Selection test answer

Approach to the problem

My approach to solving this problem is to build a web as the front end, Express to build the backend. I choose MySQL as the database. The web contains two pages: *Add New Duck Data* page and the *All Duck Data* page.

Add New Duck Data page displays a form. After the user fills the form, it validates all fields. The form is allowed to be submitted when all requirements match. It ensures scientists can collect valid data that contains all required information for their study. When the submission succeeds, React will fetch data through endpoints constructed by Express. Express communicates with MySQL and returns a response. React uses response data to update the *All Duck Data* page.

All Duck Data page lists all collected information in a table. Scientists can review all data submissions. When a form is submitted or users navigate to the *All Duck Data* page, it loads the most up-to-date data.

Technologies chosen

React

The web application has a frontend and a backend. I use React to build the front end for it’s lighter and more flexible than Angular. React uses JSX. It can develop the UI fast and dynamically. It also is less error-prone as the compiler shows errors and warnings right away. It has a large community. Thus, I can easily find solutions when encountering problems. There are plenty of npm packages that support it and provide powerful tools to solve problems efficiently. For example, I can build form and validation features very quickly with the help of Formik and Yup. Considering the test itself and the time limit, I believe React is a good option.

Bootstrap

I mainly use Bootstrap to style the application. It is a popular front-end open-source toolkit. I use its grid to manage my layout and components like Spinner to enrich the page. Using Bootstrap's CSS classes enables me to spend less time on CSS and focus on other tasks. That's the main reason I use it.

Express

Express is a minimal, flexible and powerful Node.js framework. I am familiar with it. It can build a simple while robust backend that supports RESTful API in a few hours. Furthermore, it uses JavaScript. It means the front end and the backend uses the same language. It decreases the extra learning effort.

MySQL

I choose MySQL as the database. It's open-sourced relational database, which makes it quite suitable for this project. Because form data has strong relationships, storing and querying form data in a relational database can be very efficient.
There are two tables in the database. The *location* table stores geographic information. The *duck table* stores duck and food information. The location Id, a foreign key of the *duck table, links the duck data* table with the *location* table. It has several advantages. First, each table won't be too large. It supports querying duck information and location separately. It's handy for data analysis. Second, it reduces data duplication. Columns like city, state and country are repeatable and countable. Therefore, in the future, they can be stored in different tables if data grows up. By using foreign keys, they can link to the location table.

Spent Hour

Front end

- Features ~2.5 hrs
- Tests ~1.5 hrs (I am unfamiliar with tests, hence, it took more than I planned.)

Back end and database ~3 hrs
Readme and answer 1-1.5 hr

In total: ~8 hrs

Jiali Jin