTYPE:0
BEGIN:VALARM
TRIGGER:-PT15M
ACTION:DISPLAY
DESCRIPTION:Reminder
```

```
END:VALARM
END:VEVENT
END:VCALENDAR
```

A.8.2 New itinerary iCalendar plaintext

```
BEGIN:VCALENDAR
METHOD:REQUEST
VERSION:2.0
PRODID:Driving schedule
BEGIN:VTIMEZONE
TZID:W. Europe Standard Time
BEGIN:STANDARD
DTSTART:16011028T030000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZOFFSETFROM:+0200
TZOFFSETTO:+0100
END:STANDARD
BEGIN:DAYLIGHT
DTSTART:16010325T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=3
TZOFFSETFROM:+0100
TZOFFSETTO:+0200
END:DAYLIGHT
END:VTIMEZONE
BEGIN:VEVENT
ORGANIZER;CN=CPO/EMSP/App Handler:MAILTO:usecase5@datenTanken.de
ATTENDEE;ROLE=REQ-PARTICIPANT;PARTSTAT=NEEDS-ACTION;RSVP=TRUE;CN=EV
   User from SAG:MAILTO:evuser123@sag.de
UID:8b6c8a4f-bd69-478c-bd52-463bcf9317cc
SUMMARY;LANGUAGE=en-DE:Your Driving Schedule
LOCATION;LANGUAGE=en-DE:Max-Planck-Straße 2, 64807 Dieburg
DTSTART;TZID=W. Europe Standard Time:20220217T084900
DTEND;TZID=W. Europe Standard Time:20220217T094500
X-ALT-DESC;FMTTYPE=text/html:<html><body><p>Dear EV User,<br><br>below
   is the travel itinerary for your appointment scheduled on
   2022-02-17T10:00 at Max-Planck-Straße 2, 64807 Dieburg, Germany.<br
   >Source: SoftwareAG Headquarters.<br>Destination: Max-Planck-Straße
    2, 64807 Dieburg, Germany.<br><a href=https://www.google.com/maps/
   dir/49.81435569634197,8.637055512872708/49.8296696,
   8.702313799999956/49.8978119,8.836467999999968/49.90162,8.85516>
   Directions</a><br>Total travel distance: 23,5 km<br>Total travel
   duration including recharging and authentication overhead at
   charging stations: 56 mins<br></p><div><ul><li>Leg 1<ul><li>
   SoftwareAG Headquarters ===>> Bahnhofstraße</li><li>Travel duration
   : 13 mins</li><li>Travel distance: 6.8 km</li><li>Charging duration
   : 5 mins</li><li>Charging Station Details:</li><ul><li>Name:
   Bahnhofstraße</li><li>Address: Bahnhofstraße 68, 64367 Mühltal</li
   ><li>Connector profile:</li><ul><li>Power type: AC_3_PHASE</li><li>
```

```
    Socket power: 22 kW</li></ul><li>Approximate Energy consumption:
    1,92 kWh</li><li>Charging timeslot to be reserved! Start time:
    2022-02-17T09:02 End time: 2022-02-17T09:12</li></ul></ul></li><li>
    Leg 2<ul><li>Bahnhofstraße ===>> An der Brückenmühle</li><li>Travel
     duration: 21 mins</li><li>Travel distance: 14.8 km</li><li>
    Charging duration: 2 mins</li><li>Charging Station Details:</li><ul
    ><li>Name: An der Brückenmühle</li><li>Address: An der Brückenmühle
     2, 64807 Dieburg</li><li>Connector profile:</li><ul><li>Power type
    : AC_3_PHASE</li><li>Socket power: 22 kW</li></ul><li>Approximate
    Energy consumption: 0,86 kWh</li><li>Charging timeslot to be
    reserved! Start time: 2022-02-17T09:33 End time: 2022-02-17T09:40</
    li></ul></ul></li><li>Leg 3<ul><li>An der Brückenmühle ===>> Max-
    Planck-Straße 2, 64807 Dieburg, Germany</li><li>Travel duration: 4
    mins</li><li>Travel distance: 1.9 km</li></ul></li></ul></div></
    body></html>
X-MICROSOFT-CDO-BUSYSTATUS:WORKINGELSEWHERE
BEGIN:VALARM
DESCRIPTION:REMINDER
TRIGGER;RELATED=START:-PT15M
ACTION:DISPLAY
END:VALARM
END:VEVENT
END:VCALENDAR
```

A.9 Use case 3 - iCalendar I/O files

A.9.1 Input iCalendar plaintext

```
BEGIN:VCALENDAR
PRODID:-//Microsoft Corporation//Outlook 16.0 MIMEDIR//EN
VERSION:2.0
METHOD:PUBLISH
X-MS-OLK-FORCEINSPECTOROPEN:TRUE
BEGIN:VTIMEZONE
TZID:W. Europe Standard Time
BEGIN:STANDARD
DTSTART:16011028T030000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZOFFSETFROM:+0200
TZOFFSETTO:+0100
END:STANDARD
BEGIN:DAYLIGHT
DTSTART:16010325T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=3
TZOFFSETFROM:+0100
TZOFFSETTO:+0200
END:DAYLIGHT
END:VTIMEZONE
```

```
BEGIN:VEVENT
CLASS:PUBLIC
CREATED:20220219T172914Z
DESCRIPTION: \n
DTEND;TZID="W. Europe Standard Time":20220220T210000
DTSTAMP:20220219T172914Z
DTSTART;TZID="W. Europe Standard Time":20220220T200000
LAST-MODIFIED:20220219T172914Z
LOCATION:Konrad-Zuse-Straße 1, 56075 Koblenz
PRIORITY:5
SEQUENCE:0
SUMMARY;LANGUAGE=en-de:Meeting at Koblenz University of Applied
    Sciences
TRANSP:OPAQUE
UID:040000008200
    E00074C5B7101A82E00800000000E0416376BE25D801000000000000000
        010000000CBFF627C5430FB488E048BEB2F482387
        01000000090950853656F4D4BA9DBF996315E1AE6
X-MICROSOFT-CDO-BUSYSTATUS:BUSY
X-MICROSOFT-CDO-IMPORTANCE:1
X-MICROSOFT-DISALLOW-COUNTER:FALSE
X-MS-OLK-AUTOFILLLOCATION:FALSE
X-MS-OLK-CONFTYPE:0
BEGIN:VALARM
TRIGGER:-PT15M
ACTION:DISPLAY
DESCRIPTION:Reminder
END:VALARM
END:VEVENT
END:VCALENDAR
```

A.9.2 New itinerary iCalendar plaintext

```
BEGIN:VCALENDAR
METHOD:REQUEST
VERSION:2.0
PRODID:Driving schedule
BEGIN:VTIMEZONE
TZID:W. Europe Standard Time
BEGIN:STANDARD
DTSTART:16011028T030000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=10
TZOFFSETFROM:+0200
TZOFFSETTO:+0100
END:STANDARD
BEGIN:DAYLIGHT
DTSTART:16010325T020000
RRULE:FREQ=YEARLY;BYDAY=-1SU;BYMONTH=3
TZOFFSETFROM:+0100
```

```
TZOFFSETTO:+0200
END:DAYLIGHT
END:VTIMEZONE
BEGIN:VEVENT
ORGANIZER;CN=CPO/EMSP/App Handler:MAILTO:usecase5@datenTanken.de
ATTENDEE;ROLE=REQ-PARTICIPANT;PARTSTAT=NEEDS-ACTION;RSVP=TRUE;CN=EV
   User from SAG:MAILTO:evuser123@sag.de
UID:c485a188-87c5-4671-b3d5-b8ec806efe71
SUMMARY;LANGUAGE=en-DE:Your Driving Schedule
LOCATION;LANGUAGE=en-DE:Konrad-Zuse-Straße 1, 56075 Koblenz
DTSTART;TZID=W. Europe Standard Time:20220220T165500
DTEND;TZID=W. Europe Standard Time:20220220T194500
X-ALT-DESC;FMTTYPE=text/html:<html><body><p>Dear EV User,<br><br>below
   is the travel itinerary for your appointment scheduled on
   2022-02-20T20:00 at Konrad-Zuse-Straße 1, 56075 Koblenz, Germany.<
   br>Source: SoftwareAG Headquarters.<br>Destination: Konrad-Zuse-
   Straße 1, 56075 Koblenz, Germany.<br><a href=https://www.google.com
   /maps/dir
   /49.81435569634197,8.637055512872708/50.028491870893205,8.128391501782962/50.3360
   Directions</a><br>Total travel distance: 134,7 km<br>Total travel
   duration including recharging and authentication overhead at
   charging stations: 2 hours 50 mins<br></p><div><ul><li>Leg 1<ul><li
   >SoftwareAG Headquarters ===>> Maschinenbau Bsullak</li><li>Travel
   duration: 50 mins</li><li>Travel distance: 64.1 km</li><li>Charging
    duration: 43 mins</li><li>Charging Station Details:</li><ul><li>
   Name: Maschinenbau Bsullak</li><li>Address: Rohrbergstraße 11,
   65343 Eltville am Rhein</li><li>Connector profile:</li><ul><li>
   Power type: AC_3_PHASE</li><li>Socket power: 22 kW</li></ul><li>
   Approximate Energy consumption: 14,20 kWh</li><li>Charging timeslot
    to be reserved! Start time: 2022-02-20T17:45 End time: 2022-02-20
   T18:33</li></ul></ul></li><li>Leg 2<ul><li>Maschinenbau Bsullak
   ===>> Konrad-Zuse-Straße 1, 56075 Koblenz, Germany</li><li>Travel
   duration: 1 hour 12 mins</li><li>Travel distance: 70.6 km</li></ul
   ></li></ul></div></body></html>
X-MICROSOFT-CDO-BUSYSTATUS:WORKINGELSEWHERE
BEGIN:VALARM
DESCRIPTION:REMINDER
TRIGGER;RELATED=START:-PT15M
ACTION:DISPLAY
END:VALARM
END:VEVENT
END:VCALENDAR
```

Bibliography

[1] *Assessment of Future Vehicle Transportation Options and Their Impact on the Electric Grid*. https://netl.doe.gov/sites/default/files/Smartgrid/repVehicle_ElecGridImpact.pdf, . – (Accessed on 02/07/2022)

[2] *Bellman–Ford algorithm - Wikipedia*. https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm, . – (Accessed on 01/24/2022)

[3] *Charging station map for electric cars | Chargemap*. https://chargemap.com/map, . – (Accessed on 01/24/2022)

[4] *Charging stations directory | GoingElectric.de*. https://www.goingelectric.de/stromtankstellen/, . – (Accessed on 01/23/2022)

[5] *Circle & Coordinate Geometry - mathscard online*. https://www.mathscard.co.uk/online/circle-coordinate-geometry/, . – (Accessed on 01/11/2022)

[6] *DE.DIGITAL - Daten Tanken - intelligente Ladeinfrastruktur in Dresden*. https://www.de.digital/DIGITAL/Redaktion/DE/Stadt.Land.Digital/Beitraege/WettbewerbStadtLandDigital/daten-tanken-intelligente-ladeinfrastruktur-in-dresden.html, . – (Accessed on 02/24/2022)

[7] *Dokumentation der Stromtankstellen API | GoingElectric.de*. https://www.goingelectric.de/stromtankstellen/api/docs/, . – (Accessed on 01/23/2022)

[8] *Geocoding API requests and responses | Google Developers*. https://developers.google.com/maps/documentation/geocoding/requests-geocoding, . – (Accessed on 01/11/2022)

[9] *Getting directions through the Directions API | Google Developers*. https://developers.google.com/maps/documentation/directions/get-directions, . – (Accessed on 01/11/2022)

[10] *Google Maps Platform Documentation | Google Developers.* https://developers.google.com/maps/documentation, . – (Accessed on 01/11/2022)

[11] *Install MongoDB Community Edition — MongoDB Manual.* https://docs.mongodb.com/manual/administration/install-community/, . – (Accessed on 01/26/2022)

[12] *Load Management Requirements in Electric Ear Charging | EVSE.* https://evse.com.au/blog/load-management-electric-car-charging/#:~:text=Remember%20an%20EV%20charging%20station,commercial%20grade%20air%20conditioning%20system., . – (Accessed on 02/03/2022)

[13] *Open Charge Map - API Documentation.* https://openchargemap.org/site/develop/api#/operations/get-poi, . – (Accessed on 01/11/2022)

[14] *Open Charge Map - Electric Vehicle Charging Locations Near You.* https://map.openchargemap.io/#/search, . – (Accessed on 01/24/2022)

[15] *Overview | Distance Matrix API | Google Developers.* https://developers.google.com/maps/documentation/distance-matrix/overview, . – (Accessed on 01/11/2022)

[16] *Power factor - Wikipedia.* https://en.wikipedia.org/wiki/Power_factor, . – (Accessed on 02/03/2022)

[17] *What are the different methods of charging an electric vehicle? | Kia Go Electric.* https://www.kia.com/eu/about-kia/experience-kia/technology/electrification/charging-methods-for-electric-cars/, . – (Accessed on 02/14/2022)

[18] *Open Charge Point Interface/Open Charge Point Protocol (OCPI/OCPP) - Funding & tenders.* https://ec.europa.eu/info/funding-tenders/opportunities/portal/screen/opportunities/horizon-results-platform/20914;needList=10,11,12, 02 2020. – (Accessed on 01/13/2022)

[19] *GitHub - ocpi/ocpi: The Open Charge Point Interface (OCPI).* https://github.com/ocpi/ocpi, 07 2021. – (Accessed on 01/13/2022)

[20] *MongoDB Java Driver Overview (driver-sync 4.4.0 API).* https://docs.mongodb.com/drivers/java/sync/current/quick-start/, 2021. – (Accessed on 01/10/2022)

[21] *MongoDB Manual*. https://docs.mongodb.com/manual/, 2021. – (Accessed on 01/10/2022)

[22] *ocpi/mod_locations.asciidoc at master · ocpi/ocpi · GitHub*. https://github.com/ocpi/ocpi/blob/master/mod_locations.asciidoc, 09 2021. – (Accessed on 01/13/2022)

[23] *Pythagorean theorem - Wikipedia*. https://en.wikipedia.org/wiki/Pythagorean_theorem, 2021. – (Accessed on 01/12/2022)

[24] *Small circle - Circle of a sphere - Wikipedia*. https://en.wikipedia.org/wiki/Circle_of_a_sphere#/media/File:Small_circle.svg, 03 2021. – (Accessed on 01/13/2022)

[25] *JSON*. https://www.json.org/json-en.html, 2022. – (Accessed on 01/10/2022)

[26] Artmeier, Andreas ; Haselmayr, Julian ; Leucker, Martin ; Sachenbacher, Martin: The Shortest Path Problem Revisited: Optimal Routing for Electric Vehicles, 2010. – ISBN 978–3–642–16110–0, S. 309–316

[27] Basmadjian, Robert ; Kirpes, Benedikt ; Mrkos, Jan ; Cuchý, Marek: A Reference Architecture for Interoperable Reservation Systems in Electric Vehicle Charging. In: *Smart Cities* 3 (2020), Nr. 4, 1405–1427. http://dx.doi.org/10.3390/smartcities3040067. – DOI 10.3390/smartcities3040067. – ISSN 2624–6511

[28] Bedogni, Luca ; Bononi, Luciano ; D'Elia, Alfredo ; Di Felice, Marco ; Di Nicola, Marco ; Cinotti, Tullio S.: Driving without anxiety: A route planner service with range prediction for the electric vehicles. In: *2014 International Conference on Connected Vehicles and Expo (ICCVE)*, 2014, S. 199–206

[29] Bedogni, Luca ; Bononi, Luciano ; Di Felice, Marco ; D'Elia, Alfredo ; Cinotti, Tullio S.: A Route Planner Service with Recharging Reservation: Electric Itinerary with a Click. In: *IEEE Intelligent Transportation Systems Magazine* 8 (2016), Nr. 3, S. 75–84. http://dx.doi.org/10.1109/MITS.2016.2573418. – DOI 10.1109/MITS.2016.2573418

[30] Chale-Gongora, Hugo G. ; Weck, Olivier de ; Doufene, Abdelkrim ; Ishimatsu, Takuto ; Krob, Daniel: Planning an itinerary for an electric vehicle. In: *2014 IEEE International Energy Conference (ENERGYCON)*, 2014, S. 1385–1391

[31] Falchetta, Giacomo ; Noussan, Michel: Electric vehicle charging network in Europe: An accessibility and deployment trends analysis. In: *Transportation Research Part D: Transport and Environment* 94 (2021), 102813. http://dx.doi.org/https://doi.org/10.1016/j.trd.2021.102813. – DOI https://doi.org/10.1016/j.trd.2021.102813. – ISSN 1361–9209

[32] Haidar, Ahmed M. A. ; Muttaqi, Kashem M.: Behavioral Characterization of Electric Vehicle Charging Loads in a Distribution Power Grid Through Modeling of Battery Chargers. In: *IEEE Transactions on Industry Applications* 52 (2016), Nr. 1, S. 483–492. http://dx.doi.org/10.1109/TIA.2015.2483705. – DOI 10.1109/TIA.2015.2483705

[33] Hoch, Nicklas ; Zemmer, Kevin ; Werther, Bernd ; Siegwart, Roland Y.: Electric vehicle travel optimization-customer satisfaction despite resource constraints. In: *2012 IEEE Intelligent Vehicles Symposium*, 2012, S. 172–177

[34] Kettle, Simon: *Distance on a sphere: The Haversine Formula - Esri Community*. https://community.esri.com/t5/coordinate-reference-systems-blog/distance-on-a-sphere-the-haversine-formula/ba-p/902128, 2017 . – (Accessed on 01/10/2022)

[35] Kettle, Simon: *Geodesic distances - Esri Community*. https://community.esri.com/t5/coordinate-reference-systems-blog/geodesic-distances-how-long-is-that-line-again/ba-p/902188, 2014. – (Accessed on 01/10/2022)

[36] Mies, Jerome J. ; Helmus, Jurjen R. ; Hoed, Robert Van d.: Estimating the Charging Profile of Individual Charge Sessions of Electric Vehicles in The Netherlands. In: *World Electric Vehicle Journal* 9 (2018), Nr. 2. http://dx.doi.org/10.3390/wevj9020017. – DOI 10.3390/wevj9020017. – ISSN 2032–6653

[37] Orcioni, Simone ; Conti, Massimo: EV Smart Charging with Advance Reservation Extension to the OCPP Standard. In: *Energies* 13 (2020), Nr. 12. http://dx.doi.org/10.3390/en13123263. – DOI 10.3390/en13123263. – ISSN 1996–1073

[38] Sryheni, Said: *Difference Between BFS and Dijkstra's Algorithms*. https://www.baeldung.com/cs/graph-algorithms-bfs-dijkstra, 2021. – (Accessed on 01/12/2022)

[39] Yong, Jia Y. ; Ramachandaramurthy, Vigna K. ; Tan, Kang ; Nadarajah, Mithulananthan: A review on the state-of-the-art technologies of electric

vehicle, its impacts and prospects. In: *Renewable and Sustainable Energy Reviews* 49 (2015), 04. http://dx.doi.org/10.1016/j.rser.2015.04.130. – DOI 10.1016/j.rser.2015.04.130