Deploying Fullstack Web App to AWS & integrating Dynatrace 💻

We will not be doing a local deployment of this App (on your machine). Instead we'll go straight to Cloud (aws) deployment. If you'd like to spin it up locally first, please make sure that you have a local instance of Postgresql server running. You can use DBeaver/PgAdmin to interact with the underlying database.

1. Web App overview 🔍

- A fullstack **Inventory Management Dashboard**
  - built with Next.js
  - styled with Tailwind css
  - featuring data visualization for inventory, products, expenses, and users personal setting.
  - Utilizes Node.js + PostgreSQL for the backend
  - deployed on AWS: EC2 (backend hosting) + RDS (psql) + S3 (graphical assets) + Amplify (frontend CICD)

2. **Tech Stack** 🛠️

Frontend:

- **Next.js** (with TypeScript)
- **Tailwind CSS** – for styling.
- **Redux Toolkit** – for state management.
- **Redux Toolkit Query** – for handling API calls and data fetching.
- **Material UI Data Grid** – for creating dynamic grid tables.
- **Recharts** – for rendering interactive charts and graphs.

Backend:

- **Node.js** with **Express** – for building the REST API.
- **Prisma** – for database modeling and communication with **PostgreSQL**.

AWS Services:

- **AWS EC2** – for hosting the backend.
- **AWS RDS** – for managing the PostgreSQL database.
- **AWS API Gateway** – for routing and managing API requests.
- **AWS Amplify** – for deploying and hosting the frontend.
- **AWS S3** – for file storage (e.g., product images, assets).

3. **Features** 🌟

Frontend (Client-Side):

1. **Data Dashboard**:

- Visualizes data using **Recharts** (bar charts, line graphs, pie charts).
- Provides a comprehensive view of the inventory, expenses, and user data.

1. **Inventory Page**:

    - Displays a list of all inventory items.
    - Allows for filtering and searching through inventory.

2. **Product Page**:

    - Shows detailed information about products in a **grid layout** using **Material UI Data Grid**.
    - Allows users to **create new products** directly from the dashboard.

3. **User Page**:

    - Displays a list of users using **Material UI Grid**.
    - Includes search and filter capabilities.

4. **Expense Page**:

    - Provides a **dynamic pie chart** to display the expense summary.
    - Allows filtering by expense category for better analysis.

5. **Settings Page**:

    - Mock page for personal settings, including dark mode, profile updates, etc.
    - Dark mode and other states are controlled using **Redux**.

7.**Responsive Design**:

- middle screen size
- small screen size