Architecture

For Maria's web services solution, a microservices architecture is the most suitable approach. Microservices is a software development approach that’s breaks down the application small, independent, and loosely coupled services. Each service focus on a specific functionality, such as user authentication, crop data management, or analytics.

Here are some the reason for choosing Microservices Architecture –

- Scalability – Each microservice can be scaled independently based on its specific needs, ensuring optimal resource utilization.
- Flexibility – It allows different people or teams to work separate parts using several types of technology.
- Ease of Maintenance – It is smaller, focused services. So, it is easy to understand.
- Time to Deployment – Microservice allows us to update and deploy the independent services without affecting the entire system, improve release cycles.

Tools

Without tools it going to be hard to deploy the architecture. For this project I will choose these tools –

- Visual Studio Code – It is an IDE (Integrated Development Environment). Without IDE it will be hard to write and manage code. The reason I choose this IDE is because it is lightweight, user-friendly, customizability and built-in features.
- Postman – For testing API I will choose Postman. It supports multiple API types (REST, SOAP, GraphQL), automates tests, offers collaboration tools. The best part of this service is its free for everyone.
- Docker – It will be used for containerizing each microservice, ensuring consistency across development, testing, and production environments. It is better than VM (Virtual Machine) because it uses lightweight containers that share the host OS, making it faster, more resource-efficient, and portable compared to full virtual machines.
- Kubernetes – Kubernetes provides automated resource utilization across different systems, and it is better since it guarantees high availability, scalability, and resource utilization at minimal cost.
- GitHub and Git – I will use this service for version control and push the code to the online.

Technologies

To build an efficient services solution, selecting the right technology is the crucial thing. These technologies will guarantee that these services function robustly, be secure for users, and be integrated easily from service to service. Here are the technologies, I will use for this solution –

- Node.js – Node.js more efficient, especially for making many concurrent requests at once, as it is an asynchronous function.
- Express.js – A lightweight framework providing tools to build scalable and fast APIs for individual microservices.
- MongoDB – I need a database to store the data about crop; I will choose MongoDB. It is a NoSQL database service that suits for semi-structured nature of crop data. It is flexible as it aligns with JSON-based data.
- JWT – JWT (JSON web token) used for secure and stateless authentication across services, ensuring robust security.
- Redis – For caching data such as frequently accessed crop information or user sessions, improving response times and reducing database load.
- Swagger – For API documentation, I will choose this service. It is easier to integrate and easy to maintain.