# WDD 330 Personal Project

This document serves as your final course assessment.

## **Introduction**

**Name**: [Adetokunbo Olutola Osibo]

**Video Link**: [Insert your video link here]

**Working Application Link**: https://aleejotrips.netlify.app/

**GitHub Source URL**: [Tolaji/AleejoTrips: Travel Website](https://github.com/Tolaji/AleejoTrips)

**Trello Board URL**: [Travel Planner Web App | Trello](https://trello.com/b/HRk9vNJz/travel-planner-web-app)

## **Course Outcomes**

The following are the course outcomes of WDD 330:

1. Become more efficient at applying your innate curiosity and creativity.
2. Become more dexterous at exploring your environment.
3. Become a person who enjoys helping and learning from others.
4. Use a divide and conquer approach to design solutions for programming problems.
5. Finding and troubleshooting bugs you and others will have in the code you write.
6. Developing and debugging HTML, CSS, and JavaScript programs that use medium complexity web technologies.

To complete this course, you need to demonstrate your skill in these areas. Outcomes #1-5 demonstrate your personal development and are most easily shown through self-assessment and sharing experiences. Outcome #6 demonstrates your programming skill and is shown through code and experience in projects.

## **Skill Development Outcome**

*Developing and debugging HTML, CSS, and JavaScript programs that use medium complexity web technologies*.

This outcome is demonstrated by your skill in the following learning objectives:

|  |  |  |
| --- | --- | --- |
| **Objective** | **%** | **Description** |
| JavaScript | 25% | Robust programming logic is demonstrated.  For example, validating the screen data, looping through an array of JSON data to display to the screen, creating and using events, changing element styles with JS, changing element classes to use different CSS rules. |
| Third-party APIs | 15% | APIs are used effectively, including APIs that provide rich JSON data. |
| JSON | 15% | Demonstrate skill processing JSON data to dynamically update the website. |
| CSS | 15% | Appropriate use of Transforms and Transitions. For example: Add round the edges to DIV, add shadows. enlarge an input field on focus and shrink it on blur, Add borders. CSS should subtly add style to a page. |
| Events | 15% | Use events to enhance the user experience. For example, increase the size of the input field on focus or add a shadow. React to a button click. Initialized the page with data once the onload event triggers. |
| Local Storage | 5% | Local storage is used effectively. |

These learning objectives are rated on the following scale:

|  |  |
| --- | --- |
| **Rating** | **Description** |
| Unsatisfactory | Very little if any work was shown in this area. |
| Developing | The learning objective was shown in very basic ways. |
| Proficient | Effective use of the learning objective was shown in multiple places. |
| Mastery | Extensive use of the learning objective was shown in non-trivial ways in many places in the code. |

For each learning objective, discuss how the topic was used in your application. List several examples of places where the topics are demonstrated.

The following is an example of what is expected:

|  |  |  |
| --- | --- | --- |
| **Learning Objective** | **Description** | **Where can this be seen in your application?** |
| CSS | *I spent a lot of time choosing colors that would complement each other.*  *I used CSS to make the input field bigger when it received the focus and to shrink it when it lost focus.* | *This can be seen on the home screen for each input field.* |
| *Images are enlarged on hover.* | *The recipe detail pages have this effect.* |
| The search results have alternating colors for the rows for readability. | See the home page after a search is successfully run. |

In the following table:

1. Describe how the topics are used.

Have someone test your links to make sure they are accessible by the grader. These links will be to your final personal project.

Feel free to add more rows to this table if needed.

|  |  |  |
| --- | --- | --- |
| **Learning Objective** | **Description** | **Where can this be seen in your final personal project application?** |
| JavaScript | Wrote robust programming logic to handle live tracking, update distance in real time, and gracefully handle GPS/IP changes. Used loops to process arrays of location data and dynamically update the DOM. Implemented event listeners to track map movements and distance updates. | In server.js, the client listens for locationUpdate via Socket.IO, loops through incoming session data, and updates the distance display every second. Styles change for active/idle status using JS class toggles. |
|  |  |
|  |  |
| Third-party APIs | Integrated multiple APIs: IP Geolocation API (to locate users via IP), Browser Geolocation API (for GPS accuracy), and Leaflet.js (for map rendering). | Map rendering and markers are powered by Leaflet.js. IP Geolocation API is used when GPS is unavailable. |
|  |  |
|  |  |
| JSON | Parsed and processed JSON payloads from server to update map positions, session states, and distance values dynamically. | Every Socket.IO event sends JSON (latitude, longitude, UUID, timestamp) which is parsed and used to reposition map markers and update the on-screen distance counter. |
|  |  |
|  |  |
| CSS | Used transforms and transitions for a smooth, responsive UI. Input fields grow on focus, buttons have hover effects, and map markers have subtle shadows for depth. | The dashboard CSS uses transform: scale(1.05) on button hover, rounded edges and box shadows for status panels, and smooth transitions for distance text updates. |
|  |  |
|  |  |
| Events | Leveraged multiple event types: click, focus, blur, socket message events, and map marker click events. | Clicking a session link triggers joining a session, focus on the name input enlarges it, clicking a map marker shows device info, onload triggers initial map rendering. |
|  |  |
|  |  |
| Local Storage | Used to store the UUID and session token so the device remains linked even after page reload or IP change. | When a user joins a session, UUID is saved in localStorage so reconnections automatically restore session state without needing to re-enter details. |
|  |  |
|  |  |

So sorry I couldn’t get the video done, the location tracker took most my time.