Hands-on Lab: Travel Recommendation Web Application

Estimated time needed: 90 minutes

Introduction

In this final project, you will create a webpage layout for a travel recommendation platform.

This website will include:

- A navigation bar with links to the home page, "About Us" and "Contact Us" pages
- A search bar allowing users to input destinations or keywords to search for recommendations
- A content section that showcases an enticing background image, social media links, and a headline emphasizing exploration and destination discovery
- A brief description promoting cultural exploration
- A "Book Now" button

You will be able to implement the knowledge and skills you have gained from working on the practice project to help you handle dynamic functionalities, like search recommendations and result presentation.

Learning objectives

After completing this lab, you will be able to:

- **Enhanced user navigation:** Implement an intuitive navigation bar with clear links to the Home, About Us, and Contact Us sections, ensuring a user friendly website.
- **Search functionality integration:** Develop a search feature allowing users to input destinations or keywords, enabling the 'Search' and 'Clear' buttons to retrieve and reset dynamically displayed recommendations promptly.
- **Visual appeal and brand representation:** Create an aesthetically pleasing landing page with visually engaging elements, including a logo, social media icons, and captivating visuals, to enhance brand recognition and user engagement.
- **JavaScript functionality:** Utilize JavaScript to enable dynamic functionalities, such as live search recommendation fetching, interactivity for user inputs, or potential API integrations, enhancing the website's functionality and user experience.

Prerequisites

- Basic knowledge of HTML and GitHub
- Basic understanding of JavaScript functions, the Fetch API method, strings, arrays
- Web browser with a console (Chrome DevTools, Firefox Console, and so on)

Setting up the environment

1. You need to create one blank online GitHub repository and name it according to the name of your travel recommendation website.

   Important! Make sure your repository is public because your peers will need to review your work.
   Also, ensure that the name of the repository and the folder you are going to create in Point 3 should be the same.

2. Do not create README.md files or others not included in this project.

3. On the window to the right, click on **Explorer** as shown at **1** in the screenshot below. Click on the project folder and again click on the icon highlighted in red at **2** in the screenshot. Enter the folder name **travelRecommendation**. It will create a folder for you. Then select the **travelRecommendation** folder shown at **3**, right-click, and select **New File**. Enter the file name **travel_recommendation.html** and click **OK**. It will create your HTML file.

4. Again, right click on the **travelRecommendation** folder and select a **New File**. Enter the file name **travel_recommendation.js** and click **OK**. It will create your JavaScript file.

5. To create a JSON file, right-click on the **travelRecommendation** folder again and select a **New File**. Enter the file name **travel_recommendation_api.json** and click **OK**. It will create your JSON file.

6. Click on this link travel_recommendation_api and copy data, then paste into the **travel_recommendation_api.json** file and then save it.

   Note: This data will act as API data where details for travel-based recommendation information are available.

Task list

Your required tasks for this project are listed below. Your peers will assess you on each of them.

Task 1: HTML template

Create a basic HTML template structure in the **travel_recommendation.html** file. It should include these tags:

- `<html>`
- `<head>`
- `<body>`
- Within the `<head>` tag include a `<title>` tag.

Write your website name in the `<title>` tag to set up the title of each web page.

Task 2: Navbar

For this task, you need to create a navigation bar. The navbar should contain the following:

- **Home** page link using anchor tag `<a>` to navigate to home page
- **About us** page link using anchor tag `<a>` to navigate to about us page
- **Contact Us** page link using anchor tag `<a>` to navigate to contact us page
- A **search bar** to allow users to enter keywords to search for recommendations
- A **Search** button to execute the search after the user clicks on it
- A **Reset** button to clear results.

The screenshot below shows a sample navigation bar.

You can also include your website name and logo if you wish.

Task 3: Home page

In this task, you will design a **Home** page which must include:

- A background image
- An introduction to this website

You can see an example home page with these elements in this screenshot:

You can also include icons on the home page for social media platforms.

> **Note:** You can also include CSS in the HTML code based on the themes and the images you choose.

Task 4: About us

In this task, you need to create an **About Us** page, which should include:

- Information about the company
- An introduction to the team members, displaying their names and their designated roles

> **Note**: Make sure that you include the code for the navbar. The navbar on this page should only include Home, About Us, and Contact Us menu items, *not* the search bar and buttons.

The screenshot below shows a sample **About Us** page.

Task 5: Contact us

In this task, you need to create a **Contact Us** page, which should contain the following:

- A form for users in case they want to reach out
- The form should contain:
  - `<input>` boxes for name and email
  - `<textarea>` for users to write their message
  - **Submit** button to submit the form

Task 6: Recommendation results

Now, you need to create logic in your JavaScript file to show results for your recommendations.

> **Note:** You should check the output of your code while developing your JavaScript. Directions to view your output are on the next page of these instructions.

- Fetch data from the **travel_recommendation_api.json** file using the **fetch** API method, from there you can fetch travel-related details, such as the name of the place. You need to have your own images for every imageUrl in the JSON file.

- Fetch the data from the JSON using the `fetch API()` method. To check if you can access the data, you use `console.log` to see if the result is displayed.
- It is good if the `console.log` logs the data. Otherwise, you need to look for a different API.

Task 7: Keyword searches

In this task, you will write JavaScript to accept these keywords and variations the user will enter in the search field in your navigation bar on the home page.

- For example, if the user enters "beach," or "beaches," "Beach" or "BEACH," then you need to write JavaScript code so that it accepts all variations of this keyword.
- For uppercase letters in the keyword, you can convert them to lowercase in your JavaScript using the string manipulation `toLowerCase()` method.
- Similarly, you need to create logic to match keywords entered for **temples** and **countries**.
- The website should display results only after the user clicks the **Search** button.

Task 8: Recommendations

In this task, you need to fetch the details of the places you recommend based on which keyword the user enters: *beach, temple,* or *country.*

For each of these three keywords, your results should display at least two recommendations, an image, and a description. Example screenshot is shown below.

Task 9: Clear button

Create logic in your JavaScript file for a **clear** button to clear the results. To implement this feature, you can create a function that will be called after clicking on the **clear** button in the navbar.

Task 10: Country date and time (optional)

In this optional task, you can create logic in your JavaScript to display the time in the country you recommend.

For Example:

```
const options = { timeZone: 'America/New_York', hour12: true, hour: 'numeric', minute: 'numeric', second: 'numeric' };
const newYorkTime = new Date().toLocaleTimeString('en-US', options);
console.log("Current time in New York:", newYorkTime);
```

- This JavaScript code retrieves the current time in the New York time zone using the Date object and the toLocaleTimeString method.
  - `const options = { timeZone: 'America/New_York', hour12: true, hour: 'numeric', minute: 'numeric', second: 'numeric' };`
    - timeZone: ‘America/New_York’ sets the time zone to New York. hour12: true specifies that the time should be displayed in 12-hour format (AM/PM).
    - hour: ‘numeric’, minute: ‘numeric’, and second: ‘numeric’ indicate that the hour, minute, and second components of the time should be displayed numerically.
  - `const newYorkTime = new Date().toLocaleTimeString('en-US', options);`
    - new Date() creates a new Date object representing the current date and time.

        - toLocaleTimeString('en-US', options) formats the time according to the specified options in the 'en-US' locale (English - United States) and the provided options object (options). This method returns a string representing the time in the specified format.
    - console.log outputs the string "Current time in New York:" along with the formatted time retrieved in the newYorkTime variable.

Check the output

1. To view your HTML page in your browser, use the built-in Live Server extension. Select the file **travel_recommendation.html** within the project folder and right-click on it. Choose **Open with Live Server**.
2. A notification will appear at the bottom right, indicating that the server started on port 5500.
3. Click on the **Skills Network** icon button on the left, shown at 1 in the screenshot below. It will open the "Skills Network Toolbox" menu. Click the Other, then Launch Application shown at 2 in the screenshot below. From there, enter the port number `5500` shown at 3.
4. Click on this button .
5. Your default browser will open. You will see the **travelRecommendation** folder. Click on that folder name and then click again on the **travel_recommendation.html** file.
6. It will open the front page, and you can see the output.

Perform git commands

1. You need to save all your files to perform git commands.
2. Perform `git add`, `git commit`, and `git push` commands to update changes inside the **travelRecommendation** folder inside your GitHub repository that you created at the start for proper code management.

   Note: If you're pushing code without cloning from a GitHub repository, remember to utilize "origin2" for the `git push` command. If you've logged out and want to resume work on the project, clone the repository from the GitHub where you previously pushed the code. After cloning, you can simply use `git push origin` without the need to execute `git remote add...` to push your changes directly.

Create git pages

Create git pages for your website as practiced in the **Practice Project**.

> **Important!** Make sure that your repository is public and that you have saved the URL for both the GitHub repository and for the link, which you will generate after creating Git pages for your website for graded review.

Instructions to Verify Your Site on GitHub Pages

If you are using `index.html` file:

1. Open your browser and navigate to the URL:
   `https://<your-username>.github.io/<your-repo-name>/`
2. Verify that the site loads correctly and displays the content from your `index.html` file and automatically loads the homepage of the website.

If you are using `travel_recommendation.html` file:

1. Open your browser and navigate to the URL and append the name of the html file you are using, it will automatically load the homepafge for you.

`https://<your-username>.github.io/<your-repo-name>/travel_recommendation.html`

2. Check that the `travel_recommendation.html` file loads correctly and displays the expected content.

Summary

1. **HTML structure:** The HTML file establishes a structure comprising navigation links to **About Us** and **Ccontact Us** pages, and the **Home** page.

2. **JavaScript logic:** The embedded JavaScript code manages the user's query to display related information.

3. **Fetch data analysis capabilities:** The code includes functionalities to filter the user query keyword to fetch the data, which recommends travel options.

4. **Event handling:** Event listeners are set up to respond to user interactions with the navigation links and then apply event listeners for search and clear buttons.