Documentation &

Project Diary

Innovation Lab 1

Year 2025

Project: Qualitiy of Life - Web

Team: Bettina Kovac

1. General Information

**Project name:** Qualitiy of Life - Web

**Supervisor:** Wahl

Innovation Lab *1, summer term 2025*

**Projectteam:**

Bettina Kovac if23b230@technikum-wien.at

**Management Summary of the Project**

The project "Quality of Life - Web" aims to create a user-friendly website that calculates and explains the quality of life at a given address in Vienna, Austria. The main priorities are: (1) identifying factors that define quality of life using open data, (2) designing an algorithm to evaluate it per address, (3) supporting different user types such as students and parents with small children, (4) visualizing differences between two addresses including routes and distances, and (5) ensuring a robust, error-tolerant user experience. The website will enable users to compare two addresses, receive personalized insights, and understand the logic behind the evaluation.

**Framework Conditions and Project Environment**

The project relies on publicly available Open Data from the city of Vienna. Instead of using live APIs, the datasets were downloaded and self-hosted in a Docker container environment to ensure stability and local control over the data. Git was used for version control throughout the development process. The website frontend was built using HTML, CSS, and JavaScript, and the Live Server extension was used to support local development and rendering of the map. Google Maps API was integrated to display interactive maps and calculate routes and distances between two addresses.

The algorithm calculates a quality of life score tailored to different user categories (e.g., students or parents with small children) and offers transparent explanations of how this score is derived. System reliability and error handling were key concerns, especially for invalid address inputs and unavailable or incomplete data.

**Semester-Roadmap**

**Project Roadmap and Effort Estimation**

The project was implemented over the course of 4 weeks during a single semester. It was developed by one person, with an estimated average workload of 5 hours per day, totaling approximately 100 person-hours.

**Effort Estimation Method**  
The effort was estimated and tracked in real working hours. No story point system was used. Development started by processing local JSON data, with functionality built step by step. In the final week, the project was containerized using Docker for local hosting. Version control was managed with Git and GitHub Desktop.

**Week-by-Week Roadmap**

**Week 1**

* Requirements analysis
* Open Data research
* Defined user categories
* Designed initial JSON data structure  
  **Estimated Hours:** 20  
  **Notes:** Initial setup and planning

**Week 2**

* Implemented algorithm to evaluate JSON-based data
* Developed first frontend version (HTML/CSS/JS)
* Integrated category-based scoring logic  
  **Estimated Hours:** 25  
  **Notes:** Focused on backend logic

**Week 3**

* Added Google Maps API
* Implemented address input, route display, and distance calculations
* Continued frontend work  
  **Estimated Hours:** 25  
  **Notes:** Focused on interactivity

**Week 4**

* Set up Docker container for local hosting
* Implemented error handling (invalid input, data issues)
* Used GitHub Desktop for version control  
  **Estimated Hours:** 30  
  **Notes:** Final deployment & polish

**Total Estimated Effort:** ~100 hours

**Reflection**  
The project followed a structured development flow. Starting from static JSON files, the main logic and user interface were developed first. In the final week, Docker was introduced to host the project locally, ensuring consistency and portability. GitHub Desktop supported version control. All core requirements were implemented within the available timeframe.

**Collaboration & Tooling**

Collaboration & Tooling

The project was developed individually but used standard tools for collaboration, version control, and deployment, in alignment with modern software development practices.

* Version Control:  
  Git was used throughout the project for version control. GitHub Desktop was utilized to manage commits, branches, and synchronization with the remote repository.  
  GitHub Repository: Link muss noch eingefügt werden
* Development Environment:
  + Frontend: HTML, CSS, JavaScript
  + Map Integration: Google Maps API
  + Live Development Server: Visual Studio Code with the “Live Server” extension
  + Backend & Hosting: Local development and deployment using Docker containers

1. **Brief Description of the Project**

The project aims to develop a web application that calculates and visualizes the quality of life at specific addresses in Vienna, based on publicly available Open Data. Users can input one or two addresses and receive a computed quality-of-life score for each location. The system will display these results, explain how the score was calculated, and provide a visual route between the two addresses—including both the airline (straight-line) distance and the actual route distance via Google Maps.

The solution is tailored to support different user categories, specifically:

* Students
* Parents with young children
* Seniors
* Peopel how are out of work

Each user group receives a personalized evaluation, as their priorities (e.g., access to education vs. safety and green areas) can differ. The user can also view the detailed criteria influencing the quality score at each address.

**Goals and Deliverables**

The following project outcomes (deliverables) were defined:

* A responsive, browser-based application built with HTML, CSS, and JavaScript
* An algorithm that calculates a quality-of-life score using structured Open Data (JSON format) from the City of Vienna
* Google Maps integration to:
  + Display the entered addresses
  + Calculate and show the route between two points
  + Show the distance (airline and route)
* Error handling for:
  + Invalid or incomplete address input
  + Missing or temporarily unavailable Open Data
* A Docker-based local hosting setup
* A simple, intuitive UI with an option to explain the score composition

The implementation also includes version control via Git, managed using GitHub Desktop, and a live preview environment using the Live Server extension in Visual Studio Code.

**Greatest Challenges**

* Integrating the Open Data into the Docker container and ensuring it could be accessed correctly by the application
* Completing the core functionality within a short development timeframe of 4 weeks, including UI, logic, and deployment

**Greatest Added Value for Users**

* Quickly understanding how livable an address is based on meaningful, personalized metrics
* Being able to compare two addresses side-by-side, including a route, distance, and scoring breakdown
* Making informed decisions (e.g., about where to live or move) using publicly available, transparent data

**Scope of the Project**

**Included:**

* Address input and validation
* Calculation of quality-of-life scores
* Custom scoring logic based on user category
* Explanation of score details
* Route mapping, distance calculations
* Local deployment using Docker

**Not included (Non-goals):**

* Full mobile app implementation (only web-based solution)
* User login or profile storage
* Large-scale scalability or cloud deployment (focus is on local prototype)
* Real-time API querying for the quality-of-life algorithm (the system uses downloaded Open Data for this, except for the Google Maps API used for routing and maps)

**Implementation Approach**

* All Open Data was downloaded and structured into local JSON files, which populate the Docker container with the necessary data during deployment.
* The Google Maps API was used for map rendering, routing, and geolocation.
* The frontend was built using HTML, CSS, and JavaScript, with the Live Server extension in Visual Studio Code for local development and rapid preview.
* In the final week, the project was containerized using Docker, ensuring consistent deployment and environment setup.

1. **Specification of the Solution**

**System Environment**

The solution is a web-based application designed to run in modern browsers. It relies on locally hosted Open Data within a Docker container for quality-of-life calculations and integrates with external services (Google Maps API) for geolocation, mapping, and routing. The system boundaries are limited to:

* Accepting user input addresses within Vienna
* Processing quality-of-life scoring using predefined datasets
* Displaying results and routes on an interactive map
* Handling errors gracefully for invalid input or missing data  
  No backend server beyond the Docker container is required, and the application runs fully client-side except for calls to Google Maps API.

**Features (Functional Requirements)**

The core features are structured as user stories:

* **User Story 1: Compare Two Addresses**  
  As a user, I want to enter a second address to compare the quality-of-life scores between two locations.
* **User Story 2: View Route and Distances**  
  As a user, I want to see the route between the two addresses on a map along with the air distance and route distance.
* **User Story 3: Understand Score Calculation**  
  As a user, I want to see an explanation of how the quality-of-life score is calculated based on different data categories.
* **User Story 4: Support Different User Types**  
  As a user, I want to specify whether I am a student or a parent with small children to receive a tailored quality-of-life evaluation.
* **User Story 5: Error Handling**  
  As a user, I want to be informed if my input address is invalid or if required data is temporarily unavailable.

**Interfaces**

* **User Interface:** Responsive web UI developed in HTML, CSS, and JavaScript, accessible via desktop and tablet browsers.
* **External API Interface:** Google Maps API for address geocoding, routing, map rendering, and distance calculation.

**Quality Characteristics and Technical Requirements**

* **Performance:** The application should respond within 2 seconds for quality-of-life calculation and map rendering after address input.
* **Availability:** Local hosting via Docker ensures data availability independent of external sources except for Google Maps services.
* **Usability:** Simple, intuitive UI with clear feedback, tailored for both students and parents. Accessibility standards for basic keyboard navigation and screen reader compatibility are considered.
* **Architecture:** Client-side single-page application architecture with Docker container hosting the data backend.
* **Expandability:** Modular design of the algorithm to allow future inclusion of additional user types or new Open Data sources.
* **Reliability:** Robust error handling for invalid inputs and missing or unreachable data sources.

1. **Effort Estimation**

**Week-by-Week Roadmap**

**Week 1**

* Requirements analysis
* Open Data research
* Defined user categories
* Designed initial JSON data structure  
  **Estimated Hours:** 20  
  **Notes:** Initial setup and planning

**Week 2**

* Implemented algorithm to evaluate JSON-based data
* Developed first frontend version (HTML/CSS/JS)
* Integrated category-based scoring logic  
  **Estimated Hours:** 25  
  **Notes:** Focused on backend logic

**Week 3**

* Added Google Maps API
* Implemented address input, route display, and distance calculations
* Continued frontend work  
  **Estimated Hours:** 25  
  **Notes:** Focused on interactivity

**Week 4**

* Set up Docker container for local hosting
* Implemented error handling (invalid input, data issues)
* Used GitHub Desktop for version control  
  **Estimated Hours:** 30  
  **Notes:** Final deployment & polish

**Total Estimated Effort:** ~100 hours

1. **Delivery**

The delivery of the project includes all necessary components to enable the customer or another software team to deploy, operate, and maintain the solution.

* **Final Solution:**  
  The complete source code of the web application, including frontend files (HTML, CSS, JavaScript), Docker configuration files, and local Open Data JSON files, is available.
* **System Architecture:**  
  The solution is a client-side web application hosted locally via Docker. The architecture includes a Docker container hosting the Open Data files and integration with the Google Maps API for mapping and routing.
* **Licenses and Copyrights:**  
  The project uses publicly available Open Data from the City of Vienna under applicable open data licenses. External libraries and APIs, such as Google Maps API, are used under their respective terms of service. No proprietary or paid third-party software is included in the deliverables.
* **Hardware Specifications:**  
  No special hardware is required; the solution runs on any modern computer with a web browser and Docker installed.
* **Additional Deliverables:**  
  The project diary documenting the development process, a presentation slide deck, and a video demonstration of the solution are also provided for handover and knowledge transfer.

1. **Our Project Diary**

This section of the project diary was only discovered during the last week of the project, which is why most of the entries and reflections come from that final phase.

During this last week, Docker setup and integration were explored and implemented for the first time. This caused significant stress because only one week remained before the project deadline.

Several challenges occurred:

* **Docker Installation and Setup:**  
  My laptop had recently been reset, so Docker Desktop needed to be reinstalled. This process was time-consuming and stressful, especially since pgAdmin also required a fresh setup.
* **Version Control Requirement:**  
  I needed to install Docker Desktop to meet version control requirements, adding pressure close to the deadline.
* **Lack of Clear Documentation:**  
  Although we learned about Docker during the third semester at school, the course lacked sufficient documentation or manuals to follow. This made setting up and integrating Docker especially challenging. After uploading everything into Docker, I had only three days to rewrite my code to accept input from the Docker-hosted data instead of local files.

Despite these difficulties, I managed to complete the necessary work. This diary entry was written on the project’s final day to reflect on the intense last week.