Home Finder

TEAM 11

Client/Advisor:
Dr. Timothy Bigelow

Undergraduate Team:
Christian Boughton
Daniel Chrisman
Michael Wieland
Lith Almadani
Ella Knott

Resources:
sddec23-12@iastate.edu
https://sddec23-11.sd.ece.iastate.edu/

Revised: 1 DEC 2023/V2

Executive Summary

Development Standards & Practices Used

IEEE 802.11 ac, g, b, a,n are required for universal wifi access to our application. Because our application will process information server side to limit user resource use, this 802.11 suite of standards will be required for access

IEEE 802.3 defines the Ethernet standard. wired or ethernet-based connectivity will be required for our application to be used by the maximum number of individuals.

IEEE 4003. This standard defines the ability for data from constellations of satellites to be used in navigation. This is crucial to our application as the global positioning system, governed under the Global Navigation Satellite System (IEEE 4003), identifies locations, aids in calculating commute times, and provides the data necessary to create the ultimate goal of this project: the weighted heat map of ideal areas to live.

Summary of Requirements

- The application will be fully accessible on web-connected mobile phones, desktops, and laptops.
- The server side will process inputted information within 5 seconds of form submission (constraint)
- Creating a heat map based on a list of data input by users like locations, frequency, and time.
- Creating a distance calculator that takes an input of locations and calculates the distance.
- The ability to create accounts and log in to existing ones. The ability to create, save, access, and delete records.
- Ability to access a web-based browser for viewing the application
- A hosting platform will be required to display our web page and allow interaction.
- The application must be visually accessible to users, including text size, font, color contrast, etc.
- The application must be responsive to ensure operability ubiquitously across browsers and platforms.

Applicable Courses from Iowa State University Curriculum

- COM S 319: Construction of User Interfaces
- CPR E 231: Cyber Security Concepts and Tools
- SE 329: Software Project Management
- COM S 309: Software Development Practices

New Skills/Knowledge acquired that was not taught in courses

1. Developing front-end applications
2. Utilizing API calls for data processing
3. Creating web applications with HTML, CSS, JS
4. Working with backend frameworks
5. Database management
6. Web hosting and cloud consoles
7. Implementing network security configurations
8. Interaction with clients in an Agile methodology

Table of Contents

List of Figures and Tables

1 Team

1.1 Team Members

Christian Boughton: Cyber Security Engineering

Michael Wieland: Electrical Engineering

Daniel Chrisman: Computer Engineering

Ella Knott: Cyber Security Engineering

Lith Almadani: Software Engineering

1.2 Required Skill Sets for Your Project

- A working knowledge of front-end languages (i.e., JavaScript, CSS, and HTML)
- A working knowledge of backend technologies and frameworks
- Knowledge of cloud infrastructure for hosting
- The ability to allow communication between the client side and server side
- The ability to design a user-friendly graphic interface (webpage)
- The ability to utilize API calls

1.3 Skill Sets covered by the Team

- Visual Design: Christian Boughton, Michael Wieland
- Back-end coding: Lith Almadani, Daniel Chrisman
- Front-end coding: Ella Knott, Michael Wieland
- Database Management: Lith Almadani, Michael Wieland
- Error Handling: Christian Boughton, Lith Almadani
- Data Integration: Christian Boughton, Daniel Chrisman
- Functions and API calls: Christian Boughton, Ella Knott
- Security Testing: Ella Knott

1.4 Project Management Style Adopted by the team

We are using Agile as a software project management style. We are using Agile because it is more dynamic and decentralized. With our project expected to take two semesters, the agile framework will allow us to procure pieces of the project early on for review. If, at any point, the client decides to add, remove, or change a part of our implementation, this can be done easier than with a waterfall style. Because each piece is being worked on simultaneously, the team is more involved and able to make each section their own, versus a centralized waterfall design.

1.5 Project Management Roles

- Christian Boughton — Team Lead
- Ella Knott — Security/Client Lead
- Daniel Chrisman — Back-End Lead
- Lith Almadani — Front-End Lead
- Michael Wieland — Development Lead

2 Introduction

2.1 Problem Statement

Searching for a new home can be exhausting and time-consuming. It requires endless hours of browsing numerous listings and websites to find the ideal area for residency. The process can be even more overwhelming if you are unfamiliar with the area. This is where our platform can make things easier. Our platform simplifies the home search process by allowing you to input your preferences, including your frequently visited places and the maximum desired travel time between them. Our application generates a heat map for the area surrounding your inputs, making it easier for you to identify houses within the desired area. You will be able to observe your average daily commute and plan your busy schedule more effectively. Additionally, the platform can help you locate an expansion for your private business. Ideal for daycares, physicians and dentists, and recurring customer base entities. This is done by calculating the average commute time for your recurring clientele through saved addresses.

2.2 Requirements & Constraints

Functional Requirements:

- The application will be fully accessible on web-connected devices
- The server side will process inputted information within 5 seconds of form submission
- Create a heat-map based on a list of data input by user.
- Utilize APIs to calculate drive times
- The ability to create accounts and log in to existing ones.
- The ability to create, save, access, and delete records.

Resource Requirements:

- Ability to access a web-based browser for viewing the application.
- A hosting platform will be required to display our web page and allow interaction.

Qualitative Aesthetic Requirements:

- The application needs to be visually accessible to users, including text size, font, color contrast, etc.

Economic Requirements:

- All members of the team are required to have access to a web-enabled device for program sharing.
- Requirement of a cloud hosting service for deployment
- Requirement of API keys for map display, drive time calculations, locations autofill, and address conversions.

UI requirements:

- The application will be designed and tested with the user experience at the forefront of development. By focusing on what users might find unnecessary and stripping much of the visual frills associated with other housing applications, our application will be easier to use.

2.3 Engineering Standards

IEEE 802.11 ac, g, b, a, n are required for universal wifi access to our application. Because our application will process information server-side to limit user resource use, the 802.11 suite of standards will be required for access.

IEEE 802.3 defines the Ethernet standard. wired or ethernet-based connectivity will be required for our application to be used by the maximum number of individuals.

IEEE 4003. This standard defines the ability for data from constellations of satellites to be used in navigation. This is crucial to our application as the global positioning system, governed under the Global Navigation Satellite System (IEEE 4003), identifies locations, aids in calculating commute times, and provides the data necessary to create the ultimate goal of this project: the weighted heat map of ideal areas to live.

2.4 Intended Users and Uses

Our project is intended for people in the market for new housing who may not want to disrupt their normal routine. Many real estate applications only display a map of available housing and leave users to calculate their daily commutes independently. With our application, users will be provided a heat map of neighborhoods that have the least commute time based on their frequented locations and their level of importance.

Some example use cases for our application include:

1. A family of three is expecting to add a new member to their family. They need to upgrade to a larger home, but they don't want to pull their child out of their current school system. With HomeFinder, they can find neighborhoods close to the school and home that matches their size needs.
2. A businessman is promoted to a position that requires frequent travel. They decided they would like to move closer to the airport to reduce their commute time to the airport. This user can input the airport as an important location, and HomeFinder will highlight more neighborhoods surrounding the airport.
3. A college student is having trouble with her current roommate situation. She decides she would like to move to a different apartment, but she doesn't want to give up the

convenience of her living arrangement. HomeFinder considers that the student frequently visits the central campus, the mall, and her favorite pho restaurant and generates a heat map of neighborhoods located short distances between these locations. The student finds an apartment that cuts her weekly commute time by almost a third.

3 Project Plan

3.1 Project Management/Tracking Procedures

We are using Agile as a software project management style. We are using Agile because it is more dynamic and decentralized. With our project expected to take two semesters, the agile framework will allow us to procure pieces of the project early on for review. If, at any point, the client decides to add, remove, or change a part of our implementation, this can be done easier than with a waterfall style. Because each piece is being worked on simultaneously, the team is more involved and able to make each section their own, versus a centralized waterfall design.

The goals of our project are to; produce a functioning product that fits the description of the project proposal, learn to work as an engineering team would in the field, and ensure each team member contributes to the actual engineering of the final product. The agile methodology was selected to better accomplish these goals by providing greater flexibility to our final implementation, providing a more dynamic role for each team member, and splitting the engineering aspects of the project so that each member would practice their trade.

Our team uses a combination of GitLab and Discord to track our progress in this project. GitLab allows us to create project boards inside of our repository to track progress on functional components, features, bugs, and incidents. It also provides a method for peer code review by requiring another member of the team to sign off on code before it is merged. Communication, announcements, and planning take place on Discord as our team has few available open spots in our schedule to meet.

3.2 Task Decomposition

Linear Breakdown of all tasks

1. Planning and Acquisition
    a. Research relevant APIs and obtain keys
    b. Research Cloud Hosting Platforms and create accounts
    c. Install software and create shared repositories
2. Create Initial Front-End Only Version for Proof of Concept
    a. Create a visual design for the web application in CSS and HTML
    b. Add functionality with JavaScript
    c. Research and obtain relevant APIs
    d. Connect APIs to process data
    e. Add additional processing and display heat-map overlay to mutable graph

    f. Receive feedback from client
3. Create A Full Stack version of the web application
    a. Modify front-end to send API requests to the back-end
    b. Implement services, controllers, objects, and repositories for data handling
    c. Initially use non persistent H2 database for testing
    d. Receive feedback from client
4. Integrate a Relational SQL Database
    a. Research optimal database for the web applications requirements
    b. Ensure database is supported by the cloud hosting service
    c. Develop schema and connect to back-end
    d. Install database management system
5. Deploy
    a. Create a virtual machine on the cloud hosting service
    b. Set network controls to allow connections only from authorized services
    c. Back-end
        i. Install code management software
        ii. Create and connect cloud database
        iii. Update Schemas
        iv. Upload code and run
        v. Verify functionality with postman
    d. Front-end
        i. Install Apache web server
        ii. Configure internal files for public access
        iii. Upload code and files
        iv. Test with browser stack to ensure portability and responsiveness
    e. Receive feedback from client
6. Alpha Testing
    a. Internally run the application on multiple platforms to catch bugs
    b. Bug Fixes
7. Beta Testing
    a. Distribute application to small group and receive feedback
    b. Visual and functional rework
8. Establish Persistence
    a. Modify web server files to ensure the web application does not go offline
9. Prep for Future Work
    a. Ensure proper documentation for code
    b. Create document with instructions on how to run code
    c. Create documents outlining deployment and server alterations
    d. Contact web hosted reality services to obtain sponsorship
    e. Zip all files and submit to client

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

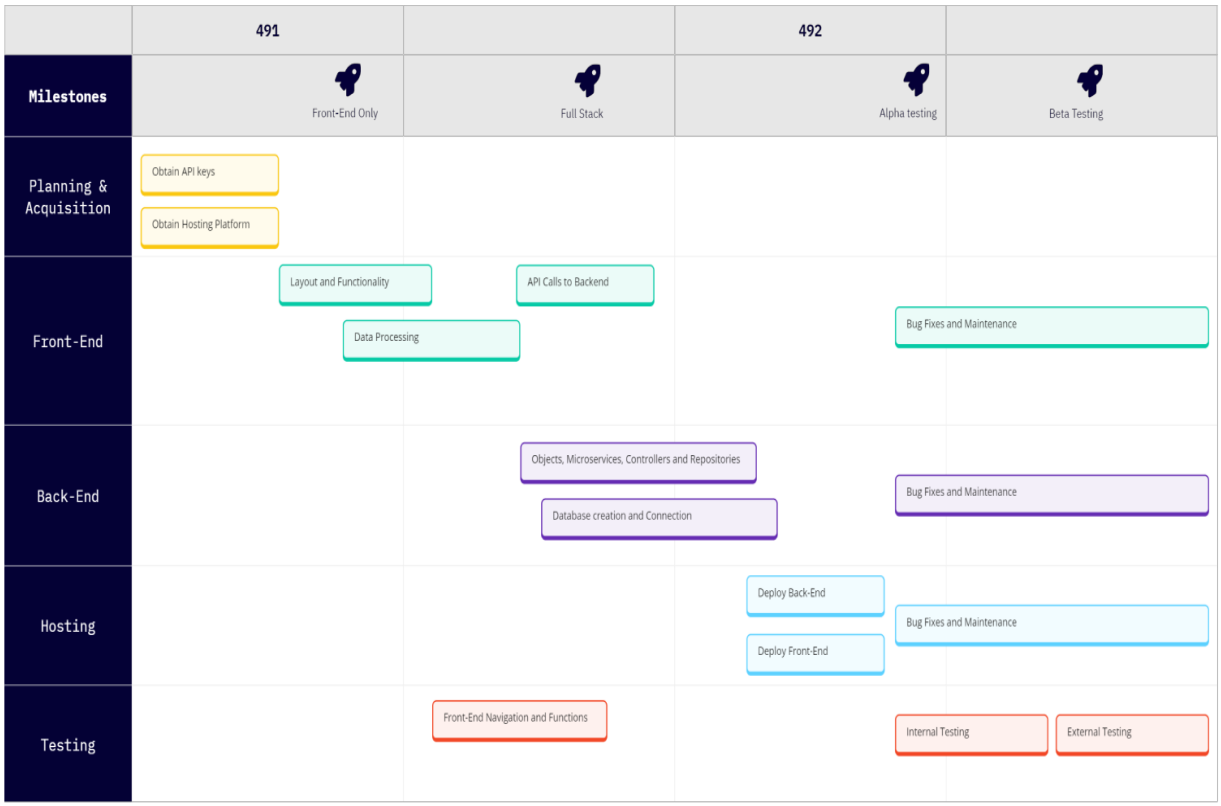


Figure 1 Project Milestones

3.4 Project Timeline/Schedule

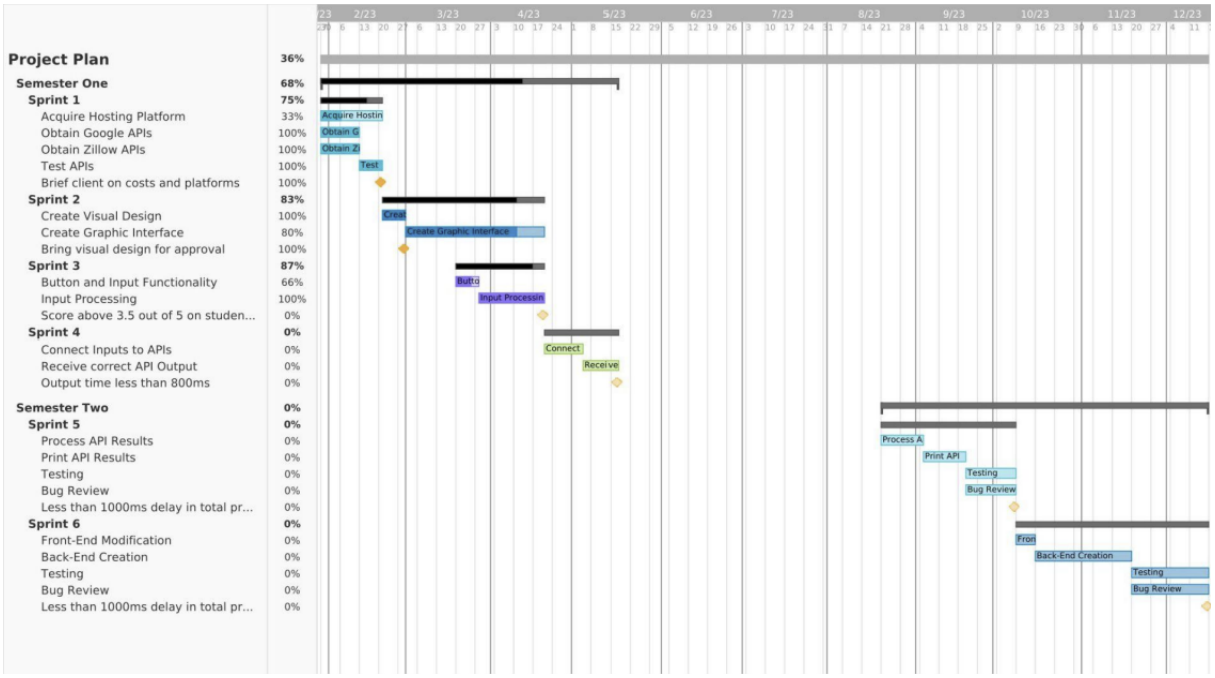


Figure 2 Gantt Chart

3.5 Risks And Risk Management/Mitigation

Table 1 Risks and Mitigations

Risk	Mitigation
High API Cost	Develope a low resolution heat-map
Security Breaches	Use well known hosting platform with default security measures
Network Data Leaks	Require all traffic to use HTTPS

3.6 Personnel Effort Requirements

Table 2 Time Requirements

Task	Hours
Research and Acquisition	20
Create Front-End Only Version	100
Create Full Stack Version	100
Integrate SQL Database	10
Deploy	10
Alpha Testing	20
Beta Testing	20
Establish Persistence	2
Prep for Future Work	10
Bug Fixes and Maintenance	50

3.7 Other Resource Requirements

The Home Finder application runs entirely on the cloud so no physical components will be necessary outside of personal devices. Other considerations are listed below.

- Financial requirement for utilizing a cloud server while the application is running
- Financial requirement for utilizing external API calls
- Programming languages and Frameworks and Database:
    - HTML
    - CSS
    - JavaScript
    - Java
    - Spring Boot
    - MySQL relational database

4 Design

4.1 Design Context

4.1.1 Broader Context

Our design project is intended to be used by any member of the population that is searching for an ideal area to live in. The application can subsequently find ideal areas for businesses with recurring client models. The community impact of this application offers many benefits to all users: however, it affects lower-income and disabled users the most. The tools provided allow users with limited transportation or need to be within a certain distance from a care facility to better find housing that matches their needs. Our goal for this project is to solve the societal issue of locating housing based on the user's needs.

4.1.2 User Needs

Home buyers need a way to search for ideal areas to live and see the properties available in those areas because current applications fail to provide adequate search customization.

Businesses need a way to ensure they are relocating or expanding to an area their clients will be satisfied with because long commutes or inopportune areas may cause undue hardships for their clientele or revenue loss.

Commuters need a way to find living locations closer to their workplaces or daily destinations because they want to reduce the amount of time and money spent on daily transportation and improve their quality of life.

Families with children need a way to find living locations in safe and convenient neighborhoods with good schools and parks because they want to provide their children with good education and a safe environment.

University students need a way to find living locations close to their campus and other facilities because they want to reduce the time and money spent on transportation and improve their academic performance and social life.

Retirees need a way to find living locations in a calm, pleasant neighborhood with access to healthcare services and social activities because they want to enjoy their retirement and maintain their physical and mental well-being.

Real estate developers need a way to identify areas with high demand for housing and potential for profitable investments because they want to maximize their profits.out

4.1.3 Prior Work/Solutions

Finding the best places to live based on user inputs has been the subject of numerous web applications. The two that most closely align with our application are:

1. Walk Score: Walk Score is a web application that calculates the walkability of a given location by analyzing its closeness to nearby amenities such as grocery stores, restaurants, schools, and parks. One of the advantages of Walk Score is that it considers a wide range of factors that can affect the livability of a location.
   https://www.walkscore.com/

2. Trulia: Trulia is a real estate website that provides a heat map of neighborhoods based on factors such as crime rates, schools, and commute times. One of the advantages of Trulia is that it provides a broad view of different neighborhoods and their livability factors.
   https://www.trulia.com/

Our web application has several advantages compared to the existing products.

- Generated heat maps do not have an input limit
- Heat-maps are skewed to the frequency of travel to each location
- Home Finder has the ability to integrate with other applications and databases (i.e.., MLS database to provide listing in the area)

One potential disadvantage of our web application is that it may require a large amount of data and computational resources to generate accurate heat maps for a wide range of users and locations.

4.1.4 Technical Complexity

The project's technical complexity is sufficiently challenging, requiring a broad range of technical skills and expertise, including UX design, front-end, and back-end development, database management, data analysis and visualization, mathematical optimization, and networking and hosting requirements. The project's requirements match current solutions and industry standards, such as location-based applications.

1. User Interface; Our application requires a user-friendly interface that allows for account creation, input handling, and map generation with modifications. The design of such an interface requires knowledge of user experience design principles and front-end web development skills such as HTML, CSS, and JavaScript.

2. Data Storage; Our application will need to store user data securely and efficiently. This requires knowledge of database management principles, such as SQL and NoSQL databases.

3. Data Processing; Our application requires a custom algorithm to overlay a variable-sized grid and perform complex calculations to create the data necessary for the generation of the heatmaps. This requires an understanding of data structures, geometry, and general mathematics.

4. Heatmap Generation: Our application will generate heatmaps based on user inputs. This requires knowledge of data analysis and visualization, as well as back-end development skills such as Python and Google Maps APIs.

5. Optimization Algorithms: Our application will also need to implement optimization algorithms to find the optimal living location for each user based on their daily trips. This requires knowledge of mathematical optimization principles.

6. Networking and Hosting : Our application runs on a cloud server that requires a detailed knowledge of operating systems, web-server applications, networking protocols, and access management.

4.2 Design Exploration

4.2.1 Design Decisions

1. Pleasing visual web design that is minimal to ensure understanding and ease of use
2. Functional front end for processing user-inputted information coded in HTML, CSS, and JS
3. Functional back end for storing data, processing data, and connecting to APIs
4. Mainstream hosting platform with default security rules

4.2.2 Ideation

Our team used the brainstorming method with a storyboard technique to ideate a solution for structuring our web application.The storyboard gave us a visual representation of our ideas that we could build on and reference.

1. Front-end only initial design
2. Full stack design
3. Hosted on cloud server

4.2.3 Decision-Making and Trade-Off

We chose to identify the pros and cons of each idea through lists in our storyboard. We chose to first start with a front-end-only design to ensure functionality before moving to a cloud-hosted front-end UI/UX design for the final implementation. Our final implementation will allow greater user access with less overhead for our client.

After starting development of the backend with a Node.js framework, we encountered an issue with requiring separate software for database management. To reduce the required software for this project, we changed to a Spring Boot back-end. Spring Boot allows for integrated database management from the running process and increased support for API hendeling.

4.3 Proposed Design

4.3.1 Design Visual and Description

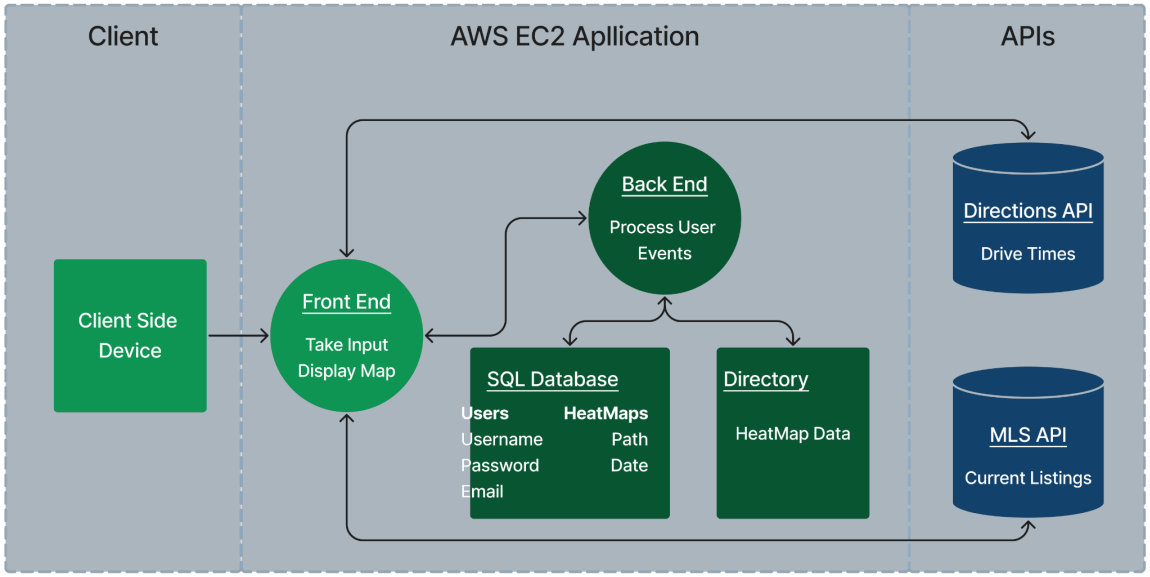


Figure 3 Design Visual

Our design is a front-end UI/UX website accessed over HTTPS, The client side is responsible for taking inputs, making API calls, and displaying processed data. information generated on the front-end can be sent to the back-end for persistence and recalling.

4.3.2 Functionality

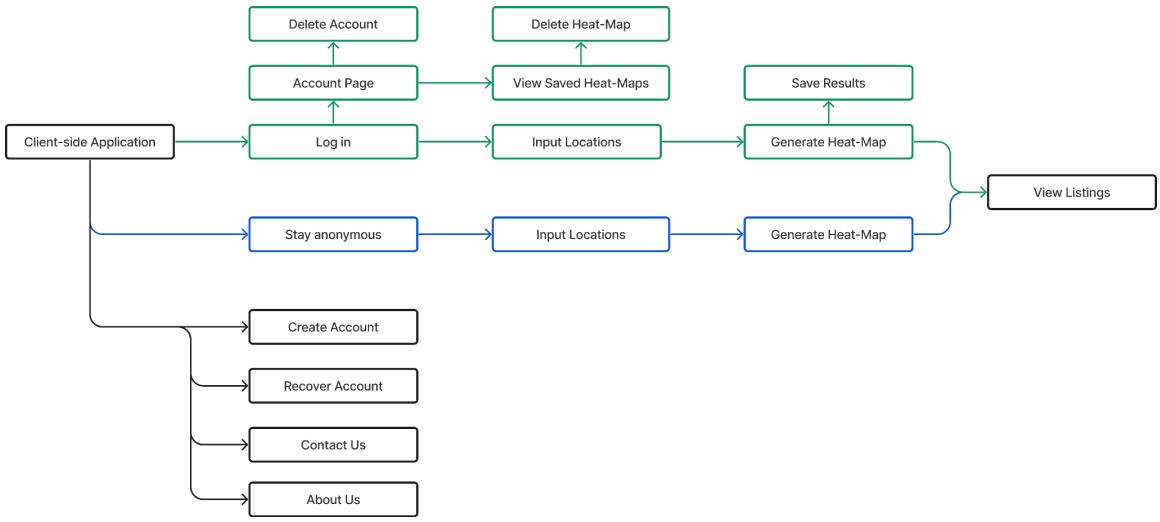


Figure 4 Functionality

4.3.3 Areas of Concern and Development

Our primary concern is the cost associated with large user requests. At a predefined number of user requests to the APIs, fees will start to accrue. Our team could not design the application without the use of the APIs because satellite and GPS information is needed to calculate drive times with traffic considerations.

To address this issue, our team has lowered the resolution of the heat-map overlay. This alteration requires less calculations and significantly improves the cost per use. Additionally, the ability to store and recall heat-map overlays reduces the overall cost by not requiring recalculations of the same inputs.

4.4 Technology Considerations

Strengths:

1. JavasScript is widely used for web development, making it easy to find resources and support.
2. Google Maps APIis a powerful tool for generating maps and manipulating location-based data, making it way easier to implement our main functionalities and add more features.
3. Spring Boot provides integrated database management and API functionality for back-end code
4. MySQL database is the most widely used and supported database on the market with ample documentation.
5. Amazon AWS EC2 web server has default security policies and allows for scaling and increased performance through Elastic Load Balancing.

Weaknesses:

1. JavaScript is a client side scripting language that can only execute code on the client's browser. This can lead to performance issues if the application is handling a lot of data, as the browser may need help to keep up with the calculations required for generating heat maps.
2. The Google Maps API has a minimal fee with each request
3. Amazon AWS Ec2 platform has a small hourly fee

Trade-offs:

1. The use of JavaScript and Google Maps API will allow us to have a high degree of customization and flexibility in the design, but it will also require us the more technical expertise to implement and maintain.
2. Using a third-party API like Google Maps means that the application relies on that provider for its core functionality. This can create dependencies and potential issues if the provider makes changes.

Possible solutions and design alternatives:

1. To reduce costs, our application could explore alternative map providers that offer similar functionality at a lower cost or for free. Examples include OpenStreetMap or Mapbox.
2. To reduce dependencies on third-party providers, our application could explore the use of open-source libraries for map generation and location-based services. like Leaflet or D3.js.

4.5 Design Analysis

The proposed design from 3.3 works as expected allowing us to meet all requirements set in the planning phase. Future designs can improve the web application by integrating GIS databases, alternative API for statistical data in geographic regions, and or use a single distance matrix for drive time durations.

4.6 Design Plan

Requirements: The website should have a user-friendly interface allowing users to input their preferences, such as location, work location, and any other frequently visited place. The website should also provide relevant information, such as current houses in the market and not ones that have already been sold.

Modules: The design plan includes separate frontend and backend modules. The frontend module is responsible for presenting the user interface and collecting user input and processing data, while the backend module provides functionality for data persistence.

Module diagram: The frontend module can work independent of the backend module, as it needs to receive information from the backend to present it to the user. The backend module runs in parallel, as it needs to process multiple requests from different users at the same time.

Interfaces: The frontend module communicates with the backend module through an API, defining the input and output parameters and their formats. JSON is used as the data type for sending information.

Module constraints: The website handles a large number of users and provides accurate and up-to-date information. The backend module is designed to handle different types of data and formats and use appropriate algorithms and databases to process the user input and provide relevant information.

5 Testing

Testing is an essential part of any software development project. It is particularly crucial for a web application like ours that aims to help users find the optimal living location by creating heat maps based on their daily trips. And to ensure that our application functions correctly and meets user requirements, we need a comprehensive testing strategy that covers all aspects of our system's design. One of the unique challenges to testing our system is the reliance on third-party APIs, particularly Google APIs, to generate the heatmaps. Since these APIs are external to our system, they can be subject to unexpected changes, impacting our application's functionality. So, we need to ensure that our testing strategy includes detailed testing of the integration with these APIs. Another challenge in testing our system is the diverse range of user inputs our application needs to handle, including daily trip frequency and timing. This makes it essential to perform extensive testing of the input validation and processing functions to ensure that the application can handle various inputs and produce accurate heatmaps. We will address these challenges and ensure the overall quality of our application by implementing a testing strategy that includes a range of testing techniques, including unit testing, integration testing, interface testing, and user acceptance testing.

5.1 Unit Testing

1. User Account Creation and Management: Create test cases for user account creation, login, logout, and password reset functionalities. Done using JUnit.
2. Input Validation Functions: Test the functions responsible for validating user inputs, including input data types, format, and range. Done using JUnit.
3. Heatmap Generation: Test generation by creating test cases for generating heat-maps based on user inputs and verifying that the heatmaps are generated correctly.
4. User Authentication and Authorization: we will test the functions responsible for user authentication and authorization. Done using frameworks such as Spring Security and Rest Assured.

When testing each unit, we will consider testing both the expected behavior and the boundary conditions, which will include testing both valid and invalid inputs and edge cases that test the limits of our application. To perform these tests, we will use various testing frameworks and tools, such as JUnit, Spring Security, and Rest Assured. These frameworks can be integrated with our build and deployment process to automate testing in a CI/CD pipeline.

5.2 Interface Testing

1. User Interface: This interface allows users to create accounts, provide input for daily trip frequency and timing, view the generated heatmaps, and manage their accounts. Testing will be done using Selenium, Cypress, or TestCafe. We have also deployed a beta version of this application for review by realtors. Our client has approved the review process.
2. API Interfaces: This interface enables our web application to integrate with Google APIs to generate heat-maps based on user inputs. Testing will be done with Postman, SoapUI, and or Rest Assured to create API tests and ensure that the API calls return the expected results.

5.3 Integration Testing

The critical path in our design is the input and output streams to and from the used APIs. This is because we do not have full control over how the data will be manipulated and what the results will be. We test this by first using a bare-bones framework to run several API requests and monitor output. Requests are generated based on normal and edge case inputs (i.e., a single location or a very large amount of locations spread over a great distance). We use tools such as JUnit and NUnit to test for exceptions by comparing the output structure to the expected structure. After ensuring the edge case and normal tests match our expected structure for data returns, we then generate heat maps from these and compare them to the hand-created ones described above.

5.4 System Testing

This is done using the BrowserStack suite of tools. These tools allow us to view the website across 300 platforms, including mobile-based ones. BrowserStack also allows us to test the website's responsiveness to ensure we maintain usability across screen and device types. Overall functionality is tested by generating heat-maps on each of the 300 browsers and comparing them manually or through image recognition to determine if any meaningful differences arise. The BrowserStack application allows us to test our design on each platform concurrently by simulating each request from a different browser type.

A beta version of this application has been reviewed by currently practicing realtors to ensure functionality meets the demands of our users.

5.5 Regression Testing

We have created our site as modularly as possible and used general CSS styles to ensure future additions will adhere to our current design. The critical features implemented in this program are the integration of the APIs. The modularity allows us to make modifications to the functions without disrupting the current progress. Front-end visual or structural changes will need to be tested as stated above on 4.4 for browser support and responsiveness. We reuse the BrowserStack suite of tools to ensure any additions to the visual or structural design in the front end do not hamper the end result.

5.6 Acceptance Testing

Our client has been involved in our acceptance testing through periodic meetings. We have met once a week to update progress made. While we display these updates, we are receiving feedback on design choices. This will help make sure that non-functional requirements are met. As development progresses, these meetings will continue. As functional features are added, we will test them and then receive feedback from the client.

Additionally, we have received information that there is interest in our product from members of the housing and real estate industry.

5.7 Security Testing

Our project is dealing with user information, including their frequented locations as well as listed housing information; because of this, we need to ensure that our user's sessions and data are properly encrypted and that our site is safe from injections and other common vulnerabilities. We test for various potential web-based security issues using various penetration testing tools, including Burp Suite, SQLMap, Nikto, and more. We use the OWASP Top Ten list of web-application vulnerabilities for a baseline of vulnerabilities to test for using these tools. This list highlights broken access control, cryptographic failures, site injections, and request forgeries as some of the most common vulnerabilities on the web.

5.8 Results

Testing has resulted in minimal bugs and a responsive web application able to be accessed via a web enabled device.

Beta testing has resulted in moderate interest for the web application from realtors and heightened interest from home and apartment searchers.

6 Implementation

6.1 Required Elements

Front-End

- Amazon AWS EC2 cloud server
- Apache2 web server
- Java Mail Sender
- VS Code editor

Back-End

- Amazon AWS EC2 cloud server
- MySQL relational database
- VS Code editor

6.2 Design Revisions

Throughout this project our team has learned a great deal about web applications and deployment. This has led to several redesigns and of both the web application and services used.

Version 1:

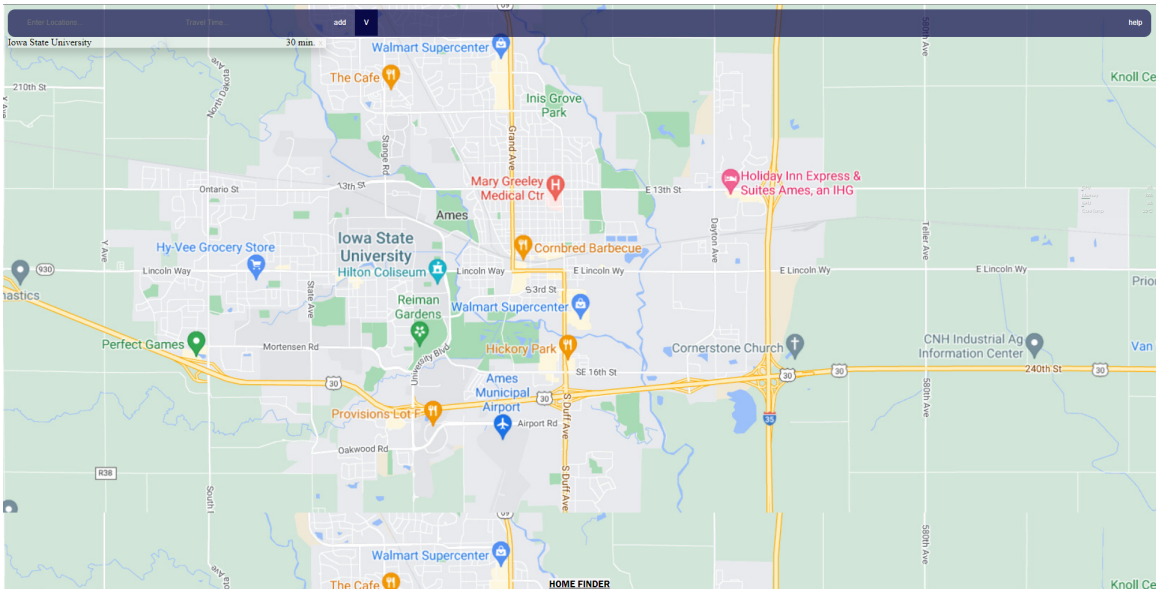


Figure 5 HomeFinder v1.0

The first iteration of the home finder web application was hosted on GitHub Pages and functioned as a standalone front-end UI that received and stored inputs locally. The initial design was developed to receive feedback from the client for how the web application should look and function. After receiving feedback, it became apparent that the single page site would be less responsive and cluttered after adding desired peripheral features.

Version 2:

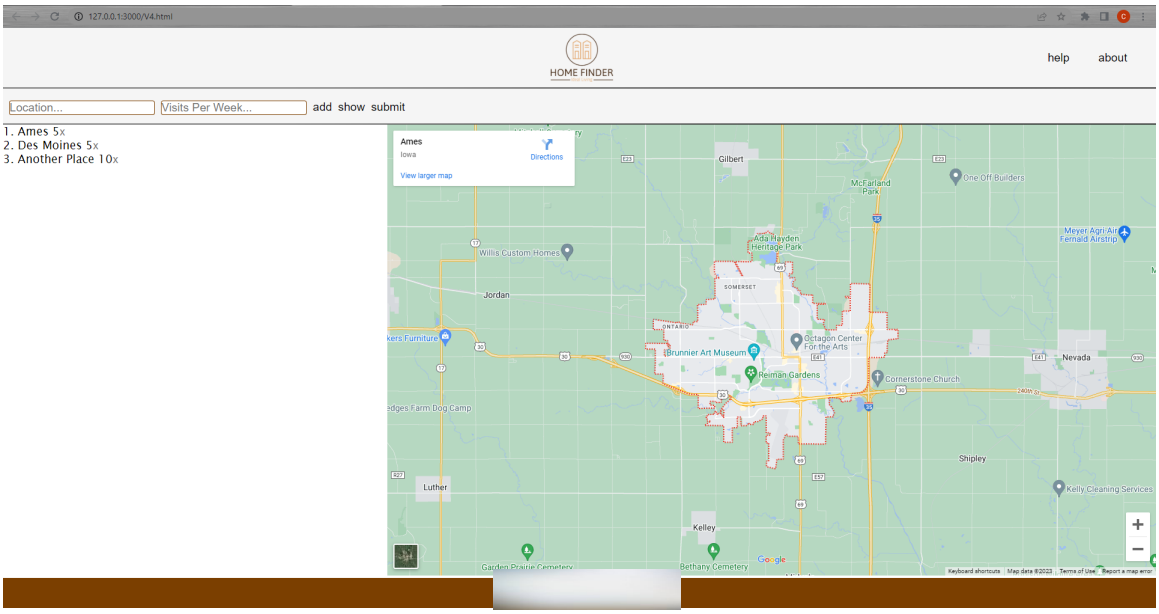


Figure 6 HomeFinder v2.0

The Second iteration of the Home Finder web application included a top navigation bar with a separate input bar. Locations and frequencies were added to the text boxes and stored as list elements. The inputs could be viewed via a collapsible side panel. Version 2 offered significant improvement over version 1; However, several bugs were encountered with the map feature. When resizing the map to accommodate the side panel, the service would reload causing high latency. Additionally, the row design decreased responsiveness creating a less than viewable screen for mobile users.

Final Version

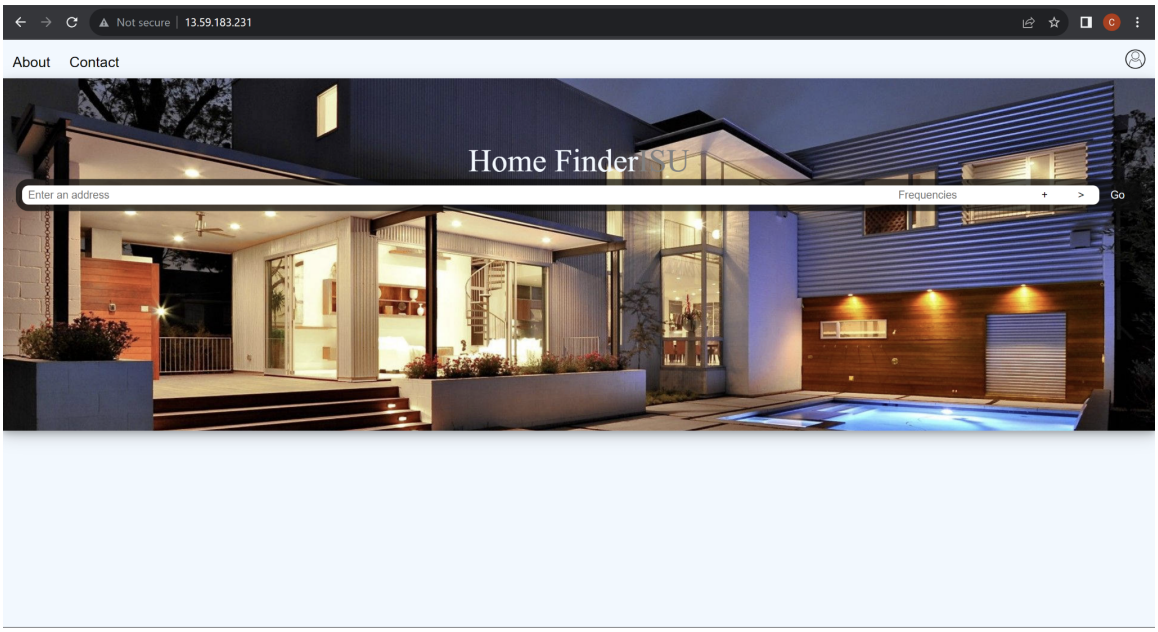


Figure 7 HomeFinder v3.0 Home Page

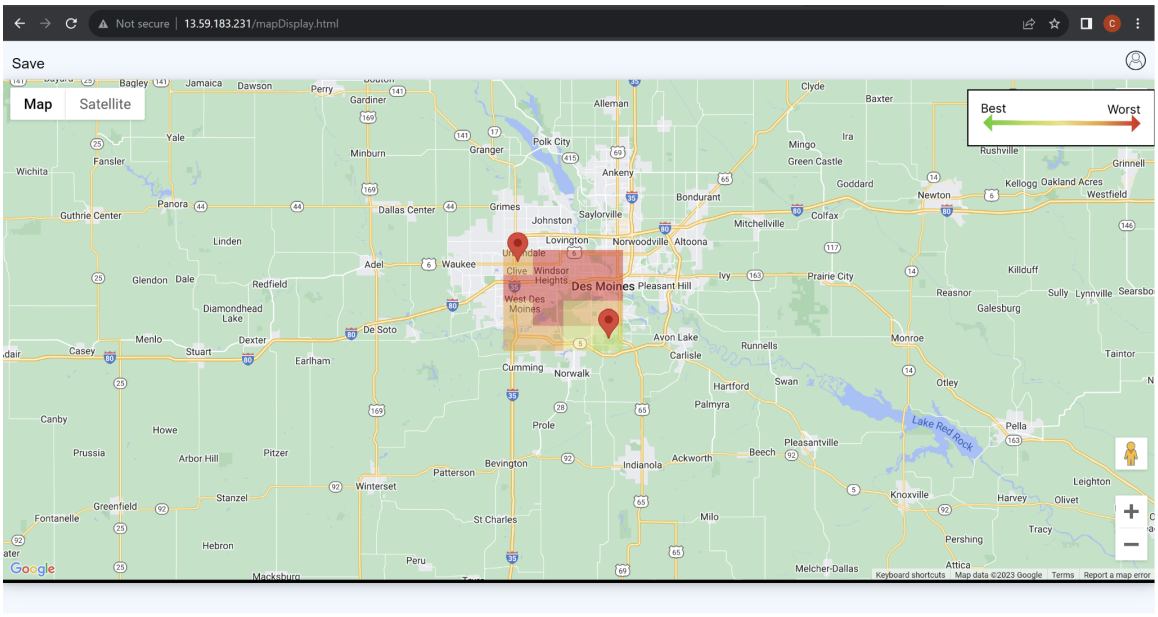


Figure 8 HomeFinder v3.0 Map Display

The final design separates the input bar from the navigation bar to indicate main functionality of the web application. The map is moved to its own page to enhance the readability. The map has also been updated to include multiple views and be mutable, autofocusing on the area of the heat-map overlay. The final version is responsive and tested on multiple mobile devices and large screens.

6.3 Web Application Implementation

6.3.1 Back-End Implementation

The back-end is hosted on the Amazon AWS EC2 platform. It is coded in Java with a Spring Boot framework that provides database management. connection to the front end is done by sending encrypted JSON packets via HTTPS and TCP.

The database is a MySQL database chosen for its relational properties and popularity. The database is hosted through Amazon's AWS RDS service that provides default security policies. It is connected to the backend through the Hibernate library.

6.3.2 Front-End Implementation

The Front end is coded in vanilla HTML, CSS and JavaScript to increase portability of the web application. It is hosted on the same AWS EC2 server as the backend running on an Apache2 web server. The web server is configured for HTTPS traffic and provides a certificate.

6.3.3 Security Policy Implementation

Security policies are primarily provided through Amazon's default security rules for cloud based services. To improve upon these rules, the EC2 and RDS service have been provided additional rules that close all unnecessary ports. Access to the database is controlled via SSH requiring a .pem RSA generated key for authenticity.

7 Professionalism

This discussion is with respect to the paper titled "Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment", *International Journal of Engineering Education* Vol. 28, No. 2, pp. 416–424, 2012

7.1 Areas of Responsibility

Public Health and Safety: The IEEE Code of Ethics addresses the responsibility by stating that engineers shall "hold paramount the safety, health, and welfare of the public and the protection of the environment." This statement emphasizes the importance of ensuring public safety and environmental protection in engineering projects.

Global and Social Impact: The IEEE Code of Ethics addresses this responsibility by stating that engineers shall "seek, accept. and offer honest criticism of technical work, to acknowledge and correct errors, and to properly credit the contributions of others." This statement emphasizes the importance of being honest and transparent in technical work.

Communication: The IEEE Code of Ethics states engineers shall "be truthful and realistic in stating claims or estimates based on available data." This shows the importance of being honest and accurate in communicating technical information.

Professional Developments: The IEEE Code of Ethics states “advance the integrity and reputation of the profession consistent with the public interest.” This emphasizes the importance of maintaining a high level of professionalism and ethical behavior in the engineering profession.

Ethical Leadership: The IEEE code of ethics addresses this responsibility by stating that Engineers shall “Avoid conflicts of interest and shall not exploit or misrepresent the work of others for personal or professional gain.” This statement emphasizes the importance of avoiding conflicts of interest and maintaining ethical behavior in all professional interactions.

Sustainability: The IEEE code of ethics addresses this responsibility by stating that engineers shall “Consider this social and environmental impact of their professional activities.” This statement emphasizes the importance of considering the long-term social environmental impact of engineering projects.

Professional Responsibility: The IEEE code of ethics addresses this responsibility by stating that engineers shall “uphold and enhance the honor, integrity and dignity of the engineering profession.” This statement emphasizes the importance of maintaining the highest standards of professionalism and ethical behavior in the engineering profession.

Compared to the NSPE code of ethics, the IEEE code of ethics is more comprehensive in addressing each of the seven professional responsibilities. While the NSPE code of ethics also covers similar areas, the IEEE code of ethics provides more specific guidance on how engineers should behave in different professional situations additionally, the IEEE code of ethics places a greater emphasis on the social environmental impact of engineering projects.

7.2 Project Specific Professional Responsibility Areas

Public health and safety: Yes, this applies to our project because the safety of the users' information must be kept in an area where it cannot be exploited by any hackers or easily accessible to the public. We are performing this at a high caliber by using reputable sites to host our site to ensure peak security.

Global and social impact: Yes, this is applicable to our project as our website is directed towards pretty much anybody so we want honest criticism in order to produce the greatest result.

Communication: Communication is very important to our project as it is what is going to create the greatest result. Without ideal communication, it will be almost impossible to complete our project.

Professional developments: This is also applicable to our website as professional development is something that we want to strive for. Professional development means user-friendly as well as an almost perfect website.

Ethical leadership: This is applicable because everybody must be held responsible for their portion inputted into a project. Leadership is something that can be very beneficial because it keeps everybody on the same task, working together towards the same goal which is very important in a project that spans two semesters.

Sustainability: Sustainability does not pertain to our project as we are only a website so there's not much sustainability to go into it.

Professional responsibility: Going hand in hand with professional development this is also applicable to our website. In order to efficiently complete this task we must all be responsible and complete our individual tasks.

7.3 Most Applicable Professional Responsibility Area

The most applicable professional responsibility area within our project is the amount of data from users that we take in. It's going to take a lot of responsibility to make sure this data is secure and not messed around with by anyone within our group or anybody outside of our group.

8 Closing Material

8.1 Discussion

The final design meets the requirements of our proposed project including all requested functionality. Additional functionality proposed by our group that was not able to be implemented was the integration of MLs data into our heat-map display. This data would allow users to connect to listed homes in their ideal areas. The functionality was omitted from the final design because the access we had received in the first semester for MLS data access was rescinded by the private corporation without cause.

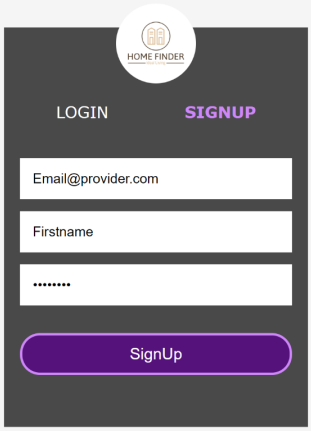
8.2 Conclusion

The technical complexity of the project encompasses various domains, including UX design, front-end and back-end development, database management, data analysis, and optimization algorithms. Our design decisions prioritize a minimal and visually pleasing interface, with a phased approach starting with a front-end-only design and later incorporating a cloud-hosted full-stack implementation. Despite challenges such as potential data and computational resource requirements, our team has devised strategies to mitigate costs and enhance efficiency. The technology considerations weigh the strengths and weaknesses of key components like JavaScript, Google Maps API, Spring Boot, MySQL database, and Amazon AWS EC2. Trade-offs include the balance between customization and technical expertise, as well as dependencies on third-party providers. The design analysis confirms the functionality of the proposed system, with future improvements suggested through the integration of GIS databases, alternative APIs, and the exploration of open-source alternatives.

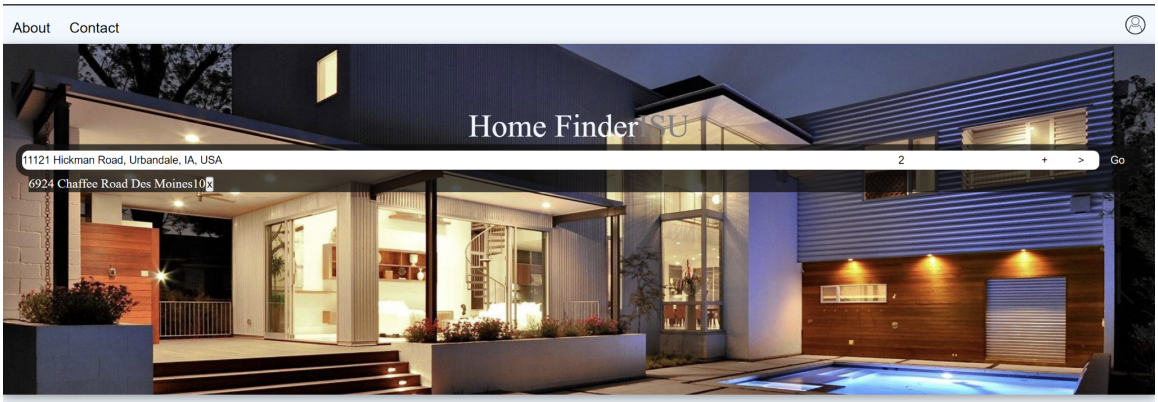
8.3 Appendices

8.3.1 Operation Manual

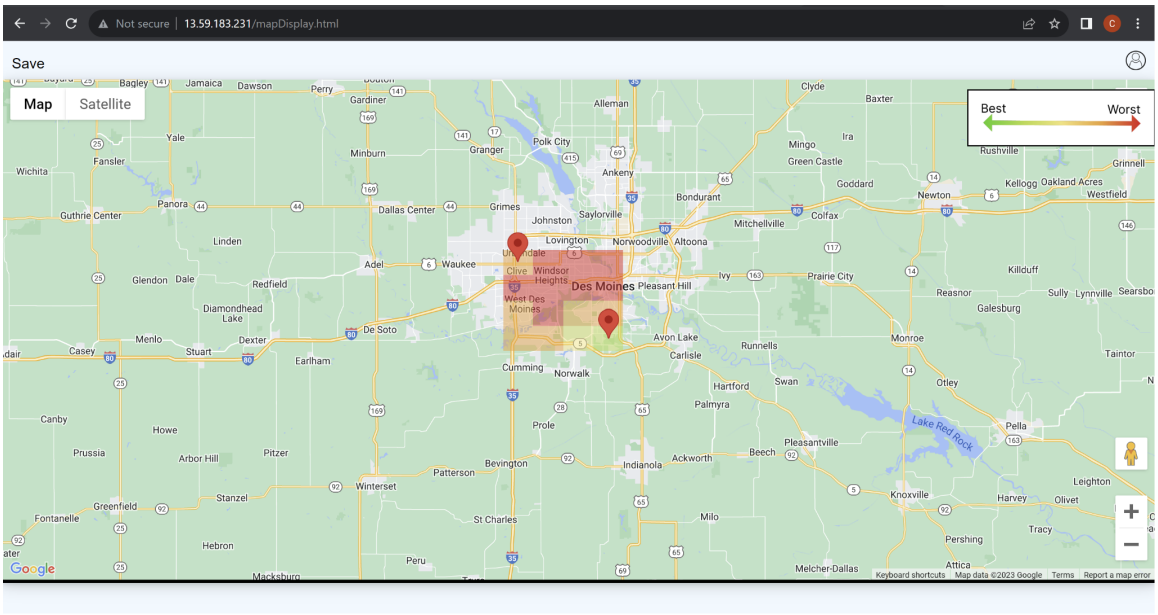
1. Connect to the Home page by navigating to http://13.59.183.231/
2. Sign up/ Login using the user icon at the top right (Optional)
   Logging in returns users to the home page.



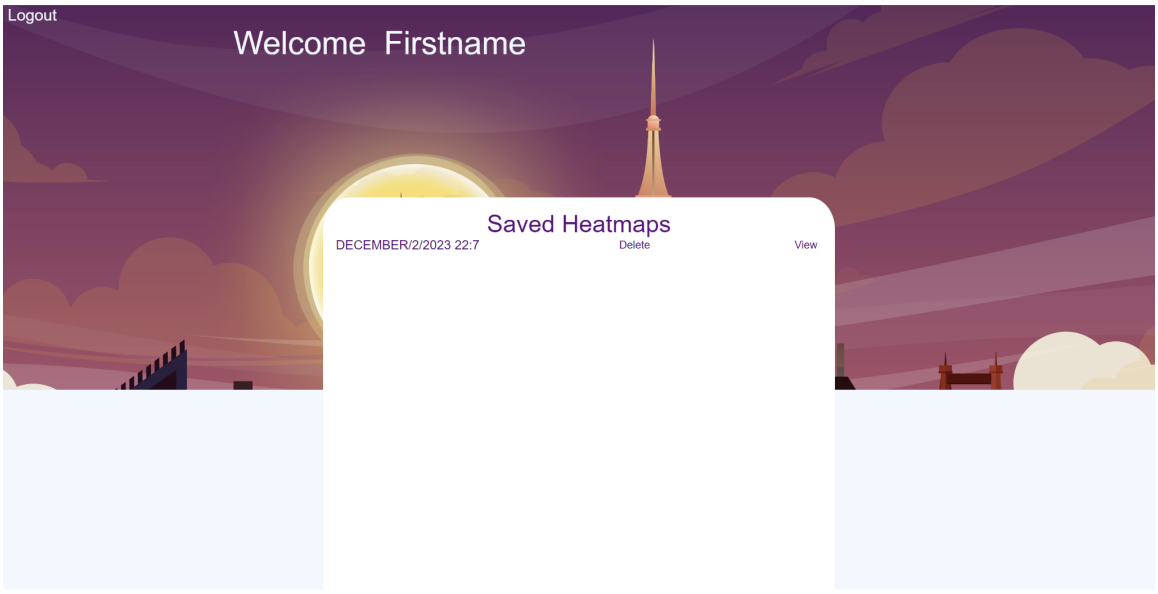
3. Enter the locations and frequencies you travel to in the input bar. As you type, locations will automatically appear under the input bar. Please select the correct location.



4. Click Go to produce the heat-map



5. If you are logged in to an account press the save button in the top left to access the heat-map again at a later date.
6. View your saved heat-maps by clicking on the icon in the top right corner. You will know you are logged in when the icon displays the first letter of your name instead of the empty user circle.



7. Click view to see the previously generated heat-map.

8.3.2 Team Contract

Team Members:

1) Christian Boughton 2) Michael Wieland

3) Daniel Chrisman 4) Lith Almadani

5) Ella Knott 6) _____

Team Procedures

Day, time, and location (face-to-face or virtual) for regular team meetings:

Group Meetings: Thursdays, 1:30 pm. ECPRE Lobby

Client Meetings: Thursdays, 2:00 pm. Advisors Office

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):

Discord - quick communication between team members, reminders, scheduling, etc
E-mail - communication with Advisor, more formal announcements

3. Decision-making policy (e.g., consensus, majority vote):

Majority vote with considerations for experience levels in relevant areas

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):

Minutes shared and archived through discord notes taken in rotations

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

Members are expected to attend all meetings, if a team member is not able to meet or plans to be late to the meeting, they should inform the rest of the team via Discord.

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

Everybody is expected to participate in team assignments and to respect and follow timelines, and deadlines. However, it is understandable that our schedules are different and our participation rate will be different from one assignment to another.

3. Expected level of communication with other team members:

On a weekly basis and everybody should communicate their absence on any team meeting or assignment

4. Expected level of commitment to team decisions and tasks:

Team members are expected to respect and commit to team decisions that were made through majority voting.

Team members are welcome to share their opinion concern and debate as long they commit to team decisions

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction,

individual component design, testing, etc.):

Client Interaction: Ella Knott

Testing: Lith Almadani

Team Organization: Christian Boughton

Documentation Organization: Michael Wieland

Development lead: Daniel Chrisman

2. Strategies for supporting and guiding the work of all team members:

Servant-leader roles are played by each individual in their field of expertise. They will use the abilities they have to focus on the growth of the group and well-being of the members. SMEs in their respective fields will lend their expertise to struggling group members in a respectable fashion for the betterment of each member.

3. Strategies for recognizing the contributions of all team members:

Attentiveness, timeliness and genuineness. Peer-to-peer recognition for work completed. Individual accomplishments in this project are posted on the group discord page for all members to see. Team accomplishments are included in lab reports and sent to the client/advisor. During our weekly meetings team members are encouraged to share their accomplishments in person.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the

team.

Christian Boughton -10 years of experience in the private field as a manager/supervisor allows me to see the big picture and day-to-day operations and how they intertwine to achieve an end goal. Previous experience in data integration and large data analysis.

Michael Wieland - Front-end design and testing, project management, and documentation experience from coursework. Currently in an internship/co-op working on front-end development that uses a geo-mapping service (MapBox). Light experience with database management.

Ella Knott - 3 years experience interning in cybersecurity. Primary focus on Vulnerability Management, light experience in front-end design, and API integration. I typically fill in the role of mediator and communicator in previous projects.

Lith Almadani- Developing, testing, and maintaining firmware for crop drying and monitoring systems. Developed Python testing scripts that use the CAN library to communicate with different modules.Experience using PSoC creator to design firmware for arm microcontroller to read, filter, and transmit different sensor data. Experience using CAN bus protocol to create a smooth network to transfer sensor data.

Daniel Chrisman - Front-end design and integration with the backend. Project management and documentation background from previous courses. Have worked with app development and work with data security. A background in field experience helps drive my communication skills within the team to ensure we are all on the same page when it comes to individual tasks to complete a larger-scale image.

2. Strategies for encouraging and support contributions and ideas from all team members:

Setting and sharing goals individually and as a team with announcements when they are reached. Healthy working environments that pay respect to each member's ideas.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will

a team member inform the team that the team environment is obstructing their

opportunity or ability to contribute?)

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

Secure access to all necessary APIs

Secure a web hosting platform

Deepen understanding of necessary tools/languages

Design the graphic interface to our website

2. Strategies for planning and assigning individual and teamwork:

Individual portions of the project will be decided during the Tuesday meeting time and will serve to accomplish the team's weekly goal.

3. Strategies for keeping on task:

Claim tasks bit by bit and work on them throughout the week. If you get stuck or overwhelmed, let the team know sooner than later.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

Communicate the infractions with the team first, and try to solve the root of the issue. If necessary, ask the advisor for help.

2. What will your team do if the infractions continue?

Communicate with Professor Shannon/Dr. Daniels depending on the semester about how to handle the uncooperative teammate.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the

consequences as stated in this contract.

1) Daniel Chrisman DATE 2/19/2023

2) Ella Knott DATE 2/19/2023

3) Michael Wieland DATE 2/19/2023

4)Lith Almadani DATE 2/19/2023

5) Christian Boughton DATE 2/19/2023