GRAND CANYON UNIVERSITY™

CST-451 Capstone Project Requirements Document

Scott Maxwell

Grand Canyon University

Instructor: Professor Mark Reha

Revision: 1

Date: July 27th 2024

ABSTRACT

CitySurf provides an interface in the form of a web application to view various statistics about different cities in the United States. To create this experience, a full stack application was created to provide a feature rich experience while having a minimal non-overwhelming interface to interact with the data. The goal is to provide a more consumable data visual tools that draw the eye to the conclusion.

Comprehensive graphs and charts are publicly available without a sign in for each city. The graphs representing the various metrics are job industry, salary, employment, health insurance, walk score, home prices, weather, rooms per household, vacancy, and possibly other data points. There will also be community contributed data which will require a login to see add contribute to. These data points include ratings (out of 5 stars) of Pollution, Safety, Friendliness, Attractions. A summary for the city will be made available near the top of the page to help readers get an overall sense of the city.

City Surf will be hosted in the cloud and use high security standards for encrypting the little user data that is collected (username and email) to lower the risk in the event of a data breach. The data itself will be collected via publicly available data sets from the Census Bureau, NOAA, and potentially other reliable sources. The summary for each City will be generated through prompting ChatGPT.

History and Signoff Sheet

Change Record

Date	Author	Revision Notes
07/27/24	Scott Maxwell	Initial draft for review/discussion

Overall Instructor Feedback/Comments

Overall Instructor Feedback/Comments

Integrated Instructor Feedback into Project Documentation

☐ Yes ☐ No

TABLE OF CONTENTS

Functional Requirements

Use Cases

Use Case or User Story	Approval Date	Justification
As a user, I would like to easily compare city statistics, so I do not need to view each city individually.	**N/a**	Users want to save cities for easy reference in the future, avoiding repeated searches.
As a user, I would like to save cities so that I can refer back to them.	**N/a**	Users want to save cities for easy reference in the future, avoiding repeated searches.
As a user, I would like to see community ratings of cities so I can get a more current sense of city's qualities.	**N/a**	Users seek up-to-date and relevant insights about cities through community ratings to make informed decisions.
As a user, I would like to type in a city to select it for viewing statistics.	**N/a**	Users need a convenient way to input and select cities by typing to quickly access desired city statistics.
As a user, I would like the typing field to correct my typos so that it finds the correct city.	**N/a**	Correcting typos in the city search field ensures users find the correct city even if they make mistakes.
As a user, I would like to select cities from an interactive map so that I can find cities I do not know the name of.	**N/a**	An interactive map allows users to discover and select cities they are unfamiliar with by visually exploring the options.
As a user, I would like to be able to request a password reset so I can recover my account.	**N/a**	Providing a password reset option helps users recover their accounts if they forget their passwords.
As a user, I would like each city to contain a small 2-3 sentence summary so that I can quickly understand the qualities of the city.	**N/a**	Users need quick and concise summaries for each city to efficiently grasp the city's key qualities at a glance.
As a User I would like to rate the city I currently live in when I sign in, so that I can contribute to the community metrics	**N/a**	Allowing users to rate their current city upon signing in encourages community participation and ensures the

		data reflects real-time user experiences.
As a user, I would like to be able to remove cities that I have saved.	**N/a**	Providing the option to remove saved cities helps users manage their preferences and keep their saved list relevant and up-to-date.
As a user, I would like to be alerted when my login credentials are invalid	**N/a**	Alerting users when their login credentials are invalid helps them quickly correct mistakes and ensures a smooth and secure login experience.
As a user, I would like to see the privacy policy, so I can know what data is collected	**N/a**	Disclosing what data is collected is important for providing transparency for users.

Non-Functional Requirements

Use Cases

Use Case or User Story	Approval Date	Justification
As a user, I want the web application to load within 3 seconds so that I can quickly access the information I need.	**N/a**	A fast loading time improves user satisfaction and reduces bounce rates.
As a system administrator, I want the application to handle up to 10,000 simultaneous users without performance degradation to ensure a smooth user experience.	**N/a**	Ensuring high performance under load prevents slowdowns and potential downtime, maintaining a positive user experience.
As a business owner, I want the application to support a 100% increase in user base annually so that it can grow with the business.	**N/a**	Supporting growth in the user base is crucial for scaling the business and capturing new market opportunities.
As a developer, I want the system to support horizontal scaling so that the load can be effectively distributed.	**N/a**	Horizontal scaling allows for the efficient handling of increased loads by adding more servers, ensuring system stability and performance.

As a user, I want the application to have an uptime of 99.9% annually so that it is available when I need it.	**N/a**	High availability ensures that users can rely on the application being accessible whenever they need it, which is critical for user trust and satisfaction.
As a system administrator, I want comprehensive logging of all significant events so that issues can be tracked and resolved.	**N/a**	Comprehensive logging is essential for diagnosing problems, ensuring security, and maintaining the integrity of the system.
As a DevOps engineer, I want real-time monitoring in place to detect and alert on performance or security issues to maintain system health.	**N/a**	Real-time monitoring allows for the immediate detection and resolution of issues, minimizing downtime and ensuring the system remains secure and performant.

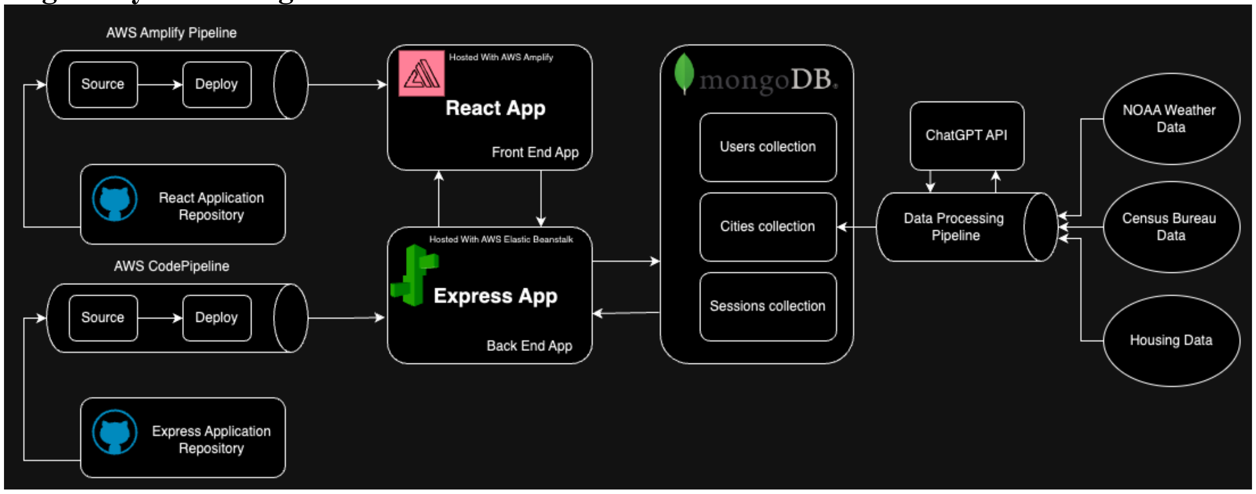
Technical Requirements

Use Cases

Tools and technologies used in the project.

Technology or Tool	Approval Date	Justification
ExpressJS (nodeJS)	**N/a**	A JavaScript framework for building the backend (API) that runs with nodeJS.
React	**N/a**	A JavaScript framework for building the frontend of the web application.
MongoDB Atlas	**N/a**	A Cloud service used to host a database for city data and user data.
Python	**N/a**	Used to automate downloading and retrieval of city data. Also used for formatting data into JSON format to be uploaded to database.
AWS Amplify	**N/a**	A cloud service used to host the frontend React application.
AWS Pipeline	**N/a**	A cloud service used to automatically deploy changes to the main branch of a Github repository to AWS Elastic Beanstalk.
AWS Elastic Beanstalk	**N/a**	A cloud service used to host the backend Express application
Plotly (JavaScript)	**N/a**	A JS library to plot graphs to display city statistics.
Mapbox	**N/a**	Used for an interactive map to select cities.
TypeScript		Utilizes static types and compiles to JavaScript. Used for both Express and React applications.
Github	**N/a**	Used for version control and Continual Integration and Deployment.
Winston & Winston-Loggly-Bulk	**N/a**	JS Libraries for Logging Backend.

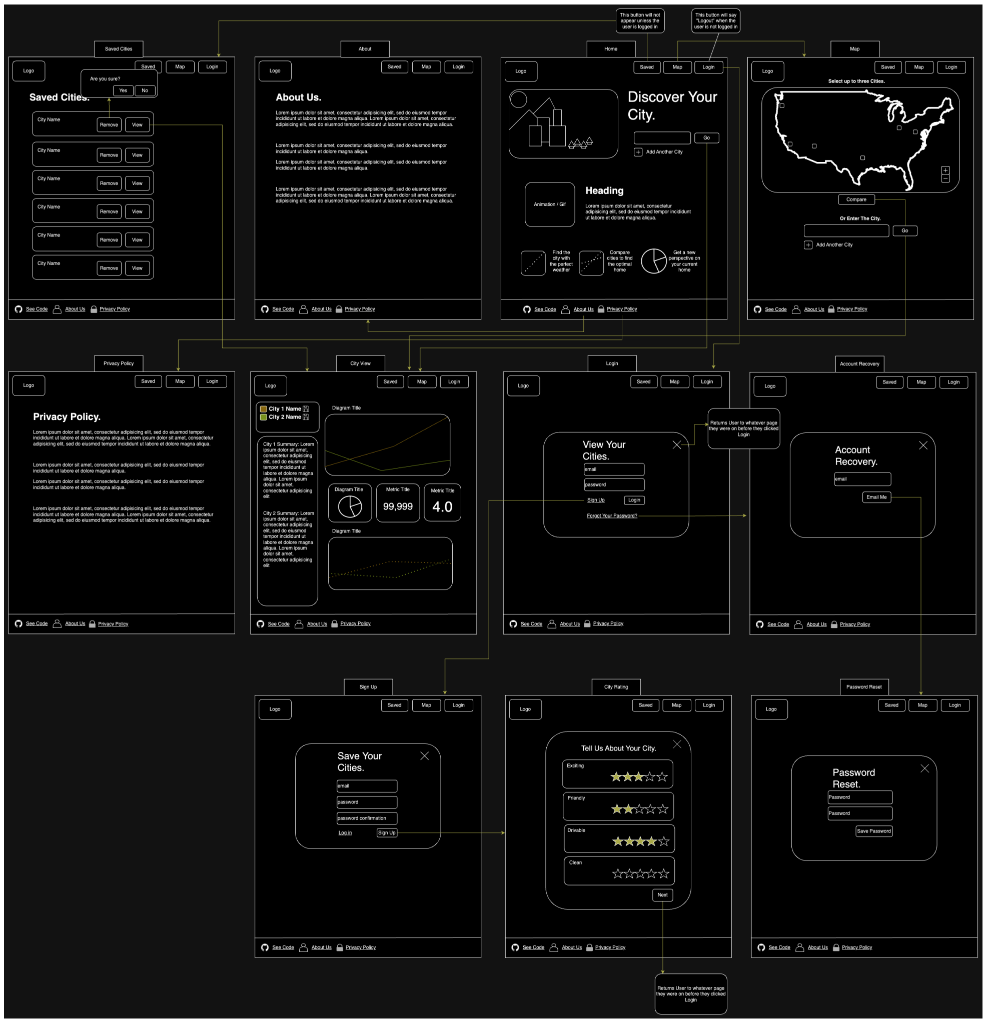
Logical System Design



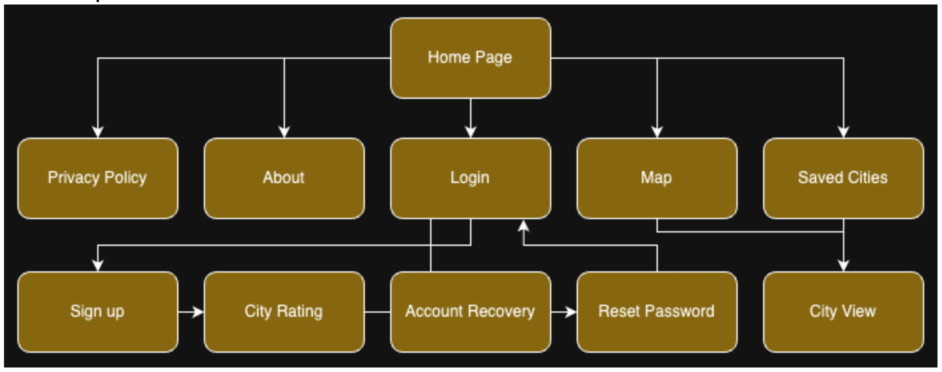
User Interface Design

See larger image here

Wireframes:



Site Map:



Reports Design

Reporting Use Case	Justification
Uptime Alerts	Uptime Robot will be used to continually monitor both Frontend and Backend applications, which will send email alerts when the service is experiencing a disruption. This will allow respective teams to respond to the disruption and investigate a fix.
Logging for Backend Access	Each time the Backend service is accessed, the application will create logs using Winston, and will integrate with Loggly.com for accessing these logs. Each API end point will provide the log and where the request is coming from. The front-end will not have any logging capabilities, as the application runs on the client side.
AWS Built-in Logging	Various logs are recorded for AWS Services that can be utilized for debugging purposes.