Overview

World Time Zones is a web-based application designed to provide users with real-time information about different countries and their respective time zones. The project integrates multiple data sources and APIs to offer accurate geographical and temporal information. Users can search for countries, filter by region, view interactive maps with timezone overlays, and track the day-night terminator in real time. The development focused on usability, efficiency, and performance optimization to create a seamless experience for users.

Development

The development of the World Time Zones project took approximately two weeks. The project was developed **page by page**, starting with the home page, where the **REST Countries API** (https://restcountries.com/) was implemented. The initial version included search bars and filters using vanilla HTML, without CSS or JavaScript.

Following this, a **pagination system** was implemented, along with additional pages such as Favorites, About, and an Error page. The **Error page** was the simplest, containing only text that prompts users to return to the home page. The **Favorites page** was more complex, requiring the implementation of JSON to locally store favorite countries. The **About page**, while straightforward, served as a hub for referencing the resources used in the project.

Once these core elements were in place, the website's overall design was developed using **CSS and JavaScript**, incorporating colors, animations, and hover effects. One of the most challenging aspects of the home page was implementing **rotating flip cards**, displaying key country details (e.g., name, flag, capital, continent, time, and time zones) on the front and additional data (e.g., bordering countries, population, area, driving side, calling code, official languages) on the back. The implementation initially had display glitches.

Next, a **Maps page** was created, integrating **Leaflet.js** (https://leafletjs.com/) with **OpenStreetMap** to display an interactive world map. The map featured zoom in/out

buttons, a **"View Time Zones" overlay toggle**, and an initial attempt at manually drawing time zone boundaries. However, this approach was unsuccessful (Figure 1).

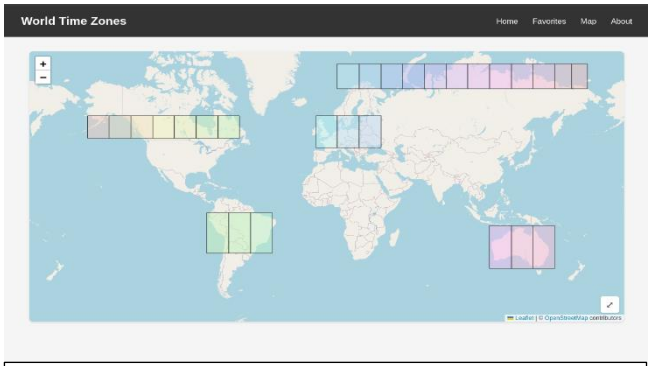

Figure 1 : Incorrect Time Zones Display

To resolve this, the **World Time Zones GeoJSON coordinates** (https://github.com/evansiroky/timezone-boundary-builder/releases) were used instead. The selected file, *timezones-with-oceans-now.geojson.zip*, was the best-suited as alternative versions had missing timezone boundaries in water areas and Antarctica.

Further enhancements included a **second overlay toggle button** to display a live **terminator line**, showing the current transition between day and night across the globe, using JavaScript.

It is important to mention that the initial idea for the project was to use the Air Quality Index API. Since there isn't enough data to create three different filters, it was changed. Nevertheless, the geography theme remained.

Enhancing Data & Performance

To provide more detailed information, the **latest Human Development Index (HDI) report** (https://hdr.undp.org/data-center/human-development-index#/indicies/HDI) was integrated. The dataset was converted from .xlsx to .csv for smoother integration, enabling additional endpoints such as:

- HDI Rank
- HDI Value
- Life Expectancy
- Average Years of Schooling
- Gross National Income per Capita

These data points were displayed under a **second tab** on the back of the flip cards, bringing the total number of endpoints displayed to **18**. The About page was updated accordingly with the new references.

Testing and Optimization During testing, two major **bugs** were identified and resolved:

1. **Misspelled Search Queries** – Displayed an appropriate error message instead of crashing.
2. **Invalid Page Number Input** – Users entering a page number beyond the maximum (21) were redirected to a custom error page instead of the "connection reset" error.

To **optimize performance**, the GeoJSON timezone coordinates file was split into **smaller, more manageable chunks** using geojsplit, a command-line tool installed via the terminal. This reduced loading time significantly.

Finalization & File Structure

The final steps involved **organizing the project's file structure**:

- **Separate folders** for .css and .js files, extracted from the original .html files.
- **GeoJSON files** stored in a dedicated folder.
- **Image assets** stored separately.
- **Splitting the main.go file into seven files** for better maintainability.

As a result, the command to run the project changed from:

`go run main.go` to: `go run *.go`

This was documented in the README.md file.

Development Approach

The project was developed organically, without a strict strategy. Features were added **spontaneously** based on new ideas and challenges that arose. The development was structured **page by page**, ensuring that each section was fully functional before moving on to the next. The entire process took approximately **2–3 weeks**, with additional minor improvements made later.