EleNA : Elevation Based Navigation
by
Team: Indentation&Semicolon

Members:

Sai Vineeth Kumar Dara
Rohith Siddhartha Reddy Bheemreddy
Tushita Singh
Rachana Ponagandla

Github repo:
https://github.com/rohithsiddhartha/Indentation-Semicolons

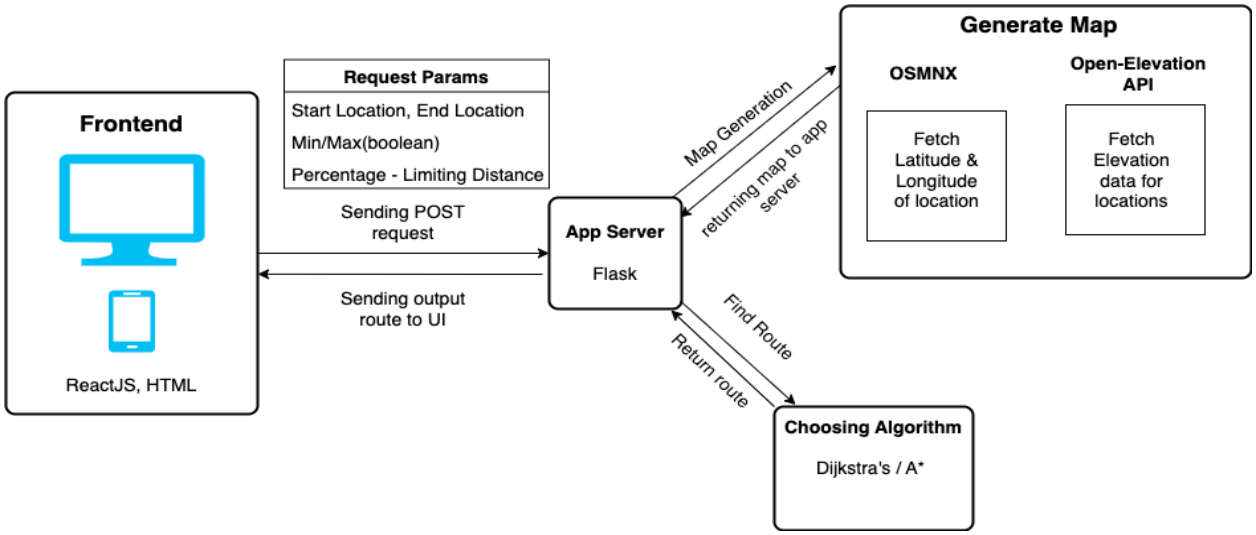
Problem Statement:

Elevation-based Navigation (EleNa) is an application that, given a start and end location, computes a route that maximizes or minimizes the elevation gain and limits the total distance between the locations to x% of the shortest path.

Maximizing the elevation gain could be useful to joggers/bikers who may be looking for an intense workout. On the other hand, minimizing the elevation gain could be useful for those who don't prefer steep climbs in the route.

Architecture:

The below is the architecture diagram of our implementation



Backend:

- App server receives the user's POST request containing source, destination, min/max(boolean) and shortest path percentage.
- Generates the Map using OSMNX(finds latitude and longitude of locations) and Open-Elevation API(provides elevation data of locations).
- App server receives the Map Information and passes it to a routing algorithm(Dijkstra's) to find the route.
- App server receives the route and sends the response to UI.

Tools used in Backend:
Flask, OSMNX, Open Elevation API

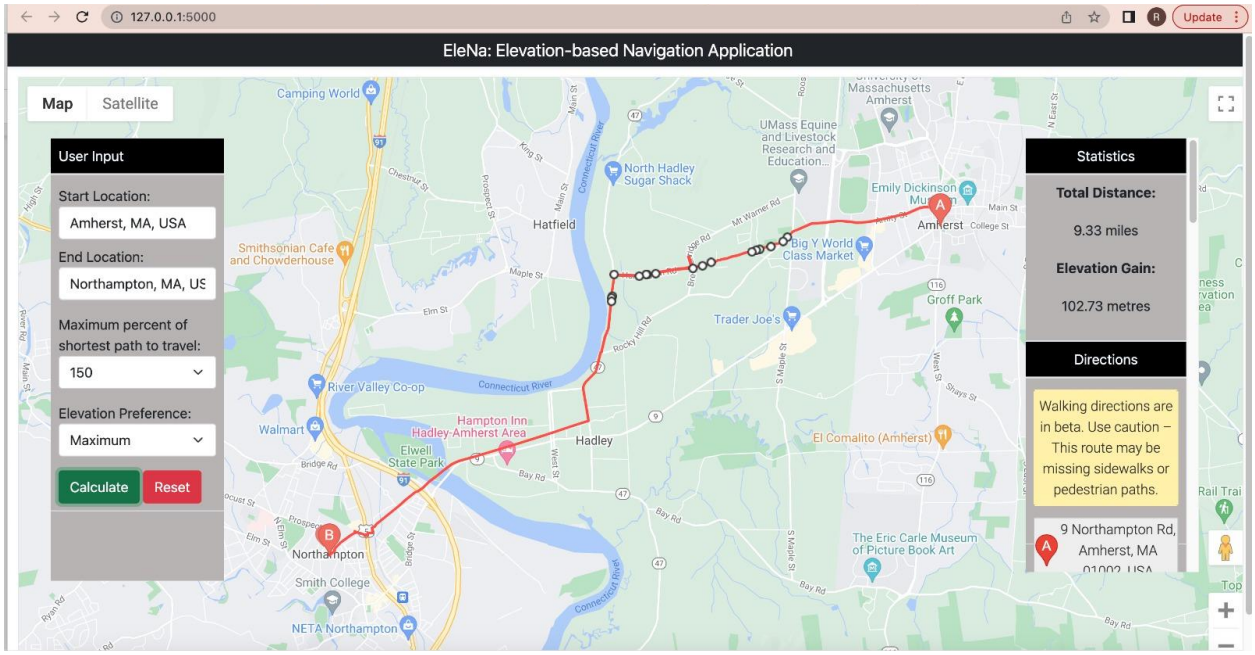
Frontend:

UI contains following components
Input Components:
- Source and destination address
- The shortest path percentage
- The max or min elevation button selection

Output Components:
- The Map Containing Output Route
- Total distance
- Elevation gain
- Directions

Tools used in frontend:
ReactJS, HTML, CSS, Bootstrap

User interface of the tool:



Evaluation:

Functional requirements:

- Given source, destination, elevation strategy and shortest distance deviation percentage, the tool should return the path satisfying the conditions.
- Algorithm: Dijkstra
- Display the route information like elevation gain, distance, directions.

Usability:

- The current implementation is valid for areas that are present in the US only.
- Users are shown the directions along with the path for better navigation.

Extensibility:

- The geographic location can be expanded to more regions by increasing the radius of the area under consideration.
- The observer pattern helps in smooth visualization of the UI making it more user friendly.

Testability:

- We have written unit test cases for verifying the correctness of class methods.

- We conducted thorough manual testing of the application for finding UI related bugs.
- Since we used the MVC architecture pattern, writing unit test cases was simple.

Best software practices used in the project:

- These are some of the best software practices we used in our project:
- Implemented MVC architecture.
- Avoided magic strings by declaring constant strings.
- Understandability: Documentation - we added comments to classes and methods to make the code more understandable. Also we added ReadMe and requirements file for easy code setup.
- Testability: Implemented unit test cases to test the tool. Also we conducted thorough manual testing.
- Appropriate naming conventions for better understanding of the code.
- While pushing the changes we also did manual review of the changes by confirming the correctness of changes from team mates.
- Conducted weekly meetings to keep everyone in the team updated regarding the project status.
- Implemented Observer design pattern: In our architecture, the model acts as observable and the view acts as observer.

Why choose our Elena tool:

Here are few interesting things we implemented thats sets us apart from other Elena systems:

- Instead of just showing the path, we are showing directions from the source to destination too.
- Our tool works for every location in USA and not just limited to a specific area (Eg.Amherst)
- Cache implementation - to make the tool more user friendly.
- Our project supports MVC architecture.
- System is tested using unit test cases and thorough manual testing.