Meet App

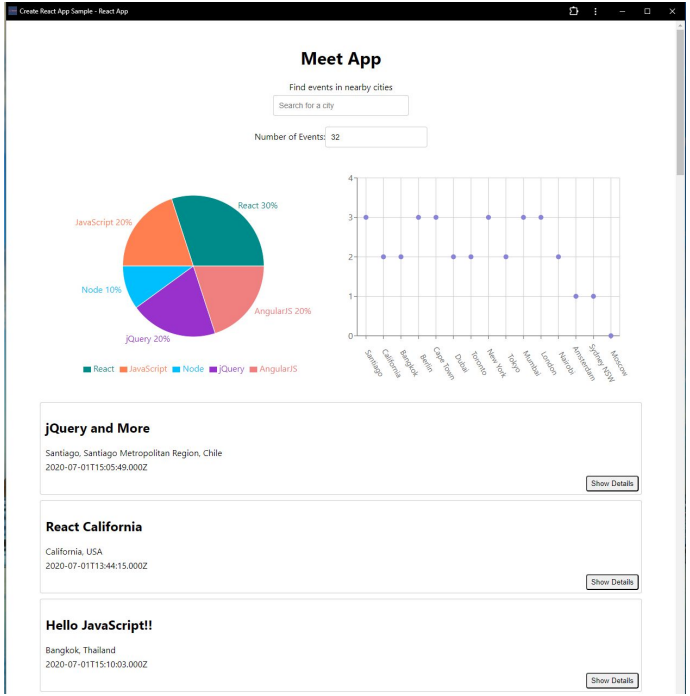
A PWA Case Study

Stephanie Leon

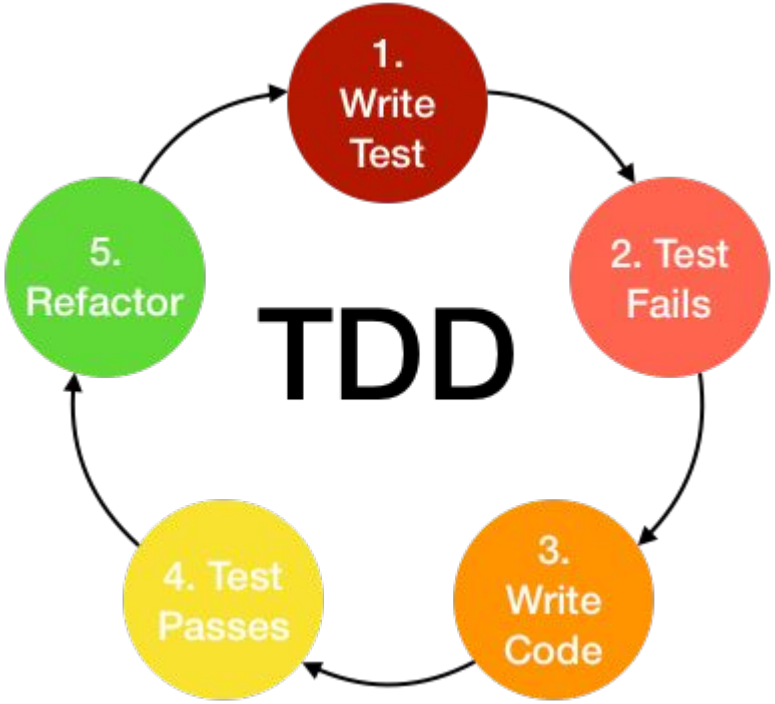
Overview

Meet is a serverless, progressive web application designed with React using a test-driven development technique. The app uses the Google Calendar API to fetch upcoming events.

Users can filter events by city, show/hide event details, specify the number of events shown, view charts visualizing event details, use the app when offline, and add an app shortcut to their home screen.



Purpose & Context



Meet was created as part of a full stack web development course I completed through CareerFoundry. This was my first opportunity to create an application specifically using a TDD approach.

Objective

The aim for this project was to combine two concepts that are growing in popularity for web development: serverless architecture and PWA.

I also practiced writing user stories based on the app's key features and writing unit tests, integration tests, acceptance tests, and doing end-to-end testing.

Feature 1: Filter events by city

```
User Story-
    As a user,
    I should be able to filter events by city
    So that I can see a list of events taking place in that city.

SCENARIO 1: When user hasn't searched for a specific city, show upcoming events from all cities.

    Given: user hasn't searched for any city;
    When: the user opens the app;
    Then: the user should see a list of upcoming events.

SCENARIO 2: User should see a list of suggestions when they search for a city.

    Given: the main page is open;
    When: user starts typing in the city textbox;
    Then: the user should receive a list of cities (suggestions) that match what they've typed.

SCENARIO 3: User can select a city from the suggested list.

    Given: user was typing "Berlin" in the city textbox AND the list of suggested cities is showing;
    When: the user selects a city (e.g., "Berlin, Germany") from the list;
    Then: their city should be changed to that city (i.e., "Berlin, Germany") AND the user should recei
```

Development Process

Technologies and Tools

- React for building an interactive and dynamic user interface
- Bootstrap for responsive design and UI components
- PWA Support to allow the app to be installed as a shortcut on mobile devices
- Jest and Puppeteer for automated testing and end-to-end test coverage
- CI/CD Pipeline was implemented using GitHub Actions to automate testing, deployment, and ensure code quality. Every pull request triggered the pipeline to run unit tests, linting, and build processes before merging.

Development Process

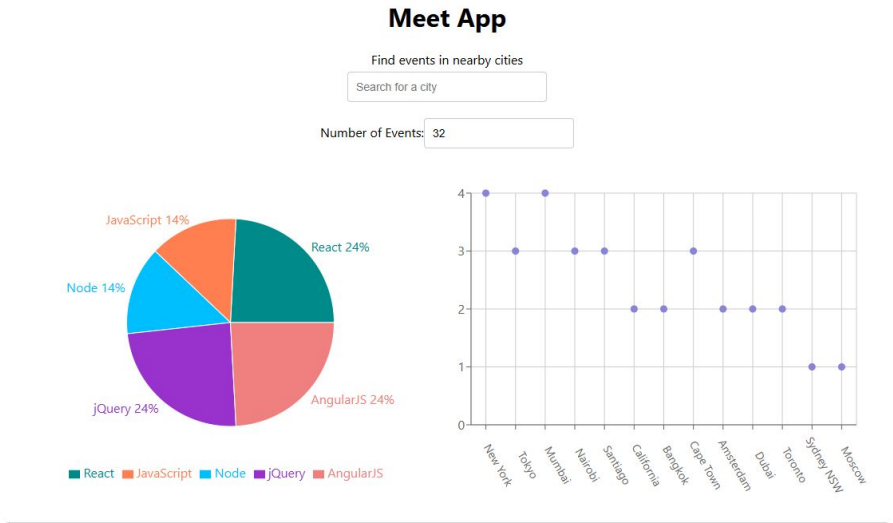
Challenges:

- API Integration: Managing third-party API rate limits and data consistency
- Offline Access: Ensuring seamless functionality without an active internet connection
- Scalability: Designing for future growth and increased user traffic

Solutions:

- Implemented caching and optimized API calls to reduce redundancy
- Used service workers to cache key data and improve accessibility
- Adopted modular code architecture and used robust libraries

Features

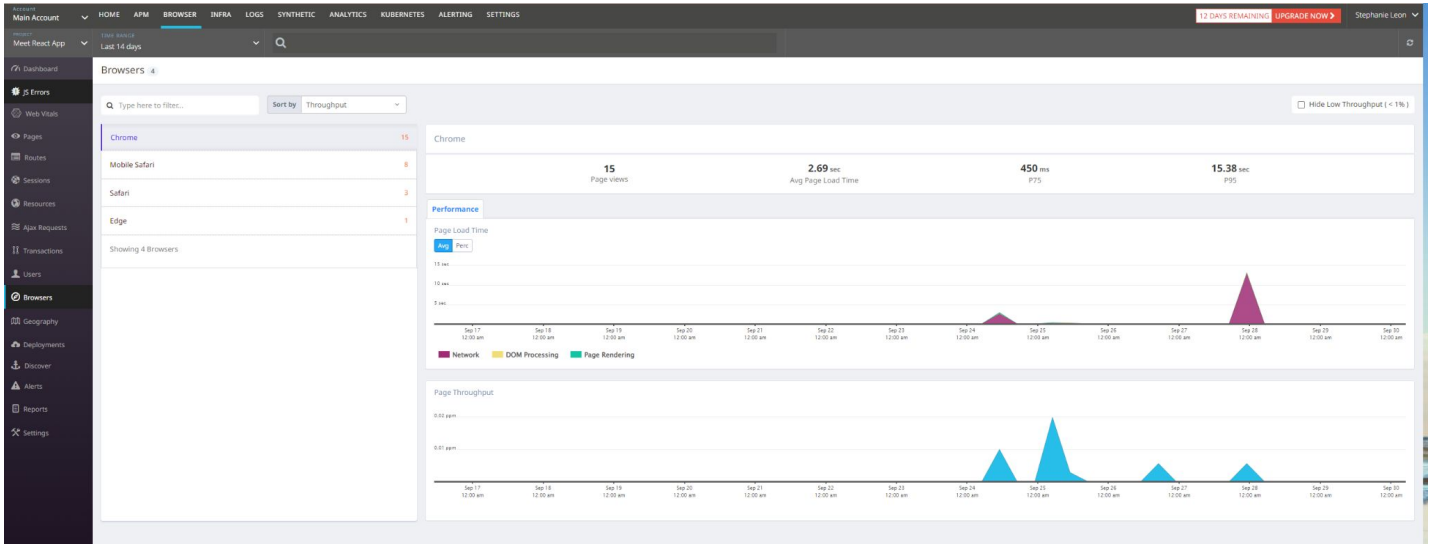


Event Search: Users can search for events by city or keyword

Responsive Design: Optimized for various screen sizes to ensure usability across devices

Event Details: Comprehensive event details, including data, time, location, and description

Features



Multiple Platforms: App must work on the latest version of Chrome, Safari, Firefox, Edge, etc

Offline Support: Users can view previously accessed events without an internet connection

CI/CD Automation: Ensures seamless delivery of updates and bug fixes

Testing Framework: A strong focus on testing guarantees reliability

Test-Driven Development

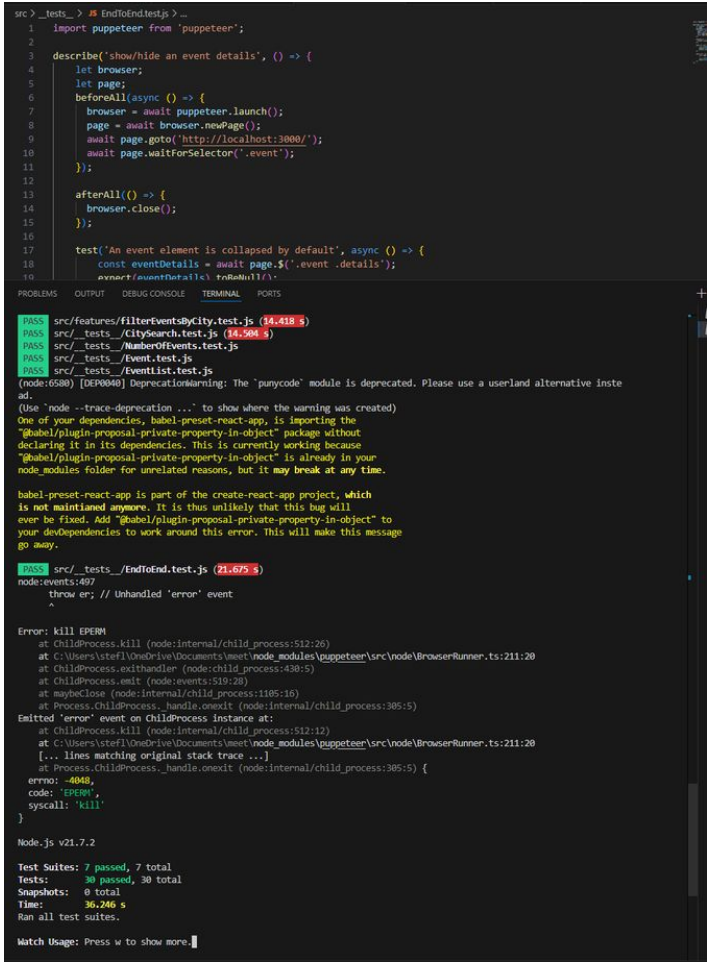
Unit Testing used to ensure individual components function as expected using Jest.

Integration Testing used to verify that multiple components work together seamlessly.

End-to-End Testing completed to simulate user interactions and to validate workflows.

Automated Testing done through the CI/CD pipeline to ensure each code commit to detect and address issues early.

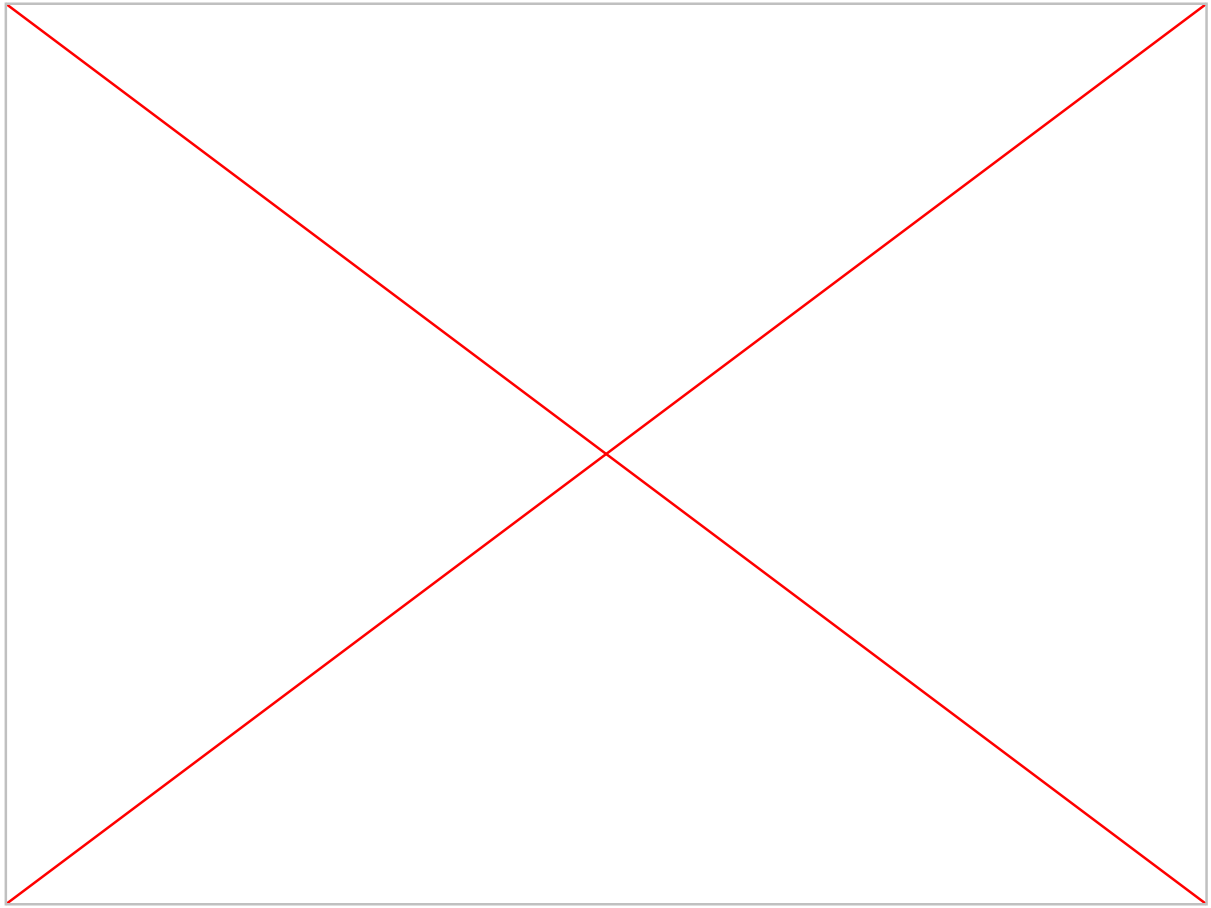
Test Coverage was over 90% and the app was monitored using Atatus.



Final Solution

A user-focused platform that successfully addresses the core challenges of event discovery and management.

Click to see a video of the app and it’s features!



Future Iterations

- Social features like users being able to connect with others attending the same events
- A notification system to send reminders for upcoming events users have selected

Duration:

This project took 3 months to complete. More time than expected was needed as my computer was unavailable for multiple weeks while being repaired and I moved states.

Credits:

Lead Developer: Stephanie Leon
Tutor: Jesus Diaz
Mentor: Shreyansh Kumar