Explore Warsaw: Interactive Travel Guide Web Application

Ricardo Saettone

December 19, 2025

Abstract

This report details the "Explore Warsaw" web application, an interactive guide designed to enhance the tourist experience in Poland's capital. The project leverages modern web technologies to provide features such as user authentication, a personalized trip planner, dynamic reviews, and real-time weather information, all managed client-side using LocalStorage.

Contents

1 Modularized Architecture **3**

2 Document Purpose and Functionality **3**
2.1 HTML Documents 3
2.2 CSS Documents 3
2.3 JavaScript Documents 3

3 Visual Overview and Features **4**
3.1 Main Page (Home) 4
3.2 User Authentication (Side-by-Side) 4
3.3 Dynamic Review System (Side-by-Side) 4

4 Conclusion **5**

1 Modularized Architecture

The application is organized into a modular structure to separate concerns and improve maintainability. The diagram below represents the project's logic flow:

- **HTML Pages**: Structural entry points for each category and the main dashboard.
- **CSS Styles**: Centralized visual rules and responsive grid systems.
- **JS Modules**:
  - *Auth*: Identity and session management.
  - *Planner*: User-specific itinerary logic.
  - *Comments*: Dynamic feedback and rating system.
  - *Weather*: External API data fetching.
- **LocalStorage**: Serves as the persistent data layer for the entire application.

2 Document Purpose and Functionality

2.1 HTML Documents

- `index.html`: The landing page. It acts as the central hub, displaying categories via cards and the "My Trip" itinerary summary.
- `restaurants.html, hotels.html, etc.`: Specialized pages that list specific venues. They provide descriptions, "Add to Trip" buttons, and the review sections.
- `guides.html`: A resource center for downloading PDF travel maps and official guides.

2.2 CSS Documents

- `style.css`: Manages the global aesthetic. It implements a fixed background image, custom color palette (#39536a and #d4192f), and the responsive grid system (`auto-fit`) that adapts the layout to mobile devices.

2.3 JavaScript Documents

- `auth.js`: Handles user registration and login. It injects a modal into the DOM and validates credentials against LocalStorage.
- `planner.js`: Manages the `myTrip` array. It prevents duplicates and renders the list dynamically on the home page.
- `comments.js`: Controls the review logic. It identifies the current venue via `data-` attributes and manages visibility based on the `currentUser` status.
- `weather.js`: Connects to the Open-Meteo API to retrieve real-time data for Warsaw (52.23N, 21.01E).

3 Visual Overview and Features

3.1 Main Page (Home)

The homepage integrates the weather widget, the category grid, and the trip planner.

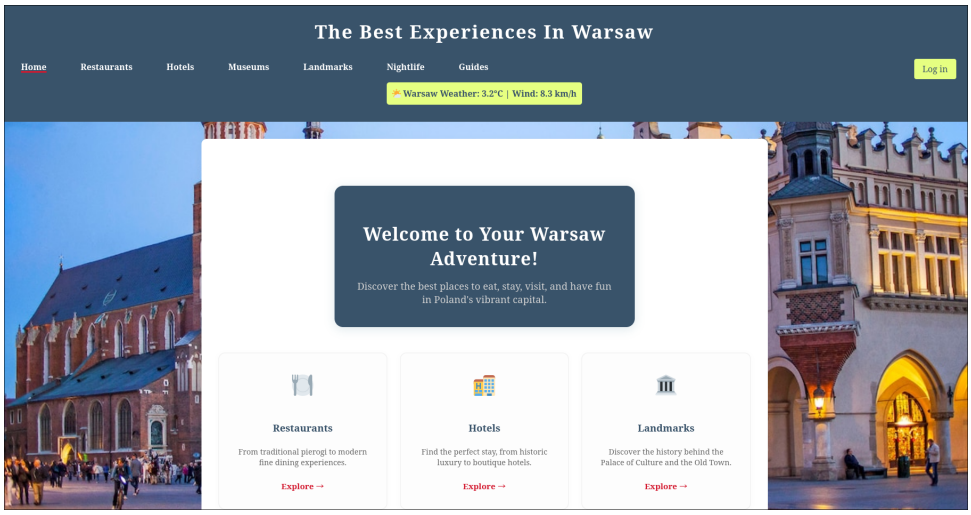

Figure 1: Main interface with the active Trip Planner.

3.2 User Authentication (Side-by-Side)

The login system provides instant feedback for failed attempts and a clean registration interface.

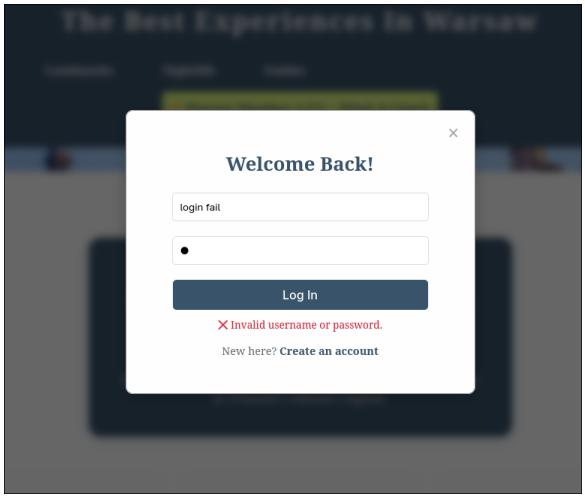

Figure 2: *
Failed Login Attempt

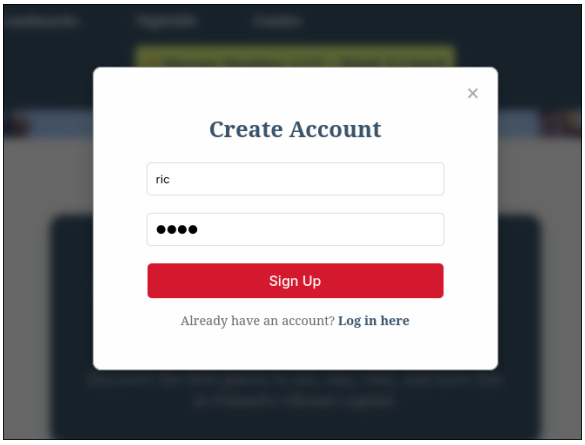

Figure 3: *
User Registration (Sign Up)

Figure 4: Comparison between the error handling in login and the registration form.

3.3 Dynamic Review System (Side-by-Side)

Access to the review form is restricted to authenticated users to ensure data quality.

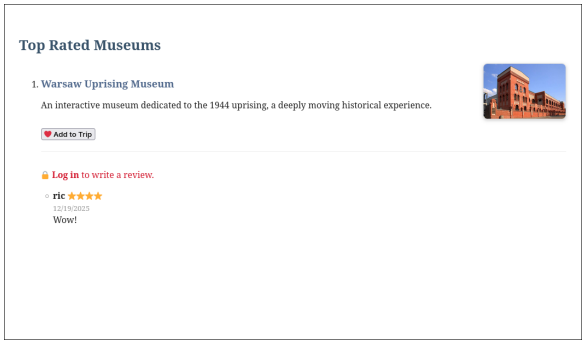

Figure 5: *
Logged Out: Locked Form

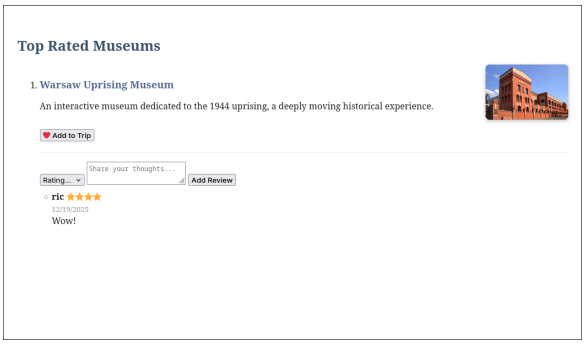

Figure 6: *
Logged In: Review Enabled

Figure 7: Review section behavior depending on authentication status.

4 Conclusion

The "Explore Warsaw" project demonstrates a complete frontend ecosystem. By modularizing the code and utilizing LocalStorage it is possible to expand the code and improve the website in the future. By using HTML, CSS and JavaScript a complex page can be created.