1. **Describe the current architecture that you planned.**

The architecture that is going to be used to implement the website that is going to share and discover recipes from different cuisines is the Microservices Architecture. Microservices architecture allows the separation of the application into smaller independent parts called services, with each part having its sperate codebase and should implement a single business requirement. In microservice architecture, services communicate with each other by using well-defined APIs and the internal implementation details of each service are hidden from other services. It supports a kind of programming style called polyglot programming where services do not need to share the same technology stack, libraries, or frameworks. This makes websites easier to scale and faster to develop based on the requirements. Also, their isolation and resilience enable the website to continue running independently, even if one component fails, providing a robust and fault-tolerant system.

1. **Compare the different architectures.**

There are other software architectures that could have been used but they have some disadvantages. For example, the monolithic architecture, which is an all-in-one architecture, wherein all aspects of the software operate as a single unit. It operates on a unified codebase, where the client-side user interface, server-side application, and data interface collaboratively interact with a single database. Its main drawbacks are the lack of scalability and flexibility. Changes and integrations need to be done to the entire application, affecting agility and innovation. Another architecture was considered which is the Service-Oriented Architecture (SOA). It has four basic principles that are common across all SOA implementations. These principles are interoperability, loose coupling, abstraction, and granularity. It involves designing the application in the form of services that behave as separate, autonomous, loosely coupled network-accessible units that communicate with each other. It was not considered as it can be heavy on performance by having an increased complexity in service communication.