Unified Butterfly Recorder Server

Members: Branden Lange, Nicole Lockard, Tyler Uhlenkamp, and Zach Miller
Client: Nathan Brockman Advisor: Dr. Diane Rover

Introduction

Butterflies are a great indicator species. Their population levels serve as a first-warning of environmental changes. Researchers collect butterfly population data both to protect butterfly species and to track problems like global warming and habitat changes.

The previous UBR senior design groups created an Android app that facilitates butterfly data collection and automatically tracks location and weather data as they are surveying. The collected data is stored on the mobile devices in CSV files, which makes it difficult for the scientific community to share and analyze the data.

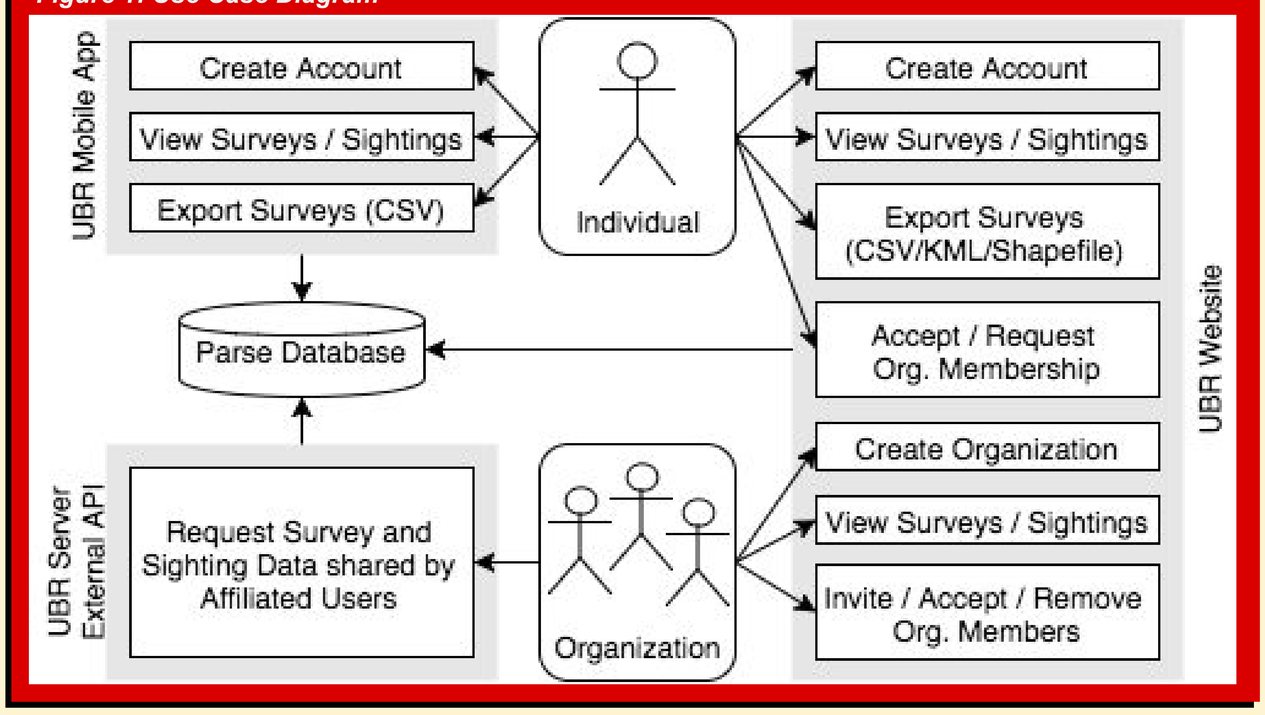
Requirements

- Secure and reliable database storage
  - Provides robust authentication system
  - Prevents unauthorized access to data
  - Protects data via storage replication
- Interface for applications to upload survey and sighting data
- Allow third-party organizations to register and collect data from users and associate data with users on external platforms
- Modern, responsive web application
  - Compatibility with all modern desktop and mobile web browsers
  - User-friendly interfaces for organization management, data analysis, and data managment
  - Professional and appealing aesthetic

Solution

- Create a database and server to which mobile applications can automatically upload survey data
- Create a public API through which third-party organizations can query the database to retrieve surveys which associated users have shared with them
- Create a website which allows users to:
  - View their surveys
  - Share them with organizations
  - Export survey data to files
  - Analyze their data using visualizations such as maps, graphs, and charts

Figure 1: Use Case Diagram



Modular Design

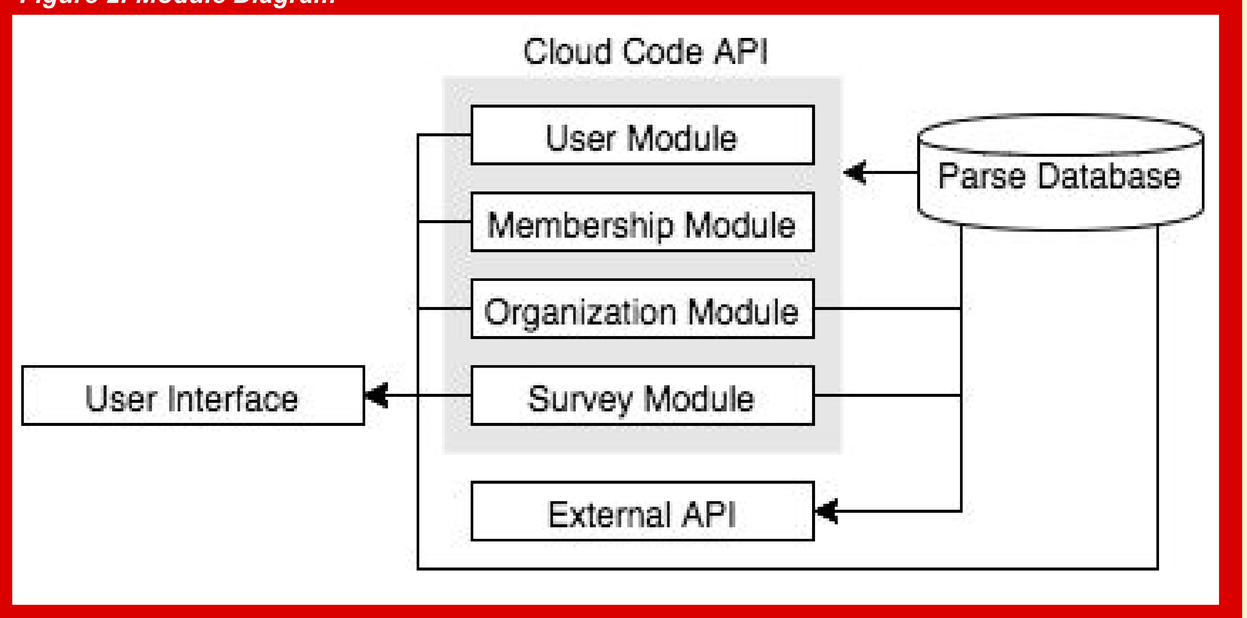
User Interface: A single-page web application composed of a hierarchy of React/JSX Components, which contain code and markup. It is comprised of modules which provide authentication, server transactions, internal routing, and cross-page messaging.

Cloud Code API: Provides private functions for authentication, authorization, and data validation in addition to operations related to organization management and the sharing of surveys.

External API: Provides public functions to retrieve survey data to authenticated requests.

Database: Comprised of tables for User, Membership, Organization, Survey, Sighting, and Breadcrumb data models.

Figure 2: Module Diagram



Development Process

- Agile / scrum development cycle with two-week sprints
- Weekly team standups
- Semi-formal code reviews facilitated via GitHub pull requests
- Weekly client meetings to assess progress, integrate feedback, and prioritize future work

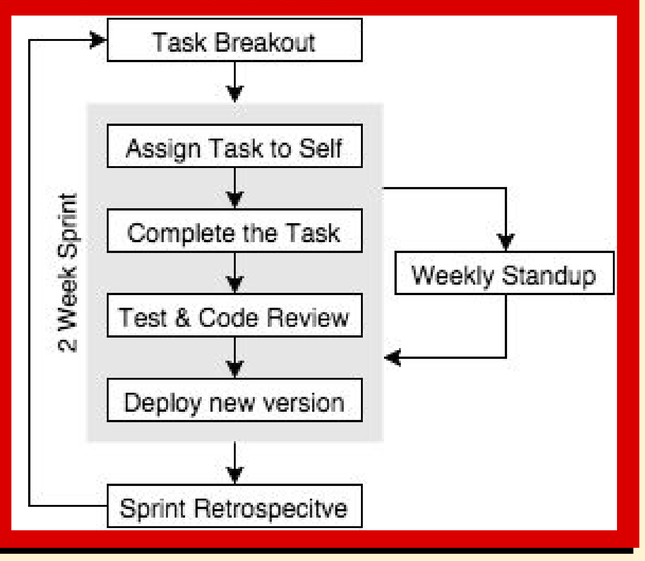


Figure 3: Process Diagram

Testing and Development Tools

- Git and GitHub for version control
- Trello for task management
- Slack for communication and meeting notes
- NodeJS for production execution
- Parse cloud code and Heroku for production hosting
- Travis-CI for continuous integration
- PhantomJS for a test environment
- Karma, Mocha, and Chai for a test framework
- Sinon and React TestUtils for extra test functionality
- ESLint, JSHint, and JSCS as linters for code quality

This project was developed by a team of computer and software engineering students (Dec15-06) in cooperation with Reiman Gardens Iowa State University and Iowa State University Department of Electrical and Computer Engineering