What was the structure of the project?

The structure of the project was mainly like any other React project, but with a few differences. We will go through the main parts of the project structure now.

Root

There are many files and folders that are not very important to the structure of the project, but we will go through the most important ones.

The root consisted of many usual files and folders like package.json (which had packages of the project), tsconfig.json (which had the Typescript configuration), .eslintrc.json (which had the ESLint configuration), .prettierrc.json (which had the Prettier configuration), .gitignore (which had the files and folders that should be ignored by Git), cypress (which had the E2E tests), docs (which had the documentation), public (which had the public files), src (which had the source files).

Documentation Folder

The documentation was put in the docs folder. Under the docs folder we also had a misc folder for images used by the docs.

tailwind.config.ts

This was the file where we put the Tailwind classes that we wanted to use in the project. This was where we could add new classes or change the existing ones.

Source/Src Folder

Pages

The project was mainly structured by the Next.js framework, more specifically the app router. The pages would therefore be put under the src/app folder. This meant that all the pages had their own layout file and a page file. If we wanted nested pages we would just create a folder with another page file inside it. If we wanted it to be dynamic we could put brackets around the page name, for example like [Id].tsx for the splat page.

API Folder

As we were using Next.js as a fullstack framework, we created api's by putting them in the src/pages/api folder. This meant that we could create a new api by just creating a new file in the api folder.

Component Folder

The components were all put under a components folder in the src folder. These components were then also put in their own respected folders, with other components that they relied on. An example of this was the widget folder, which had the Card, Checklist, Icon and Text widget components. It also had their respected popups when clicking on a placeholder in the widget. We will not be naming all the components in the project as there are many, but just take a look in the components folder to see them all.

Store Folder

The store was put in the src/store folder. This was where we stored the global state of the application. We used Zustand for this, which is a small and fast global state management library. We had a store for the Splat page and a store ContentAdder. The Splat page store was used to manage the grid, content, title, subtitle and more. The ContentAdder store was used to handle which specific widget was being chosen.

Types Folder

As we were using Typescript as our programming language, we created types to safely create functions and propogate data in the project. These can be found in the src/types folder.

Context Folder

The context was put in the src/context folder. This was where we stored the global state of the AI Form.

Styling Folder

The global styling was under the src/styles folder.

Utility Folder

The utility functions were put in the src/utils folder. These were functions that were used in multiple places in the project. An example of this was the grid utilities used by the Splat page or the Supabase utilities.

Testing Folder

We had two testing folders, one with the unit tests under the src/tests folder, and the E2E tests with Cypress under the cypress folder.

Data Folder

The data was put in the src/data folder. This was where we stored the data that was used in development, for example mock data for the Splat page. It is also used for the questions in the AI Form.