SOFTWARE REQUIREMENTS SPECIFICATION

for

Elevation-based Navigation System (EleNa)

COMPSCI 520 Final Project

Prepared by: Mudit Chaudhary (32607978)
Pragyanta Dhal (33069605)
Ayushe Gangal (33018381

Contents

1 Introduction

1.1 Purpose

This document specifies the software requirements for Elevation-based Navigation System (EleNa). This project corresponds to the CS520 Final Project.

1.2 Intended Audience and Reading Suggestions

This document is intended for developers, testers, documentation writers, and the grading staff for the project. This document does not specify the technical details, evaluation results, and user manual. The reader should read the system design document evaluation result, and user-manual document separately.

1.3 Project Scope

EleNa on higher level has the following objectives:

- Allow user to get directions between origin address and destination address.
- The provided directions should minimize the elevation gain.

Such a system will provide the following benefit(s):

- Improve user's walking and biking commute by minimizing the elevation gain.

2 Overall Description

2.1 Product Perspective

EleNa is a new, self-contained project and is not part of a larger system. It serves to the user as a web application.

2.2 Product Functions

EleNa provides the following function(s) to its users:

- Allow user to enter the origin and destination address.
- Allow user to set a tolerance i,e., how much extra are they willing to commute in comparison to the shortest non-elevation based path.
- Allow the user to change the map type (OpenStreetMap, Satellite, Roadmap, Terrain, Hybrid).
- Allow the user to specify their mode of transportation (walk, bike).
- Provide the path to user with the minimal elevation gain.
- Provide the following information to the user:
  - If a path is found based on the user's requirements.
  - Percentage elevation gain reduction w.r.t. the shortest non-elevation based path.
  - Percentage commute length increase w.r.t. the shortest non-elevation based path.

2.3 User Classes and Characteristics

We identify only two user types – application user and developer.

Application user has the following characteristics:

- User is not expected to have technical expertise.
- User is not expected to have access to the internals of the system.
- User only gets a use-only privilege for the system

Developer has the following characteristics:

- Developer is expected to have technical expertise.
- Developer is expected to have access to the internals of the system.
- Developer only gets a use/modify privilege for the system.

We seek to improve the experience of the application user.

2.4 Operating Environment

EleNa is expected to be deployed in the following operating environment:

- Linux-based system with:
  - Docker engine `v20.10.21`
  - Docker compose `v2.12.2`
  - Atleast 8GBs of RAM
  - Intel i5 or higher
  - Internet connection

EleNa is expected to be used in the following operating environment:

- Any operating system with:
  - Web browser (Chrome, Safari, Edge)
  - Internet connection

2.5 Design and Implementation Constraints

Due to computational resource limitations, we have the following design constraints:

- EleNa only supports origin and destination addresses in the United States of America.
- EleNa only supports origin and destination addresses belonging to the same state and city. Inter-city navigation is not provided.
- Cannot send concurrent very high number (> 3 per second) of requests due to free API limit of Nominatim address to coordinates API More about the API limit here.

2.6 User Documentation

We will provide the following components for user documentation:

- User manual
- Demonstration video

2.7 Assumptions and Dependencies

EleNa has the following third-party dependencies and functions based on the availability of these services:

- Google Maps Elevation API
- Nominatim Geocoding API
- OpenStreetMap API

These dependencies have their own limits based on the enrolled pricing plan. Currently, EleNa only use the free pricing plan, which limits its scalability potential.

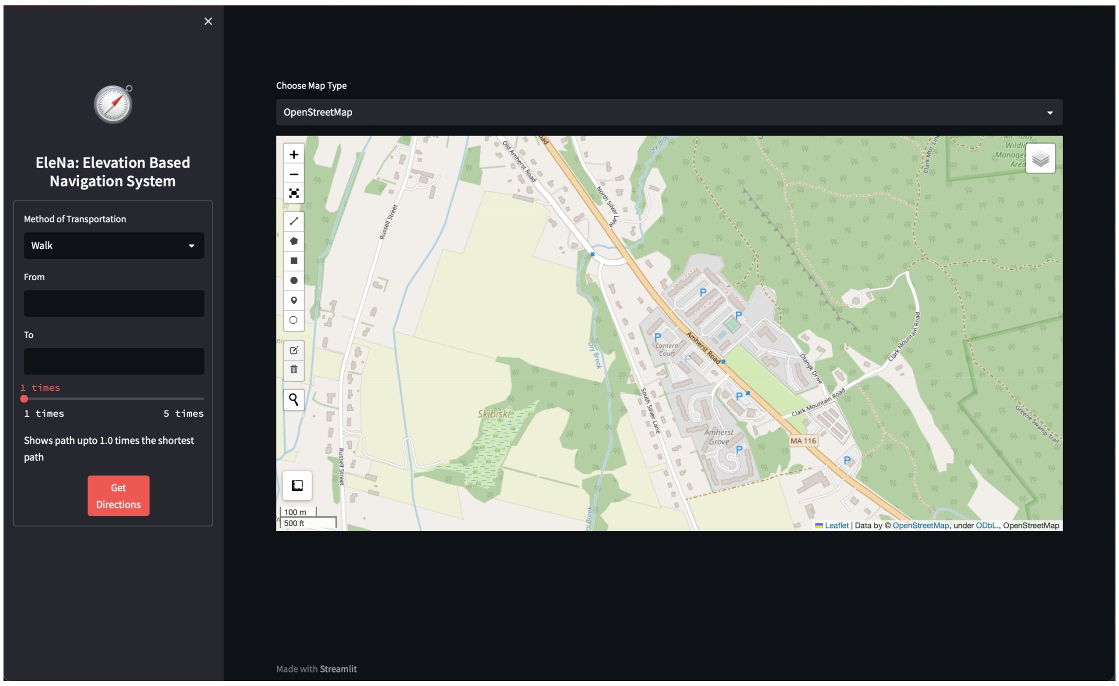
3 External Interface Requirements

3.1 User Interfaces

EleNa should provides a web application interface for the users. The user interface should have the following components:

- Form to enter:
  - Origin address
  - Destination address
  - Dropdown to select the transport mode
  - Slider to select tolerance
- Dropdown to select map type
- Map
- Button to perform validation and request directions
- Notification box for showing information such as error, success, invalid values to the user.

Fig 3.1 shows the proposed interface. Fig 3.2 shows the UML use-case diagram for the interface.



Method of Transportation
Walk
From
To
1 times
1 times 5 times
Shows path upto 1.0 times the shortest path
Choose Map Type
OpenStreetMap

Figure 3.1: Proposed interface

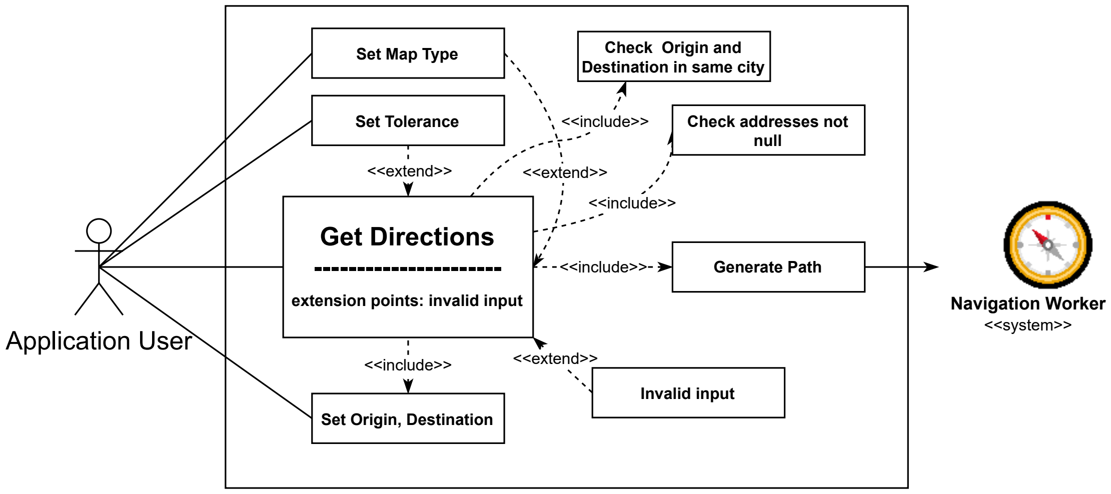


Figure 3.2: UML Use-Case diagram

4 System Features

The system should provide the following features:

4.1 Web Application Interface

Web application interface for the user to interact with EleNa.

4.1.1 Description

The system should provide a web application interface as specified in the prior section.

4.1.2 Stimulus/Response Sequences

Fig 3.2 shows the different user actions possible. The application can respond in the following ways:

- Success: Show the shortest elevation-based path with path statistics
- Path found failure: Show a message specifying that the requirements could not be met and show the next best path possible with path statistics.
- Invalid input: Show message specifying the input error.

4.1.3 Functional Requirements

The functional requirements for this system feature's UI are explained in Section 3.1. Other functional requirement(s) include:

- Serve as a web application accessible through a web browser.
- UI needs to be responsive to different screen sizes.
- Log the user requests through a logger.

4.2 Navigation Worker

Worker process to download, process, and generate the shortest path.

4.2.1 Description

Deals with downloading the graph, perform elevation grading and finding the shortest elevation based path. It is to be served as an internal web-API service.

4.2.2 Functional Requirements

Functional requirement(s) include:

- Download the map graph from OpenStreetMap API or load from cache.
- Perform elevation grading and path cost using Google Map Elevation API.
- Save the elevation graded graph to cache.
- Find origin and destination coordinates using Nominatim Geocoding API.
- Find the shortest elevation-based path using Dijikstra's algorithm with elevation cost weighting
- Calculate the statistics: elevation gain, difference w.r.t. shortest path by length.
- Serve two API endpoints – 1) Download graph, 2) Calculate shortest path.

4.3 Load Balancer

Load balancer to balance the load between different navigation worker.

4.3.1 Description

This feature allows load balancing of the internal API request to many navigation workers. It improves API throughput and reduces the failure rate.

4.3.2 Functional Requirements

Functional requirement(s) include:

- Perform round-robin load balancing to the navigation worker API.

4.4 LRU Cacher

Least recently used cache for processed graphs.

4.4.1 Description

Navigation worker caches the elevation graded graph. LRU cacher should be able to automatically delete least recently used processed graphs.

4.4.2 Functional Requirements

Functional requirement(s) include:

- Run the LRU cache cleaner every 24 hours.
- Utilize the logs from the view to find the last use date of a cached graph.
- Delete cached graphs not used within the past 1 day.

5 Nonfunctional Requirements

5.1 Performance Requirements

We expect EleNa to have a response time of < 5 seconds for cached graph and < 30 seconds for un-cached graphs.

5.2 Security Requirements

EleNa is expected to not expose the internal worker API endpoints externally. The user can only interact through the user interface.

5.3 Portability Requirements

EleNa is expected to be able to port to another system with Docker.

5.4 Scalability Requirements

EleNa is expected to be able to scale its service with demand by increasing the number of navigation workers.

5.5 Software Quality Attributes

The code is expected to have the following attributes:

- Correctness
- Reusability
- Testability
- Maintainabiltiy
- Debuggability