IT211 Project 2
Score: ______/100 v3.0

Specifications

In this project, you will create a personalized weather application that allows a user to register and log in using an email and password. The user will be able to save their zip code, view the current weather or a 5-day forecast for that location, and switch between dark mode and light mode. You will use localStorage to store user data, cookies to track the logged in user and for the theme preference (dark or light), and Bootstrap for styling.

Requirements

1. User Registration and Login:
    - Users must be able to register with an email and password.
    - Store the email, password, and zip code in localStorage as a stringified JSON object.
    - When a user logs in successfully, store a cookie to represent the logged-in user (e.g., the user's email).
2. Saving Zip Code:
    - Each user should have a unique zip code stored in localStorage.
    - Users can modify their zip code on the Settings page, and it should be updated in localStorage.
3. Weather API Integration:
    - Use https://openweathermap.org to display:
        - Current Weather for the user's saved zip code.

          https://openweathermap.org/current#zip
        - 5-day Weather Forecast for the user's saved zip code.

          https://openweathermap.org/forecast5#zip5
    - The Current Weather page should update automatically every 30 seconds.
4. Theme Selection (Dark Mode/Light Mode):
    - Allow users to switch between dark mode and light mode using a UI toggle.
    - Store the selected theme in a cookie, which should persist across sessions.
    - The theme setting should be applied globally (even if multiple users share the device, since cookies are not user-specific).
5. Navigation Bar:
    - Create a Bootstrap navigation bar with:
        - Links to the Home, Current Weather, and 5-day Forecast pages.
        - A Settings link where users can change their zip code and toggle dark/light mode.
        - A Logout link.

6. Logout:
    - The logout function should delete the login cookie to log the user out, but localStorage should remain unchanged.
    - After logging out, the user should be redirected to the Login page.
7. Login Redirection:
    - Implement a login check for every page of the application (except the Login and Register pages).
    - If a user is not logged in (i.e., the login cookie does not exist or is empty), they should be automatically redirected to the Login page.

Page Details

1. Login Page:
    - A form for users to log in with their email and password.
    - Store the user’s email in a cookie to track the current session.
    - Redirect to the Home page upon successful login.
2. Register Page:
    - A form to register new users by entering an email, password, and zip code.
    - Save this information in localStorage as a stringified JSON object.
    - After registration, log the user in by setting the login cookie and redirect to the Home page.
3. Home Page:
    - A welcome page that greets the user by their email (retrieved from the cookie) and displays their saved zip code.
    - Include buttons or links for Current Weather and 5-day Forecast.
4. Current Weather Page:
    - Display the current weather based on the user’s saved zip code.
    - The weather data should update every 30 seconds automatically.
5. 5-day Forecast Page:
    - Display the 5-day weather forecast for the user’s saved zip code.
6. Settings Page:
    - Allow users to change their zip code and update it in localStorage.
    - Include a toggle for switching between dark mode and light mode.
    - Save the theme preference in a cookie.

Additional Constraints

- The following must be used effectively:
  - CSS
  - Cookies
    - Cookies should be used only for preserving user preferences or authentication state.
  - JSON
- jQuery cannot be used.
- You must use localStorage.
  - https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
- A navigation bar (or other medium for navigation) must be used.
- When logging in, the system will verify the existence of an email or password in the localStorage. Upon successful login, some cookie's value can be straightforwardly set to the user's email to indicate that the user is authenticated. If a user is not logged in, they should be automatically redirected to the login page from any other page.
- If multiple JavaScript and CSS files are used, they should be placed in subdirectories to not crowd the project's root directory.

Extra Credit Opportunities

- (6 pts) This bonus is available for students who go beyond the standard styling requirements and use CSS and/or Bootstrap to create a visually exceptional weather website. To qualify, your website should be highly polished and professional, with design elements that rival production weather websites (e.g., clean layout, responsive design, great use of colors, typography, and icons).
- (3 pts) **Effectively** utilize Bootstrap Icons.

Submission

You will submit the commit ID for the commit you want graded. Submit the commit ID on the Canvas assignment.

Daily commits and pushes to your assignment GitHub repository are mandatory.

Tips

- Spend some time crafting your API calls using Postman.
- Ensure you understand the data returned via your API requests. In the linked openweathermap.org pages, there are sections that document the responses. For example:

  https://openweathermap.org/forecast5#fields_JSON

- The APIs should return the name of an icon (e.g. 01d). You can use this page here if you want to display those icons:

  https://openweathermap.org/weather-conditions

Academic Honesty Policy

All code submissions must be original and authored by the student. Any code sourced from another student, falsely presented as one's own, or derived from third-party websites or generated by AI, will constitute a breach of the school's academic honesty policy.

Daily commits and pushes to your assignment GitHub repository are mandatory while working on your project. Failure to comply will result in deductions from your final grade at minimum, and could lead to academic honesty violations.