Go Find KY - Project Outline

Overview

My objective for this project is to create a responsive, visually appealing website to track a user's progress in spotting, identifying, and logging their sightings of local wildlife. I aim to achieve this using data pulled from an API/ multiple APIs containing images of the animals, a short summary of them, and possibly more information.

- **Tools and Technologies:** Utilize HTML, CSS (Grid, Flexbox, Media Queries), JavaScript, Fetch API, Express.JS and Node.JS.
- **Goals:**
  - Implement responsive design using media queries.
  - Use Fetch API to retrieve data from an API.
  - Use Express.JS and Node.Js to host a web server.
  - Use local storage to store user's data.
  - Create a smoothly functioning and visually pleasing site.

Selected Features

To meet the project guidelines I plan to implement the following:

1. Analyze data stored in arrays, objects, sets or maps and display information about it.
2. Use a regular expression to validate user input, inform the user if invalid, and do not store incorrect input.
3. Persist important data to the user to local storage to make the data accessible in the app.
4. Create a Node.JS server using Express.JS.
5. Retrieve data from an API. Planning on using Cornell Labs API for my MVP if I just go with birds, but if I'm able to implement more, I'll be using Inaturalist's API for animal data, as well.

Planned features

Overview: A website containing a database of animals and birds commonly found in Kentucky or surrounding areas. Users can checkmark animals/birds as SEEN (Spotted), HEARD (Heard call, or song) or wishlist certain ones. Users can search the database specifying animal/birds, size, color, name.

1. User’s stored info

Plan on implementing local storage to save user’s data on what birds/animals they have marked as seen, or heard. Possible features considered, such as showing the user how many times they have seen or heard, and adding a comment box for each ‘log’ so they can store anything they wish within 500 characters. Also automatically store the times/most recent time they were logged.

2. Create and implement a search function

Allow users to search the database and find a certain animal/bird, based on size, color, name ect and validate user input to ensure no incorrect data is received.

3. Create a Node.JS server

The application will use a Node.JS and Express server to handle data exchange and manage sighting logs. It will also utilize an external wildlife API to display relevant species information. Handle GET and POST routes for user logs, allowing the frontend to send and retrieve data. While the backend will manage data flow, user-specific sighting information (such as seen/heard logs and comments) will be stored in the browser’s local storage. This ensures that the data persists after page reloads and remains immediately accessible to the user.

Possible future updates:

- Implementing filters for seasonal or regional sightings.
- Building a map feature to visualize sighting locations using data from users or APIs.
- Including seasonal or regional sighting highlights, such as migration patterns or common birds in the user's area.
- Allowing users to upload images or audio clips of their sightings to share evidence or enhance identification.
- Adding user authentication so users can create profiles and track their personal sightings.