The Knowledge Hub Universities

School of computing

KH6003CEM-Web API Development Report

Ibrahim Mohamed Elsayed

ID:2000803

Lecturer: Dr Nada GabAllah

Table of Contents

Recipe Share

A Platform for Sharing and Discovering Culinary Delights

Project Overview:

This project, named "Recipe Share," is a web-based platform designed to facilitate the sharing and discovery of recipes from various cuisines before initiating this project we had gone through the brief information regarding this recipe share this is very useful web application designed to be in an user friendly environment even uneducated people able to use this web application based on the appearance of this application where the image representation dynamically makes users to understand and use this very user friendly.

Features:

- User-friendly interface for easy navigation and recipe discovery.
- Personalized recipe recommendations based on user-defined criteria.
- Detailed recipe information including cooking time, difficulty level, and nutritional content.
- Community engagement through user ratings and reviews.
- Shopping list generation with price estimates for ingredients.
- Support for exploring recipes by cuisine or dish, catering to diverse culinary preferences and interests.

Frontend Technologies:

- **HTML, CSS, JavaScript:**
  - HTML, CSS, JavaScript: These are the building blocks of web development. HTML provides the structure, CSS handles styling, and JavaScript adds interactivity to the frontend interface of the application.
- **React.js, Angular**
  - These are popular frontend frameworks/libraries that enable building interactive user interfaces. React.js allows for the creation of reusable UI components, making development easier.

Backend Technologies:

- **Node.js, Django, Flask**
  - These are backend frameworks used to build server-side logic and APIs Node.js is particularly popular for its non-blocking, event-driven architecture, while Django and Flask are preferred for their simplicity and robustness in Python-based development.

- **Express.js, Fast API**

    - These are lightweight web frameworks that work well with Node.js and Python, respectively, for building RESTful APIs. They simplify routing, request handling, and response generation.

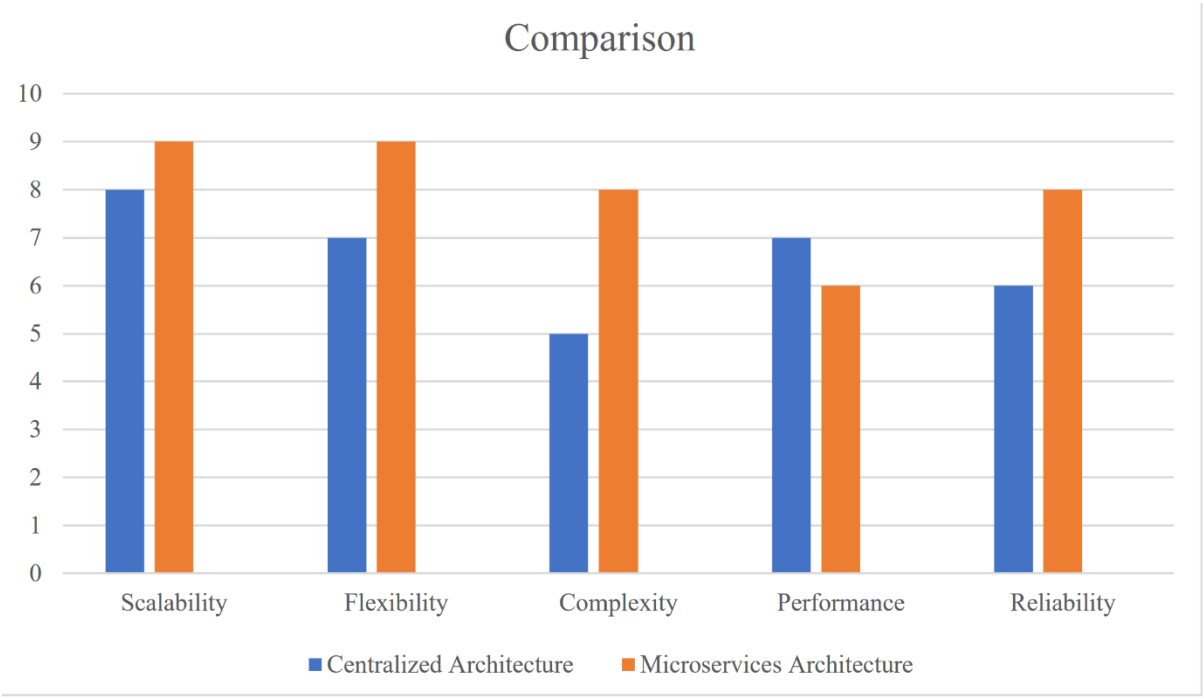
These are the most commonly used technologies in a web application development these provide very user friendly experience with the use of simple UI elements also in designing we have flexible technology called bootstrap which helps very essentially in designing a website as we focus to provide a best user experience for the users or clients who visit this website, we must aim to focus on frontend development because before showing the functionality they must love our integrity.

Architecture:

The architecture that I have planned for the recipe sharing and discovery tool is a client-server model, considering both frontend and backend components to deliver a seamless user experience.
Where front end is most appropriate for clients who visit this web application should built with more care, We have designed a centralised system with several interconnected components as the recipe sharing and discovery tool's architecture. The frontend interface is the central component of this design, offering customers a smooth and intuitive way to browse recipes, enter preferences, and interact with the system. Our backend is made up of a number of services that handle different functions. Among them is a backend logic service that oversees the basic logic required to retrieve, sort, and handle recipe data. A large number of recipes, each with comprehensive details on everything from ingredients and cooking techniques to nutritional values, are housed in a centralised location called the recipe database. Our system is enhanced by the integration of external APIs, which adds features like user reviews and nutritional analysis. Moreover, authorization and authentication are available procedures guaranteeing safe system access. Based on user preferences and previous interactions, a recommendation engine driven by machine learning algorithms offers customised recipe ideas. While analytics tools track user behaviour and system performance to continuously enhance efficiency, caching technologies are used to optimise speed by storing frequently requested data.

Architecture Comparison

We compare the unique advantages and trade-offs between our centralized system and a micro services approach. With all components installed on a single server or cloud platform, the centralized method provides simplicity and ease of management. Through APIs, interactions between components are precisely defined, guaranteeing smooth data transmission and communication. Micro services architecture, on the other hand, excels in scalability, flexibility, and agility. Micro services allow for the autonomous scalability and deployment of individual. components by decomposing functions into discrete services that each operate independently and communicate using lightweight protocols like HTTP or message queues. Better resource use, quicker development cycles, and simpler need adaption are the outcomes of this. Micro services, however, increase the complexity of managing and transferring services, necessitating the use of powerful orchestration and monitoring tools.



The theoretical implementation of the centralized architecture involves deploying the various components on a single server or cloud platform. The frontend interface interacts with backend services through well-defined APIs, while the recipe database serves as the centralized repository for recipe data. External APIs are integrated to enrich the system with additional functionalities, and authentication and authorization mechanisms ensure secure access. A recommendation engine powered by machine learning algorithms provides personalized recipe suggestions, while caching mechanisms optimize performance. Analytics tools monitor system performance and user interactions for continuous improvement.

In contrast, the micro services architecture requires breaking down the functionalities into separate services, each running independently. The user service handles authentication, authorization, and profile management, while the recipe service manages storage, retrieval, and manipulation of recipe data. A search service provides functionality for searching and filtering recipes, and a recommendation service implements the recommendation engine. External API services integrate with external APIs for additional functionalities, and analytics service, monitors system performance and user interactions. Each service communicates via lightweight protocols like HTTP or message queues, enabling independent scaling and deployment.

Conclusion

In conclusion, there are benefits and trade-offs for each architecture. Scalability, adaptability, and agility are the main advantages of the micro services architecture over the centralized architecture. The micro services design appears more appropriate in light of the varied requirements and room for expansion of our recipe sharing and discovery platform. On the other hand, the choice ought to be taken only after giving considerable thought to aspects including the scope of the project, team composition, and particular requirements.

References:

o brief react Js

o https://client-server-model

o https://centralized-architecture

o Fowler, M. (2014). Micro services: a definition of this new architectural term. Retrieved from

https://martinfowler.com/articles/microservices.html