Your task is to develop ‘My Experiences’, a web application which allows users to look up for experiences (activities) all around the world.

You are required to build ‘My Experiences’ according to the specification below. You must use Node and Express as the back-end technology, and SQLite, MySQL or MariaDB for the database.

**Database**

The database should follow the structure below. In your implementation, you may choose to use additional database tables. If you do, they must be documented, with justification.

You will be provided with an SQLite .db file.

*experiences - represents individual experiences (activities)*

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Role** |
| id | INT, PRIMARY KEY | An auto-incrementing index uniquely identifying each record |
| exp\_name | VARCHAR(255) | the name of the Experience |
| exp\_type | VARCHAR(255) | the Experience type, e.g. photography, cooking, sports, etc. |
| country | VARCHAR(255) | the country that the Experience can be found in |
| region | VARCHAR(255) | the region that the Experience can be found in, e.g. Hampshire, Normandy, Bavaria, California, etc |
| lon | FLOAT | the longitude of the Experience place |
| lat | FLOAT | the latitude of the Experience place |
| exp\_description | TEXT | the Experience's description |
| bookings | INT | number of total bookings for this Experience |

*users - represents ‘My Experiences’'s users*

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Role** |
| id | INT, PRIMARY KEY | An auto-incrementing index uniquely identifying each record |
| email | VARCHAR(255) | the email |
| username | VARCHAR(255) | the username |
| password | VARCHAR(255) | the password |

*bookings - represents bookings information of the activities*

|  |  |  |
| --- | --- | --- |
| **Column** | **Type** | **Role** |
| id | INT, PRIMARY KEY | An auto-incrementing index uniquely identifying each record |
| people | INT | Number of people for this booking |
| expID | INT, FOREIGN KEY | the ID of the Experience that this booking is for (from the experiences table) |

**Task Detail**

**Part A – Develop a very simple REST API**

You should first develop a simple REST web API using Node and Express which allows clients to:

1. Look up all activities in a given region. It should return the results as JSON.

2. Add a new Experience. This API endpoint should simply read in the Experience details as POST data, and add them to the database.

3. Book an Experience. This API endpoint should read in the Experience ID and increase the number of bookings by one for that Experience int the ‘experiences’ table.

**Part B – Develop a simple AJAX-based JavaScript front-end**

Next, you should build a simple HTML and JavaScript front-end which communicates with your REST API using AJAX (no page reload should be necessary). You can use a view engine such as EJS or Express Handlebars, or optionally use React for extra marks (see Part G, at this point you should have decided if you will be using a view engine or the React framework for your frontend).

4. Write an HTML page which allows the user to search for all activities in a given region. The user should be able to enter a region, and then, using JavaScript, and AJAX, the page should communicate with your REST API to find all activities in that region. The JSON must be parsed, and the results presented to the user in a user-friendly way.

5. Write another HTML page which allows the user to enter new Experience details. Again, using JavaScript and AJAX, the page should communicate with your web API. Finally, using a standard HTML hyperlink, link the HTML page to the task 4 HTML page.

6. Modify your code to process the search results, so that you create a “Book” button for each result. When the user clicks on this button, you should send an AJAX POST request to the REST API (task 3) to allow the user to book the Experience.

**Part C – Adding simple error-checking**

7. Add error-checking to task 2, so that if any of the Experience details are blank, an appropriate HTTP error code is sent back to the client. Then, in task 5, test for the HTTP code returned from the server and display an appropriate message to the user.

**Part D – Adding a map**

8. Using Leaflet, add an OpenStreetMap map to Task 4, so that the results are displayed as markers on the map. When a user clicks a marker, the Experience name and description should appear as a popup.

You must use Leaflet and OpenStreetMap. In particular, Google Maps is NOT acceptable.

9. Allow the user to add a new Experience by clicking on the map at a particular location. When the map is clicked, the user should be able to enter the Experience details. When a button is clicked, the data should be sent over to your REST API (task 2) and a marker should be added to the map. The marker should appear only if the REST API returns a success HTTP code. Check for errors returned from the REST API and communicate them to the user in a user-friendly way.

**Part E – logins and sessions**

10. Implement a session-based login system. A user should be able to login from the main index page (task 1) using **either their username or email**. If a user logs in successfully, a message should appear within a <div> on the index page, e.g.

Logged in as johnd

You should also implement a logout facility. Ensure the logged-in message still appears if the user reloads the page. There is no need to implement a signup facility, as the SQL file to populate the database contains existing users.

11. Change task 2 so that a user must be logged-in to add a new Experience, sending back an appropriate HTTP error if they are not. Also change task 9 so that this error is checked, and an appropriate error message displayed to the user if they are not logged in.

**Part F – Implementing a booking system**

12. Edit the Book endpoint from Task 3, allowing clients to book an Experience for multiple people. The users should be logged in for the booking functionality to be available. It should read in the ‘people’ (number of people that will attend the Experience) and ‘expID’ as POST data, and must check that the ‘expID’ exists in the database, the ‘people’ field is not blank, that the user is logged in, and store this information in the ‘bookings’ table.You must return appropriate HTTP status codes if any of these checks fail.

13. Add a booking button and an input field (for the number of people) attending to the popup from Task 8, allowing the user book an Experience. When the user presses on the booking button, the value of the ‘people’ input field must be sent, along with the ‘expID’, to the REST API (task 12). You must check for errors returned from the REST API, and display them to a user in a user-friendly way.

**Part G – Using React**

14. Use React to implement the front-end.

To fetch information from the backend and communicate with the database, you must implement AJAX using the fetch API or the Axios library.