1. Describe current architecture:

Using Microservices architecture could be the best option for designing a web application offering recipes from various cuisines. With a flexible architecture as Microservices, the main functionalities of this web application can work independently. First, a recipe management API could be integrated so it can handle the main objective of the web application which is providing recipes, cooking time and level of difficulty. Also, an API providing nutritional information and health benefits for each recipe could be integrated. Moreover, displaying the ratings of recipes and reviews from every registered user could be a functionality integrated. Last but not least, a shopping list service could be integrated as users might purchases a missing item for recipe if needed.

2. Compare 3 different architectures:

The first architecture that will be discussed is the Microservices architecture. This type of architecture divides an application into independent services allowing each service to do one objective precisely. While using the Microservices architecture, the structure of the code will be a decoupled services that communicate with each other using RESTful APIs[1]. Providing many benefits, the Microservices architecture offers scalability, flexibility, modularity, isolation and parallel development. To begin with, Microservices architecture provides scalability by creating an individual scale for each functionality based on the demand. Moving on, the use of various technologies and frameworks can be accessible and easily integrated by the Microservices architecture. Not to mention, each microservice can be developed independently which allows updates to be deployed easier which would not affect the entire system. If an issue occurred, it would not affect the rest of the services maintaining the availability of the system. Lastly, more than one team can work on different microservices simultaneously allowing a fast development process.

The second architecture that will be discussed is the Monolithic architecture. This type of architecture depends on developing the web application as single structure unlike the microservices. Each service available in a Monolithic designed web application is dependent on each other[1]. If a web application is designed using a Monolithic architecture, it will provide simplicity as the whole web application will be connected to one database. Also, debugging and testing would be easier as everything is in one place. However, integrating more than one framework would be impossible which limits the flexibility of developing the web application. Unlike the Microservices, the Monolithic architecture can not provide scalability nor maintainability as it would not be easy scale each service individually and to update one part of the web application individually.

The Final architecture that will be discussed is the Serverless architecture. By hosting the web application on a third-party service, the web application is developed in a particular way that it does not need to manage the infrastructure[2]. Developing a web application with a Serverless architecture will provide scalability by scaling the workloads on each service automatically. Also, it would be cost-efficient as companies would only pay for the actual execution time of functions. Without a doubt, any web application based on Serverless architecture would be flexible as more than one framework can be used. Although working without managing an infrastructure could be beneficial, it can also limit the use of web application. As the functions are designed to stateless, it cannot retain information. Also, third parties hosting the web application can differ from execution time and memory.

Resources:

[1]M. Aftab, “What Are the Main Types of Web Application Architecture?,” Nov. 03, 2022. https://enou.co/blog/web-application-architecture/ (accessed Feb. 23, 2024).

[2]“Serverless Architectures,” *Amazon Web Services, Inc.* https://aws.amazon.com/lambda/serverless-architectures-learn-more/#:~:text=What%20is%20a%20serverless%20architecture