Projects Overview

AWS Solution Architect Project:

Engaged in the utilization of EC2 for virtual computing, S3 for storage solutions, various databases, both for static and WordPress websites, handling HTTP/HTTPS protocols, Docker for containerization, and ALB/NLB for load balancing.

HTML-CSS Project:

At the onset of learning, a comprehensive HTML-CSS website template is developed, with a thorough exploration of HTML-CSS intricacies throughout the process.

[React-NodeJS] Projects:

Crafting a comprehensive full-stack application with React for the frontend and Node.js for the backend, incorporating diverse database solutions such as MongoDB, SQL, SQLite, integration of APIs, and more. These projects are amalgamated into a unified application, while also being deployed individually to public repository on GitHub.
this is there specification:

- [Note-taking Application]

> *A small-scale project that, in this case, does not rely on a server-side and database structure but rather on local storage (for the purpose of learning proper usage of local storage).*
> *The application allows users to create folders for notes, customizing them with chosen designs and colors. Within each folder, users can create as many notes as they desire.*
> *The application includes the following features:*
> *Folders Management: Users can create folders,customize the design of them and delete them.*
> *Notes Management: Users can create, edit, customize the design of notes, move them to different folders, and of course, delete them.*
> *The application is meticulously designed and reactive, ensuring that every action elicits a responsive feedback from the application.*

- [Airplane Dashboard Project - Air Force Project]

> *Client-Side:*
> *The web application comprises three pages:*
> *A. Input Page: For entering flight data.*
> *B. Numerical and Technical Display Page: For displaying numerical and technical data.*
> *C. Visual Representation Page: For visualizing data through various graphical representations.*
> *Server-Side:*
> *Upon data entry, the information is sent to the server, where it is stored in a dedicated data structure within a MongoDB Atlas database. Once the data is successfully saved without any errors, the server sends back the saved data to the client-side. The client-side then*

displays the data accordingly.
The client-side heavily utilizes React hooks such as useState, useEffect, useRef, useContext, etc.
Moreover, the client-side design utilizes the Tailwind CSS library and the Flowbite library.
It employs Axios to send data to the server.
The server-side is based on Node.js and utilizes the Express and Mongoose libraries.
Additionally, data security measures have been implemented using CORS (Cross-Origin Resource Sharing).

- [The All-In-One Application]

The comprehensive application encompasses all the aforementioned mini-applications along with additional features and functionalities:

A. Full Authentication System:
Complete registration and login system connected to an Express server and a SQLite database.
Incorporates all necessary security measures and encryption protocols for password storage and data security.

B. Shopping Cart Mini-Application:
Enables users to add products to their cart.
Provides detailed product information, including images fetched via API.
Options for marking items as "purchased," editing product names and quantities, and removing items from the cart.
All data is stored on the server side per user, ensuring continuity across sessions.

C. Maps Feature:
Integrates Leaflet maps functionality allowing users to search and display any location on the globe.
Utilizes unique animations and API for displaying real-time weather conditions in the area.

The application provides a seamless user experience with synchronized data storage and retrieval across sessions, ensuring personalized and secure usage.

IT-AI Enhancement Project - AppsFlayer Company Collaboration

The project aimed to enhance the IT system within AppsFlayer through a collaborative effort with the company. Within the company's framework, whenever an employee encounters an issue or requires assistance, they open a ticket in the Jira Service Management system. However, various issues arise within the company, primarily concerning the mismatch between the problem/request and its urgency or impact level. This often leaves IT personnel confused when receiving tickets.
Additionally, another issue arises when the employee filling out the ticket nears completion, as the system presents various forms from the company's knowledge base that might address the issue before submitting the ticket. However, the system's method of retrieving these forms is not intelligent; it simply searches for words or phrases in the ticket fields, sometimes offering solutions irrelevant to the problem.

> *Our project addressed these issues. Firstly, we implemented a secure system that sends ticket details to an AI system upon submission. This AI system evaluates the problem/request against the employee's urgency or impact level choices and appends a summary to the ticket as a comment. This provides IT personnel with a clear overview of the ticket details and its urgency and impact levels, enhancing response time and efficiency. Secondly, we aimed to improve the system's retrieval of proposed solutions from the knowledge base. Instead of solely relying on keywords, the system now listens to the entered information and searches through past tickets for similar issues or requests already resolved by the IT department. It then suggests these solutions alongside others from the knowledge base, potentially expanding the pool of suggested solutions and streamlining the IT department's workflow.*
> *(Note: Due to company information security policies, I am unable to provide project specifics.)*

Installation

If you intend to install any of the specified applications, you'll first need to download the respective folder and execute the following command sequence:

```
npm install
```

Navigate to the client directory and execute:

```
npm run dev
```

Navigate to the server directory and execute:

```
npm run dev
```

Or

```
npm start
```