**Micro-Services Assignment**



Session: 2021 – 2024

**Submitted by:**

Uswa Arif

2021-CS-77

**Submitted To:**

  Aatif Hussain

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

**Introduction:**

**Microservices**, or microservices architecture, is an approach to the design and implementation of enterprise applications in which a large application is built from services or apps. Each app supports a specific task and uses a well-defined communications interface, such as an application programming interface (API), to communicate with other apps and services.

**How do microservices work?**

1. In a microservices architecture, an application has only one frontend (client side).
2. In a microservices architecture, an application is divided into distinct apps and services as backend logic (Servers).
3. .Each app or service is created independently.
4. Each app or service runs a unique process and usually manages its own separate database.
5. A service can support user interfaces (UIs), handle user identification or authentication, and perform various other computing and processing tasks with same frontend but different backend.
6. The services can communicate with each other using APIs. Also, APIs can be used to communicate with components present in services itself or communicating with database.

**Microservices vs. monolithic architecture**

A monolithic architecture is a single program. It incorporates all the application or business logic required to perform work within a single stack located on a single server within the database.

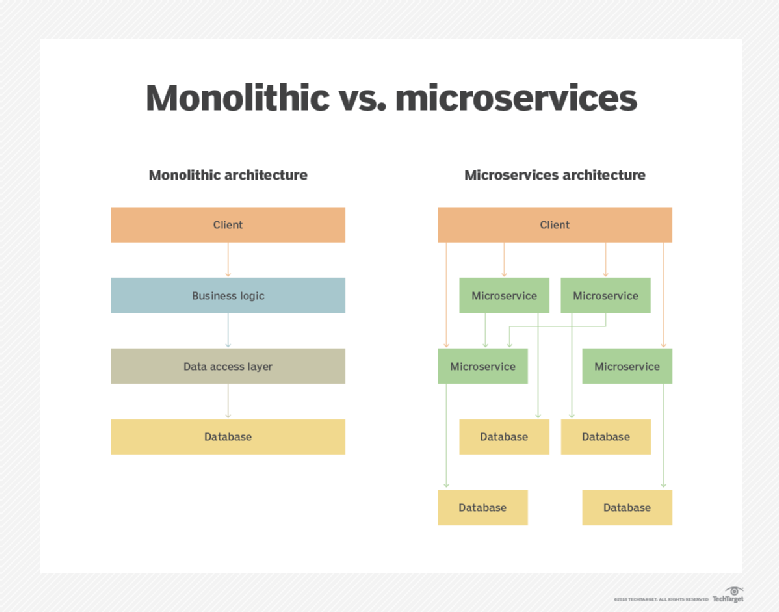
vs

A microservices architecture decomposes underlying logic into a series of different apps or services, each of which can be developed and deployed separately and communicate through an API.

**Challenges:**

Microservices also have challenges.Microservices has complex environment as there are multiple apps to handle. Each service depends on network performance and integrity, and every individual service must be attached to management, logging, monitoring and other tools.

Also, Microservices components rely on API-driven communication over a network.



**Implementation:**

**Technologies Used:**

* Frontend Development:

React js

* Backend Development:

Python, Node js

* Database:

Sql Server, Mongodb, Firebase

**Application Descriptions:**

* **App 1: User Registration and Login**

Backend: Node.js

Database Connection: MongoDB

API Integration: POST and PUT requests to store registration data in MongoDB and retrieve data for login.

* **App 2: Add Product**

Backend: Firebase Cloud Functions with Firestore

Database Connection: Firebase Cloud Firestore

API Integration: Storing emails from MongoDB into products through API and sending data to Firebase of Products.

* **App 3: Adding TaskList**

Backend: Python

Database: SQL Server

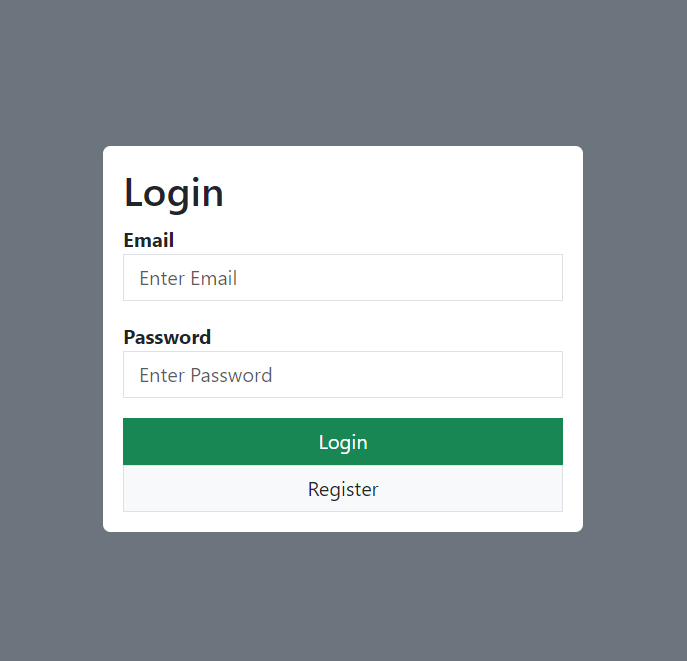
API Integration: Methods such as POST, PUT, GET, DELETE to send data to the database.

* **App 4: Weather Information App**

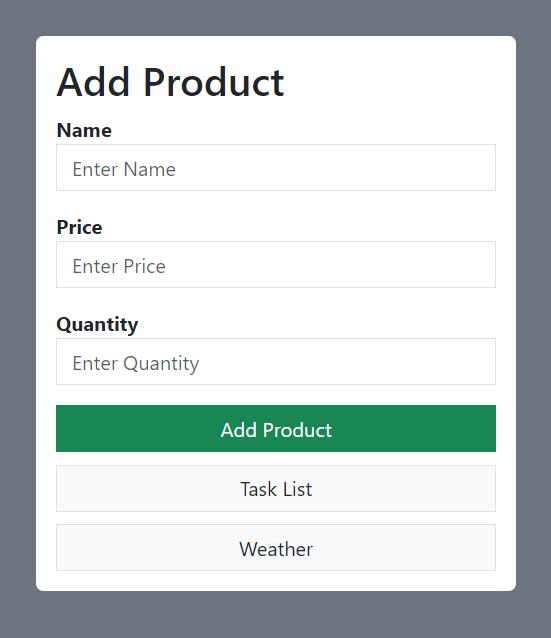
Integration with External API: Weather API

**User Interfaces of all Apps:**

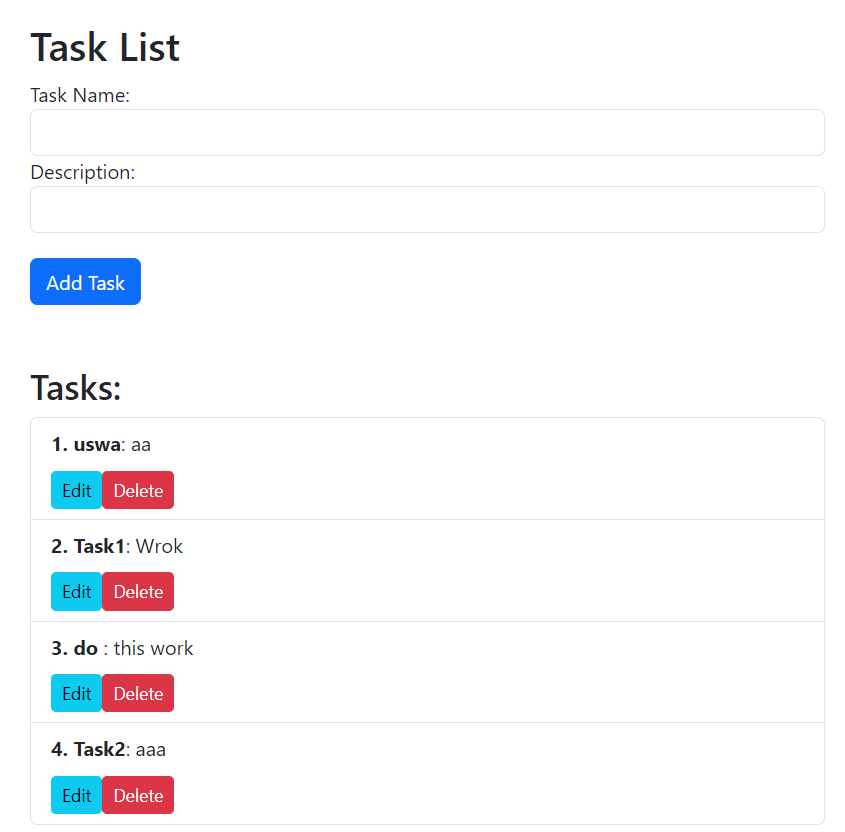
* **App 1: User Registration and Login**



* **App 2: Add Product**



* **App 3: Adding TaskList**



* **App 4: Weather Information App**

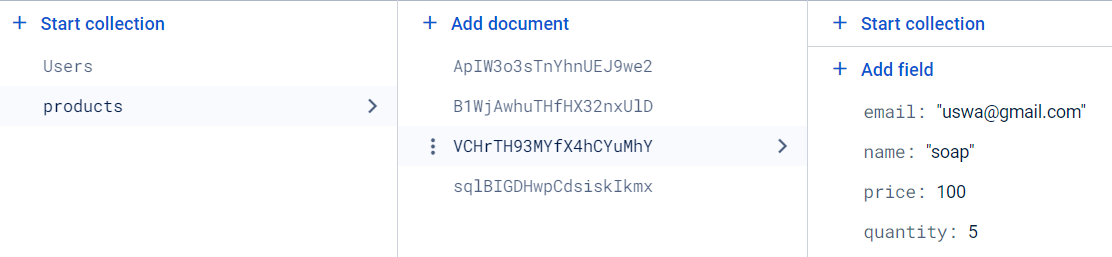


**Databases of all Apps:**

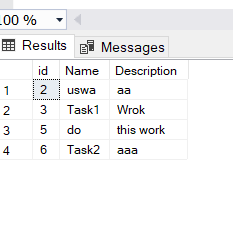
* **App 1: User Registration and Login**



* **App 2: Add Product**



* **App 3: Adding TaskList**

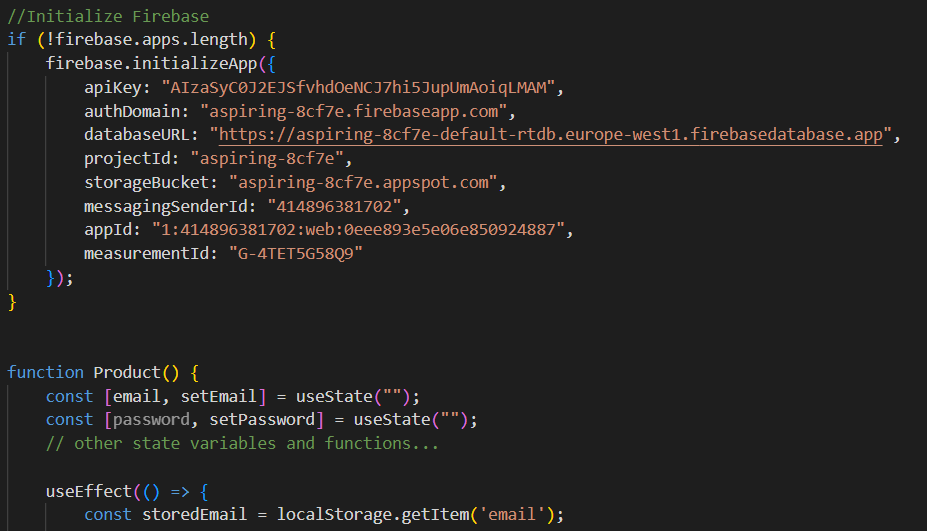


**Backend Code of all Apps:**

* **App 1: User Registration and Login**



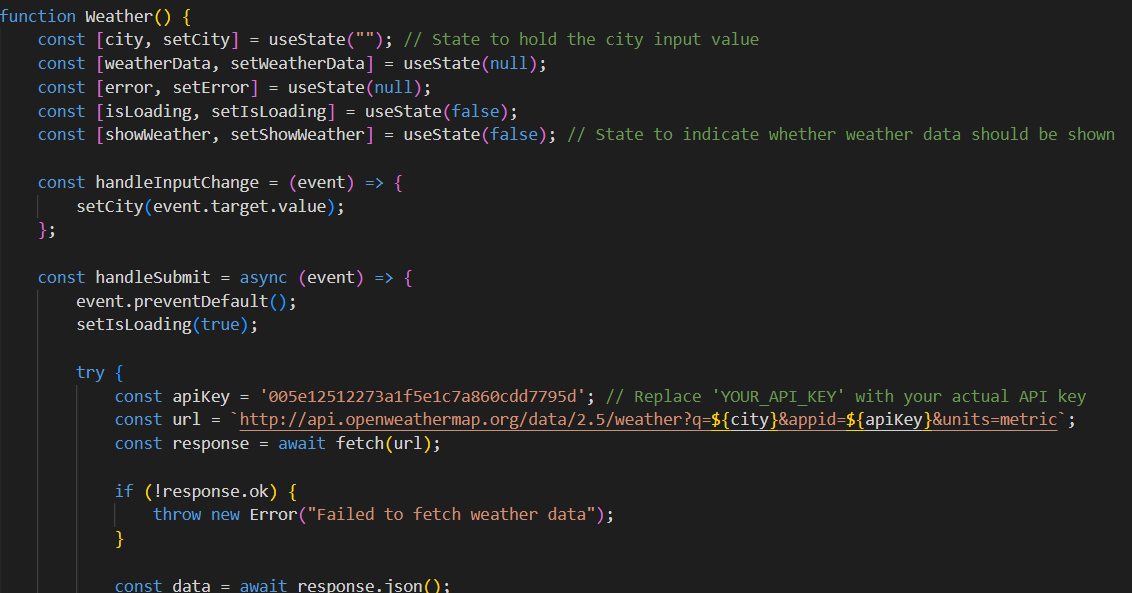
* **App 2: Add Product**



* **App 3: Adding TaskList**



* **App 4: Weather Information App**



**Conclusion:**

In conclusion, the development of the microservices architecture for this assignment has been both enlightening and rewarding. Through the creation of four distinct applications, we delved into the intricacies of modern software development practices. Utilizing React.js for frontend development provided us with the tools to craft intuitive and responsive user interfaces across all applications, ensuring a seamless user experience. Exploring different backend technologies such as Node.js with MongoDB for user registration and login, Firebase Cloud Functions with Firestore for product management, and Python with SQL Server for task list management showcased the versatility and adaptability of microservices. Despite encountering challenges in integrating various APIs and databases, such as coordinating data flow between different services and ensuring consistency across systems, the assignment offered valuable opportunities for problem-solving and implementing robust solutions.