Development of a Web-Based Platform for Biodiversity Tracking and Visualization

Ruth Mary Kurian Chandy

June 27, 2025

Abstract– This final report details the fulfillment of the final degree project focused on developing a web-based platform designed to track and visualize biodiversity data. The project consists of the operational implementation of core biodiversity tracking functionalities, with the backend infrastructure fully established using Django and MongoDB. Frontend development has completed all planned features including observation visualization, user management, and the ability for users to contribute through manual observation entry. This user-generated data complements information integrated from the iNaturalist API, with all geolocation details enriched via OpenStreetMap. This document outlines the project's objectives, context, state of the art, methodologies employed, detailed implementation outcomes with visual evidence, conclusions, and outlines future work beyond the current project scope.

Keywords– Biodiversity, Citizen Science, Web Platform, Django, Vue.js, MongoDB, Data Visualization

1 Introduction

Effective biodiversity tracking and monitoring are paramount in the face of rapid environmental changes, habitat loss, and climate change. Citizen science [6][10] plays a crucial role in this endeavor by enabling individuals to contribute to valuable biodiversity data, which is essential for scientific research and conservation efforts. This project aims to design and develop a functional and user-friendly web-based platform to facilitate the tracking, visualization, and analysis of these contributions.

Developed entirely from scratch, the platform empowers users to document biodiversity observations, explore species distributions, and engage with a community of citizen scientists. This foundational achievement enabled full customization to meet biodiversity tracking needs, resulting in an intuitive interface for data submission, interactive maps for visualizing geographic trends, and tools for users to manage their contributions. By integrating verified data from established biodiversity networks, the platform ensures reliability while encouraging public participation. These features collectively support the project's broader goal of democratizing biodiversity monitoring and fostering environmental awareness.

This report provides a comprehensive overview of the project's motivation, its significance, achievements, and is structured as follows: The Objectives section defines the project's goals and technical scope. Context and State of the Art reviews existing platforms and methodologies in biodiversity tracking. Methodology and Development Process details the implementation of the backend (Django/MongoDB) and frontend (Vue.js). Discussion and Quality of Results evaluates the deployed platform's features through visual evidence. Finally, Conclusions and Future Work summarizes achievements and proposes enhancements. The Appendix provides additional screenshots of the implemented system.

- Contact email: RuthMary.Kurian@autonoma.cat
- Mention: Software Engineering
- Supervised by: Mehmet Oguz Mulayim
- Academic Year: 2024/25

2 Objectives

The primary objective of this project is to develop a web-based platform that allows users to upload, search and track field observations and identify species, contributing to a comprehensive database of biodiversity information. Specifically, the project aims to:

- Focus initially on insect observations, as they represent one of the most diverse and well-documented taxonomic groups in biodiversity databases, making them ideal for platform testing and validation.
- Design and implement a robust backend infrastructure using Django[2] and MongoDB[8] for efficient data management.
- Develop a dynamic and intuitive frontend with Vue.js[3] for user interaction and data visualization.

- Integrate data from existing public biodiversity databases, specifically the iNaturalist API[1][14], to enrich the platform's data pool.
- Implement features for users to upload their own observations, including manual observation entry, species identification, and location data.
- Provide tools for visualizing biodiversity data, such as species distribution maps and observation trends.
- Ensure data quality through mechanisms like duplicate removal and data enrichment (e.g., using OpenStreetMap[13] for geolocation).
- Create a user-friendly interface that encourages citizen participation and contributes to biodiversity conservation efforts.

3 Context and State of the Art

Biodiversity tracking and monitoring have evolved significantly with advancements in technology and the increasing adoption of citizen science. This section reviews existing platforms, methodologies, and technologies relevant to biodiversity conservation and data management.

3.1 Existing Platforms

Platforms like iNaturalist[1], GBIF[16] (Global Biodiversity Information Facility), eBird[17], eButterfly[18], and Observation.org[19] have demonstrated the immense potential of citizen science in collecting and disseminating biodiversity data.

iNaturalist is a leading citizen science platform where users can record and share observations of plants, animals, and other organisms. It leverages a community of experts to assist with species identification, making it a valuable tool for both amateur naturalists and researchers.

GBIF operates as a global network, providing free and open access to biodiversity data aggregated from institutions worldwide. Its extensive dataset supports large-scale research and policy-making by offering standardized biodiversity records.

eBird, developed by the Cornell Lab of Ornithology, specializes in bird observations. It allows birdwatchers to log sightings while contributing to scientific studies on avian distribution, migration patterns, and population trends.

Similarly, **eButterfly** focuses exclusively on butterfly observations, offering tools for data verification and visualization tailored to lepidopterists and conservationists.

Observation.org, a Europe-based platform, supports observations of all species and features strong mobile applications and regional verification systems, making it particularly useful for localized biodiversity monitoring.

For the development and validation of our system, access to substantial biodiversity observation data was needed. Establishing a new community data collection effort would be impractical within the project timeframe. Therefore, leveraging existing platforms with publicly available data became a necessary approach to demonstrate the system's capabilities. These external sources provide immediate access to verified observations at scale, enabling us to focus on developing and testing our analytical framework rather than data collection.

For this project, iNaturalist was chosen as the primary data source due to its comprehensive nature, providing both observation and taxonomic data in a unified format. While the current implementation focuses on iNaturalist, the framework could potentially be extended to incorporate additional publicly available biodiversity data sources in future work. This approach maintains focus while allowing for future expansion of data integration capabilities.

3.2 Technologies and Methodologies

Building a solid web platform for tracking biodiversity means picking the right tools and approaches. Recent studies emphasize how crucial smart data management systems are for helping us protect and monitor biodiversity [4]. You could think of these systems as the backbone for handling all the varied and often huge amounts of data that come from observing nature.

Today, web platforms, mobile apps, and powerful data analytics aren't just nice to have; they're essential for keeping tabs on biodiversity effectively [5]. They give us the framework to collect, store, process, and display all that valuable information. Plus, using well-designed APIs (Application Programming Interfaces) is super important. They're like bridges that let different data sources talk to each other smoothly, making it easy to pull in and combine information [1]. This approach also aligns with the idea of open science, where making data accessible and reusable helps everyone involved in research [9].

Citizen science plays a massive role here, with more and more people contributing to biodiversity monitoring [6][10]. As Chao (2024) points out, citizen science initiatives are a big help in tracking biodiversity changes, especially in urban areas, by getting the public involved [10]. And Callaghan et al. (2021) even suggest that when citizen science data is handled well, it can open up exciting new avenues for biodiversity research [6]. To really make the most of these public contributions, we need robust tech frameworks behind them.

For this project, a key part of our approach involves using existing systems and tools to handle data processing and ensure quality. We are plugging into advanced features from the external data sources we use. For instance, the iNaturalist API is central to our platform. It doesn't just provide observation data; it also brings with it iNaturalist's own AI-powered species identification capabilities, where it gives you a subset of the possible species, which are a huge help for users trying to figure out what they've spotted [1]. Our system incorporates multiple validation measures to ensure data quality and accuracy. A key component of this process involves enhancing location data through integration with the OpenStreetMap Nominatim API [13]. This service enables reverse geocoding of the raw latitude and longitude coordinates provided with each observation.

The reverse geocoding process extracts structured location information at three administrative levels. First, it identifies the nearest city or populated place name to provide local context. Second, it determines the broader administrative region or state where the observation occurred. Finally, it establishes the country location, ensuring proper geographic classification of all records.

This makes sure our geographic data is precise, which is vital for creating accurate biodiversity maps. Storing location data in GeoJSON[12] format within MongoDB also lets us run native geospatial queries, which is beneficial for our interactive mapping features.

In a nutshell, bringing all these tech pieces together—efficient data management (thanks to Django and MongoDB), user-friendly interfaces (with Vue.js), powerful APIs (like iNaturalist and OpenStreetMap), and smart data processing—is what makes this platform effective. It helps us support biodiversity conservation and encourages everyone to get involved.

4 Methodology and Development Process

The project was implemented from the ground up using an agile development methodology, with all core components custom-built to meet the project's specific biodiversity tracking requirements. While the iNaturalist API serves as the primary data source, the entire processing pipeline was developed specifically for this initiative, representing a complete technical implementation rather than an adaptation of existing solutions.

The source code for this project is publicly available in a GitHub repository, which includes the backend implementation in Django, the frontend components in Vue.js, and the necessary configuration files for MongoDB and Docker. The repository can be accessed at: https://github.com/ruthie-2003/biodiversity-tracker.

4.1 Backend Development

The Django backend employs a multi-layered architecture that cleanly separates data ingestion, processing, and delivery concerns. At the data layer, MongoDB's document model accommodates the heterogeneous nature of biodiversity observations while supporting high-performance geospatial queries through its native GeoJSON implementation.

The processing layer implements custom deduplication logic using compound indexes on observation metadata, along with an asynchronous task queue for geocoding operations. For data integrity, we established referential constraints between observations, taxonomic records, and user submissions while maintaining audit trails of all data transformations.

Containerization via Docker[15] ensures consistent execution environments across development and deployment stages, with particular attention to network isolation for secure API communications. The architecture's modular design allows individual components (data ingestion pipelines, geospatial services, etc.) to scale independently based on demand.

4.2 Backend Data Validation

The data validation code follows Django's testing framework patterns while addressing MongoDB-specific requirements, creating a validation layer that spans data integrity, business logic, and security requirements.

The data validation code thoroughly validates data input requirements, ensuring proper handling of mandatory fields including geographic coordinates, observation dates, and taxonomic information. Invalid submissions are rejected with appropriate error messages, while valid observations progress through the complete processing pipeline. This includes verification of both complete taxonomic classifications (family/genus/species) and partial classifications where default values are automatically applied.

4.3 Backend Unit Testing

The backend implementation incorporates a suite of unit tests that systematically assert correct functionality under both complete and incomplete data conditions. These tests validate core components by simulating real-world scenarios where users submit observations with varying levels of detail, ensuring the system handles partial classifications, missing geolocation data, or incomplete metadata without failure.

Media handling receives special attention in the testing regimen, with dedicated test cases verifying proper processing of image and audio attachments. These tests validate the complete lifecycle of media files from upload through storage management to final URL generation. Security tests form another critical component, thoroughly validating JWT (JSON Web Token) token verification and access control mechanisms for protected endpoints.

The test implementation employs mock objects to isolate external dependencies such as the OpenStreetMap API during test execution, ensuring consistent and reproducible results. Edge case coverage includes scenarios involving missing or malformed input data, duplicate observation detection, location resolution failures, and media processing errors. This testing methodology provides strong confidence in backend reliability while maintaining the flexibility to extend system functionality in future iterations.

4.4 Frontend Development

All planned frontend features have been fully implemented using Vue.js core components. This includes the observation visualization, manual observation upload and species visualization functionalities, as well as the user management interfaces such as login, registration, profile editing, and the admin dashboard.

The visualization dashboard supports interactive filtering by taxonomic classification, geographic region, and temporal range. The front-end architecture which consists of components such as Comment Section, Filter Panel, Forgot Password, Image Gallery, Charts, Map View, Observation Card, Reset Password, Search Bar and Successful Banner, has been simplified through a component-based Vue.js implementation that emphasizes reusability.

To efficiently handle complex data transformations and prepare data for display, taxonomic relationship visualization functionality has been integrated directly into the backend using MongoDB's aggregation framework, a powerful tool that allows for multi-stage data processing pipelines, reducing client-side processing requirements.

The frontend implements rigorous data validation checks to ensure data integrity before submission. For new observation uploads, the system validates: (1) mandatory taxonomic classification (family/genus/species), (2) valid geographic coordinates through interactive map selection or OpenStreetMap geocoding, (3) observation date formats, and (4) media file types (images/audio). Invalid submissions are blocked with descriptive error messages, while the interface provides real-time feedback, such as dynamic location suggestions and coordinate auto-population from map clicks—to guide users toward complete and accurate data entry. These client-side validations complement backend checks.

4.5 System Architecture

The platform follows a client-server architecture. The Vue.js frontend interacts with the Django backend via RESTful APIs[1]. The Django backend communicates with the MongoDB database for data storage and retrieval.

The following sequence diagram shows how the system populates insect taxonomy from the iNaturalist API and later processes user observation requests. It validates species, resolves locations, and stores data in MongoDB before returning results.

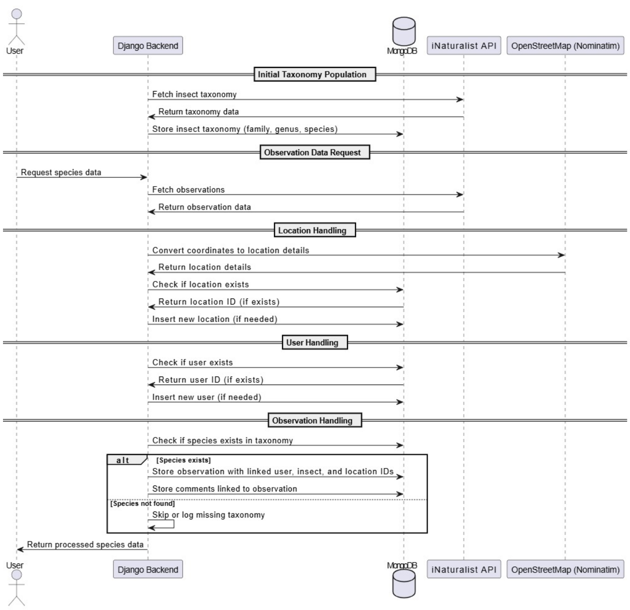


Fig. 1: Data flow sequence diagram

5 Discussion and Quality of Results

The biodiversity platform has been implemented with core components for data exploration, visualization, and community engagement. The system integrates iNaturalist API-sourced observations with interactive mapping tools and taxonomic databases to deliver a research-grade (verified) citizen science experience. Below we describe the key components of the deployed interface.

When you click on a specific observation (Figure 2), you'll find all its details neatly organized - what species it is, when and where it was spotted, and who shared it. The page keeps the original iNaturalist information and the link while making it easier to understand, with clear labels and helpful extras like a photo gallery. There's also a space for users to comment about the findings.

The home page (Figure 3) welcomes users with a friendly search box that instantly suggests species and users as you type, much like how search engines predict what you're looking for. Bright number counters show how many species, observers, and sightings the platform tracks, updating as new data comes in. A scrolling feed of recent discoveries gives a lively sense of the community's activity, with each card showing what was found, where, and by whom.

Species pages (Figure 4) collect everything known about a particular insect in one place. You can see where it lives on a map, how often it's been spotted historically, and browse recent observations of that particular species from different observers. The system organizes this information so pages load quickly, even when showing hundreds of sightings.

The maps and charts section (Figure 5) turns raw data into visual stories. The interactive map lets you zoom in to see sightings in your neighborhood or zoom out for a global view. Simple filters help narrow down by family, genus and species of insects. The timeline graphs reveal when different species appear throughout the year, helping spot seasonal patterns with just a glance. All these tools work together smoothly, updating instantly when you change your search.

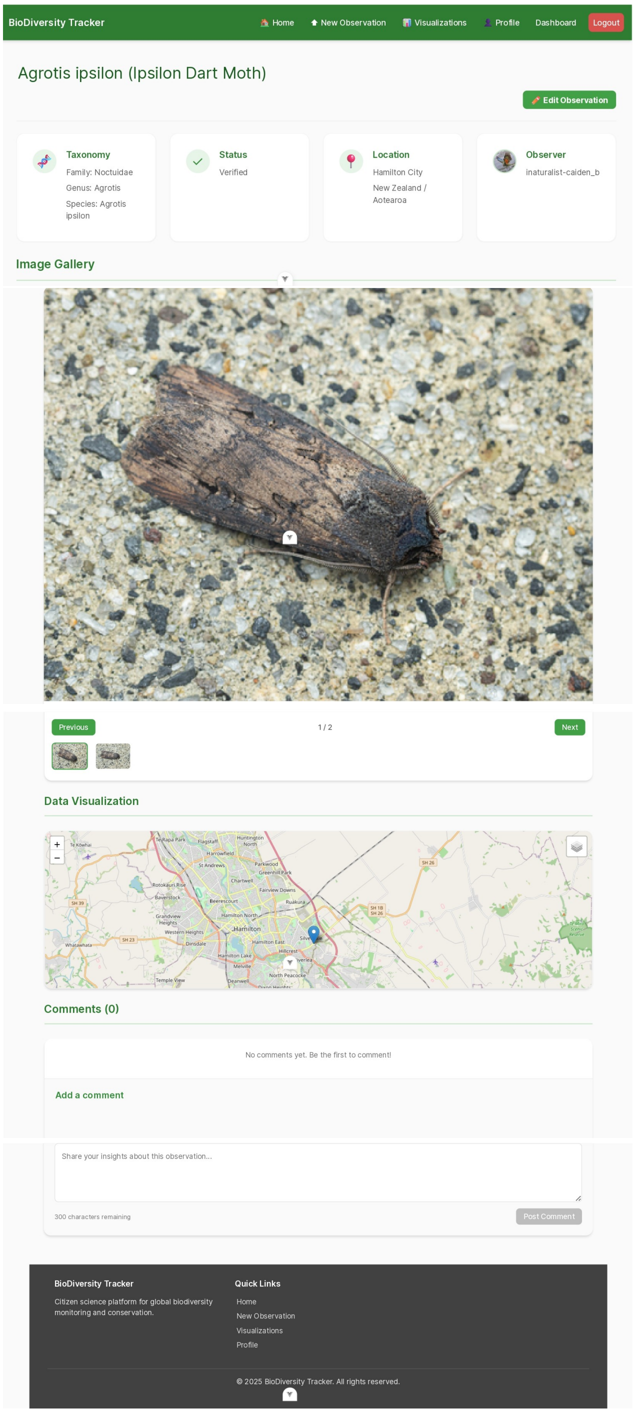


Fig. 2: Observation detail interface implementation showing complete biological record with: taxonomic classification, verification status badge, geolocation data with OpenStreetMap integration, and observer information.

The "New Observation" page (Figure 6) offers a streamlined interface for users to contribute new biodiversity records. Users can specify the species taxonomy, input the observation date, and quantity. Critically, the location input is highly flexible, allowing users to either type in a location name or directly select a point on the interactive map. The map feature automatically populates coordinates, ensuring precise spatial data capture. Users can also upload photos or audio files and add descriptive notes to their observations, making the submission process comprehensive and user-friendly.

The "User Profile" page (Figure 7) provides a personalized hub for each registered user. It displays a summary of their submitted observations, including a table with details like date, species, location, and status. Key statistics such as total observations, verified entries, and comments are prominently featured, along with a recent activity feed. Users can edit their profile information via a dedicated section, and also have access to data exporting options as CSV or JSON files for their submitted observations, with these functionalities strictly accessible only to the authenticated user or an administrator.

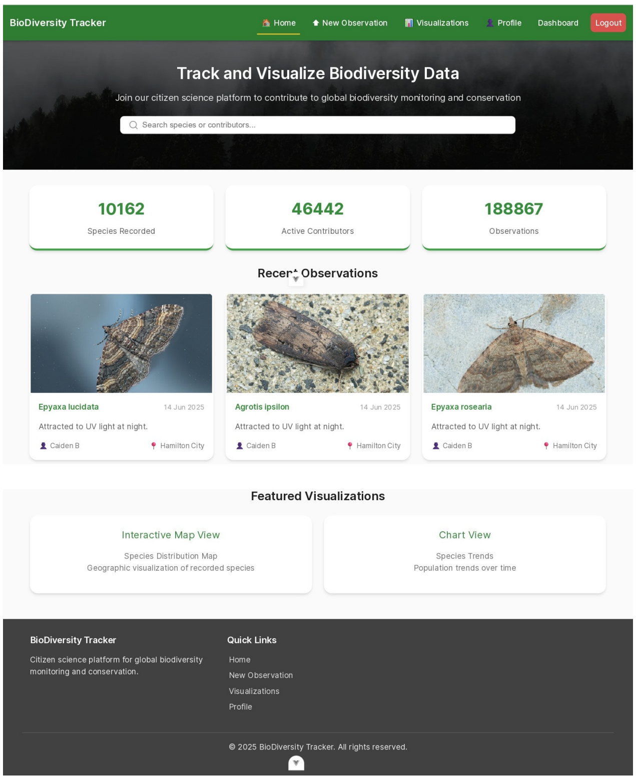


Fig. 3: Implemented home page showing search functionality, observation carousel, and primary navigation elements.

The "Edit Profile" page (Figure 8) allows authenticated users to manage their personal information. Here, users can update their display name, modify their profile photo (including options to edit or remove it), and provide a short biography to describe their interests in biodiversity. The page also prominently displays activity statistics such as the number of observations, species, and locations contributed, offering a concise overview of their engagement with the platform. All changes are saved securely, ensuring user data privacy and control.

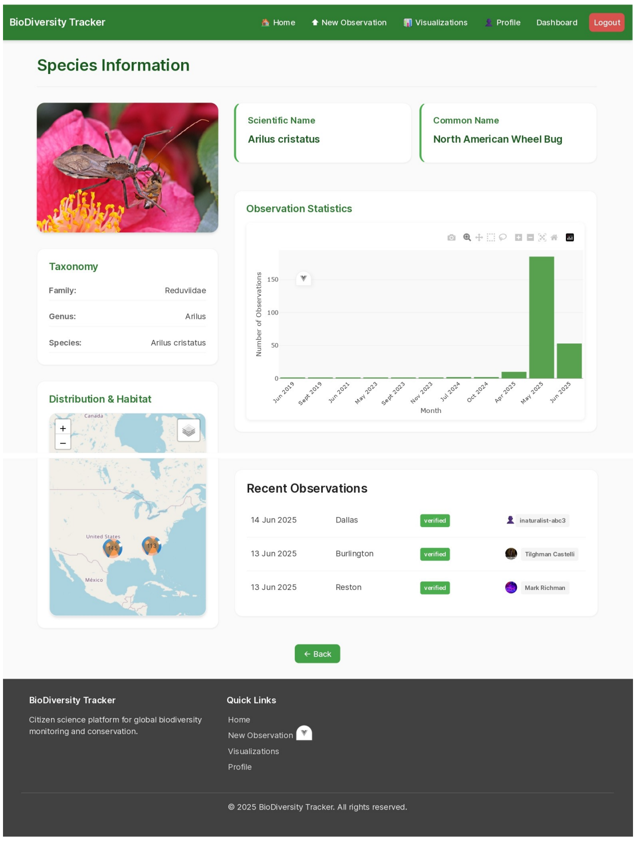


Fig. 4: Species detail page showing: scientific and common names, taxonomic hierarchy, geographic distribution map with occurrence points, and recent species observations.

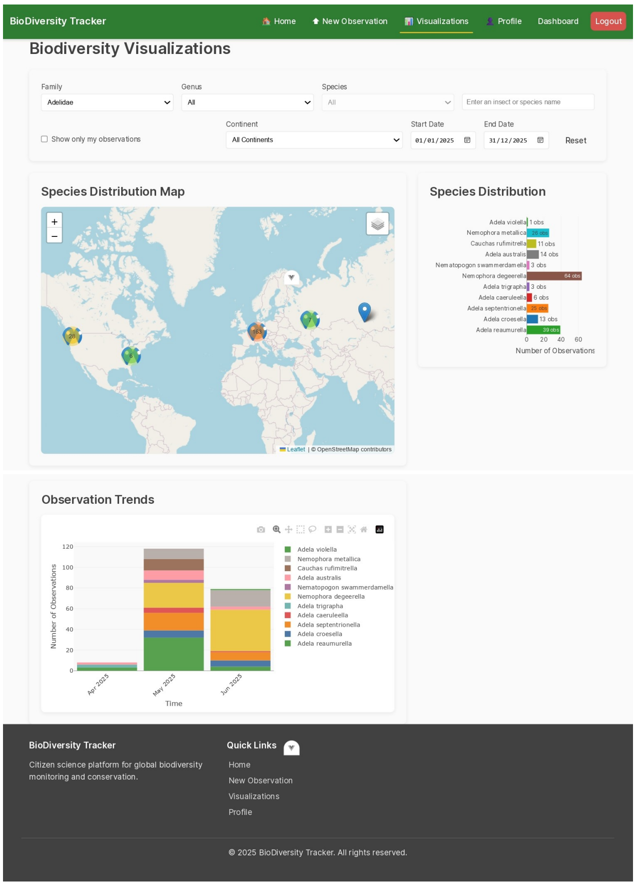


Fig. 5: Biodiversity Tracker visualization dashboard showing: species distribution map, observation trends, and filter controls.

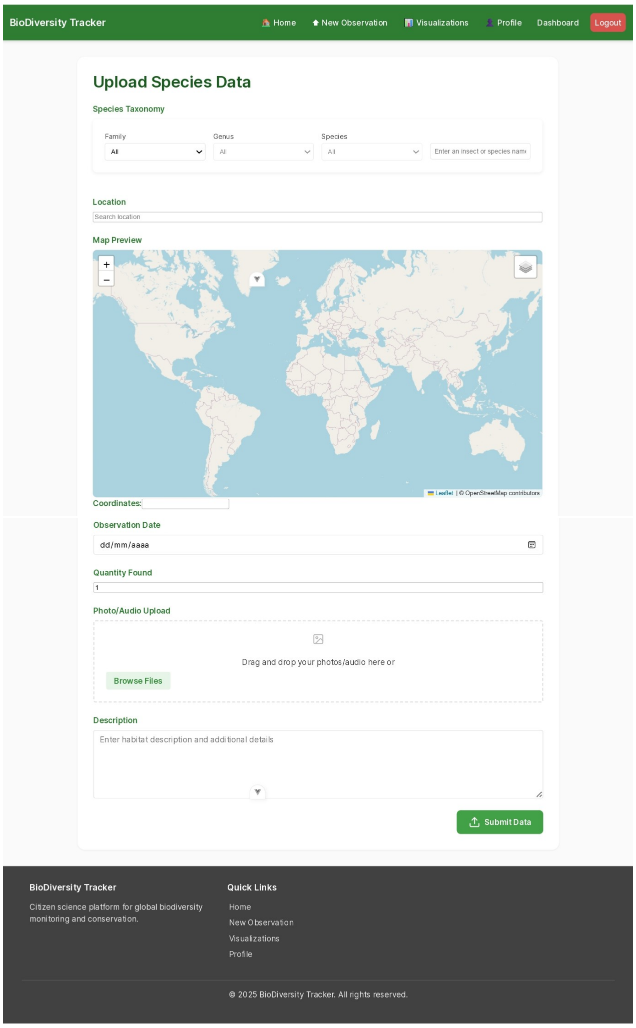


Fig. 6: New Observation interface implementation showing fields for species taxonomy, date, quantity, location input with map selection, and media uploads.

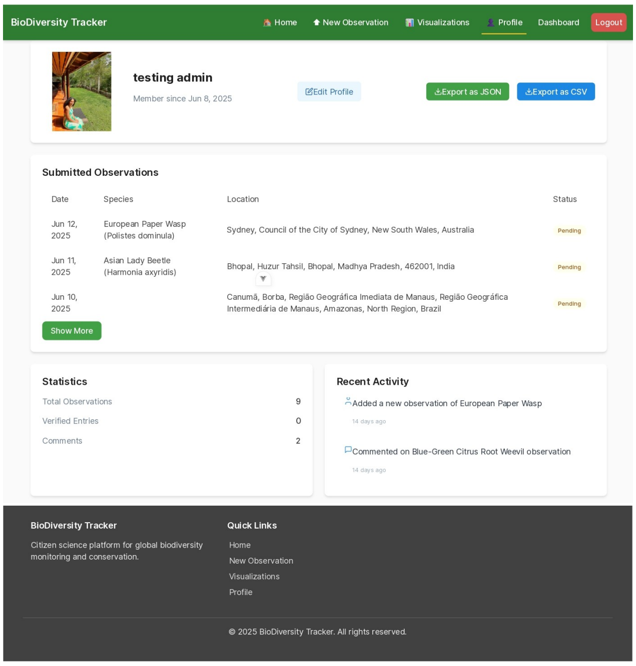


Fig. 7: User Profile interface implementation showing user's submitted observations, statistics, recent activity, and options to edit profile and export data.

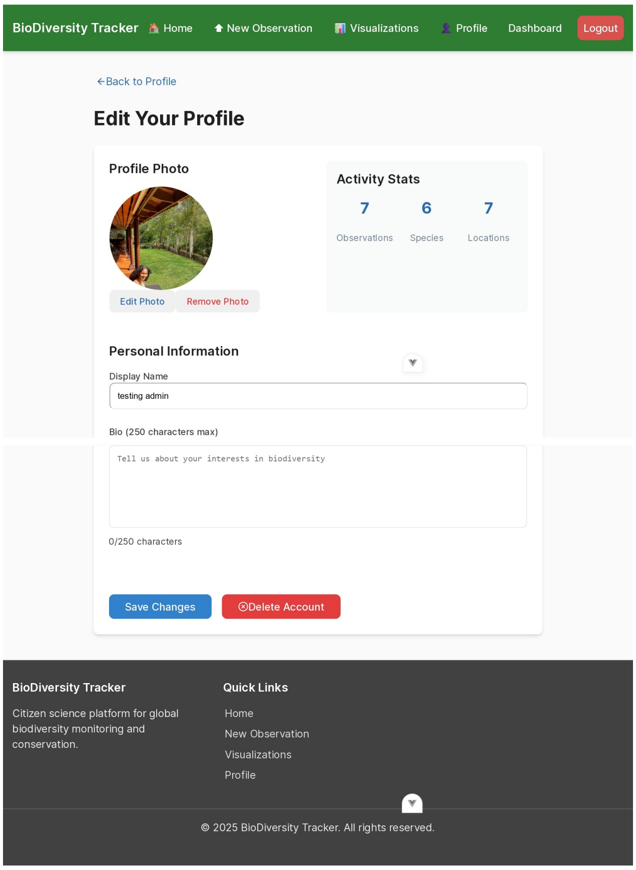


Fig. 8: Edit Profile interface implementation showing options to update profile photo, display name, biography, and view activity statistics.

The "Admin Dashboard" (Figure 9) provides a centralized control panel for platform administrators. It offers comprehensive summary statistics, including total contributions, active users, pending reviews, and data quality metrics, giving an overview of the platform's health and activity. The dashboard also features dedicated sections for user management, allowing administrators to search for and manage user accounts, and observations review, where they can oversee and moderate submitted observations. This is very crucial feature in a citizen science platform to ensure the reliability of the biodiversity data. Quick actions for exporting data in CSV or JSON format are available, facilitating administrative tasks and data analysis.

The "Login" page (Figure 10) serves as the secure entry point for registered users. It features fields for entering an email or username and a password, ensuring that only authenticated users can access personalized sections of the platform. For convenience, options are provided for users who may have forgotten their password, and a clear link is available for new users to register an account, streamlining the access and onboarding process.

The "Register" page (Figure 11) allows new users to create an account on the platform, serving as the gateway to full participation. It requires users to provide their full name, choose a unique username, enter their email address, and set a secure password with confirmation. This clear and structured form ensures that all necessary information is collected for account creation, while also providing a convenient link for users who already have an account to sign in directly.

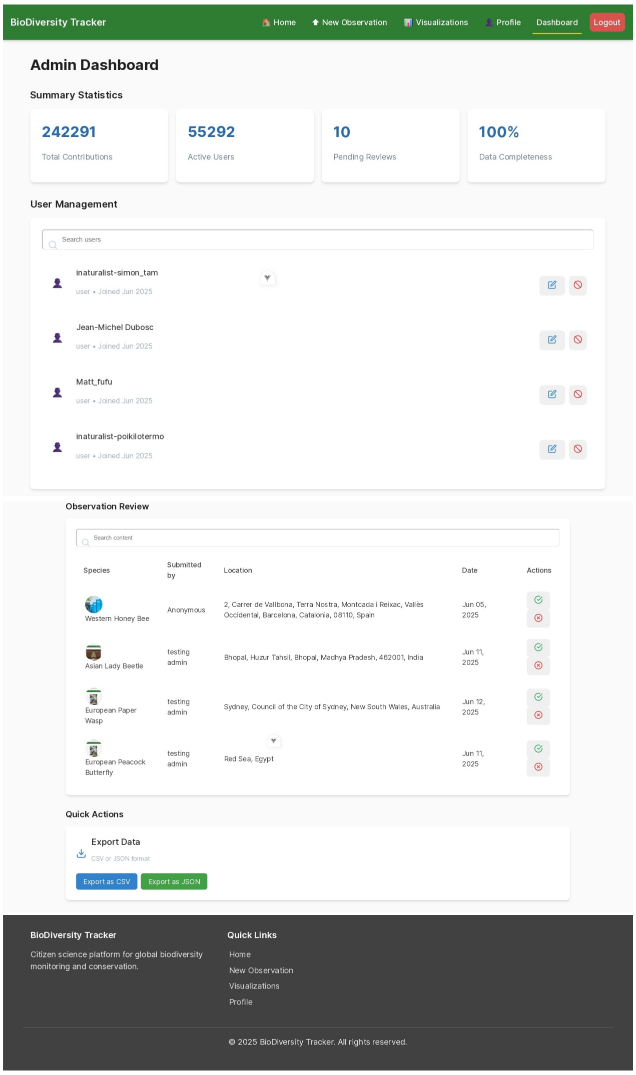


Fig. 9: Admin Dashboard interface implementation showing summary statistics, user management, content review, and data export options.

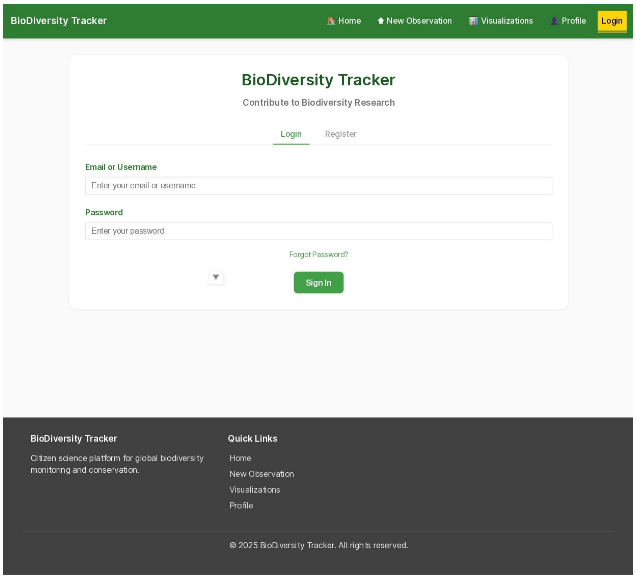


Fig. 10: Login interface implementation showing fields for email/username and password, with options for forgotten password and new user registration.

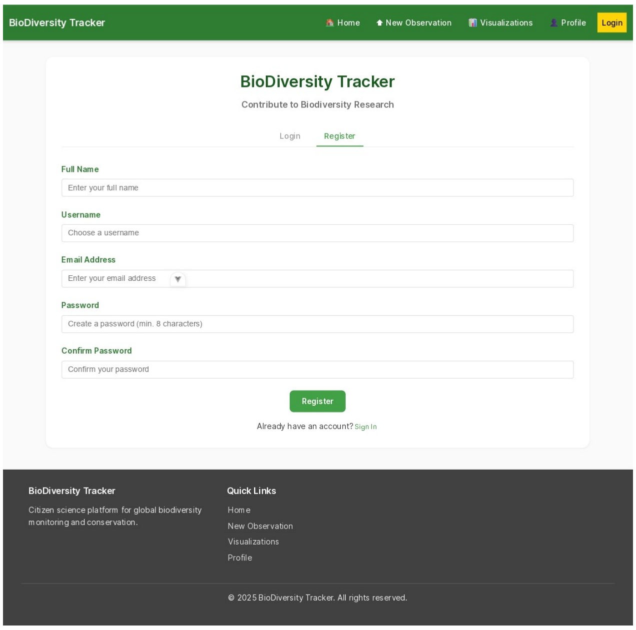


Fig. 11: Register interface implementation showing fields for full name, username, email, password, and confirm password, with a link for existing users to sign in.

6 Conclusions and Future Work

The completed system validates the technical viability of the proposed architecture, meeting all design requirements for biodiversity data processing, visualization, and user interaction. Developed entirely from scratch, this custom implementation demonstrates how tailored solutions can effectively address biodiversity tracking needs. The iNaturalist API has proven sufficiently comprehensive for implementation. MongoDB's flexible document model has effectively accommodated the heterogeneous biodiversity data while supporting performant spatial queries. The component-based Vue.js frontend has demonstrated good responsiveness even with large observation datasets, with all planned features, including user management and observation submission, fully implemented. The complete source code is available in the project's GitHub repository: https://github.com/ruthie-2003/biodiversity-tracker.

The development of the web-based biodiversity tracking and visualization platform has successfully established the core backend infrastructure and implemented all key frontend functionalities. The integration of iNaturalist data and OpenStreetMap for geolocation enrichment has created a robust data foundation. Working on the data visualization components was particularly exciting for me, as it represented new territory in my development experience. Implementing interactive maps and dynamic charts to represent biodiversity patterns proved both challenging and rewarding, and I recognize these visualization skills as increasingly crucial in today's professional environment where data-driven decision making is paramount. Extensive unit tests validate all critical functionality, including: complete and partial observation submissions, media file handling, error conditions, and user authentication. Functional testing was performed by manually entering sample data to verify the functionality of various pages and features, ensuring seamless operation across all components. The platform successfully demonstrates the potential of citizen science in contributing to biodiversity conservation and monitoring efforts.

Future work will primarily focus on refining the existing functionalities and enhancing user experience. Immediate next steps include implementing features such as improved table sorting, minor UI/UX adjustments for enhanced usability, and general performance optimizations to ensure a smoother experience. These targeted improvements aim to polish the current platform, addressing minor enhancements that will contribute to its robustness and user satisfaction.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Professor Mehmet Oguz Mulayim, for his invaluable guidance and unwavering support throughout this project. His expertise and constructive feedback were instrumental in navigating various challenges and achieving the project's objectives. Beyond his academic direction, Professor Mulayim fostered a supportive and comfortable learning environment, which greatly facilitated my work and allowed me to approach each phase with confidence, regardless of the outcomes. His mentorship has been truly appreciated.

References

[1] iNaturalist API Documentation. (2025). Retrieved from https://www.inaturalist.org/pages/api+reference

[2] Django Documentation. (2025). Retrieved from https://docs.djangoproject.com/

[3] Vue.js Guide. (2025). Retrieved from https://vuejs.org/guide/

[4] Urbano, F., Viterbi, R., Pedrotti, L. et al. (2024). Enhancing biodiversity conservation and monitoring in protected areas through efficient data management. Environ Monit Assess 196, 12. https://doi.org/10.1007/s10661-023-11851-0

[5] J, J., & Magrath, M. J. (2021). A Comprehensive Overview of Technologies for Species and Habitat Monitoring and Conservation. BioScience, 71(10), 1038-1062. https://doi.org/10.1093/biosci/biab073

[6] Callaghan, C. T., Poore, A. G., Mesaglio, T., Moles, A. T., Nakagawa, S., Roberts, C., Rowley, J. J., Vergés, A., Wilshire, J. H., & Cornwell, W. K. (2021). Three Frontiers for the Future of Biodiversity Research Using Citizen Science Data. BioScience, 71(1), 55-63. https://doi.org/10.1093/biosci/biaa131

[7] Plotly. (2025). Histograms in Python. Retrieved from https://plotly.com/python/histograms/

[8] MongoDB Documentation. (2025). Retrieved from https://www.mongodb.com/docs/

[9] Suter, S., Barrett, B., & Welden, N. (2023). "Do biodiversity monitoring citizen science surveys meet the core principles of open science practices?" *Environmental Monitoring and Assessment*, 195, Article 295. https://doi.org/10.1007/s10661-022-10887-y

[10] Chao, W. (2024). "The Role of Citizen Science in Monitoring Biodiversity Changes in Urban Areas." *Journal of Biology and Today's World*, 13(5), Article 009. https://www.iomcworld.org/articles/the-role-of-citizen-science-in-monitoring-biodiversity-changes-in-urban-areas.pdf

[11] Ullah, F., Saqib, S., & Xiong, Y.C. (2025). "Integrating artificial intelligence in biodiversity conservation: bridging classical and modern approaches." *Biodiversity and Conservation*, 34, 45–65. https://doi.org/10.1007/s10531-024-02977-9

[12] GeoJSON Specification. (2016). Retrieved from https://geojson.org/

[13] OpenStreetMap Nominatim API Documentation. (2025). Retrieved from https://nominatim.org/

[14] pyinaturalist: Python Client for iNaturalist API. (2025). Retrieved from https://pypi.org/project/pyinaturalist/

[15] Docker Documentation. (2025). Retrieved from https://docs.docker.com/

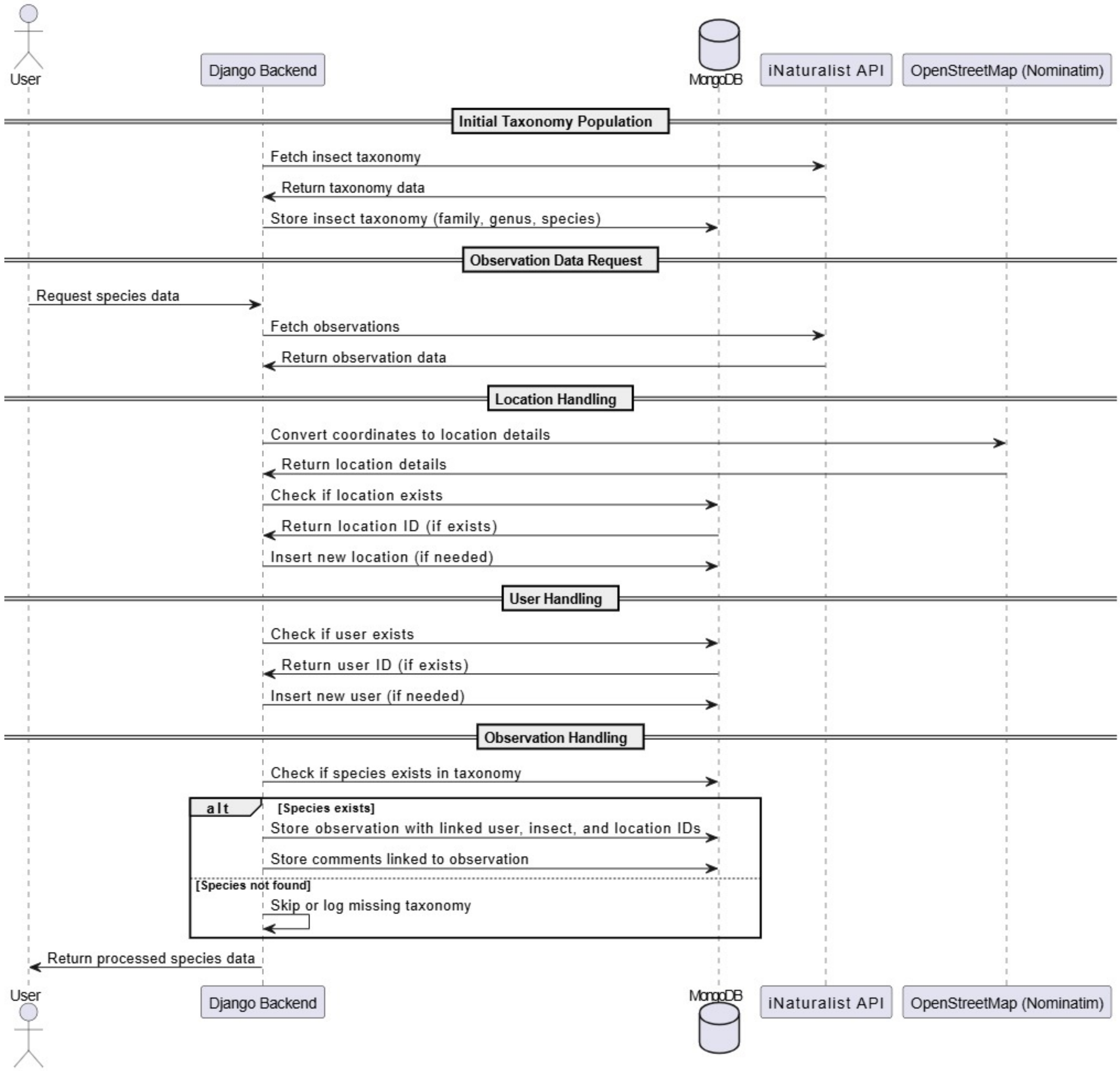
[16] GBIF API. (2023). Retrieved from https://www.gbif.org/developer/summary

[17] Cornell Lab of Ornithology. (2025). "eBird Data Use." Retrieved from https://ebird.org/science/use-ebird-data

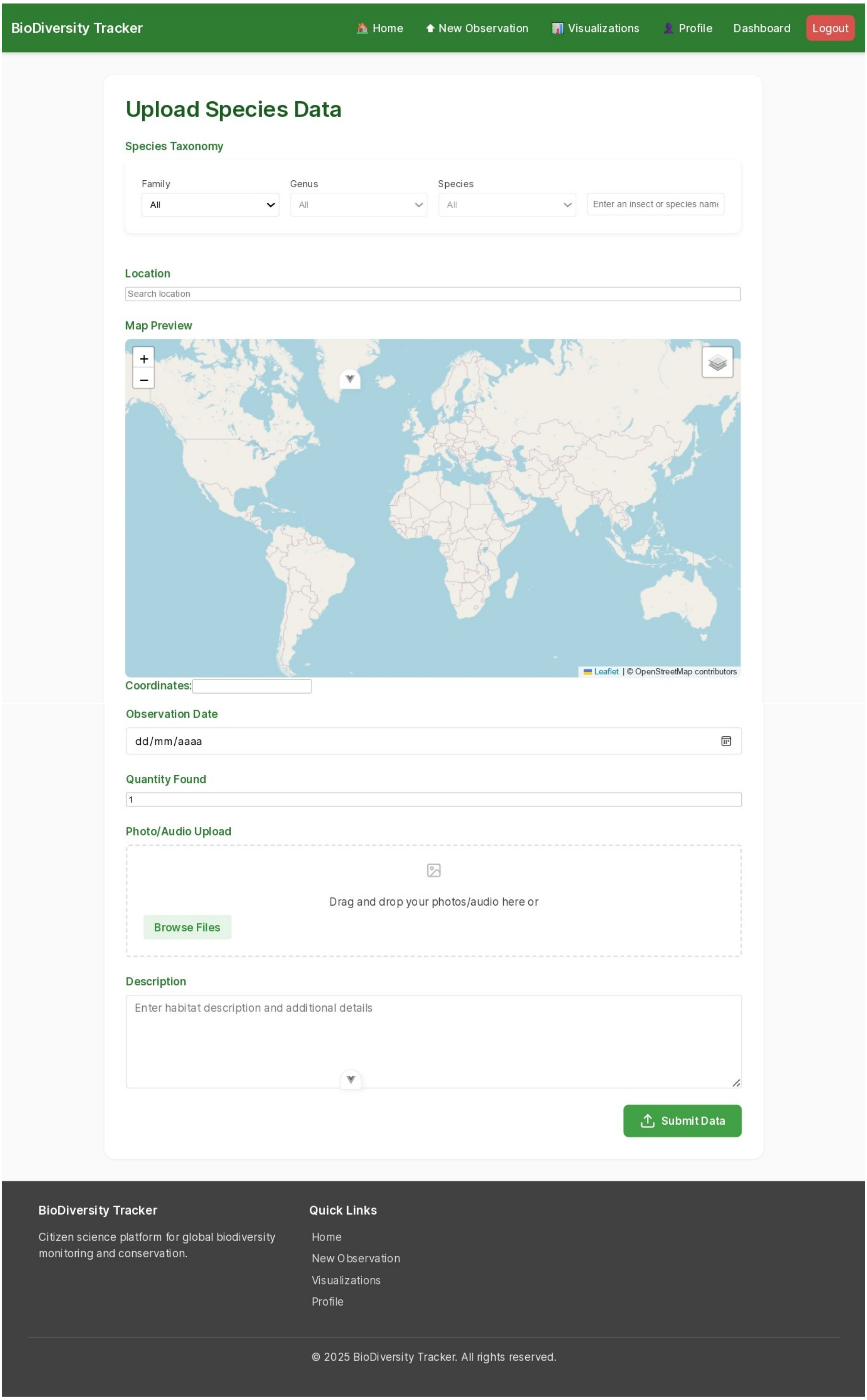
[18] eButterfly. (2022). "About eButterfly." Retrieved from https://www.e-butterfly.org/about/

[19] Observation International Foundation. (2025). "About our Platform." Retrieved from https://observation.org/pages/about/

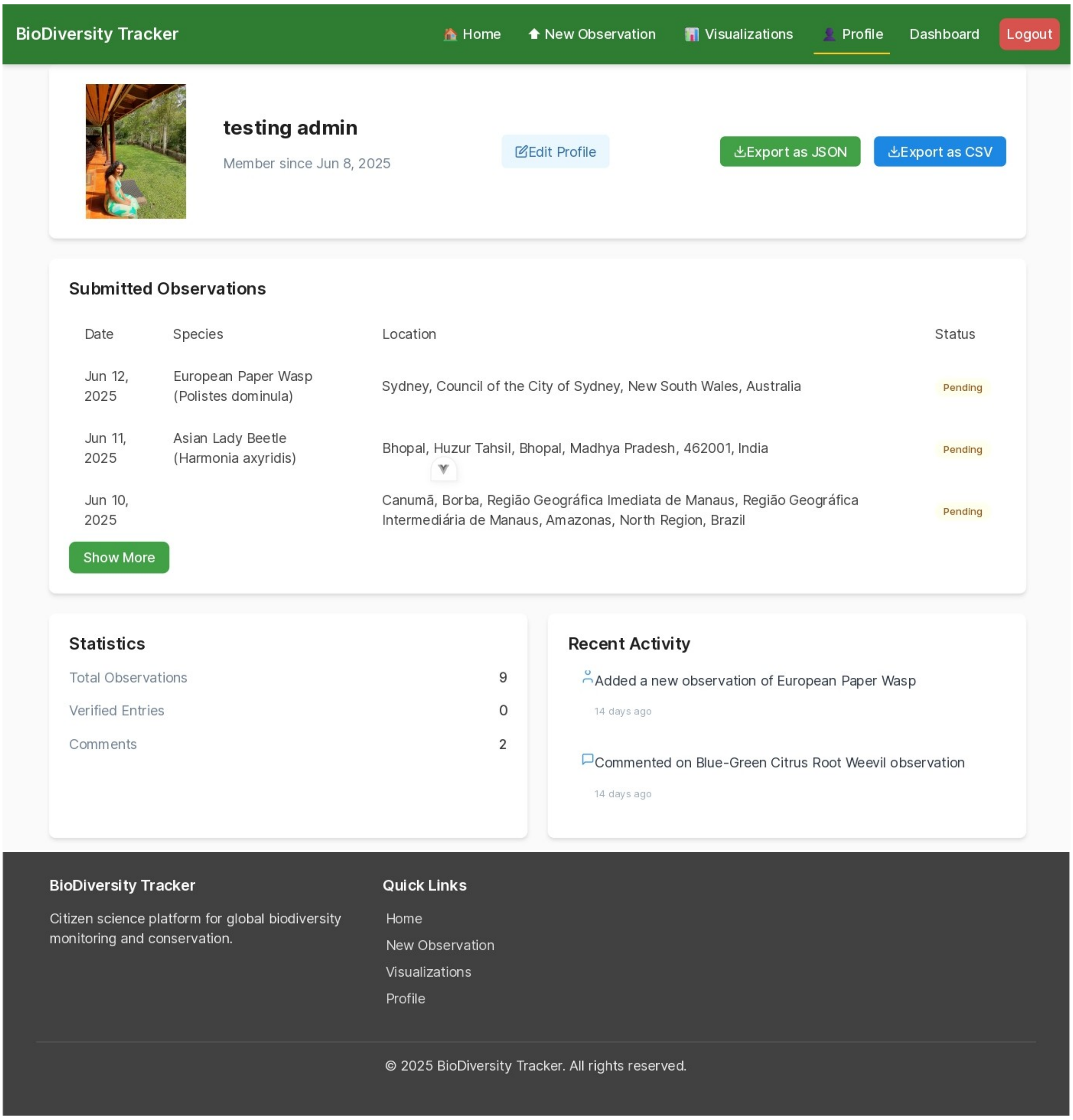
APPENDIX: IMPLEMENTATION SCREENSHOTS



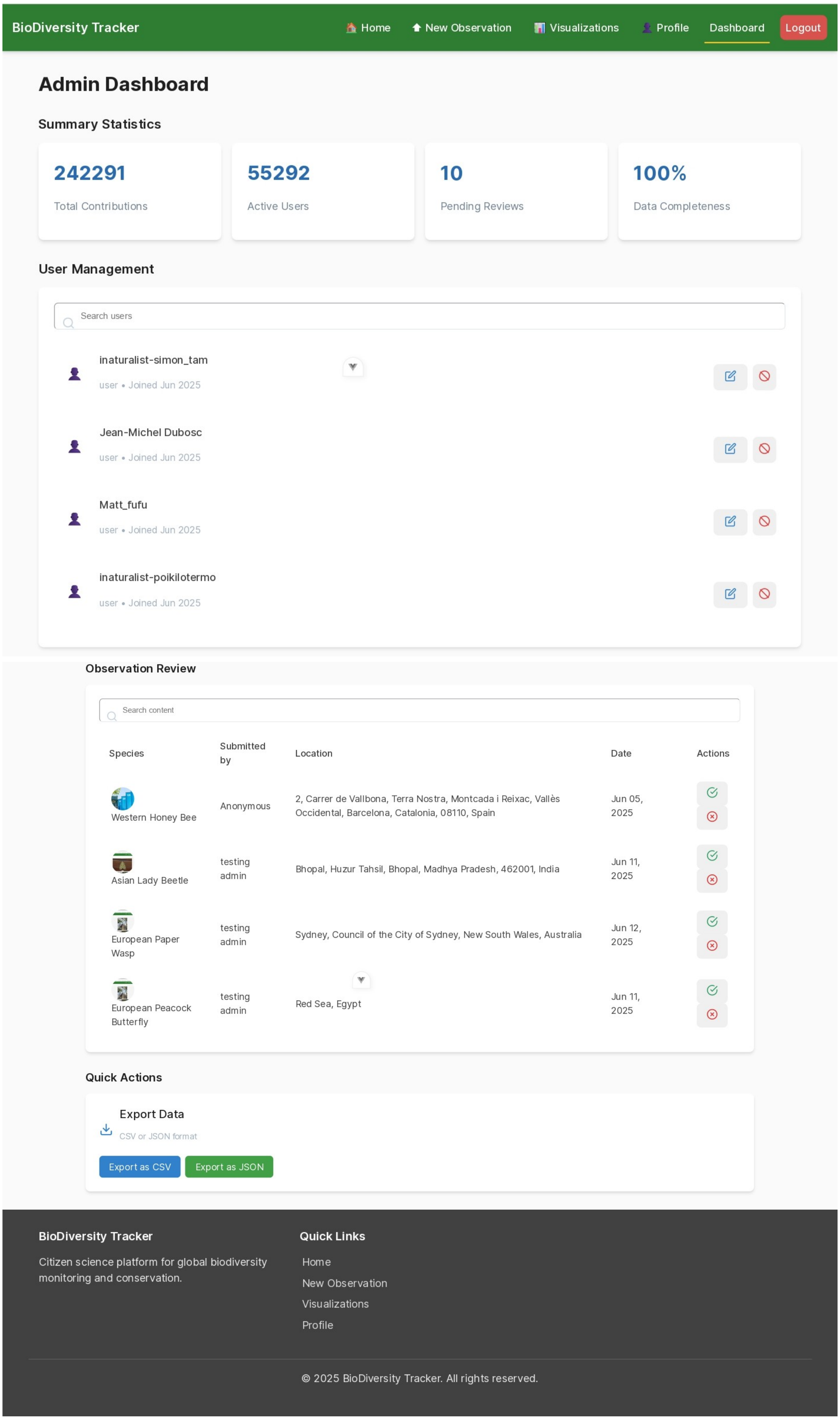
Reproduction of Figure 1: System Architecture Data Flow



Reproduction of Figure 6: New Observation Implementation



Reproduction of Figure 7: User Profile Implementation



Reproduction of Figure 9: Admin Dashboard Implementation