SYSTEM DESIGN DOCUMENT

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 Tech Stack

We use the following tech stack for EleNa:

- Python
- Docker
- Streamlit
- Nginx Load Balancer
- Gunicorn
- Flask
- OpenStreetMap
- Google Maps Elevation API
- Python unittest
- Python Logging
- Nominatim Geocoding API
- NetworkX

3 System Design

Fig. 3.1 describes the system design of EleNa. We have an MVC microservices architecture. We orchestrate our services as separate docker containers. We provide the ability to scale our application by scaling the number of workers and using load balancer to avoid worker overloading.

3.1 View

For view, we use Streamlit to generate the web application and UI. View exposes the web application end point to the user. View communicates with the workers to get the shortest path and maps for display using internal HTTP requests.

3.2 Worker

Worker services provide graph downloading, processing and finding the shortest path functionality. As it is a computationally expensive and slow process, we provide the ability to scale them horizontally. It exposes API endpoints using Gunicorn to receive HTTP requests. The workers use Dijkstra's algorithm with our custom defined elevation-cost as weight to find the optimized elevation-based path.

3.3 Load Balancer

Load balancer uses Nginx load balancer to balance internal HTTP request from view to workers using a round-robin approach.

3.4 Least Recently Used Cache Cleaner

EleNa caches the processed graphs to reduce the response time for navigation requests regarding the same geographical region. In order to reduce the memory usage, LRU cache cleaner cleans the least recently used processed graph at 24 hour interval.

3.5 Third-party APIs

We depend on Open Street Map API to get the map graphs and Google Map Elevation API to perform elevation grading and adding elevation-based cost for elevation-based navigation.

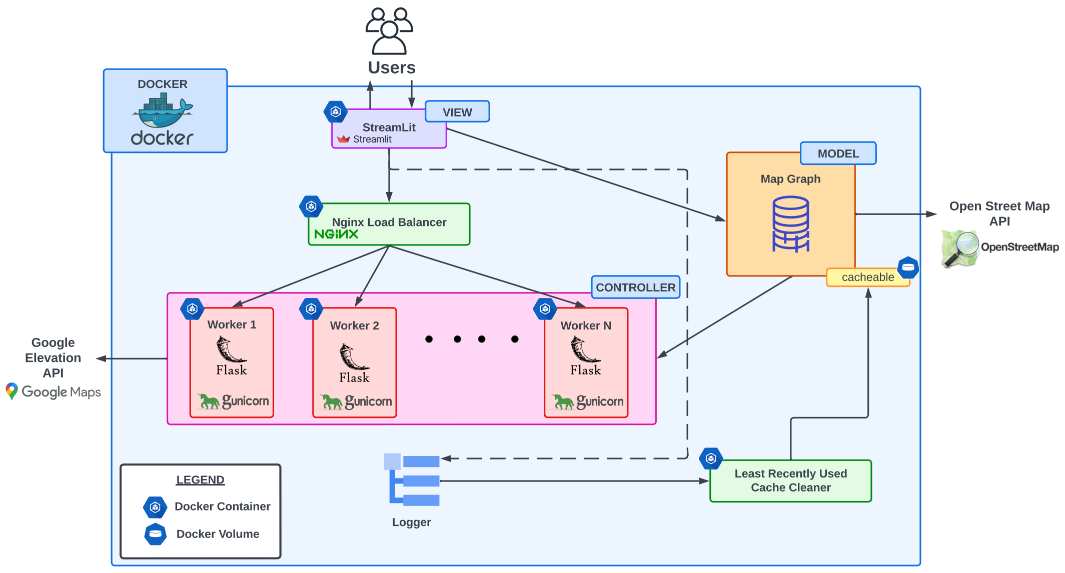

Figure 3.1: EleNa System Design