MSE806.2 Group Project
Electric vehicle route planner

Team Member:
Guangyao Li
Jiajun Liu
Changsheng Tian

Date: July 14, 2024

Table of Contents

Introduction

Background: The issue of global warming is getting worse. People want to contribute to solving the problem of global warming. Electric vehicles with less carbon emission have become a popular choice when they consider buying a vehicle.

Our group project is about route planning for electric vehicles (EVs). The authenticated users input the departure location and destination to get the optimized route. At the same time, if the EV does not have sufficient electricity to arrive at the destination, the router planner will plan the route to the closest charging station. We will collect behavior data for all users, including the addresses entered by users and each click on charging stations. For logged-in users, we will also collect data on the vehicles they add and the charging stations they bookmark. These four types of data will be stored in Amazon DynamoDB. Additionally, we have established a backend management system to analyze the collected data.

To finish this assignment, we create three projects, including ev-route-planner-client, ev-route-planner-admin, and ev-route-planner-api. The address of the ev-route-planner-client is https://github.com/alexlili/ev-route-planner-client. The address of the ev-route-planner-admin is https://github.com/alexlili/ev-route-planner-admin. The address of the ev-route-planner-api is https://github.com/alexlili/ev-route-planner-api. The ev-route-planner-client is open to the public. All projects have been deployed on the AWS cloud via Amplify Hosting. The website of the ev-route-planner-client is https://dev.d1pghmqcjpsb6.amplifyapp.com/. The website of the ev-route-planner-admin is https://dev.d2uxghi9675sr8.amplifyapp.com/. Please click these two websites to check the details and take an overview.

Project Target

To create a website including admin website and client website while using the OpenChargeMap API and the OpenMap API. These software systems will be deployed on AWS, using Amplify Hosting. The administrator can log in to the admin website to check and manage user's information, generating some reports about routes that have been finished by all users. In terms of the client website named "route planner", users can use it to find the shortest route, inputting some parameters like departure location, arrival location, and the information of car battery. Then, our project can be used around the world, and the addresses around the world can be identified on the map.

Project Description

- **Methodology**: The shortest route is generated by the OpenStreetMap API. Then, the system will check whether EVs can arrive at the destination based on the remaining electricity level and route distance. If EVs cannot arrive at the destination due to the current battery situation. Then, the system will recommend the closest charging station with the OpenChargeMap API, if there is insufficient electricity level to arrive at the charging station. The system will notify it.

- **Scope**:
  EV route planner: The login user only needs to input the departure location and destination to utilize the route planner. The EV information will be directly obtained from the EV database. The user without login needs to input the distance the EV operates if fully charged and the current battery situation apart from the departure location and destination.

  User information management: The registered user can input information about more than one EV, such as the distance the EV can operate while fully charged.
- **Deliverables**:
  We will categorize the project into various phases, from initial planning to final deployment.
  1. Planning and Analysis phase:
     Planning the project, writing a document outlining the project scope, timeline, resources, stakeholders, and a detailed list of functional requirements based on user needs.
  2. Design phase:
     design the user interface, database design, and API specifications.
  3. Development phase:
     well-documented and modular source code for the route planning algorithms, user interfaces, and backend services.
  4. Testing phase:
     Detailed test cases and automated test scripts for functional, and performance testing.
  5. Documentation phase:
     Step-by-step instructions for deploying the route planner. Automated scripts for setting up and configuring the production environment and documentation detailing the features.
- **Timeline**: A schedule of key milestones and deadlines.

Project Team

- **Team Members**: Guangyao Li, Jiajun Liu, Changsheng Tian
- **Responsibilities**:

  Team leader: Guangyao Li, responsible for designing the whole architecture of this project, creating the framework for two software systems and the API project, developing key components, and deploying projects. The detail of the whole architecture is displayed in the part named ‘System Architecture’. To be specific, what I have done in this assignment contains as below: the initialization of three projects(client, admin, and API), integrating the amplify into the client, integrating the amplify into admin, implementing the data model in the amplify studio according to the ERD designed by Jiajun Liu, all API serving for client and admin, developing all pages and the components of admin, the deploy of three projects, and the planning of development environment.

Frontend and Backend developer: Jiajun Liu, responsible for designing ERD, and developing functional components and pages, including the home page of the client, the function of search address, the display of charging stations, the detail of every charging station, adding car, the function of collecting charginng station, the component of planning route, the integration of the login UI, the study of OpenChargerAPI, the study of OpenstreetMap API and all components of client.

Algorithms specialist: Changsheng Tian, responsible for studying the shortest route algorithm and implementing it using javascript language.

Technology

React framework: Building the front-end websites.
AWS Amplify: Initializing the project in a short time, providing authentication, API, data storage, hosting, and publishing.
OpenChargeMap API: Providing data about EV charging stations.
OpenStreetMap API: Providing detailed data about map and routing capabilities.

System Architecture

Frontend Development: REACT

React.js: Used to build the user interface, the component-based development approach makes the code more maintainable and reusable.

React Router: Manages navigation between different pages and components.

Redux: Used for global state management, especially when the application becomes complex. In our project, we use it to manage the login state.

Backend Services: AWS Amplify

Amplify CLI: Used to set up and configure Amplify projects, supports command-line operations to initialize projects, and add features (such as authentication, API, and storage). In our projects, we use aws-authentication and the functionality that can generate API automatically.

Amplify Libraries: Provides integration with AWS services, simplifies communication with the backend, and has built-in support for authentication, authorization, storage, data synchronization, and more. In this assignment, we build three projects: one for client, one for admin, and the other for API. The client application and Admin application are supported by the API application.

Hosting: Amplify also provides easy-to-use hosting services for quick deployment and hosting of front-end applications.

Database: AWS AppSync (GraphQL API)

AWS AppSync: Used to build scalable GraphQL APIs, supports real-time updates and offline operations.

GraphQL Schema: Defines the data models and query, mutation operations, allowing the frontend to flexibly fetch the required data.

Resolvers: Connects GraphQL queries to backend data sources, and can use DynamoDB.

Data Security: AWS Authority

AWS Cognito: Used for user authentication and authorization, supports user registration, login, multi-factor authentication (MFA), etc.

Amplify Auth: Integrated with Cognito, simplifies the authentication process, and provides easy-to-use APIs and UI components.

Route Planning: OpenChargeMap API and OpenStreetMap API

OpenChargeMap API: Provides information about charging stations, which is used for route planning to find and navigate to charging points.

OpenStreetMap API: Provides map data and services, used in conjunction with OpenChargeMap for route planning.

Address Input and Geocoding: Google Maps API

Google Maps API Key: Obtained to enable fuzzy query matching for global addresses when inputting addresses.

Geocoding: Uses Google Maps API to quickly find the latitude and longitude information of input addresses.

Workflow

Initialize Project: Use Amplify CLI to initialize the project and configure the required services such as authentication and API. In this assignment, we initialize three projects, including ev-route-planner-client, ev-route-planner-admin and ev-route-planner-api.

Define GraphQL Schema: Define the GraphQL schema in AppSync, including data types and operations.

Create React Components: Use React to create UI components and design the user interaction interface.

Integrate Amplify Libraries: Introduce Amplify Libraries in the React application to interact with backend services.

Authentication and Authorization: Configure Cognito and use Amplify Auth in the front end to implement user authentication and authorization.

Integrate APIs for Route Planning: Use OpenChargeMap API and OpenStreetMap API to plan routes and find charging stations.

Implement Address Geocoding: Use Google Maps API to enable fuzzy address search and geocoding for latitude and longitude information.

Deploy Application: Use Amplify Hosting to deploy and host the frontend application.

Advantages

Rapid Development: Amplify provides many out-of-the-box features, accelerating the development process.

Flexibility: GraphQL offers flexible data querying capabilities, allowing the front end to fetch exactly the needed data.

Security: Ensures application security through services like IAM and Cognito.

Comprehensive Route Planning: By integrating OpenChargeMap and OpenStreetMap APIs, the application can provide detailed route planning for charging stations.

Accurate Geocoding: Google Maps API allows for accurate and quick retrieval of latitude and longitude information for addresses, enhancing the user experience with fuzzy search capabilities.

Scalability: AWS cloud services provide high scalability, allowing dynamic adjustment of resources according to needs.

This architecture combines the efficient frontend development capabilities of React with the powerful backend services of AWS Amplify, and the additional functionalities provided by OpenChargeMap, OpenStreetMap, and Google Maps APIs, ensuring high performance, security, and comprehensive route planning during the development, deployment, and operation of the application.

Functional Requirements

- The website can automatically identify the user's position after opening the website

- User sign-up and sign-in:
    - The user can create an account with an email and password

    - The user can sign in with the created account

- User Dashboard:
    - After the sign-in, the user can add their frequent route plan to the user dashboard

    - After the sign-in, the user can add EV information to the user dashboard, such as vehicle brand, model, the battery autonomy of the EV, the distance covered under the fully-charged condition, and the type of charging port.

- Admin Dashboard:
    - The Admin can sign in with the authorized account to access the information about users and generate the graph or data analysis tools to get insight from users.

- Charging station searching
    - When the user opens the website, they can input the address or city to search the charging stations around the inputted address or city. By default, the search results will be 30 kilometers around the inputted address or city. This distance can be set by users in the future.

    - When the user clicks on one of the charging stations, they can learn about its information, like address, status: operational or closed, price, and general comments.

    - At the same time, the user can collect the charging stations to the user dashboard after signing in.

- User Input:
    - The user inputs the departure location and destination.

    - The user inputs the battery autonomy, which means the distance the EV can cover under the fully-charging condition

    - The user inputs the current battery condition (the percentage of battery left).

- Route Calculation:
    - Calculate and plan the route from departure to destination.
    - Evaluate the potential demand for charging stations based on the current battery condition and battery autonomy. Our project will by default recommend the user charging stations within the distance the current battery level can cover along the planned route if the EV can not reach the destination. Then, if another charging station is needed, another charging station will be recommended until being able to get the endpoint.

- Integrating charging station:
    - If the charging is needed, fetch and identify the nearby charging stations along the planned route.

    - The route from the nearby charging station to the destination is planned.

- Output display:
    - Display the planned route on the map

- Display the map spots, like charging stations, departure spots, and destination

Database

Electric Vehicle List: VehicleID, Car, model, brand, battery situation (odometer if fully charged), type of charging port.
Customer List: CustomerID, Email, Password, VehicleID
Admin User List: AdminID, Email, Password
User Favorite Charger Station List: StationAddress, UserId
User Search Address List: Address, ListId
User Click Charger Station List: Address, ListId

Efficiency Analysis of Intelligent Transportation Systems

1. Route Optimization

   KPI: Planned route travel time and distance

   The system calculates the most efficient route by considering the electric vehicle's range and charging stations' location. The route planner minimizes detours and (charging times) and minimizes the travel time and distance

   The area for Improvement: The lack of real-time traffic data

   Solution: By integrating the OpenAPIs with the real-time traffic data with the route planner, the real-time traffic flows and a more accurate estimated arrival time can be calculated. This will increase the system's responsiveness to the shifting traffic circumstances.

2. Charging station availability

   KPI: The charging station spots availability

   By evaluating the availability of spots for charging stations, the project can recommend the charging station with available spots and calculate the travel time and distance.

   The area for improvement: The lack of real-time data about spot of availability of charging stations

Solution: By collaborating with the charging station company to get the real-time availability of charging stations, the route planner can recommend the closest charging stations with available spots to reduce the potential waiting time.

3. Electricity charging level and time

   KPI: The charging time

   The project will by default recommend the users charging stations within the distance the current battery level can cover along the planned route if the EV can not reach the destination. Then, the EV gets charged at charging stations until the battery level reaches the current battery level inputted by the user. For example, if the distance from the start point to the end point is 500 kilometers, and the current battery level can only cover 100 kilometers, the charging station within 100 kilometers will be shown along the planned route. Then, if another charging station is needed, another charging station within 100 kilometers will be provided until the EV arrives at the endpoint.

   The area for improvement: The lack of charging time optimization and choice

   Solution: The user can input the amount of electricity charged they want. The algorithm can be used to recommend the charging solution for users with the optimized charging time and the optimized overall time spent between the start point and the endpoint.

4. User Experience:

   KPI: User Satisfaction and Ease of Use

   To improve customer satisfaction, the interactive map and user-friendly interface are designed to increase the ease of use and enable users to access the information they need

   The area for improvement: Gathering user feedback

   Solution: Create a feedback mechanism to gather users' experiences and any suggestions for improvements to satisfy user’s needs better.

Integration of Emerging Technologies

1. OpenCharing API Integration

The data about charging stations: OpenCharing API provides extensive data about electric vehicle charging stations worldwide, such as charging station locations, availability, types, and electric vehicle compatibility.

Real-data access: OpenCharing API provides real-time data about electric vehicle charging stations for EV route planners to recommend the EV charging station based on the EV's range and travel distance.

OpenCharing API integration with EV route planners can help to provide more accurate planning routes for users based on real-time data about charging stations. Then, it brings convenience and reduces the potential latency for users to provide the optimized charging plan for users to have enough electricity level to arrive at their destination.

2. OpenStreetMap API with Leaflet Library

   The data about the map and locations: OpenStreetMap API provides extensive data about map spots and locations. Then, Leadlet Library is a JavaScript Library to supports displaying the route plan and user interaction

   Optimized route plan: The map information from OpenStreetMap API supports route planning to optimize the route plan for users. Then, Leaflet Library supports the route planner to get input from users about departure location and destination and display users the optimized planning route.

   Location identification: OpenStreetMap API supports the route planner to identify and show departure and destination map points to display the planning route.

   OpenStreetMap API supports route planners to plan the optimized route for users based on map information. Then, the Leaflet library offers users an interactive interface to get necessary inputs and display planning routes between departure points to destination map points.

System Reliability, Security, and Sustainability

System Reliability: The website smoothly provides the departure address and destination identification and the route plan, provides the charging stations along the planned route if needed, and provides the nearby charging station without disruption and bugs. This ensures a smooth user experience. Then, the error handling method is used to deal with the potential bug or API downtimes to support continuous and reliable operation.

System Security: Reliable Authentication through sign-in ensures only sign-in users can access information in their user dashboard. Only the admin with the authorized account can sign in to access and manage the user dashboard, this prevents the potential and unauthorized data access and data breach. However, to show the details of admin, we do not open the login modal. It means everyone can see the dashboard and the data we have collected. In our process of authentication, we use **Amplify Auth**. It provides easy-to-use methods for user sign-up, sign-in, and password recovery. It also supports authentication through social identity providers like Google, Facebook, and Amazon. We use **AWS IAM** to ensure that users and services have the minimum level of access necessary to perform their functions.

System Sustainability: The fewer charging stations and the lack of information about charging stations may cause some inconvenience to EV drivers. This website can provide EV drivers with real-time information about charging stations and plan their EV route to bring convenience to encourage more people to choose EV as their EV choice. As more people choose the EV, the amount of carbon emission produced by petrol vehicles can be reduced. This is great for the environment. At the same time, the optimized route plan can help EV drivers minimize their energy consumption to reduce the impact of EVs on the environment and optimize electricity consumption to reduce the charging cost.

Algorithm

In the realm of route planning and optimization, algorithms play a crucial role in determining the most efficient path from a start point to an endpoint. Two notable algorithms are Dijkstra's algorithm and the route planning algorithm designed specifically for electric vehicles (EVs), as Hybrid Algorithm (Dynamic Programming + Greedy).

Dijkstra's algorithm (Nitin Gupta, Kapil Mangla, Anand Kumar Jha, Md. Umar, 2016, 122-124), developed by Edsger W. Dijkstra in 1956, is a graph search algorithm that finds the shortest path between nodes in a graph. It is widely used in network routing and geographical mapping due to its efficiency and reliability.

Key Features of Dijkstra's Algorithm (Abhishek Goyal, Prateek Mogha, Rishabh Luthra, Ms. Neeti Sangwan, 2014, 15):

1. Initialization: Assigns an initial distance value of zero to the starting node and infinity to all other nodes. All nodes are marked as unvisited.

2. Iteration: Selects the unvisited node with the smallest known distance, calculates the tentative distances through this node to its neighbors, and updates the shortest path if a shorter path is found.

3. Completion: Repeats the iteration until all nodes have been visited or the shortest path to the destination node is determined.

Strengths:

- The algorithm is straightforward and easy to implement.
- With a time complexity of $O(V^2)$, or $O(E\ log(V)$ when using a priority queue, it efficiently finds the shortest path in graphs with non-negative weights.
- It is versatile and can be applied to various types of networks.

Limitations:

- It does not consider dynamic factors such as real-time traffic conditions or external constraints like vehicle battery life.
- The sole focus is on finding the shortest path, without regard to other considerations such as energy consumption or charging needs.

Hybrid Algorithm (Dynamic Programming + Greedy (Kurt Mehlhorn and Peter Sanders, 2007, 239-242))

The route planning algorithm is tailored for electric vehicles, considering factors unique to EVs, such as battery autonomy and the availability of charging stations.

Key Features of Hybrid Algorithm:

1. Initialization: Sets parameters for the car’s autonomy and battery usage.

2. Initial Route Calculation: Uses mapping services to determine the initial route from the start to the endpoint.

3. Battery Consideration: Calculates distances between route coordinates to check if the journey can be completed with the current battery charge.

4. Charging Station Integration: If the route exceeds the car’s autonomy, the algorithm identifies potential charging stations along the way by segmenting the route and searching within a specified radius around segment midpoints.

5. Dynamic Adjustment: Validates the feasibility of charging stations and adjusts the route dynamically to include necessary charging stops.

Strengths:

- It specifically addresses the needs of EVs, ensuring that routes are feasible given the constraints of battery life and charging infrastructure.
- It adjusts the route in real time based on the availability of charging stations and the vehicle's current battery status.
- It integrates multiple objectives, such as minimizing travel time and ensuring the vehicle can complete the journey without running out of charge.

Limitations:

- More complex to implement than Dijkstra’s algorithm due to the need for real-time data fetching and dynamic adjustments.
- Relies on mapping APIs and real-time data from charging station databases.

Comparison and Justification for the Specialized EV Algorithm

- While Dijkstra’s algorithm excels in finding the shortest path, it does not account for the practical limitations faced by EVs. The specialized EV route planning algorithm

aligns better with the objectives of ensuring the vehicle can complete the journey without running out of charge.
- The EV algorithm’s ability to dynamically integrate charging stations makes it more suitable for real-world electric vehicle scenarios. This adaptability is crucial for ensuring that the planned route remains feasible under varying conditions.
- By considering battery autonomy and the necessity for charging stops, the EV algorithm offers a more holistic solution. This approach reduces the risk of an EV running out of charge mid-journey, a critical concern not addressed by Dijkstra’s algorithm.
- Although the EV algorithm may involve more complex calculations and real-time data integration, its practical benefits outweigh these challenges. Ensuring a feasible route for EVs in real-world scenarios is paramount, making the additional complexity justifiable.

In summary, while Dijkstra’s algorithm is a powerful tool for general shortest-path problems, the specialized route planning algorithm for electric vehicles offers a more practical solution for EV route planning. Its ability to integrate charging stations dynamically and consider battery limitations makes it the preferred choice for applications involving electric vehicles. By addressing real-world constraints, the EV algorithm ensures that routes are not only optimal in terms of distance but also feasible and safe for electric vehicle journeys.

References

Abhishek Goyal, Prateek Mogha, Rishabh Luthra, Ms. Neeti Sangwan. (2014). PATH FINDING: A* OR DIJKSTRA’S? *indianjournals*, *2*(1), 15. https://www.indianjournals.com/ijor.aspx?target=ijor:ijie&volume=2&issue=1&article=001

Andreas Artmeier, Julian Haselmayr, Martin Leucker, Martin Sachenbacher. (2010). The Shortest Path Problem Revisited: Optimal Routing for Electric Vehicles. *researchgate*. https://www.researchgate.net/publication/221562703_The_Shortest_Path_Problem_Revisited_Optimal_Routing_for_Electric_Vehicles?enrichId=rgreq-ae8773049fb1b4e2c08975d7d4947f82-

XXX&enrichSource=Y292ZXJQYWdlOzIyMTU2MjcwMztBUzoxNjk1NDI4NTY3NDkw
NTZAMTQxNzQzMzU3NT

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). *Introduction to Algorithms, Fourth Edition*. MIT Press.

Kurt Mehlhorn and Peter Sanders. (2007). *Algorithms and Data Structures*. Kurt Mehlhorn and Peter Sanders.
https://users.dcc.uchile.cl/~voyanede/cc4102/mehlhorn-Sanders-Toolbox.pdf

Nitin Gupta, Kapil Mangla, Anand Kumar Jha, Md. Umar. (2016). Applying Dijkstra’s Algorithm in Routing Process. *International Journal of New Technology and Research*, 122-124.
https://www.academia.edu/download/50979673/IJNTR02050040.pdf