CEREO Living Atlas

Project Solution Approach

Center for Environmental Research, Education, and Outreach (CEREO)

February 13, 2025

Living Atlas Development Team

Garoutte, Zachary

Joseph, Jonathan

Gao, Yaru

TABLE OF CONTENTS

I. Introduction

The CEREO Living Atlas is a geospatial web application designed to visualize and share environmental data, with a primary focus on water quality in the Columbia River Basin while also having the capability to extend beyond this scope. The goal of this tool is to serve as a critical resource for researchers, tribal communities, and government agencies, enabling them to contribute and access geo-tagged data stored in a database through an interactive map interface. By fostering collaboration and transparency, the Living Atlas supports informed decision-making on critical environmental issues.

Initially developed by student teams at WSU over multiple years, the application provides basic functionality but struggles with performance and scalability, particularly under high user traffic. Usability challenges and a manual data input process also limit its effectiveness for broader adoption. This project seeks to address these limitations by enhancing the system's overall performance, automating workflows, and refining the user interface for a more convenient interaction. The goal is to deliver a robust and user-friendly tool that can support a growing user base and expanded functionality.

II. System Overview

The functionality of the application's system should be focused on providing a user-friendly experience for all users that will be navigating through the Living Atlas, while also accommodating support for features for more technical users, such as adding complex geospatial data types to the database. To achieve an intuitive user interface, we will implement features found on other GIS-based websites like Zillow which achieve seamless search and filtering for items on a map while being easy-to-use and compatible across all devices. For users adding data to the database, the system should support upload of shapefiles and/or GeoJSON files to represent the geospatial data as shown on the map. The system will be designed such that these geospatial data files can be transferred from a user upload at the application frontend to the backend and into the database, which the frontend will read from and display its data on the map.

III. Architecture Design

III.1. Overview

The Living Atlas is designed with a three-layer architecture, comprising the frontend, backend, and database. The frontend, developed using React.js, delivers an interactive and user-friendly interface, enabling seamless map visualization and intuitive user interactions. On the backend, FastAPI acts as the core intermediary, efficiently managing incoming API requests, authentication, and data processing. Supporting this all is a PostgreSQL database, enhanced with PostGIS, which securely and effectively stores geospatial data, user credentials, and metadata related to uploaded datasets. This modular approach ensures that each component can be updated or modified independently, enhancing the system's scalability, flexibility, and long-term maintainability.

A diagram of the basic subsystem layers of the architecture can be found in Appendix A.

III.2. Subsystem Decomposition

The system is structured into three core subsystems: the frontend, backend, and database. The frontend handles data visualization, processes user interactions, and provides tools for data submission. To enhance mapping capabilities, it leverages libraries such as Mapbox and Leaflet.js. The backend manages user authentication, enforces role-based access, validates data uploads, and delivers geospatial data through API endpoints. Meanwhile, the database efficiently organizes structured data, ensuring rapid retrieval and optimized querying through spatial indexing.

Frontend Subsystem

The frontend subsystem is responsible for managing user interactions and rendering geospatial data. It consists of key components such as the homepage, an interactive map, and data submission forms. By leveraging API calls, it seamlessly communicates with the backend to ensure smooth data processing and storage.

A diagram of the frontend subsystem can be found in Appendix B.

Backend Subsystem

Serving as the system's processing core, the backend manages user authentication, enforces access control, and validates geospatial data submissions. Positioned as the bridge between the frontend and the database, it ensures secure and efficient data flow. To optimize performance, various algorithms are implemented to streamline API request handling and minimize response times.

A diagram of the backend subsystem can be found in Appendix C.

Database Subsystem

The database subsystem is responsible for securely storing crucial system data, including user credentials, authentication records, and geospatial datasets. Built on PostgreSQL with PostGIS extensions, it is optimized for handling spatial queries with high efficiency. Its scalable architecture enables rapid data retrieval, advanced filtering, and indexing to ensure seamless access to stored information.

A diagram of the database subsystem can be found in Appendix D.

IV. Data design

The database design consists of structured tables to store geospatial data, user information, and role-based permissions. The **Geospatial Data Table** contains dataset entries, including spatial attributes, metadata, and associated files (e.g., shapefiles, GeoJSON). The **User Table** maintains account credentials, authentication tokens, and access levels. The **Permissions Table** defines role-based access to ensure only authorized users can upload or modify data. Data indexing and relational integrity are enforced through PostgreSQL with PostGIS, allowing for optimized querying and retrieval of large-scale environmental datasets.

There will be two major data structures present in the system database: The data structure containing all geospatial data points (including attached card information), and the data structure containing the account information of all users. Each object in the geospatial data structure will contain a shapefile so that the data can be displayed properly on the map in the correct location, and the card information for that data point, which can include the name of the user who added the data, the user's email, the organization who collected study data, a link to the study dataset, the title of the linked dataset, a description of the data set, the data category, and search tags. Each object in the account information data structure will contain a user ID, username, email, encrypted password, and the user's authorization level (view only, authorized to add data, or admin).

V. User Interface Design

The CEREO Living Atlas provides an interactive web interface for users to explore and manage environmental data. The homepage serves as the main point of interaction. Users can search, filter, and navigate data efficiently, while authorized users can contribute and manage geospatial information. The features described here align with those listed in the Use Cases section.

The homepage offers several key features. Users can search for data using filters in the top navigation bar. Geospatial data can be added through the "Add" button in the cards section. Viewing data is straightforward, with coordinates and shapes displayed on the left-side map and corresponding cards on the right. Users can also remove data from the same card section. Bookmarked data is accessible in this area as well. For diagrams illustrating these features, refer to Appendix E.

Editing data is available by clicking "Learn More" on a card, which opens a detailed window containing an edit option. Additionally, this window also allows authorized users to delete a card, update information and attached files as needed. For visual references of this feature, see Appendices F and G.

The user center includes account management functions. Password resets require entering the current password and a new one. This ensures security while keeping the process simple. A diagram of this feature can be found in Appendix H.

The homepage navbar provides access to key pages, including the registration page and administrator dashboard. The registration page allows non-logged-in users to fill out a form requesting an account. Submitted requests are sent to the admin panel, where an administrator can approve or deny them. For a diagram of this process, see Appendix I and J, respectively.

The administrator dashboard displays a list of all registered users in the Living Atlas. Admins can manage user accounts by modifying access levels or deleting accounts. This ensures proper role-based access control within the system. For a detailed view of the admin dashboard, refer to Appendix K.

VI. Glossary

API (Application Programming Interface) – A set of rules and protocols that allow different software applications to communicate with each other.

Backend – The server-side logic of an application responsible for processing requests, handling data storage, and managing business logic.

Columbia River Basin – A geographic region covering the watershed of the Columbia River, which is the primary focus of the Living Atlas for water quality data.

Database – A structured collection of data stored electronically, which in this case includes geospatial data, user credentials, and metadata.

FastAPI – A modern web framework for building APIs with Python, known for its speed and ease of use.

Frontend – The part of the application that users interact with directly, typically consisting of a graphical user interface (GUI).

Geospatial Data – Information that includes geographic location attributes, often represented using coordinates and spatial features.

GeoJSON – A format for encoding geographical data structures using JSON.

GIS (Geographic Information System) – A system designed to capture, store, manipulate, analyze, and display spatial or geographic data.

Leaflet.js – An open-source JavaScript library used for interactive maps.

Living Atlas – A geospatial web application designed for visualizing and sharing environmental data, focusing on water quality in the Columbia River Basin.

Mapbox – A mapping platform providing tools for designing and implementing custom maps.

Metadata – Data that provides information about other data, such as descriptions, timestamps, and ownership details.

PostGIS – A spatial database extender for PostgreSQL that enables advanced geospatial queries.

PostgreSQL – An open-source relational database system used for managing structured data.

Role-Based Access Control (RBAC) – A security mechanism that restricts access to system functions based on user roles.

Scalability – The ability of a system to handle growth in users, data, or computational workload without performance degradation.

Shapefile – A common geospatial vector data format used for geographic information system (GIS) applications.

Spatial Indexing – A technique used in databases to optimize spatial queries, improving the efficiency of geographic data retrieval.

User Authentication – The process of verifying the identity of users before granting access to the system.

VII. References

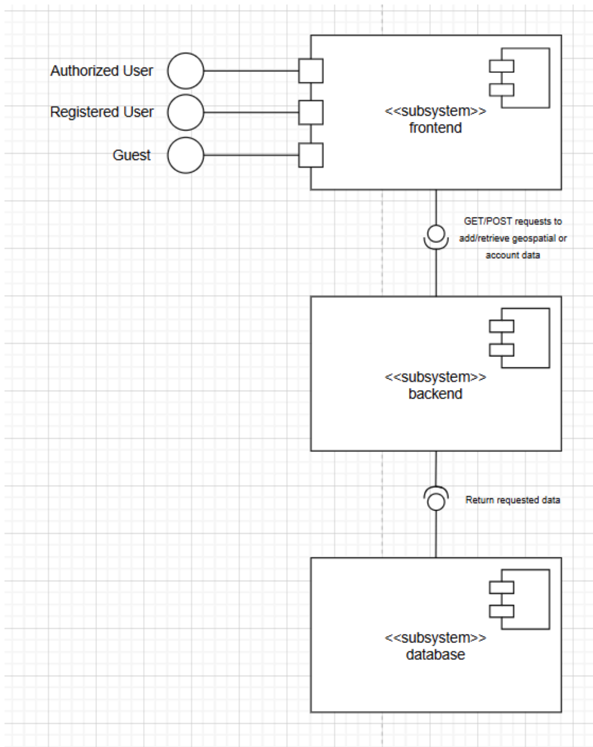
Bruegge, Bernd., Dutoit, Allen H.. Object-oriented Software Engineering: Using UML, Patterns, and Java. United Kingdom: Prentice Hall, 2010.

VIII. Appendices

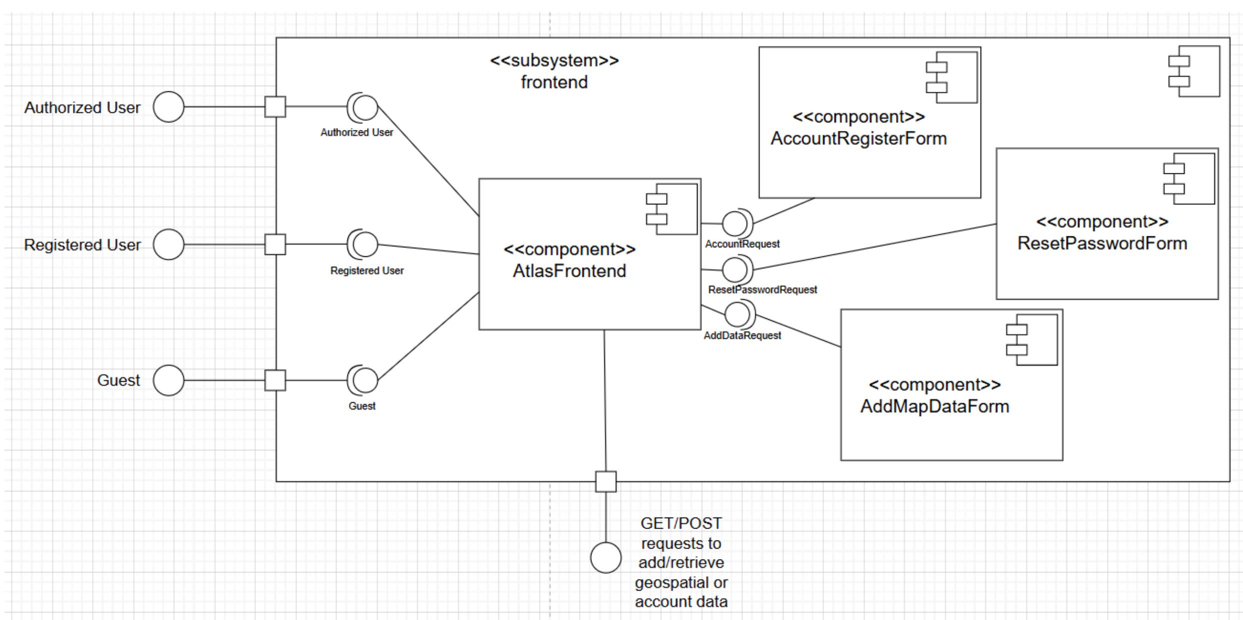
Any larger images, charts, or external materials should be put into appendices. These are attached at the end of the document, so the main materials are kept closer together and the overall flow of the document is preserved. If you include 4 pages of spreadsheets in the middle of a section, it makes it very difficult to track the flow of your presentation. Instead, those sheets go in Appendix [X] and are referred to by the earlier document.

You may have as many appendices as you need for the document to make sense.

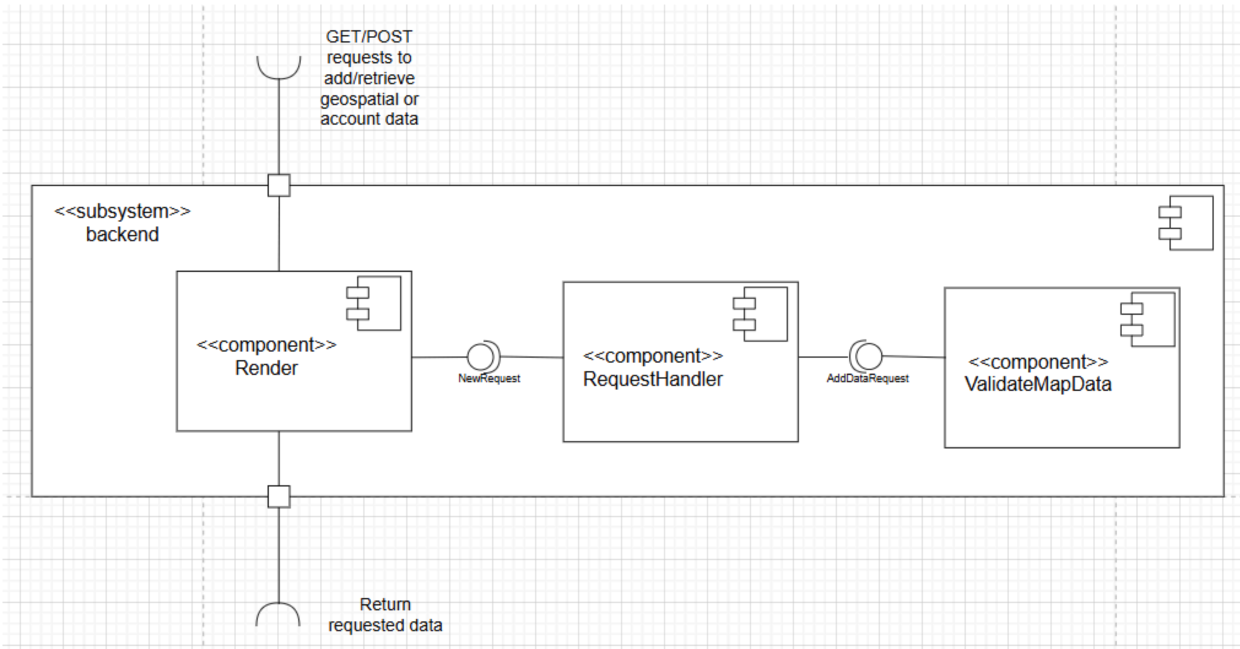
Appendix A: Architecture Design Overview Diagram



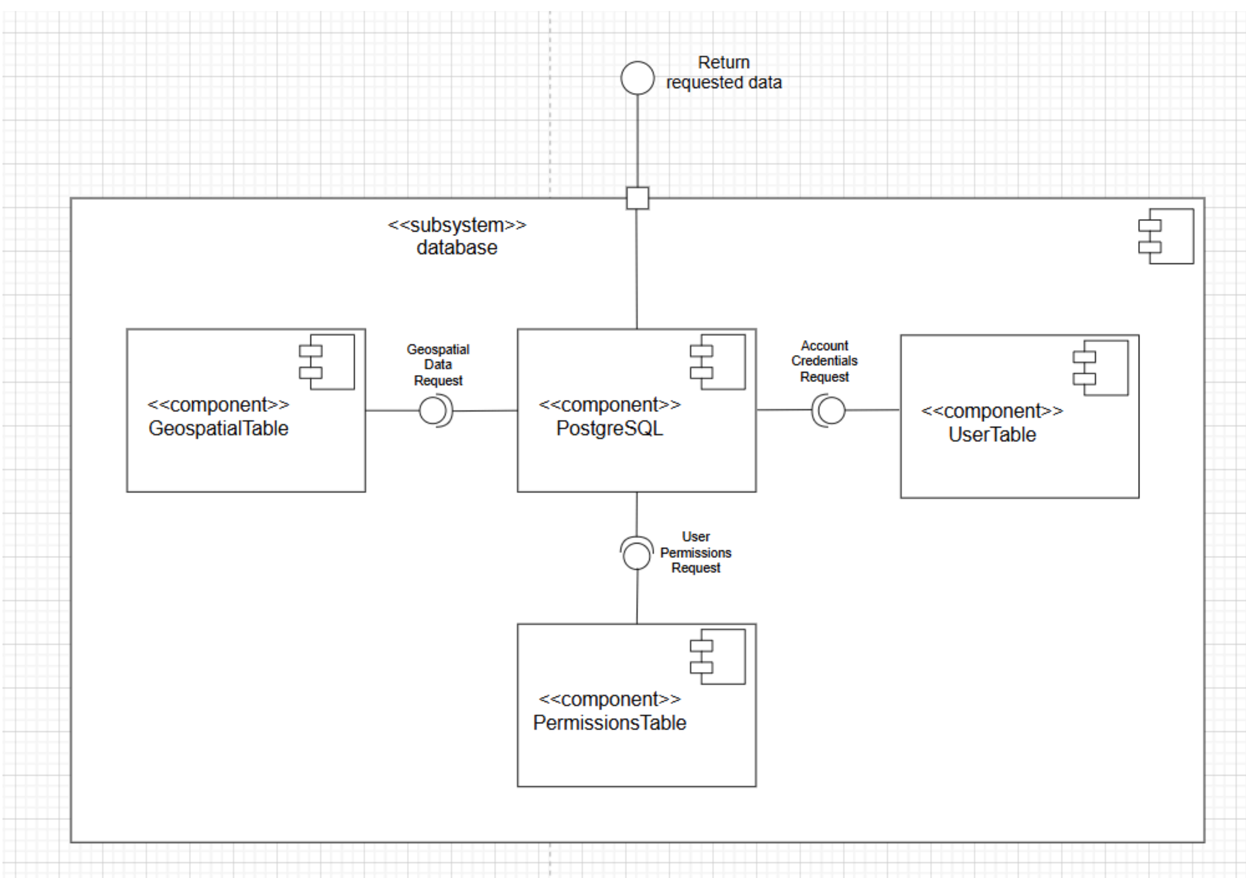
Appendix B: Frontend Subsystem Diagram



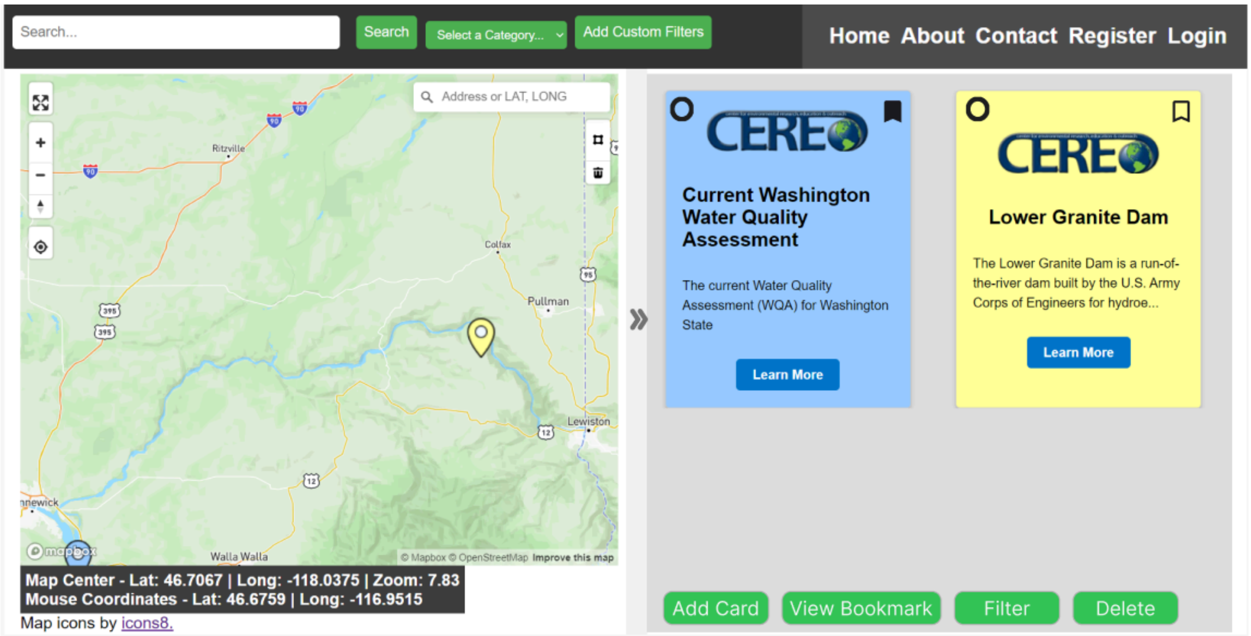
Appendix C: Backend Subsystem Diagram



Appendix D: Database Subsystem Diagram

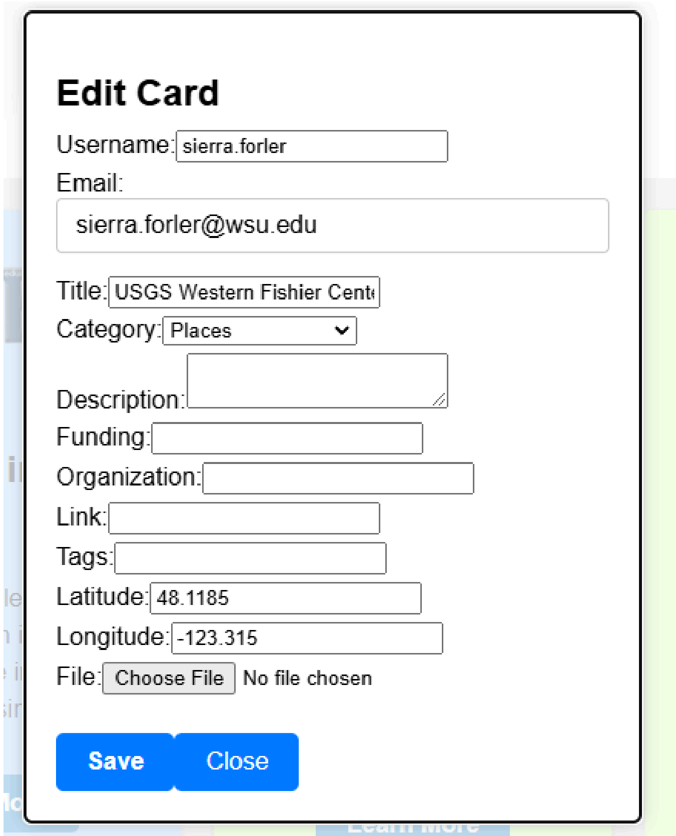


Appendix E: Mock-up diagram of the homepage, illustrating search, filtering, data addition, viewing, removal, and bookmark access.

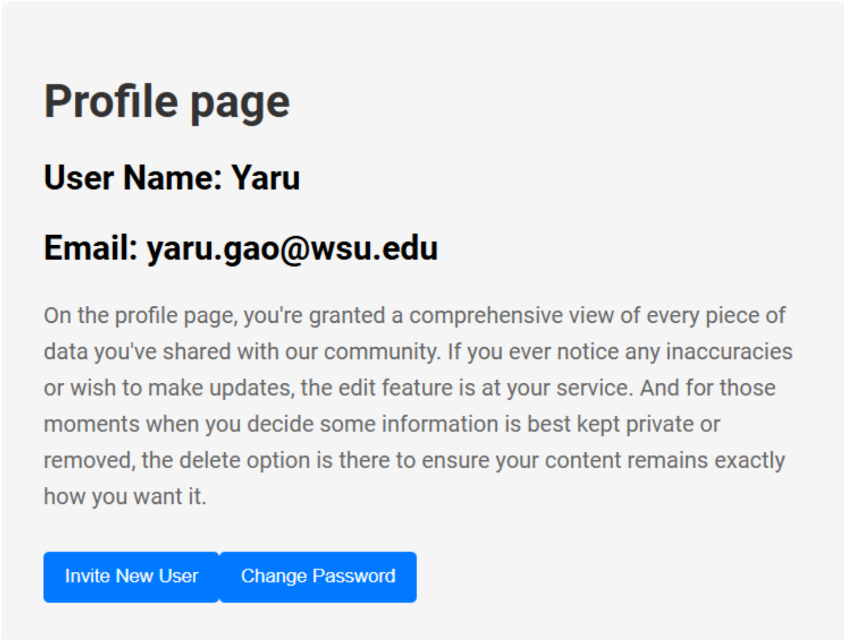


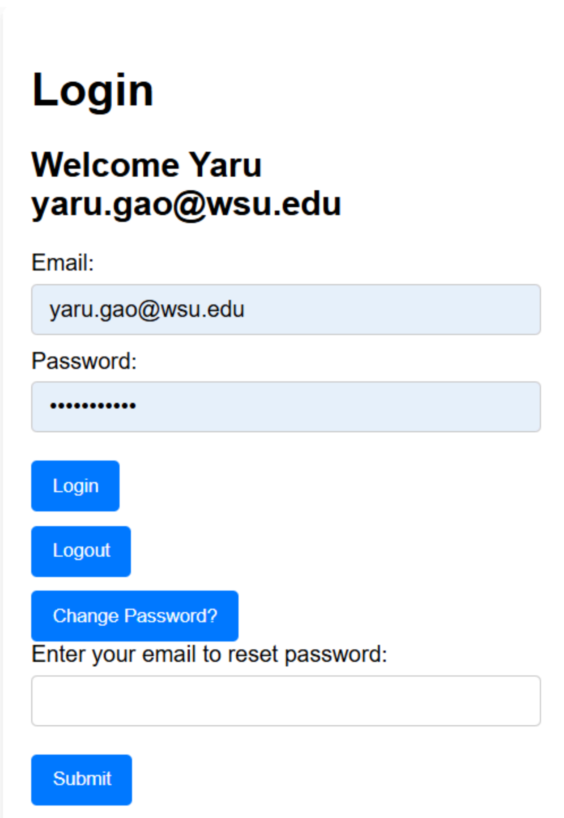
Appendix F: Mockup diagram of the "Learn More" window, showing the edit option for modifying geospatial data.

Appendix G: Mockup diagram of the edit functionality within the "Learn More" window.

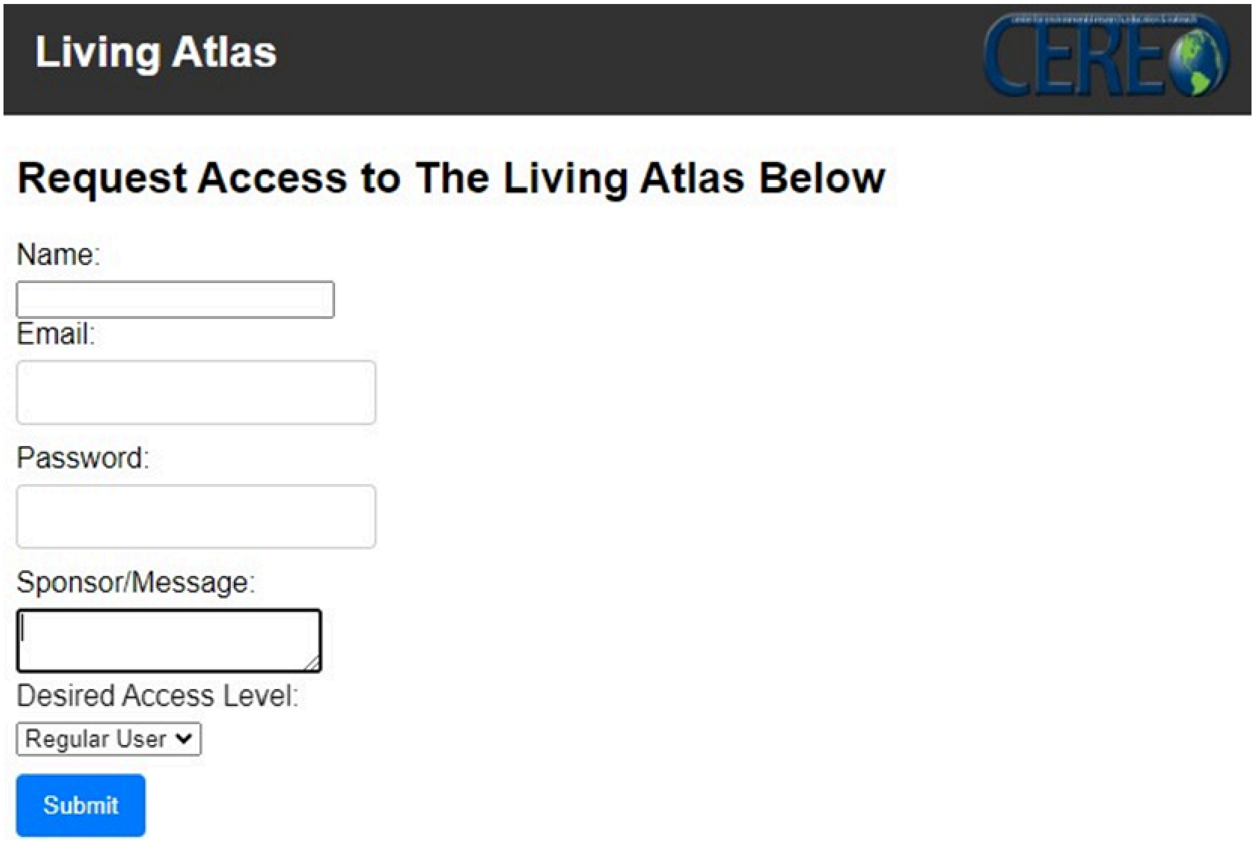


Appendix H.1 and H.2: Mockup diagram of the user center, depicting the password reset process.

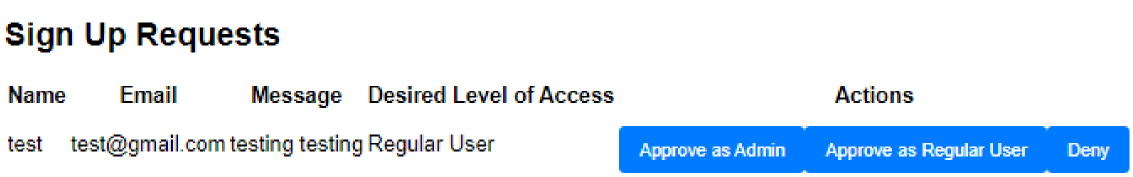




Appendix I: Mockup diagram of the registration page, showing the user account request form.



Appendix J: Mockup diagram of the administrator panel, illustrating user request approvals.



Appendix K: Mockup diagram of the administrator dashboard, showing user management features for modifying access levels and deleting accounts.

Josh	joshua.long@wsu.edu	Regular User	Change Role	Delete
Sierra	sierra.svetlik@wsu.edu	Regular User	Change Role	Delete
Mitchell	mitchell.kolb@wsu.edu	Regular User	Change Role	Delete
userj	j@j	Regular User	Change Role	Delete
usec	c@c	Regular User	Change Role	Delete
userk	k@k	Regular User	Change Role	Delete
userg	g@g	Regular User	Change Role	Delete
Jan	j.boll@wsu.edu	Regular User	Change Role	Delete
userf	f@f	Regular User	Change Role	Delete
Bryce	bryce.moser@wsu.edu	Regular User	Change Role	Delete
zuser	z@z.com	Regular User	Change Role	Delete
Julie	julie.padowski@wsu.edu	Regular User	Change Role	Delete
Silas	silas.peterson@wsu.edu	Admin	Change Role	