Foodiee

Next.js/TypeScript/GraphQL migration

Zacharias Christos Argyropoulos

SAE Athens 2022

TOC

About Foodiee

Foodie is **food ordering delivery application** where users can order food from multiple restaurants by adding various items in a overall cart and get them all together at the same time.

The main technologies that were used to build this project when it was first implemented was **Node.js and Express** for the backend (microservice) and **React** for the frontend.

The main goal of this final project was to further investigate and explore state of the art technologies and frameworks as well as refactoring existing code. Eventually this conversion leaded to better performance and SEO.

Why choose Next.js/TypeScript/GraphQL

- **Next.js** is a framework that allows us to write React components as usual that run on the server which ultimately gives us more rendering capabilities than the typical client side rendering. (SSG/SSR/ISR and CSR when needed)
- With **TypeScript** we can define types on our data and components, which catches many errors and mistakes in development and gives us better autocompletion in the editor.
- **GraphQL** will work as a middleware in the existing Node.js application and will reside besides the already defined REST APIs. With GraphQL the client can query exactly the data that it needs, which reduces over fetching data (when some properties in the response body won't be used at all) and under fetching data (having to make multiple separate API calls to get the final desired structured data).

Service refactoring and implementation

01 Overall refactoring on the db collections. It was decided to split the store collection into several ones for better scaling and for GraphQL (resolvers) to make more sense.

02 To follow a more MVC-like project architecture and best practices an additional directory with controllers has been introduced in the project code structure.

03 A new GraphQL endpoint was introduced that was later used in the client side to query and consume data. The types of each data property and the connection between all of them was defined to structure the final graph/schema. Try it out

Client Next.js/TypeScript migration process

- Used the existing project structure and and added the new TS/Next.js dependencies.
- Started from the index.html page migrated the meta/link tags to the appropriate app.tsx, _document.tsx pages.
- Changed the cloudinary image component to Next’s image component while still being integrated with the Cloudinary SDK.
- Changed the links to use Next’s link component.
- Decided what generation and rendering method to use in each page of the application and gradually started migrating those into the application.
- Introduce Redux Toolkit for state management for the components that needed to change
- Strongly defined types for state, props, functions etc. in components that were modified during this process with TypeScript.

Client Rest API/GraphQL queries

- For the graphql queries and mutations that was taking place on the backend a custom `gqlQuery` hook was implemented that is basically a wrapper the apollo client query function to mimic the functionality of the useQuery hook.
- For the client querying the useQuery hook exported from the apollo client package was used.
- For the existing REST APIs a custom SWRAxiosFetcher hook was implemented to wrap the existing client fetching functions and integrate them with swr for better performance. For the API calls in the backend the existing axios fetch function were used.

Deployment

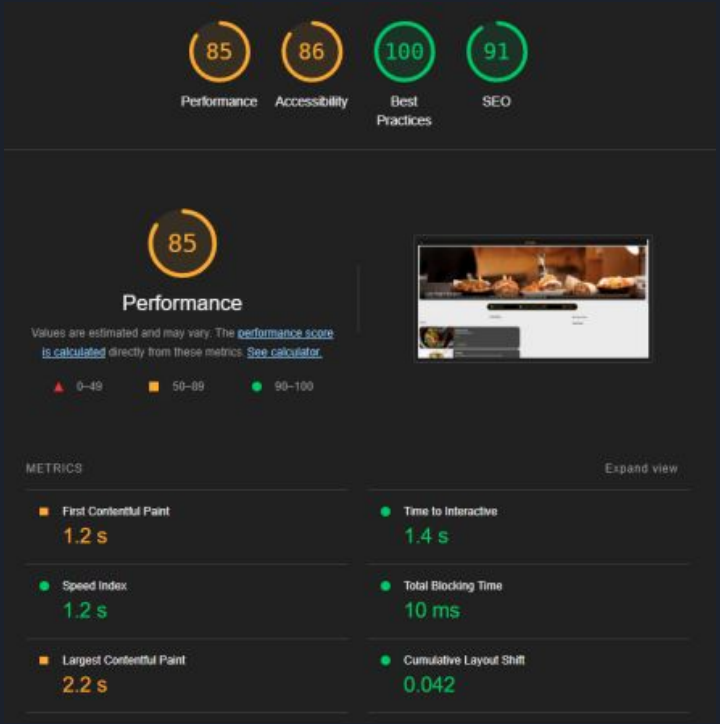
- Both were deployed on Vercel and used ci/cd on the GitHub repo to automatically build and deploy the application
- Some minor configuration was needed to the service to make it work as a standalone express app. This was due to the fact that Vercel suggest to use the Next.js API system to create serverless functions.

Links

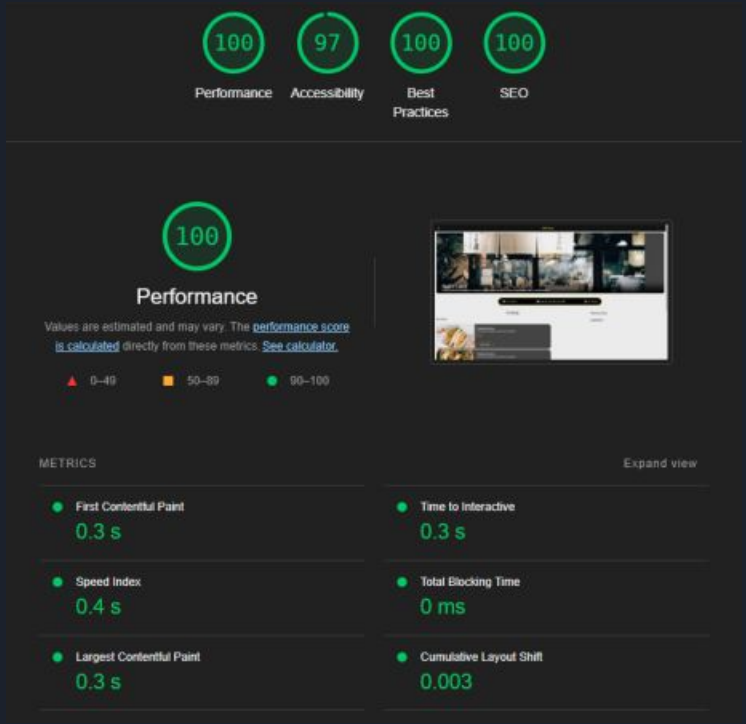
https://foodiee.vercel.app/

https://foodiee-server.vercel.app/

Metrics (Desktop)

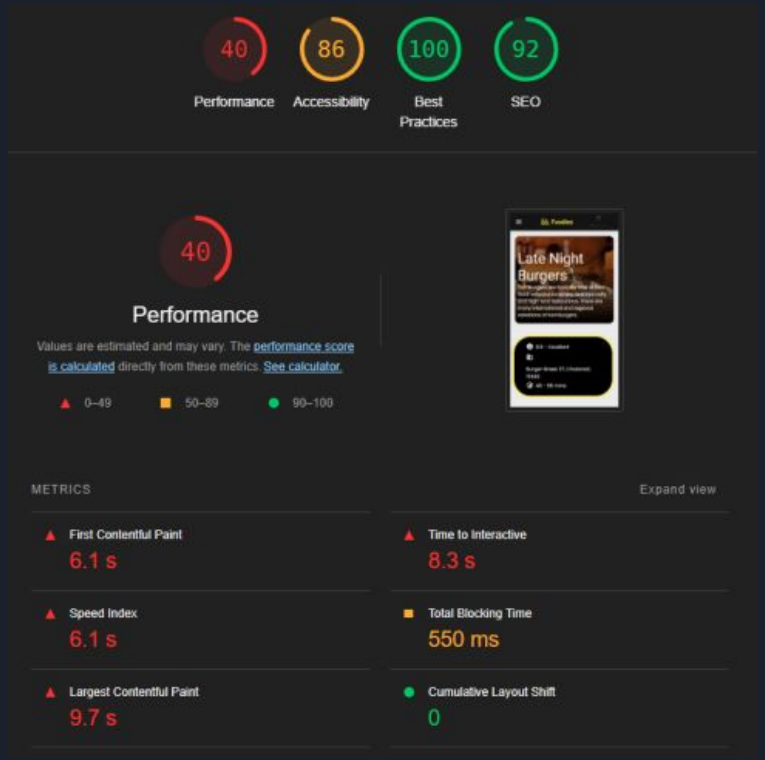


Before (CSR)

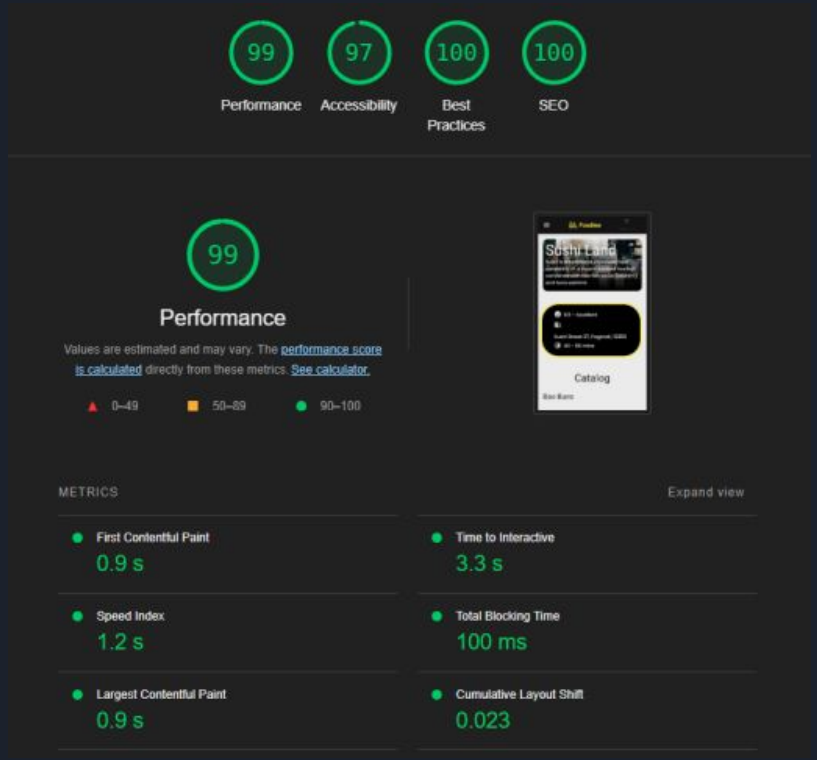


After (ISR)

Metrics (Mobile)

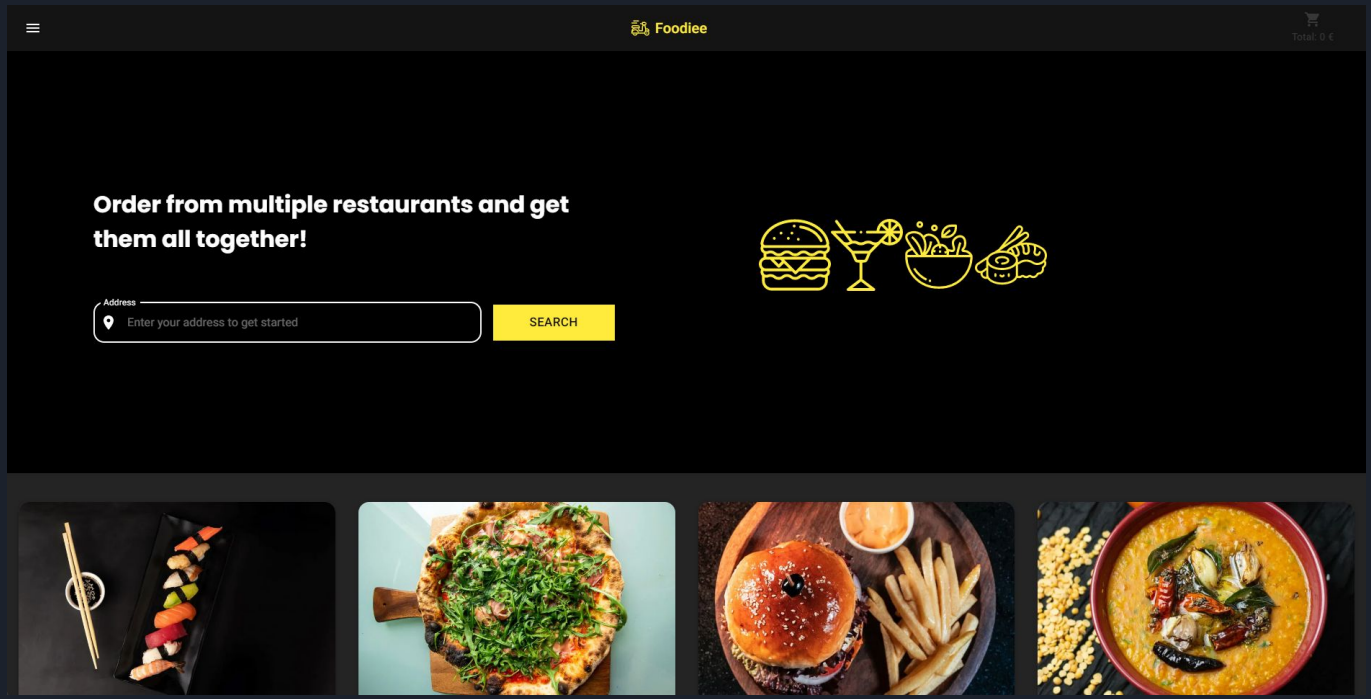


Before (CSR)



After (ISR)

Quick demo



https://foodiee.vercel.app/

What’s next

- Next.js 13
- Top level ‘use’ (await hook) for React components
- React components being SSR first
- TypeScript ‘by design’
- GraphQL on the rise

Thank you!

https://www.linkedin.com/in/hrsargyropoulos

https://www.github.com/z-argyropoulos