**SDLC Phase Document**

**Phase 1: Planning**

* **Goal**: Build a real-time search interface for travel destinations that fetches data from an API and shows the current time of each destination.
* **Scope**: The system will display destination information dynamically as the user types. Data will be fetched asynchronously from an API and displayed in real-time.

**Phase 2: Requirement Analysis**

* **Functional Requirements**:
  + A search bar that filters results based on user input (place name, city, or country).
  + Integration with a data source (static JSON or API).
  + Display of the current local time for each destination based on its timezone.
* **Non-Functional Requirements**:
  + The system must handle large datasets efficiently.
  + Real-time search should be responsive with debounce logic.
  + The UI should be simple and easy to navigate.

**Phase 3: System Design**

* **Architecture**:
  + The frontend fetches data from a local or remote JSON file.
  + User input will trigger the filtering of data, showing results that match the search term.
  + Each destination result will display the current time based on its timezone.
* **Detailed Design**:
  + **Debounced Search**: Implement debounced search to prevent excessive API calls.
  + **Fetch Data**: Use fetch() to retrieve the JSON data.
  + **Timezone Conversion**: Use Intl.DateTimeFormat to display the current time for the destination.

**Phase 4: Development**

* **Implement Core Functions**:
  + fetchData(): Fetches data from the JSON file or API.
  + createAndFilterData(): Filters the fetched data based on user input.
  + createCard(): Creates HTML cards to display the destination's details and current time.
* **Timezone Handling**: Use Intl.DateTimeFormat to display the current time dynamically for each destination based on its timezone.

**Phase 5: Testing**

* Test **real-time search** functionality with various inputs.
* Check if data fetching and filtering work as expected.
* Verify that the current time is displayed correctly for each destination based on the timezone.

**Phase 6: Deployment**

* Deploy the app on a local or public server for testing.

**Phase 7: Maintenance**

* Fix any bugs identified during testing.
* Plan for potential future updates, such as adding a backend API or supporting more destinations.