Butterfly, Group 7

COM S 402C Section 1

(Spring 2024)

Richard Bach

Joseph Jennings

Tze Yik Ong

On Wave Tiong

Kaden Wehrenberg

Client:

Nathan Brockman

Instructor:

Simanta Mitra, Ph.D.

TAs:

Eshita Zaman

Md Mahbubur Rahman

Md Rayhanul

(2) TAs to attach - a letter from the client expressing their thoughts/feelings on the team's work.

Requirements:

In our first meeting with our client at Reiman gardens we went inside of their butterfly enclosure and their butterfly lab. There the client explained to us what he wanted to do and his ideas to accomplish it. The Client uses an Alphacode system to designate a unique code to each of their butterflies. Before Reiman Gardens releases a butterfly into their enclosure, they will attach a small yellow sticker to the butterfly with a code written on it. For example “AAA”, “ADC”, “HOB”. We were then tasked with creating a website to fulfill 3 main purposes listed below.

1) Allow Reiman Gardens to enter butterflies into a database
2) Allow Volunteers and public users to look up a butterfly by its code and collect data about these look ups
3) Allow Reiman Gardens to moderate and generate a report about user data

In talking to the client we have expanded on these 3 core purposes to be able to design specifically what our Client has in mind.

1) Allow Reiman Gardens to enter butterflies into a database
    a) A database must be created to store all the data the client wants us to store
        i) Data 1
        ii) Data 2
        iii) Data 3

- b) Admins/Reiman Garden Staff are able to add butterflies to that database by pairing a code with a butterfly species already in the database
- c) Admins/Reiman Garden Staff are able to add new butterfly species to the database

2) Allow Volunteers and public users to look up a butterfly by its code and collect data about these look ups
   - a) Users will enter the 3 digit code they see on a butterfly
   - b) When entered the webpage will redirect to the name and scientific name of the butterfly
   - c) Internally, we will log the info of their entry for later viewing
3) Allow Reiman Gardens to moderate and generate a report about user data
   - a) On the webpage there will be a page dedicated to admin users being able to generate a report filtering user data
   - b) Admins will also be able to remove data from the database to filter out bad data.
   - c) Admins will have multiple filters to be able to specify data they want.

(4) Chapter on Design (includes API):

Frontend:

Frontend design was centered around accommodating the needs of our client and making it as user friendly as possible. Our website is the user interface for the project and is going to be primarily used by visitors to Reiman gardens and the workers therein. Following the path a public user would follow, our webpage is accessible via a QR code. This allows public users to easily access the website on their phone. Our website is purposely designed to be usable from mobile devices to facilitate this traffic. All of the HTML elements on our website are setup to display the website on both mobile and desktop devices. Once the public accesses the website they are tasked with a login screen where they will enter a username of their choosing and be redirected to our home page. The public’s home page contains an image tutorial for what to look at on a butterfly and a medium sized icon that directs the public to their next screen, reporting a seen butterfly. The report butterfly page contains a large and easy to access textbox that can only contain a maximum of 3 characters as well as a submit button to submit the sighting. When the user clicks submit the page is redirected to our display facts page where information about the reported butterfly is displayed to the user. From here the user is directed back to the home page for the public, completing a loop for the public to follow until they are done with the website.

Similar to public users, Admin users login, have a home page, report butterfly page, and display fact page containing all the same features as the public versions of these pages do. The Admin Home page contains several more options to facilitate all the functionality our client requested. All buttons have rounded corners due to client request. All pages have a logout button in the top right, and the Reiman gardens logo in the top left. Clicking the logo redirects to the relative home page for the user.

First off, we have the register butterfly page. This page is designed to be as user friendly as possible to facilitate entry of butterflies to our database. Because the workers at Reiman Gardens can physically hold up to four butterflies, we have a radio select to choose how many butterflies the worker wants to enter at once. Next we have an autofill functionality for the alphacode entry. If a worker fills out the alphacodes manually, the next time they enter more butterflies the alphacodes will be filled out for them. Furthermore, The workers select a butterfly species from a dropdown that they can type into to search for the species they are looking for. This database contains both the scientific names and common names for butterfly species. Finally, we have a manual date entry that allows the worker to select the date of entry for the butterfly, if left empty, it will default to the current date and time.

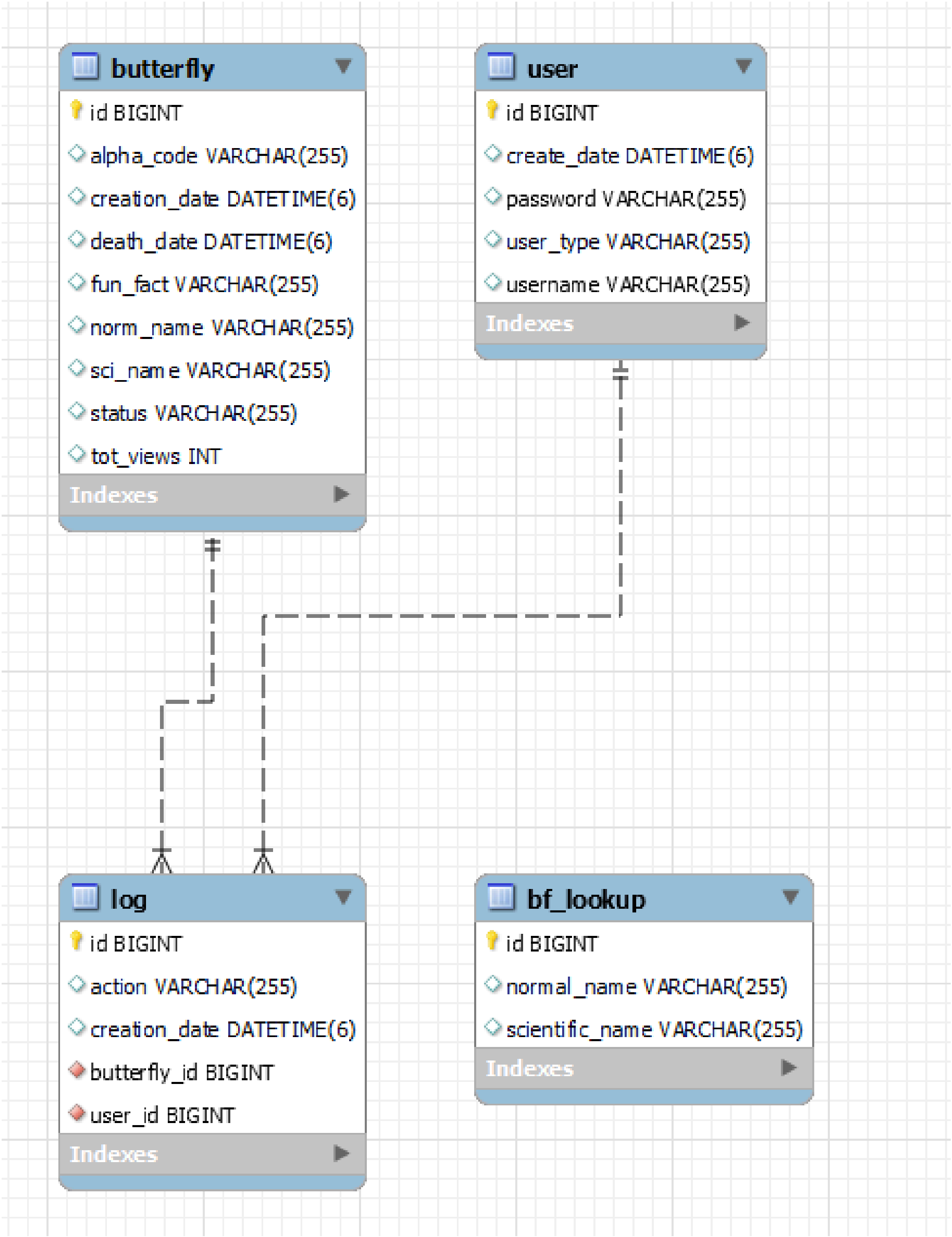
Second, we have the add species page. This page is very simple and allows the workers to enter a scientific name and a common name and click a submit button. This enters the pairing into the backend database and allows the workers to find it in any dropdown that searches the name database.

Third, we have the access database page. This page contains the logs for every butterfly report entry in the database. The entries alternate in background color between a dark gray and the default white. This was done upon client request to allow the client to easily differentiate the lines. We have sort options that allow the admin user to filter and sort based on alphacode, who it was reported by, the date the butterfly was registered, and the species of the butterfly. We can also order things by oldest or newest. In the bottom right corner there is a button that will download the entire butterfly logs database to a CSV file. Each entry has a button that directs to our edit entry page. The edit entry page allows the admin user to manually edit any individual entry and update its data.

Finally we have a generate report page, this allows the admin users to generate a report showing data on various topics and export it to a CSV file. These options include an individual butterfly, a species of butterfly, all butterflies registered on a certain date, all butterflies tagged on a certain date, the status of a butterfly, all butterflies reported by a certain user, which group of user (Admin, Docent, Public) reported it. Each of these options contains multiple subcategories to alter and edit the report generation.

To conclude front end design. Everything is designed to allow our client to interface with the databases we have created for them and make that interface as user friendly as possible.

Backend:



butterfly
- id BIGINT
- alpha_code VARCHAR(255)
- creation_date DATETIME(6)
- death_date DATETIME(6)
- fun_fact VARCHAR(255)
- norm_name VARCHAR(255)
- sci_name VARCHAR(255)
- status VARCHAR(255)
- tot_views INT

user
- id BIGINT
- create_date DATETIME(6)
- password VARCHAR(255)
- user_type VARCHAR(255)
- username VARCHAR(255)

log
- id BIGINT
- action VARCHAR(255)
- creation_date DATETIME(6)
- butterfly_id BIGINT
- user_id BIGINT

bf_lookup
- id BIGINT
- normal_name VARCHAR(255)
- scientific_name VARCHAR(255)

Above is the ERD, or entity relationship diagram, for our backend tables. We have 4 tables. One of which, bf_lookup, is separated from the others. This table acts as a database for our scientific names and normal names. It is only used for looking up this information. Our other three tables are butterfly, user, and log. Logs are automatically created when a user creates a new butterfly or enters a public sighting. These logs tie the user who entered the information within the butterfly information it is associated with. On the user side, we collect information about the day the user was made, their password, username, and user type: admin, docent, or public. Finally, we have the butterfly database. This houses our information about the butterflies. The alpha code, creation date, death date, a fun fact, the normal name, scientific name, status (alive/dead), and the number of times it has been viewed.

For our API documentation, we conveniently have a publicly available swagger UI page to view how it works. See here for more details, link. At a high level, we have 5 endpoints for user-related work. We have 9 endpoints for log creation-related work. We have 8 endpoints for butterfly-related work. Finally, we have 3 endpoints for butterfly lookup-related work. The link above will show the endpoints and what they do.

Work done:

Kaden Wehrenberg worked primarily on the frontend of the project. Kaden completed the autofill for selecting a butterfly species, in particular making the frontend create a call to the backend and paring the backend’s response. Kaden helped set up the general functionality for submitting a butterfly as well as connecting the pages to one another. Kaden worked on multiple pages that needed to connect to the backend as well as making sure they could properly take input from users and save that input between pages. Kaden also worked on writing the final project report

Joseph Jennings took the lead role of maintaining and developing the backend for our project. Joseph wrote the database repositories and endpoints for the frontend needs. He also helped with the frontend in navigating HTTP requests. In addition, Joseph wrote functionality to anonymize the user logs for proper data security and added automatic logging for actions. He also deployed the frontend and backend application on the AWS servers while making it publicly available for the client. Finally, he created the AWS servers for a production and development enivronment.

Richard Bach styled a majority of our webpage and helped set up routing between. In addition to styling Richard implemented the mobile formatting for our page. All of the CSS and HTML styling for all of our pages was done by Richard. Richard implemented the frontend functionality for adding new species of butterflies to the database. Richard also assisted in implementing functionality for reporting butterflies creating the input fields and styling them. Finally, Richard handled a large amount of contact with the client and did a fantastic job getting the info the group needed for work on our project.

Tze Yik Ong worked on both the frontend and backend of the project, working on frontend for demos 1 and 2, and backend for demo 3. Tze Yik helped to implement the login functionality for frontend users and the registration of new butterflies. On the backend Tze Yik worked to help prevent duplicate entries of alpha codes and usernames. When a user would enter a name already in the system it would spit out a new username instead. In addition, Alpha Codes could not be registered if an existing one of the same code was already in the system.

On Wave Tiong helped implement the login functionality for the frontend users. In addition, Wave assisted Richard in styling frontend screens. Wave worked on large portions of our senior design website and gathered the information needed for the website. Wave also implemented the validation for the admin password, so admin username and password needed to be correct.

Results achieved:

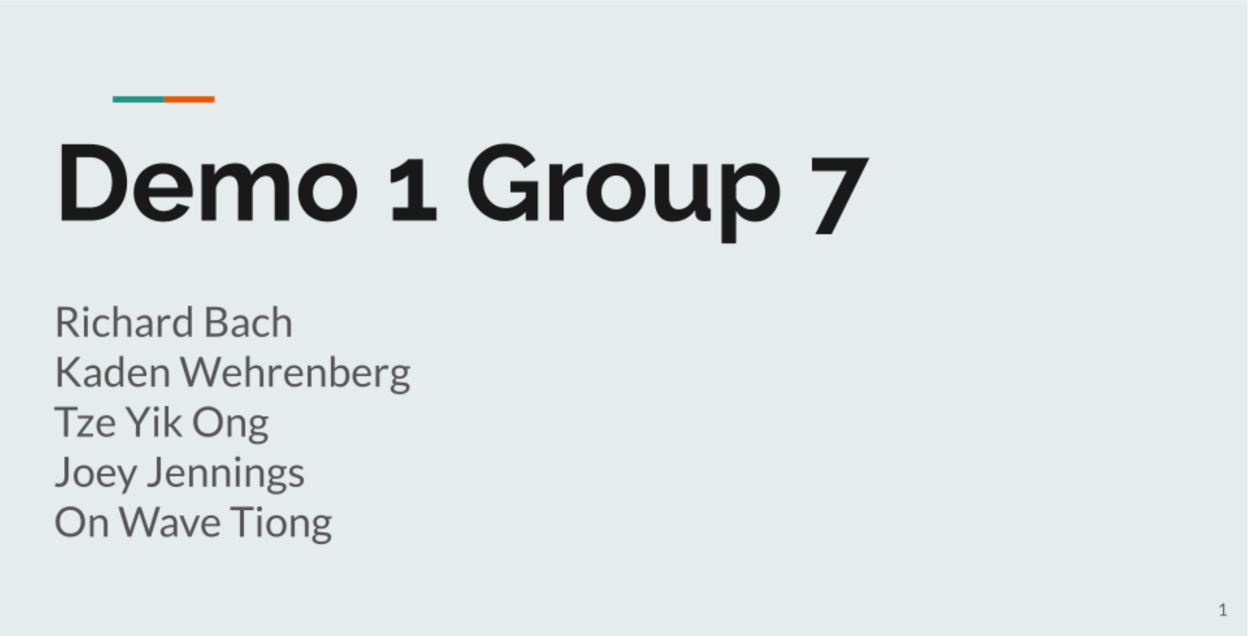
We managed to create a working website fulfilling our requirements outlined previously. Our website handles multiple public users who can access our website, type in a username, type in the alphacode of a butterfly they see and be redirected to a page with the scientific name and common name of the butterfly. This sends a request to our backend which collects data about the submission. This data is then stored for later use.

We have a separate set of pages with similar functionality to the public ones for the volunteers at Reiman Gardens called docents. They will similarly log in and type in the alphacode of a butterfly to find out its name.

Admin users are required to log in to the admin account we created to access the admin functionalities. Admins can do everything public and docents can do, but can also add butterflies to the database, add new butterfly species to the database, generate a report on data in the database, export that report to a .csv file.

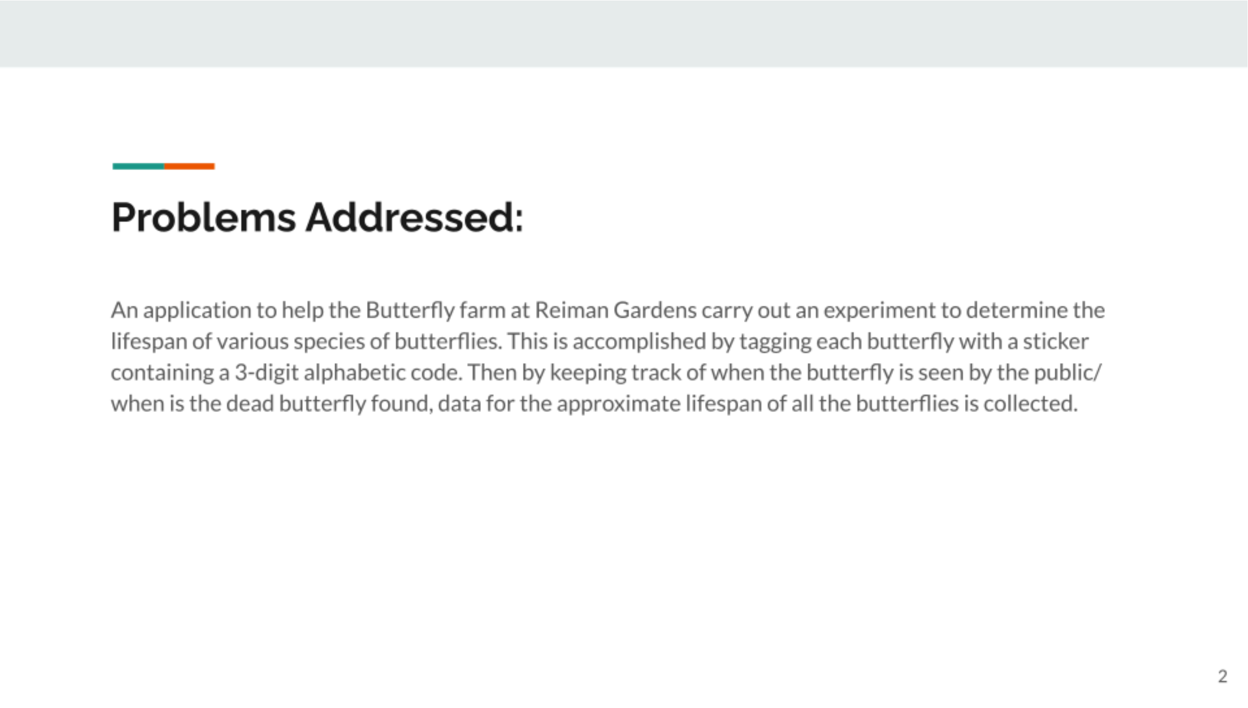
All webpages made for the public and docent users have formatting that works on both desktop and mobile devices, we have multiple safety checks in to both prevent malicious users and to account for when users commit errors or enter data incorrectly. We handle all of, but not limited to duplicate name entries, duplicate alphacode entries, duplicate butterfly species, etc.

Appendix:



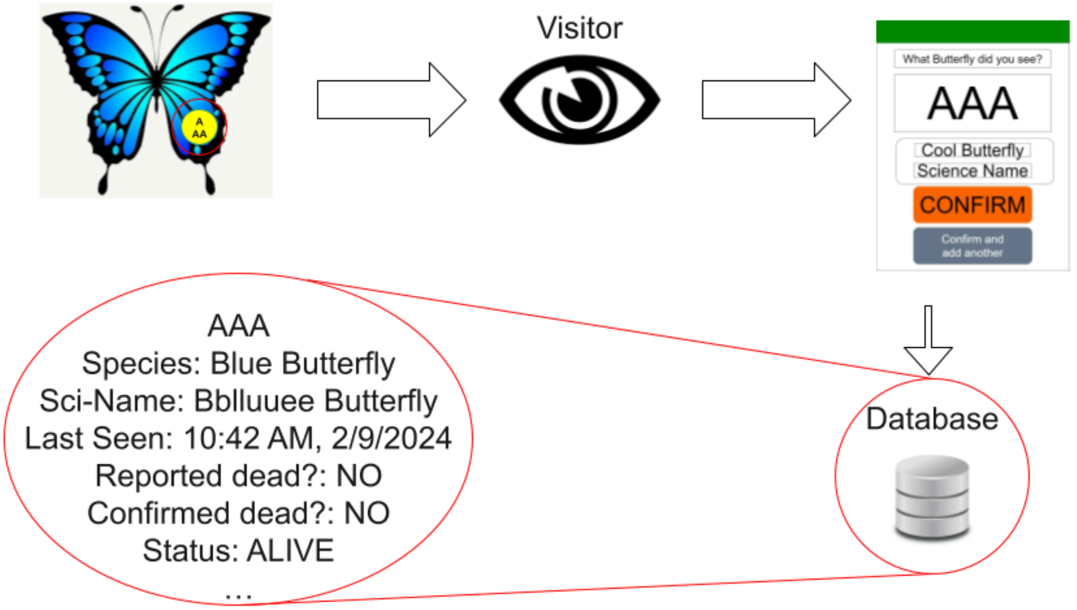
Demo 1 Group 7

Richard Bach
Kaden Wehrenberg
Tze Yik Ong
Joey Jennings
On Wave Tiong



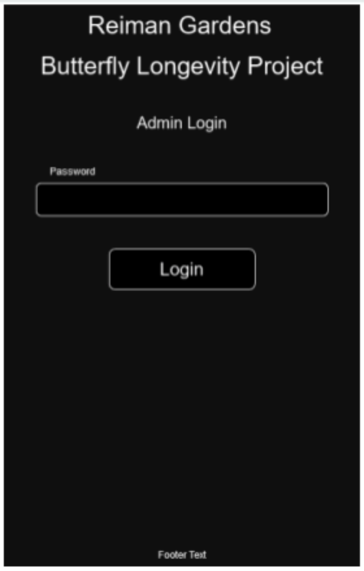
Problems Addressed:

An application to help the Butterfly farm at Reiman Gardens carry out an experiment to determine the lifespan of various species of butterflies. This is accomplished by tagging each butterfly with a sticker containing a 3-digit alphabetic code. Then by keeping track of when the butterfly is seen by the public/ when is the dead butterfly found, data for the approximate lifespan of all the butterflies is collected.



AAA
Species: Blue Butterfly
Sci-Name: Bblluuee Butterfly
Last Seen: 10:42 AM, 2/9/2024
Reported dead?: NO
Confirmed dead?: NO
Status: ALIVE
...

Admin Screens

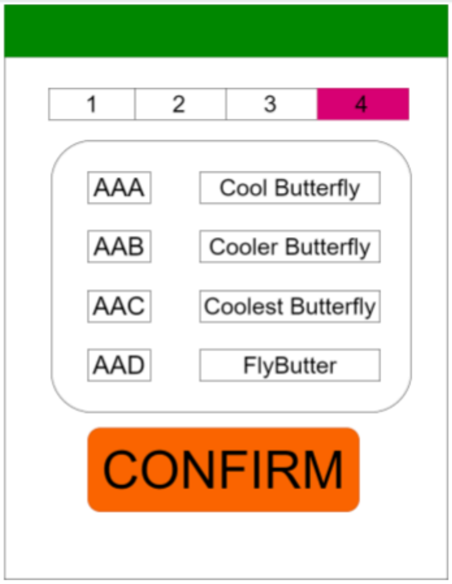


Reiman Gardens
Butterfly Longevity Project

Admin Login

Password

Login for admin

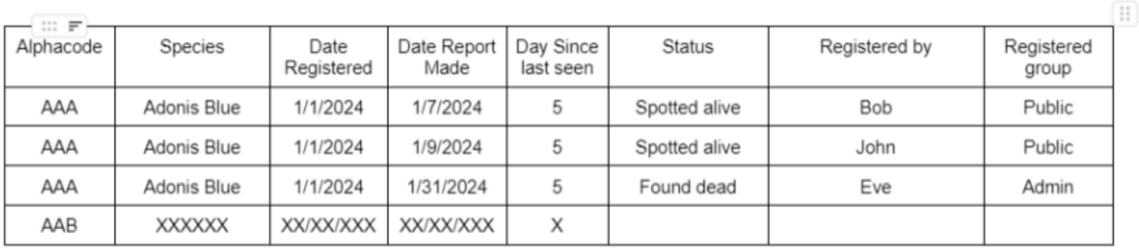


1	2	3	4

AAA	Cool Butterfly
AAB	Cooler Butterfly
AAC	Coolest Butterfly
AAD	FlyButter

Registering new butterflies

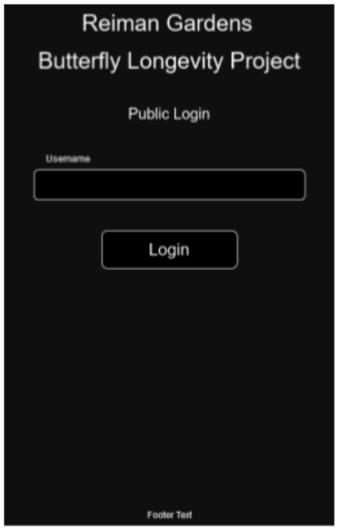
Data Screens



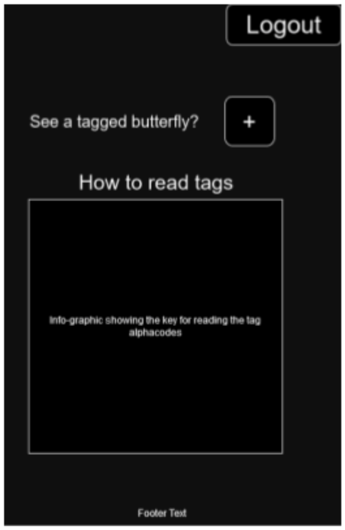
Alphacode	Species	Date Registered	Date Report Made	Day Since last seen	Status	Registered by	Registered group
AAA	Adonis Blue	1/1/2024	1/7/2024	5	Spotted alive	Bob	Public
AAA	Adonis Blue	1/1/2024	1/9/2024	5	Spotted alive	John	Public
AAA	Adonis Blue	1/1/2024	1/31/2024	5	Found dead	Eve	Admin
AAB	XXXXXX	XX/XX/XXX	XX/XX/XXX	X			

Generated Report Example

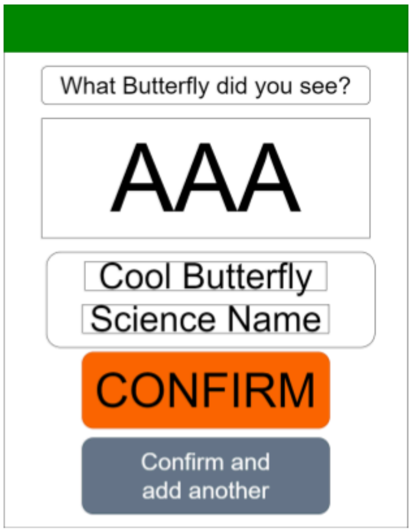
Public screens



Public login

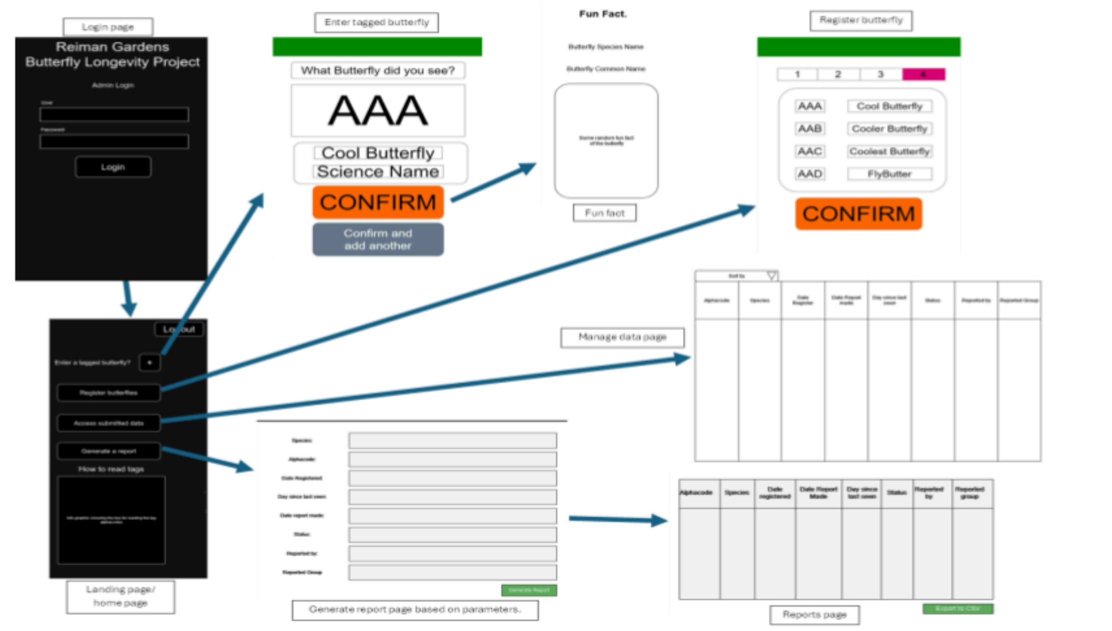


Landing page/Home page



Inputting the alphacode of the butterfly

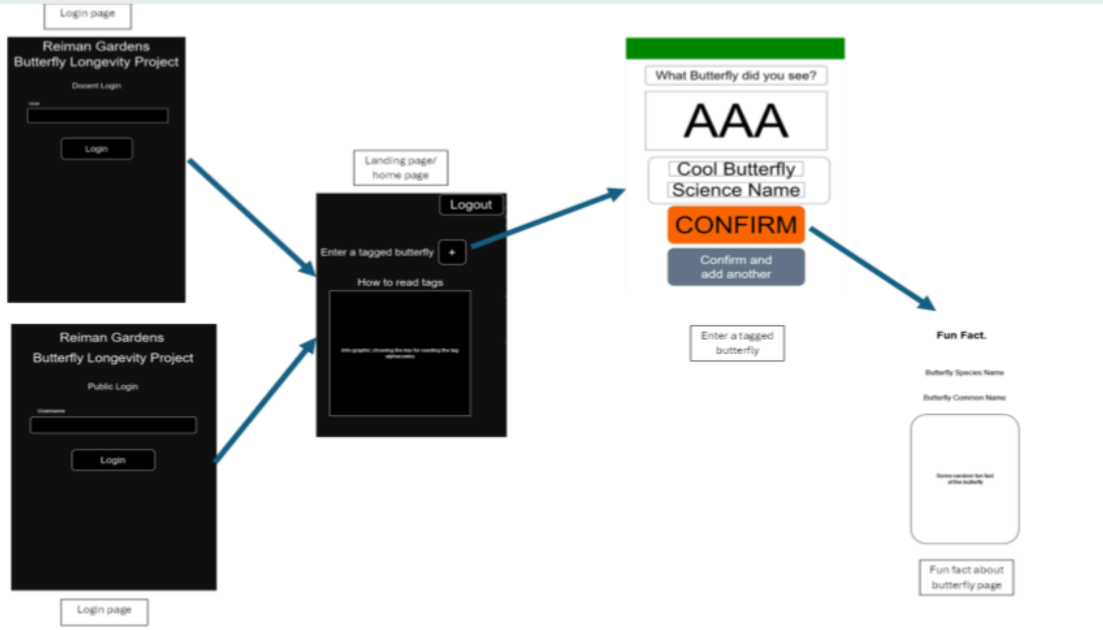
Admin Screen Flow

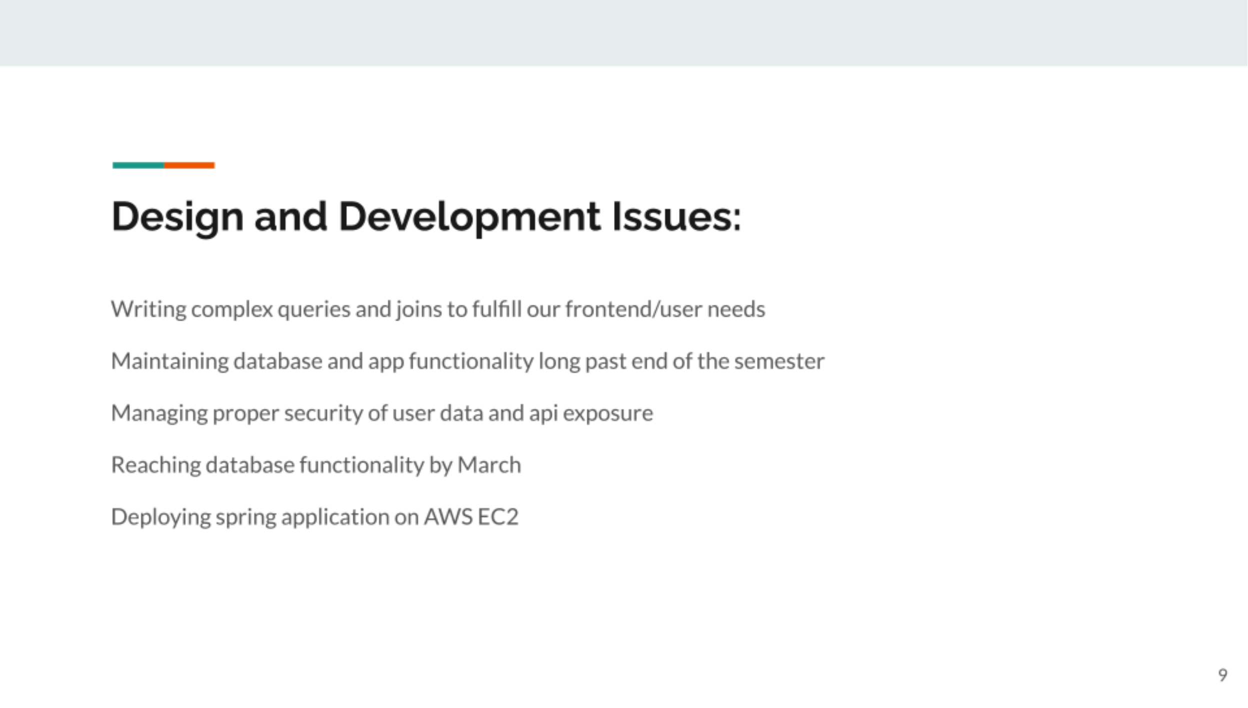


Alphacode	Species	Date Register	Date Report made	Day since last seen	Status	Reported by	Reported Group

Alphacode	Species	Date registered	Date Report Made	Day since last seen	Status	Reported by	Reported group

Public Screen Flow





Design and Development Issues:

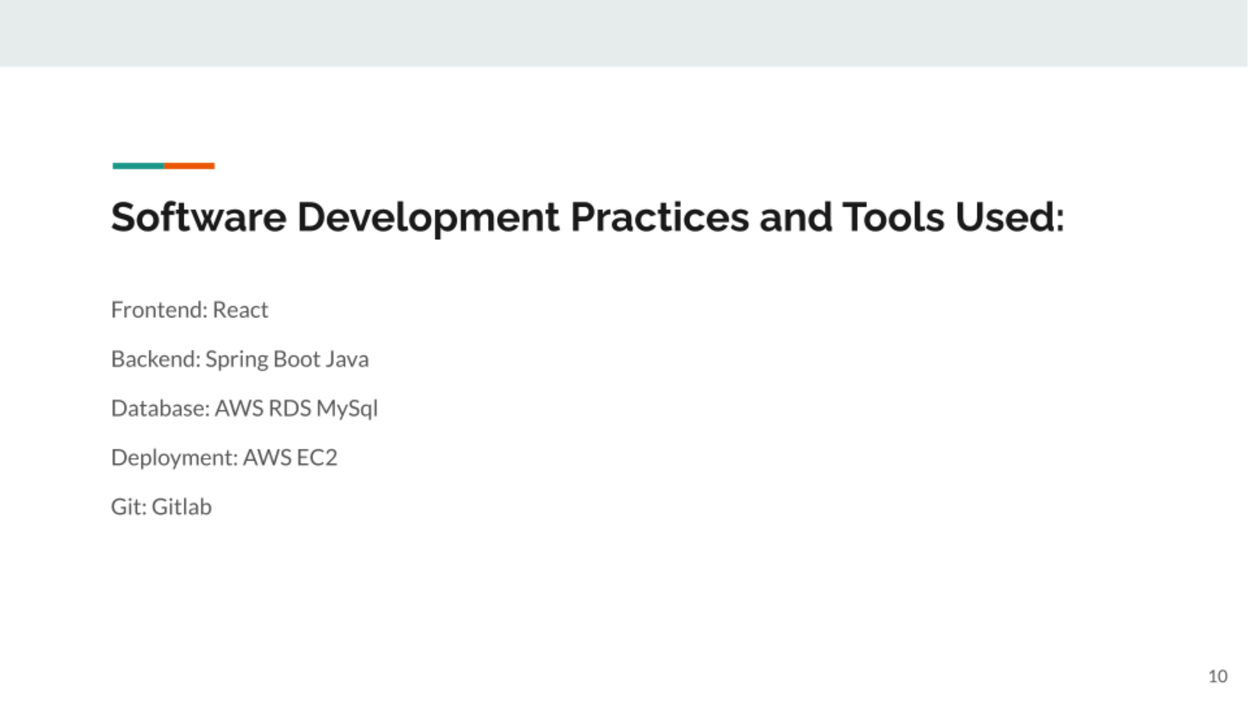
Writing complex queries and joins to fulfill our frontend/user needs

Maintaining database and app functionality long past end of the semester

Managing proper security of user data and api exposure

Reaching database functionality by March

Deploying spring application on AWS EC2



Software Development Practices and Tools Used:

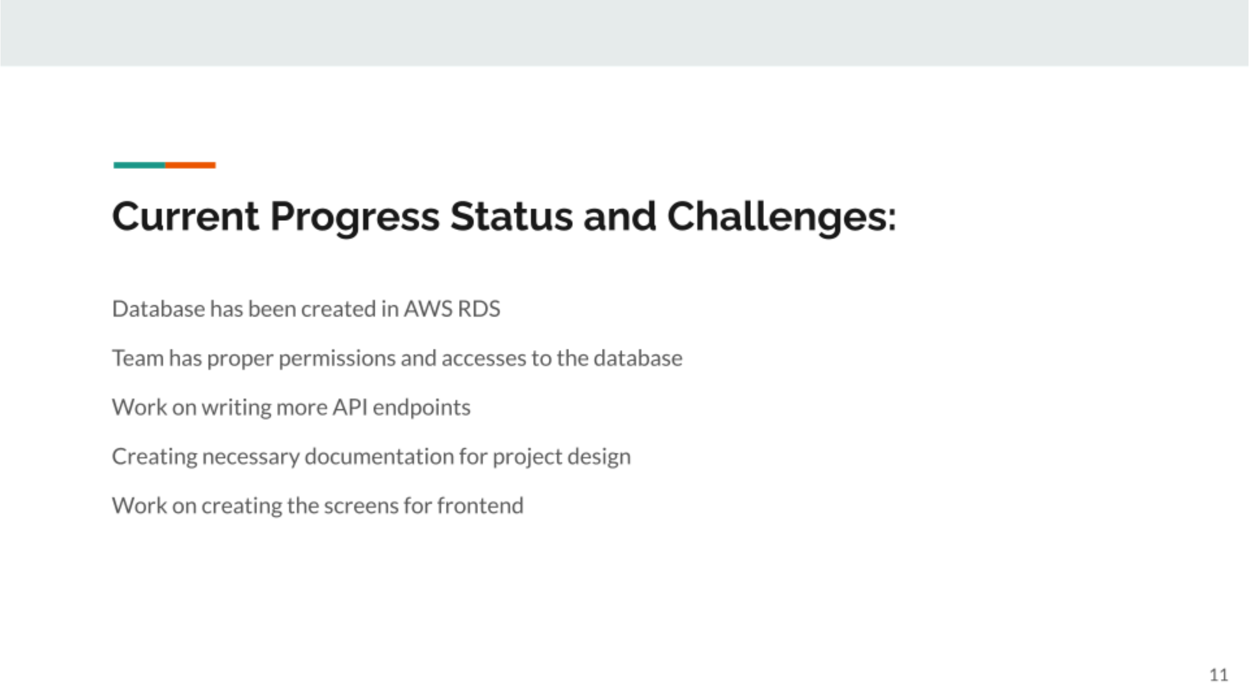
Frontend: React

Backend: Spring Boot Java

Database: AWS RDS MySql

Deployment: AWS EC2

Git: Gitlab



Current Progress Status and Challenges:

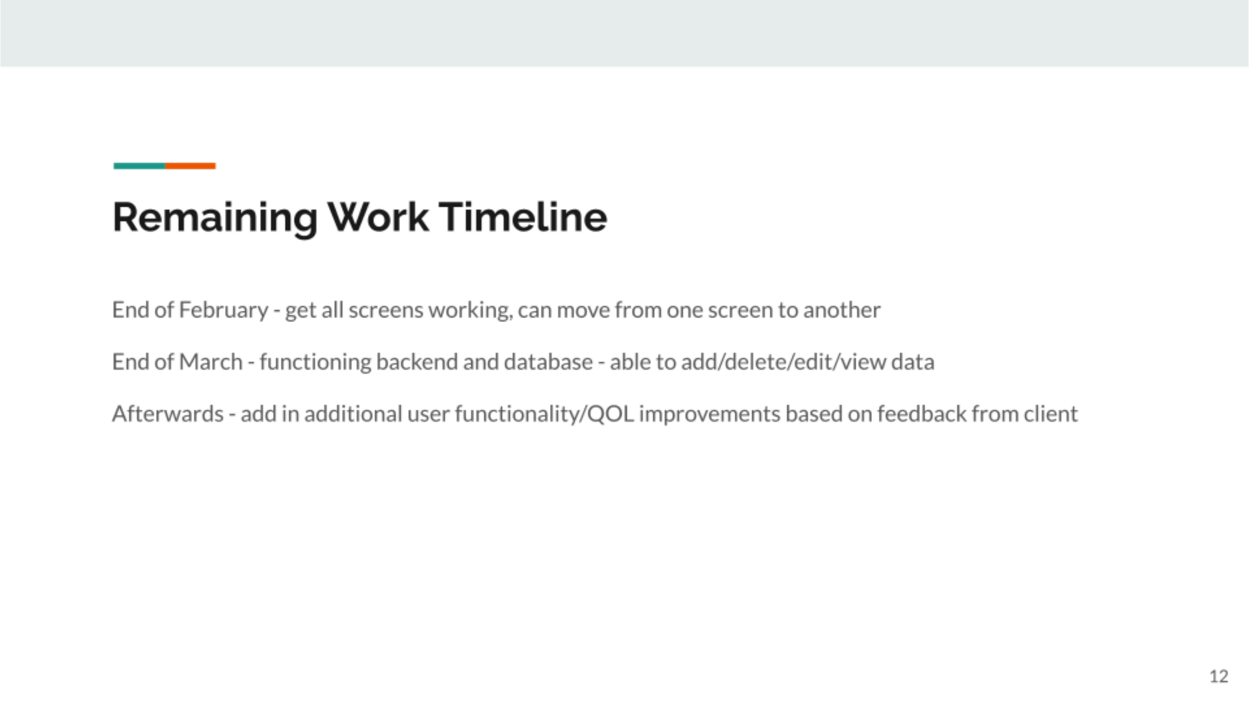
Database has been created in AWS RDS

Team has proper permissions and accesses to the database

Work on writing more API endpoints

Creating necessary documentation for project design

Work on creating the screens for frontend



Remaining Work Timeline

End of February - get all screens working, can move from one screen to another

End of March - functioning backend and database - able to add/delete/edit/view data

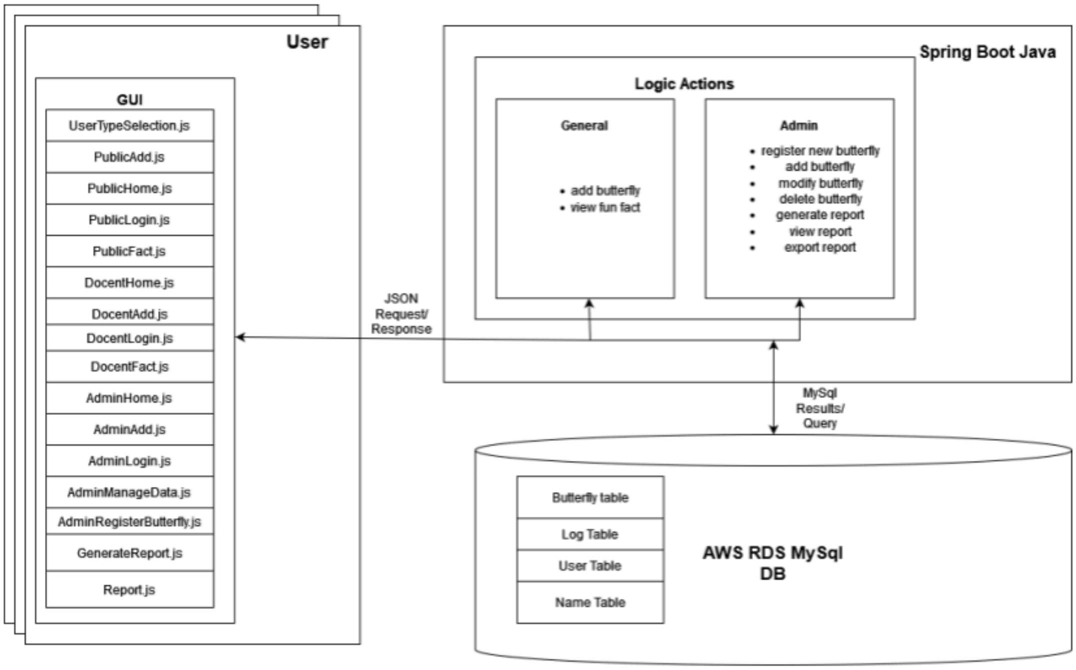
Afterwards - add in additional user functionality/QOL improvements based on feedback from client

Demo 2 Group 7

Richard Bach
Kaden Wehrenberg
Tze Yik Ong
Joey Jennings
On Wave Tiong

Progress since Demo 1

- Created dummy screens for all pages present in our screen sketch
- Provided navigation between pages
- Styled pages for a consistent look throughout the web application
- Implemented Butterfly Registration
- Implemented autocomplete for butterfly names on the Butterfly Registration page
- Design Document



User

GUI

UserTypeSelection.js

PublicAdd.js

PublicHome.js

PublicLogin.js

PublicFact.js

DocentHome.js

DocentAdd.js

DocentLogin.js

DocentFact.js

AdminHome.js

AdminAdd.js

AdminLogin.js

AdminManageData.js

AdminRegisterButterfly.js

GenerateReport.js

Report.js

JSON Request/ Response

Spring Boot Java

Logic Actions

General

- add butterfly
- view fun fact

Admin

- register new butterfly
- add butterfly
- modify butterfly
- delete butterfly
- generate report
- view report
- export report

MySql Results/ Query

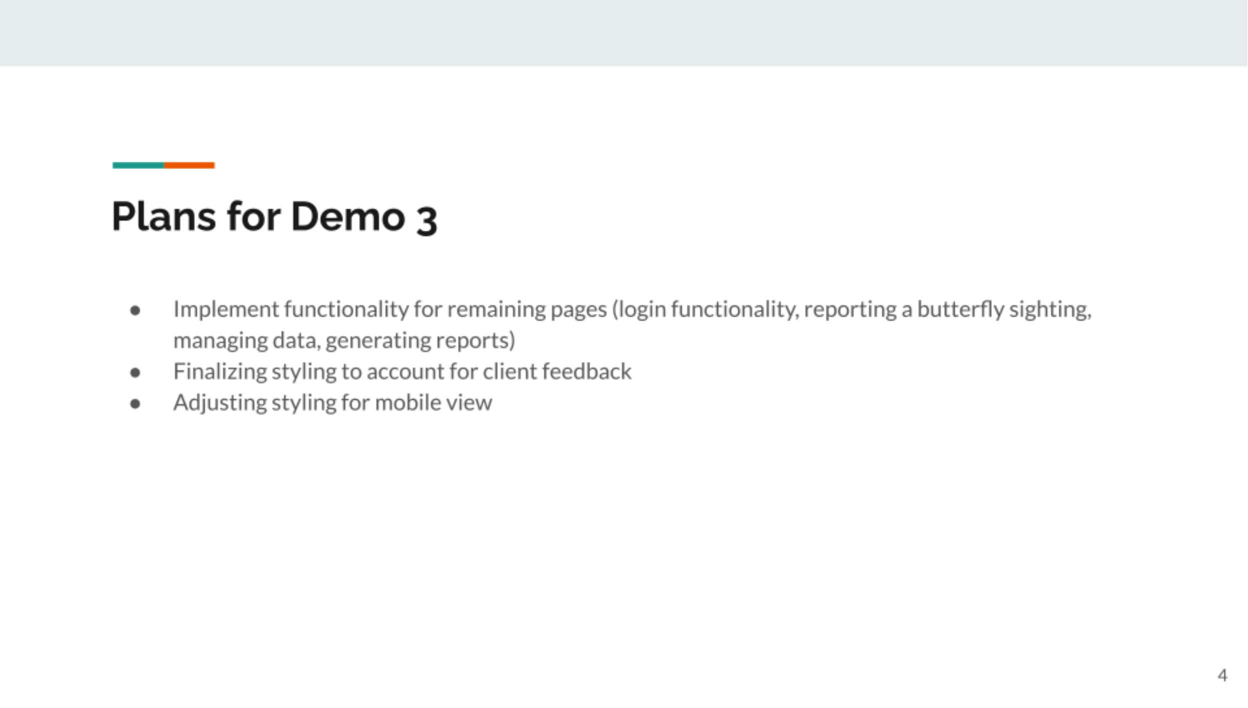
Butterfly table

Log Table

User Table

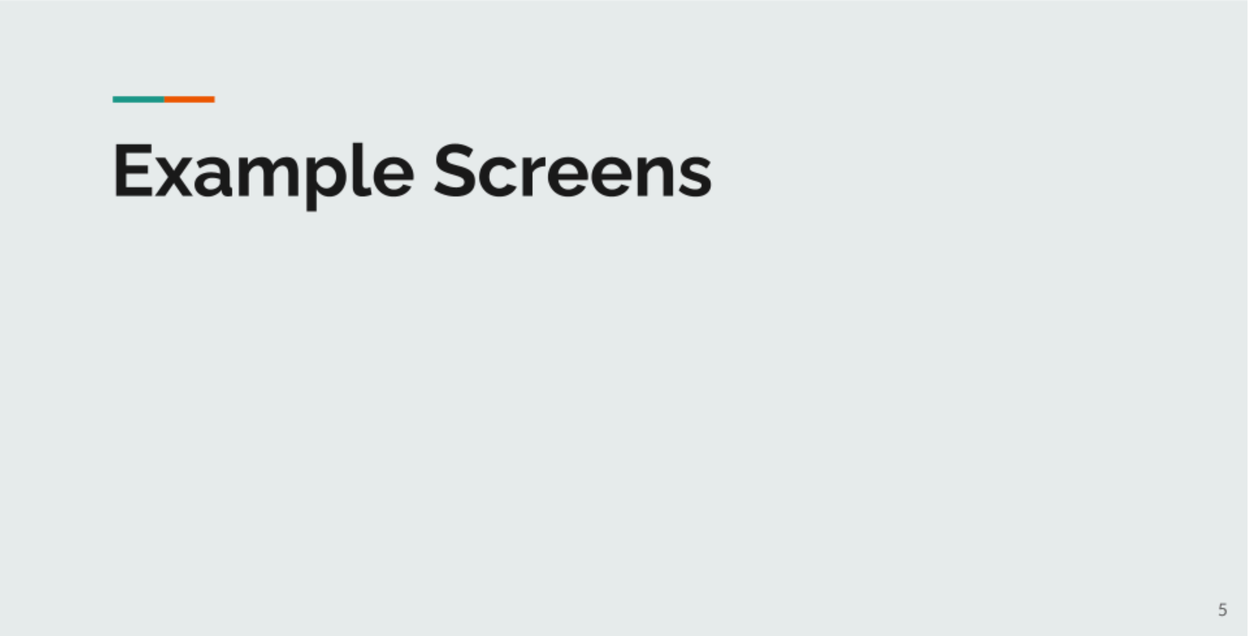
Name Table

AWS RDS MySql DB

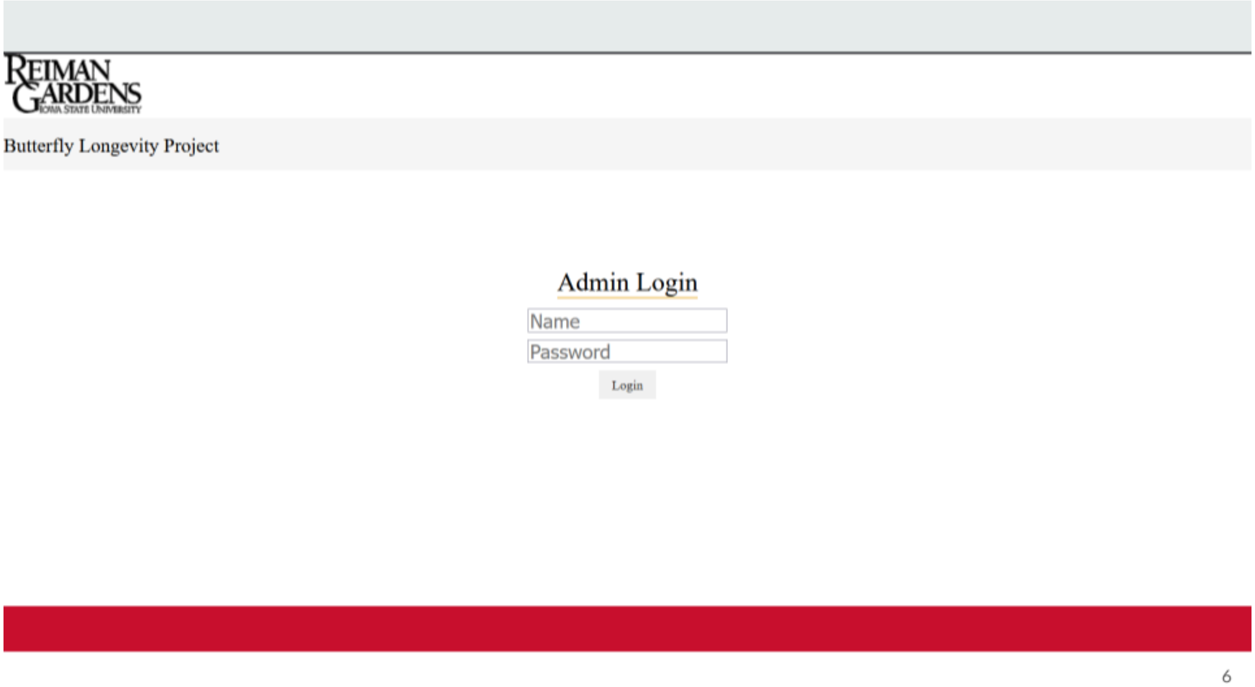


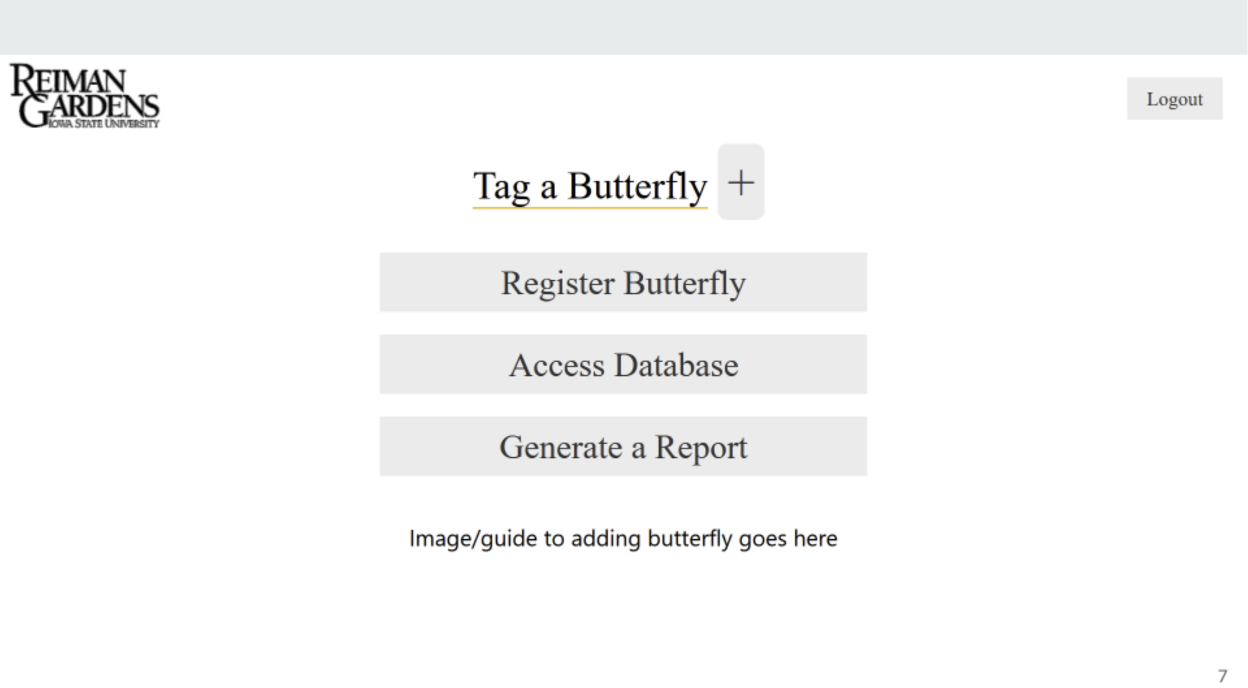
Plans for Demo 3

- Implement functionality for remaining pages (login functionality, reporting a butterfly sighting, managing data, generating reports)
- Finalizing styling to account for client feedback
- Adjusting styling for mobile view

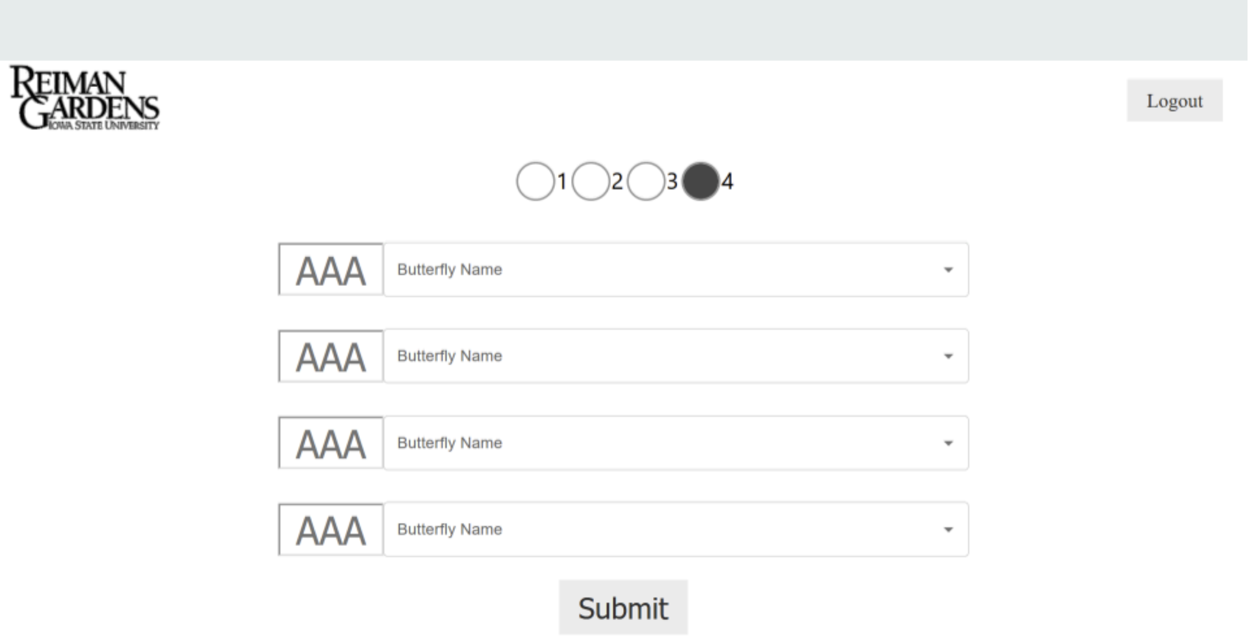


Example Screens

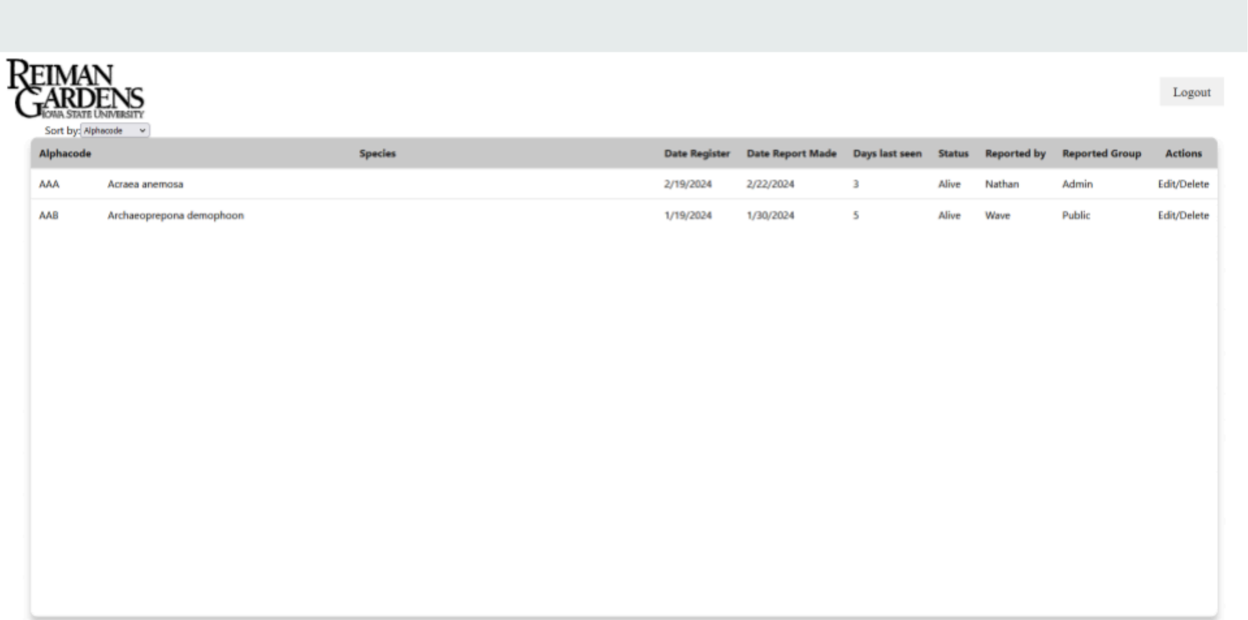




Image/guide to adding butterfly goes here



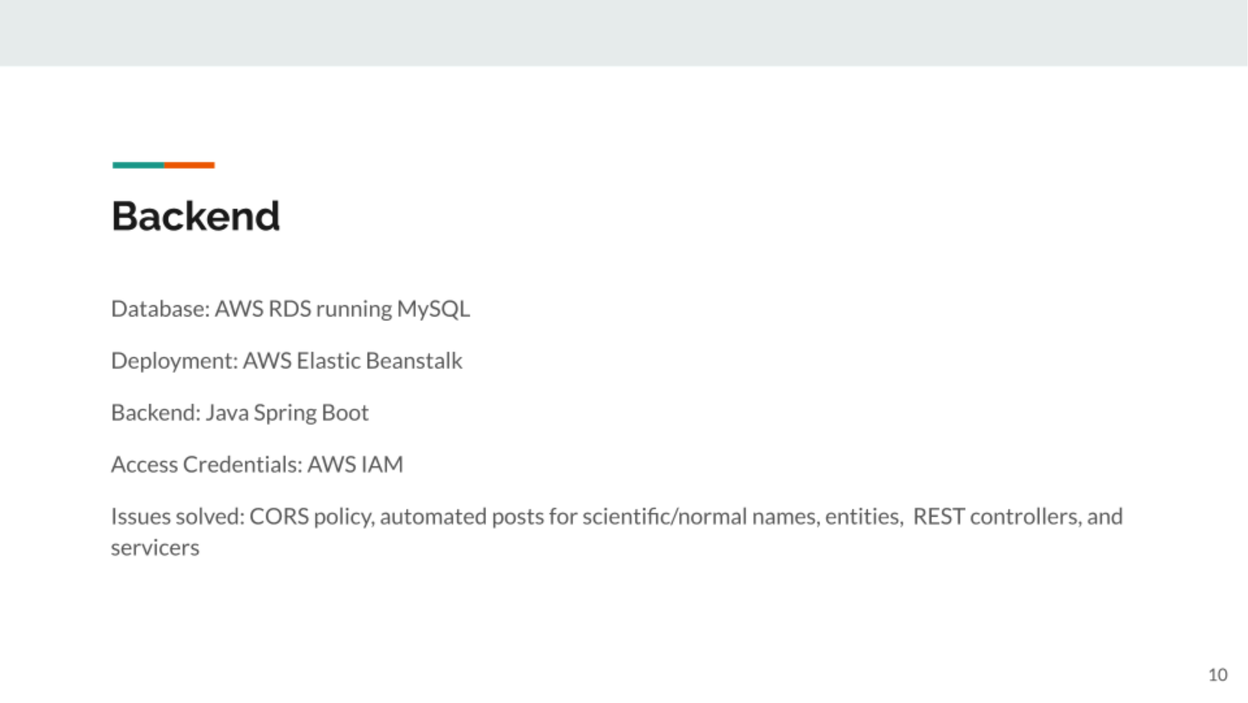
8



Sort by: Alphacode

Alphacode	Species	Date Register	Date Report Made	Days last seen	Status	Reported by	Reported Group	Actions
AAA	Acraea anemosa	2/19/2024	2/22/2024	3	Alive	Nathan	Admin	Edit/Delete
AAB	Archaeoprepona demophoon	1/19/2024	1/30/2024	5	Alive	Wave	Public	Edit/Delete

9



Backend

Database: AWS RDS running MySQL

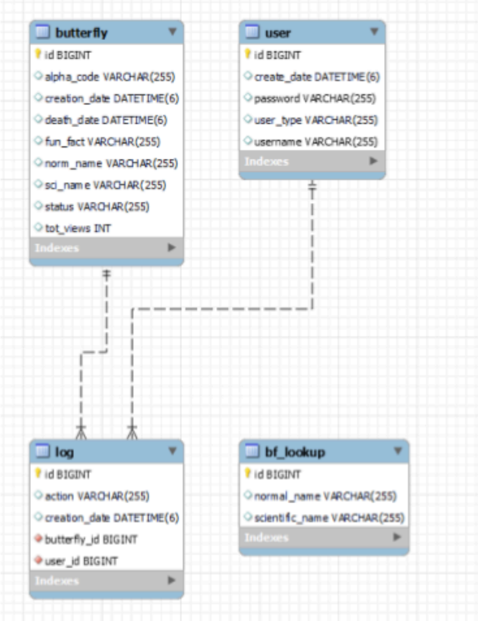
Deployment: AWS Elastic Beanstalk

Backend: Java Spring Boot

Access Credentials: AWS IAM

Issues solved: CORS policy, automated posts for scientific/normal names, entities, REST controllers, and servicers

ERD



butterfly
- id BIGINT
- alpha_code VARCHAR(255)
- creation_date DATETIME(6)
- death_date DATETIME(6)
- fun_fact VARCHAR(255)
- norm_name VARCHAR(255)
- sci_name VARCHAR(255)
- status VARCHAR(255)
- tot_views INT

user
- id BIGINT
- create_date DATETIME(6)
- password VARCHAR(255)
- user_type VARCHAR(255)
- username VARCHAR(255)

log
- id BIGINT
- action VARCHAR(255)
- creation_date DATETIME(6)
- butterfly_id BIGINT
- user_id BIGINT

bf_lookup
- id BIGINT
- normal_name VARCHAR(255)
- scientific_name VARCHAR(255)

Functionality Demo