Scenic Navigation App

Kevin Buss, Jacob Marvel, Nicholas Santone, Robert Silver

CSC 468 – Group 1

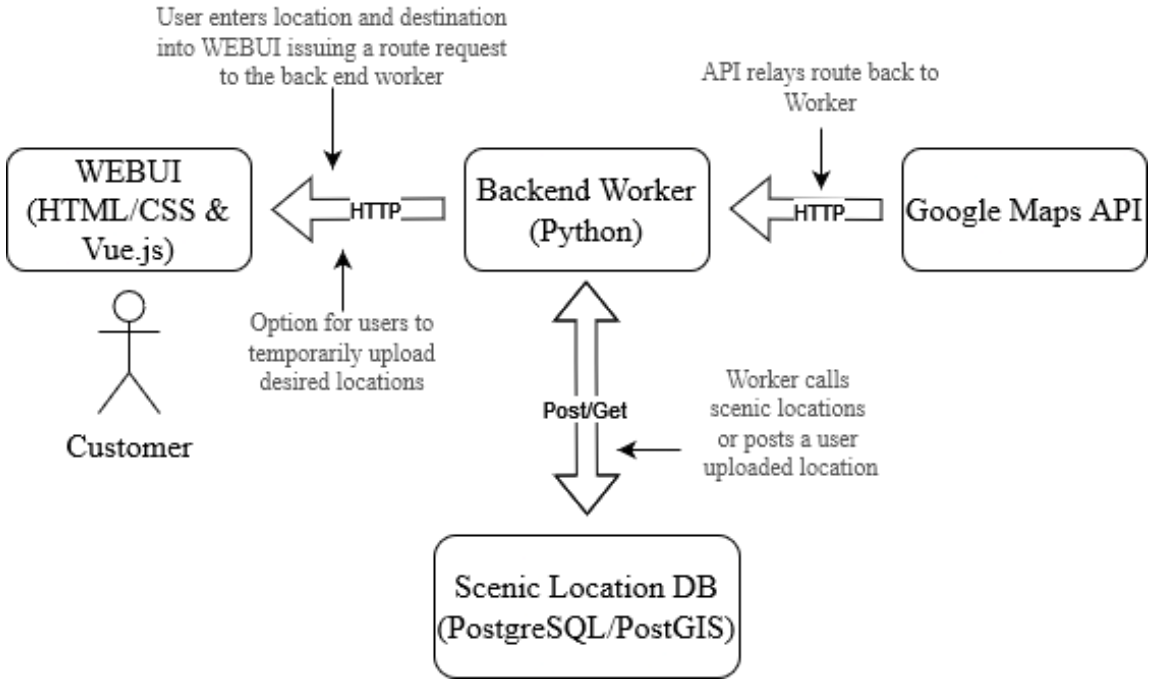
Chapter 1: Team Vision

1.1 Motivation

"It's not about the destination, it's about the journey." With our Scenic Navigation App, we take this adage to heart, transforming mundane routes into something memorable. Today's navigation apps are optimized to reach your destination as quickly as possible. We aim to reimagine everyday travel by prioritizing enjoyment over speed. Input your destination, and our app crafts a personalized journey, weaving through picturesque roads and landmarks to achieve this goal.

1.2 System Architecture Overview

Our cloud-based application follows a modern, microservices-oriented architecture. The frontend, built with Vue.js, provides an intuitive user interface for entering locations and viewing scenic routes. The backend, powered by Python, handles route generation logic and interfaces with the Google Maps API and our scenic locations database. The database, managed with PostgreSQL and PostGIS, stores curated scenic points of interest. The entire system is containerized using Docker for portability and scalability, and will be orchestrated using Kubernetes. Continuous Integration and Continuous Deployment (CI/CD) pipelines ensure smooth development and deployment workflows.



Chapter 2: Implementation Proposal

2.1 Web UI

The Scenic Navigation App's frontend will be developed using Vue.js. The user interface will allow users to input their desired start and end locations, and display the generated scenic route with turn-by-turn directions. The map component will be integrated using the Google Maps API, enabling features such as autocomplete location search and dynamic route rendering.

2.2 Backend Development and Data Management

The backend server, written in Python, will serve as the central hub for the application's functionality. It will handle incoming requests from the frontend, retrieve scenic coordinates from the database, and communicate with the Google Maps API to generate optimized scenic routes. The server will also manage the storage and retrieval of user preferences and route history.

2.3 Containerization with Docker

Each component of the application will be containerized using Docker to ensure portability and ease of deployment. Dockerfiles will be created for the frontend, backend, and database components, specifying the necessary dependencies and configurations. Multi-stage builds will be employed to optimize image sizes and improve build efficiency.

```
# Stage 1: Building Vue.js
FROM node:lts-alpine as build-stage
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
RUN npm run build

# Stage 2: Start the http-server
FROM node:lts-alpine
WORKDIR /scenic-navigation-app
COPY --from=build-stage /app/dist /scenic-navigation-app
RUN npm install -g http-server
EXPOSE 8080
CMD ["http-server", "/scenic-navigation-app", "-p", "8080"]
```

2.4 Cloud Integration and CI/CD Pipelines

The application will be deployed on CloudLab infrastructure. CI/CD pipelines will be set up to automate the build, test, and deployment processes. Code changes will trigger the CI pipeline, which will build Docker images, run automated tests, and push the validated images to a container registry. The CD pipeline will then deploy these images to the CloudLab environment.

2.5 Database Development

The database will be developed using PostgreSQL with the PostGIS extension to support spatial data. The logical schema will include tables for storing scenic locations, user preferences, and route history. The database will be accessed by the backend server through a Data Access Object (DAO) pattern, which encapsulates the database operations and provides a clean interface for data retrieval and manipulation.

Chapter 3: Intermediate Milestones

3.1 Building Docker Images and Testing Docker

Images for each component will be built using Dockerfiles. Challenges may include managing dependencies, optimizing image sizes, and ensuring proper configuration. Testing will involve running the containers and verifying their functionality through unit tests and integration tests. Health checks will be implemented to monitor the containers' status and ensure their reliability.

3.2 Data Collection and Management

Initially, data on scenic locations around West Chester will be collected manually by finding coordinates of relevant landmarks using Google Maps. These coordinates will be inserted into the PostgreSQL database table to populate the app's initial dataset. Additional data collection methods will allow users to add scenic points through the app's interface.

The backend server will interact with the scenic locations database through a data access object (DAO) layer, which will encapsulate the logic for querying and inserting location records. This will provide a clean interface for the server to request or upload scenic waypoints when generating routes or users uploading scenic points.

3.3 Limitations and Concerns:

The primary limitation is the tight deadline for completing the project, which may pose challenges in terms of balancing development efforts with other commitments. However, the utilization of the Google Maps API and its generous free usage tier is expected to alleviate some of the technical challenges. Security concerns, particularly related to API key management, will be addressed through secure key storage and rotation practices.

Chapter 4: Final Results

4.1 Successful implementation and MVP Achievement

The primary goal of the Scenic Navigation App was to offer an alternative navigation experience that incorporated scenic locations into a route. This objective was successfully met through the development of a route algorithm that utilizes a database of scenic locations and the Google Maps API. The app provides turn-by-turn directions along routes that include scenic locations along the way.

4.2 Significant Technical Achievement

The development of the scenic waypoint selection algorithm represents a significant technical achievement for this project. It is responsible for gathering scenic waypoints from a database, determining which ones are relevant to the route, and utilizing the Google Maps API to generate a route. The waypoint filtering works by calculating their proximity to a theoretical straight line path between the start and end locations. Below is an excerpt from the code that illustrates this process:

```
def get_scenic_waypoints(start, end, max_detour_distance=1000):
    # Retrieve scenic locations from the database

    # Parse the start and end coordinate strings
    start_lat, start_lng = map(float, start.split(','))
    end_lat, end_lng = map(float, end.split(','))

    # Create Points for start and end
    start_point = Point(start_lng, start_lat)
    end_point = Point(end_lng, end_lat)

    # Generate a LineString from start to end
```

```
    route_line = LineString([start_point, end_point])

    # Filter locations within a specified max_detour_distance from the route
    nearby_scenic_locations = [
        location for location in scenic_locations
        if is_location_near_route(route_line, Point(location['longitude'],
location['latitude']), max_detour_distance)
    ]

    # Convert filtered locations to waypoints
    waypoints = [f"{location['latitude']},{location['longitude']}" for location
in nearby_scenic_locations]

    return waypoints
```

4.3 Challenge: HTTPS Server Configuration on CloudLab

During the development of a scenic navigation application using the Google Maps API, several technical challenges were encountered, particularly related to Kubernetes security restrictions and the API's handling of requests from unsecure HTTP hosts.

It was discovered that the Google Maps API often blocks requests originating from unsecure HTTP hosts, which posed a significant problem since the application was initially deployed on a non-HTTPS server. To overcome this obstacle, an HTTPS server was attempted to be configured to ensure secure communication between the application and the API. However, setting up the HTTPS server proved to be challenging due to the occupancy of port 443, the default port for HTTPS traffic. This port was already in use by Kubernetes services running on the same server, causing a conflict, and preventing the successful configuration of the HTTPS server.

To further address this issue, the front-end deployment was simplified, and a clear separation between the frontend and backend components of the application was established. By decoupling these two aspects, the frontend was deployed on a separate server with a dedicated HTTPS configuration, eliminating the port conflict. This separation also facilitated better integration between the frontend and backend, allowing for smoother communication and data exchange. Although this solution required additional effort in terms of infrastructure setup and deployment, it ultimately resolved the HTTPS server configuration challenge and enabled the scenic navigation application to securely interact with the Google Maps API.

Chapter 5: Conclusion

5.1 Lessons Learned

Through the development of this application, our team faced numerous challenges that tested our technical skills and problem-solving abilities. One of the key lessons learned from this project was the critical importance of early integration and regular communication. This experience emphasized the value of small incremental updates over large, compartmentalized ones.

5.2 Technical Skills and Professional Development

This project served as an excellent platform for all team members to enhance their technical expertise, particularly in areas relating to Python programming, Kubernetes deployment, and frontend development with Vue.js. Additionally, working on a cloud-based platform like Cloudlab provided experience in managing and deploying cloud applications.

5.3 Future Direction

Looking ahead, there are several avenues for further development of the Scenic Navigation App. We aim to expand the app's database to include user-generated content, allowing users to submit and rate scenic locations. This feature could transform our app into a more interactive and community-driven platform. Furthermore, improving the algorithm for route generation to include more complex parameters like historical traffic patterns and user preferences could significantly enhance the app's functionality and user satisfaction. These enhancements will help in transitioning the project from a primarily academic exercise to a potentially marketable product.

5.4 Concluding Thoughts

In conclusion, the Scenic Navigation App project was not only a test of our technical abilities but also a valuable lesson in teamwork and project management. The project allowed us to practically apply our coursework in a real-world-like scenario, bridging the gap between theoretical knowledge and practical application. As we move forward, the insights gained from this project will undoubtedly influence our approach to future software development projects, emphasizing the importance of adaptability, continuous learning, and user-centered design.

Kevin Buss

kevinbbuss@gmail.com | github.com/KevBuss

Education

West Chester University — West Chester, PA
Bachelor of Science in Computer Science — *Sep. 2022 – est. May 2025*

Montgomery County Community College — Blue Bell, PA
Associate's of Science in Computer Science — *Sep. 2019 – May 2021*

Experience

Software Engineer Asc. — Sep. 2023 – Present
Lockheed Martin — *King of Prussia, PA*

- Collaborate with a software development team to design, develop, and debug software applications, focusing on service-to-service communications, frontend, and backend development
- Demonstrate technical accomplishments to key stakeholders
- Acted as Scrum Master, facilitating daily stand-ups, sprint planning, and retrospectives
- Improved proficiency in Java, JavaScript, Python, SQL, HTML, and CSS

Programmer — *Oct. 2021 – Sep 2023*

- Contributed to the automation of end-to-end testing processes using Playwright, significantly improving testing efficiency and reliability
- Developed and implemented comprehensive testing strategies, ensuring robustness and high quality of software applications

Information Technology Consultant — Dec. 2020 – Apr. 2021
Geeks-on-site — *Coral Gables, FL*

- Provided on-site IT solutions, diagnosing and solving software-related issues
- Managed and scheduled regular consultations with clients
- Recommended and purchased essential software and hardware equipment

Website Designer — May 2019 – July 2019
Academy of the New Church Secondary Schools — *Bryn Athyn, PA*

- Worked closely with stakeholders to ensure the website reflected the institution's branding and messaging
- Led the design, build, and maintenance of the institution's website using advanced HTML and CSS
- Regularly updated website content, ensuring accuracy and relevance

Technical Skills

Languages: Java, Python, SQL (Postgres), JavaScript, HTML/CSS
Frameworks: React, Node.js, Flask, JUnit, WordPress, Material-UI, FastAPI
Developer Tools: Git, Docker,VS Code, Eclipse
Methodologies: Agile, Scrum, Kanban

Jake Marvel

nicholassantone@gmail.com | https://github.com/NS977255

Education

West Chester University — West Chester, PA
BS in Computer Science, Cyber Security Certificate — Aug. 2021 - May 2024

West Chester East High School — West Chester, PA
High School Diploma, Academic — Sep. 2016 – June 2020

Coursework

Computer Science I – Intro to Java — Aug. – Dec. 2020
- Variables, data types, I/O, loops

Computer Science II – Java — Jan. – May 2021
- Arrays, sorting algorithms, GUI

Computer Science III – Java — Aug. – Dec. 2021
- Classes, methods, polymorphism, OOP

Computer Systems – C — Aug. – Dec. 2021
- Memory allocation, bitwise, hexadecimal

Foundations of Computer Science — Jan. – May. 2022
- Optimizing variable usage, function reusability, problem solving

Data Structures and Algorithms – Java — Jan. – May. 2022
- Linked lists, queues, BFS and DFS

Computer Security & Ethics — Aug. – Dec. 2022
- Understanding cyber defense structures and ethical dilemmas

Programming Language Concepts and Paradigms - Haskell — Jan. – May. 2023
- Underlying foundation of programming languages

Software Engineering - Java — Jan. – May. 2023
- Object-oriented programming, inheritance, polymorphism

Technical Skills

Languages: Java, Python, C, SQL, Haskell, Linux
Developer Tools: Git, Docker, jGrasp, VS Code, IntelliJ
General Skills: Problem-solving, openness to other ideas, following directions, teamwork
Interests: Video games, Drawing, Music, Writing

Nicholas Santone

| nicholassantone@gmail.com | https://github.com/NS977255

Education

West Chester University — West Chester, PA
BS in Computer Science, Cyber Security Certificate — Aug. 2021 - May 2025

Bensalem High School — Bensalem, PA
High School Diploma, Academic — Sep. 2017 – June 2021

Coursework

Computer Science I – Intro to Java — Aug. 2021 – Dec. 2021
- Variables, data types, I/O, loops

Computer Science II – Java — Dec. 2021 – May 2022
- Arrays, sorting algorithms, GUI

Computer Science III – Java — Aug. 2022 – Dec. 2022
- Classes, methods, polymorphism, OOP

Data Structures – Java — Dec. 2022 – May. 2023
- Linked lists, queues, BFS and DFS

Computer Systems – C — Dec. 2022 – May 2023
- Memory allocation, bitwise, hexadecimal

Foundations of Computer Science — Aug. 2023 – Dec. 2023
- Optimizing variable usage, function reusability, problem solving

Technical Skills

Languages: Java, C/C++, OOP
Developer Tools: Git, Visual Studio, jGrasp
General Skills: Problem solving, teamwork

Interests: Camping/Hiking, Martial arts

Robert Silver
RobertSilver011@outlook.com

EDUCATION

West Chester University of Pennsylvania — West Chester, PA
Bachelor of Science, Computer Science — Expected Graduation May 2026
Recipient of Academic Excellence Scholarship — 2022-2026
Recipient of Dean's List Academic Achievement — 2022-2023

COURSEWORK

Computer Science 1 - Intro to Java — Aug 2022 – Dec 2022
• Program design, control flow, input/output, variables, data types, and string manipulation

Computer Science 2 - Java — Jan 2023 – May 2023
• Implemented recursive algorithms to solve multilayered computational problems
• Utilized arrays to store and manipulate datasets, increasing processing speed
• Implemented search and sorting algorithms to sort a large data set or search for a value within it
• Developed a GUI Java calculator application capable of evaluating required arithmetic functions

Computer Science 3 - Java — Aug 2023 – Dec 2023
• Advanced Object-oriented Programming: inheritance, polymorphism, abstraction, exception handling, random access files, serialization, and fundamental data structures

Computer Systems – Intro to C — Aug 2023 – Dec 2023
• Analyzed CPU architecture and memory systems to grasp system execution
• Utilized parallel and distributed programming techniques to reduce processing time

SKILLS & INTERESTS

Skills: Object-oriented programming, Java, C, jGrasp/VSCode, GitHub, Terminal/CLI Use, Collaboration, Problem-Solving, Self-Driven, Mechanically Inclined, Writing Shells and Testing Programs
Interests: Automotive Maintenance, Electronic Upgrades/Repairs, Hiking/Outdoors

LEADERSHIP

• Eagle Scout Rank Achievement & Leadership Project — Jul 2020
• Reading Berks Science and Engineering Fair (RBSEF) – Honorable Mention — 2019 & 2020
• Kennedy House Soup Kitchen – Prepared/served meals, clean up, delivered donations — 2016 - 2020
• Habit for Humanity – Collect and organized donations for sale at Habitat for Humanity — 2018-2022

PERSONAL EXPERIENCES

• Basic vehicle maintenance: changing oil, coolant, belts, power steering fluid, tires, alternator, battery
• Vehicle Bodywork: drilling out spotwelds, grinding/cutting metal, mounting engines
• Personal 3D modeling with SolidWorks and printing with Creality Ender 3 S1
• iPhone screen, battery, back glass, and earpiece assembly replacements
• Laptop Ram/SSD upgrades/replacement and internal cleaning for dust buildup
• Booting removable OS from BIOS, Disk Partitions, Configuring Storage Devices