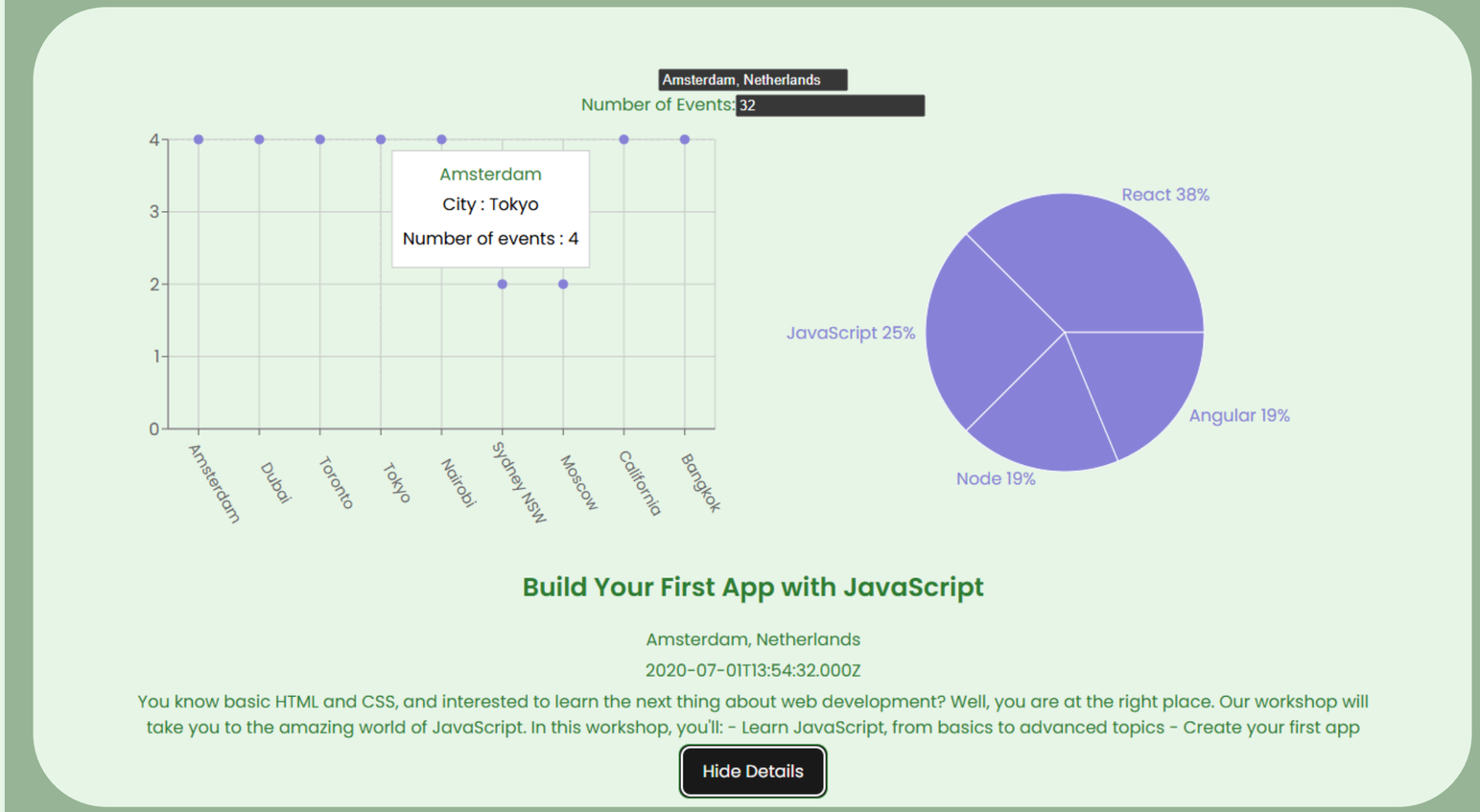
Full Stack Developer Case Study

by Giancarlo Boas

Overview



Check the site

Meet is a serverless Progressive Web App (PWA) built with React, following a test-driven development (TDD) approach. Its main purpose is to display tech-related events happening in various cities. The app integrates the Google Calendar API to fetch event data and presents it through interactive charts and a user-friendly interface..

Context

This project was created as part of the CareerFoundry Web Development Program. It highlights full-stack JavaScript skills, API integration, and data visualization while following best practices in modern web development.

Objective

The primary goal was to build a robust, scalable application using tools like React, Recharts, and GitHub Pages. The project serves as a professional showcase of practical experience with contemporary web technologies.

Hard Work, Great enviorement and the right tools

Duration

This project took approximately two weeks to complete. I had a great time working on it—especially exploring more about APIs. I look forward to building more serverless applications with React in the future.

Methodologies

Core features include offline functionality, push notifications, responsive design, and cross-platform compatibility. These ensure a smooth and consistent experience across both desktop and mobile devices.

Credits

Lead Dev: Giancarlo Boas
Tutor: Marwa Jawad
Mentor: Renish Bhaskaran

Serverless Functions

The backend functionality is powered by serverless architecture, specifically using AWS Lambda. These functions handle user authentication with the Google Calendar API and securely retrieve event data. By leveraging serverless technology, the app scales automatically based on demand, lowers operational costs, and ensures high availability without the need for manual server management.

Technical Approach

The app uses the Google Calendar API to fetch event data and displays it in a clear, organized list. A behavior-driven development (BDD) methodology was used, with structured tests written in Jest-Cucumber, and component rendering and user interactions handled with React Testing Library.

Retrospective

What went well?

I successfully implemented a serverless architecture using AWS Lambda, which initially felt quite challenging and the integration with Google Calendar API works great

What didn’t go well?

Synchronizing backend data with the React UI led to some tricky bugs and I struggled to understand how to properly authenticate with the Google Calendar API

Future Steps

1. Improve the interface with subtle animations and a more polished visual design.
2. Add features allowing users to select favorite cities and filter events by category.
3. Deploy the app on a custom domain and work on basic SEO improvements.

Final Thoughts

This project was a key learning experience in my development journey. It challenged me with real-world technical problems and allowed me to apply modern tools and best practices. What I value most is how much I learned during the process and how this project reflects my growth as a developer.