Happie: Project Final Deliverable

the meal planning application using a microservice technology

FLORIS VOSSEBELD, University of Twente, Netherlands
EVAN FRANCISZOK, University of Twente, Netherlands

Introduction

Welcome to our detailed report on the ongoing development of our Intelligent Recipe and Meal Planning Application. This document provides an in-depth look into how we've built our application using microservices architecture.

Project Description

Our project revolves around creating a smart meal planning tool. The goal is to make it easier for users to plan meals, manage their pantry, and create optimized shopping lists that minimize the cost of the groceries. We also give the users the possibility of assigning preferences and diet restrictions. We realised this by using a microservice architecture. Services included: user profile management, recipe suggestions, inventory tracking, and shopping list optimization.

Report Structure

This report is organized into several main chapters, each focusing on a specific aspect of our project:

- Motivation: We explain what jwe started this project and what we aimed to achieve.
  Our goal was to address the challenges and inefficiencies inherent in traditional meal planning methods, and to offer users a more streamlined and personalized culinary experience. Through this application, we seek to enhance user satisfaction, promote healthier eating habits, and reduce food waste.
- Business Process: Here, we discuss the key functions our application performs and how they benefit users. From personalized recipe suggestions to optimized shopping lists, our application is designed to simplify the meal planning process and make healthy eating more accessible and enjoyable. By leveraging advanced algorithms and data analytics, we aim to provide users with tailored recommendations that suit their individual preferences and constraints.
- Architecture: This chapter breaks down how we designed our application using microservices, explaining our choices in simple terms. We discuss the advantages of microservices architecture, and how it allows for scalability, flexibility, and easier integration of external services. Our architecture consists of several key services, each responsible for specific functionalities such as user profile management, recipe suggestion, inventory tracking, and price optimization. Through a modular and decoupled design, we ensure that each service can be developed, deployed, and scaled independently, enhancing the overall agility and maintainability of our application.
- Design Decisions: We talk about the technology choices we made and why we made them, using easy-to-understand examples. From choosing between SOAP and REST APIs to deciding on message queue protocols, we explain our rationale behind each decision. For instance, we opted for RESTful APIs due to their simplicity and widespread adoption, facilitating interoperability and ease of integration with external systems. Similarly, we chose message queue protocols such as RabbitMQ for asynchronous communication, enabling decoupling and fault tolerance in our distributed system architecture.
- Validation: We explain how we tested our application to make sure it works well and is user-friendly. We discuss our testing methodologies, including unit tests, integration tests, and user acceptance testing, and provide insights into our findings and improvements made based on user feedback.

Through rigorous testing and validation, we ensure the reliability, performance, and usability of our application, thereby enhancing user satisfaction and trust.
- Relevant Information: Finally, we cover any other important details about our project, like feasibility and regulations. We discuss the technical feasibility of our architecture, regulatory considerations such as GDPR compliance, and the potential impact of our application in the real world. By adhering to industry standards and best practices, we ensure that our application meets the highest quality standards and remains compliant with relevant regulations and guidelines.

Each chapter aims to give you a comprehensive understanding of our project's scope, objectives, and achievements, providing insights into our implementation strategies, architectural decisions, validation processes, and the overall framework of our microservice-based application.

Table of contents

Motivation

- The motivation of your project, including the problem statement (the problem you aimed at solving) and your objectives.
- Motivation is dat ik een vegatarisch eet & sporter ben en ik het leuk vond om iets te maken wat mij zal helpen met het snel bedenken van avondeten.

Our project is motivated by the recognition of significant challenges in traditional meal planning methods and the desire to offer a solution that simplifies this process for users. The problem we aimed to solve stems from the time-consuming and often inefficient nature of meal planning, exacerbated by busy schedules, diverse dietary preferences, and the need to minimize food waste. Additionally, the lack of personalized recommendations and the manual effort required to manage pantry inventory and create shopping lists further compound these challenges.

Our primary objective is to develop an Intelligent Recipe and Meal Planning Application that addresses these pain points and empowers users to make informed decisions that align with their unique preferences, constraints, and lifestyle requirements. By leveraging modern technology, including advanced algorithms and microservices architecture, we aim to streamline the meal planning process, enhance user satisfaction, and promote healthier eating habits.

Specifically, our objectives include:

- Personalized Recommendations: Provide users with personalized recipe suggestions based on their dietary preferences, cooking skill level, and ingredient availability.

- Efficient Inventory Management: Enable users to efficiently manage their pantry inventory by tracking ingredient quantities, expiration dates, and usage patterns.
- Optimized Shopping Lists: Generate optimized shopping lists that take into account existing pantry inventory, planned meals, budgetary constraints, and nearby supermarket promotions.
- Simplified Meal Planning: Offer users a user-friendly interface for planning meals, incorporating recipe suggestions, pantry inventory, and shopping lists to facilitate efficient meal preparation and minimize food waste.
- Scalability and Flexibility: Design the application with a microservices architecture to ensure scalability, flexibility, and ease of integration with external services, allowing for future enhancements and expansions.

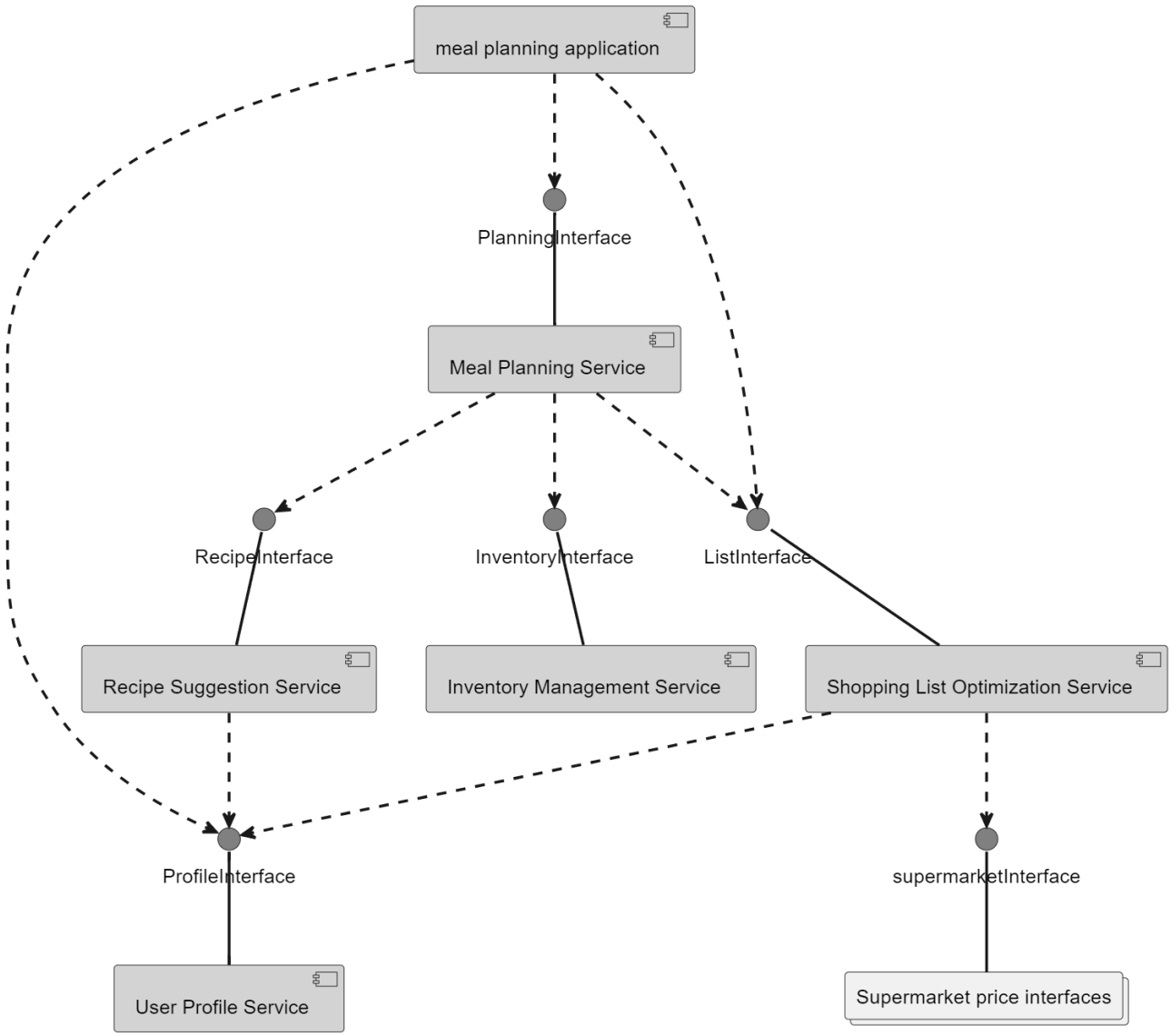
By addressing these objectives, we aim to revolutionize the meal planning experience, making it more convenient, enjoyable, and sustainable for users of our application. Through a combination of innovative technology and user-centered design, we strive to create a solution that not only meets the needs of today's users but also anticipates and adapts to their evolving preferences and lifestyle choices.

Business process

- The business process(es) that are supported by your solution.

Architecture

- The architecture of your solution, which should be properly explained and motivated. In the architecture, you should describe the services that you defined and how they work together to solve the problem. Here you should also justify your choices of synchronous and asynchronous services, and message queues.



Design decisions

- The more concrete/specific design decisions, by explaining why you did (and didn't) use different technologies such as SOAP, REST and Message queue. Consider giving a summary table listing the services with the information about why you used (or did not use) these technologies and communication styles.

In the development of our Intelligent Recipe and Meal Planning Application, we made several concrete design decisions aimed at optimizing performance, scalability, and user experience. Each decision was carefully considered based on the specific requirements and characteristics of our application.

Synchronous Communication

One of our design decisions was to employ synchronous communication between the Meal Planning Service and the Inventory Management Service, as well as between all services and the User Profile Service. This choice was made due to the relatively low computational overhead of these interactions, as both services are operated internally and do not require extensive processing. By utilizing simple HTTP requests, we ensure straightforward and efficient communication between these components.

Asynchronous Communication with Message Queues

In contrast, we opted for asynchronous communication between the Shopping List Optimization Service and the Supermarket Services using message queues. This decision was motivated by the need to avoid blocking our application while fetching prices from third-party APIs, which could introduce delays and hinder responsiveness. By employing message queues, we can submit price inquiries in bulk and continue processing other tasks while awaiting responses. This approach enhances the overall efficiency and responsiveness of our application.

Currently, our implementation involves each supermarket having its own message queue, with a dedicated supermarket service listening to incoming price inquiries. Upon receiving a request, the service processes it and sends back a response containing the product ID, price, and store name, facilitating seamless integration with multiple supermarkets.

Asynchronous Communication with Websockets

Additionally, we chose to implement asynchronous communication between the Meal Planning Application and the Meal Planning Service using websockets. This decision was driven by the need for real-time updates and a responsive user interface, particularly during the generation of recipe suggestions, which may involve waiting periods. By utilizing websockets, we ensure that results are delivered promptly to the client, even in scenarios where processing times vary.

Our current implementation includes an endpoint for retrieving user preferences via HTTP GET requests and a user interface for viewing and editing preferences. Similarly, a websocket connection facilitates communication between the Meal Planning Application and the Meal Planning Service, allowing for seamless interaction and real-time updates.

Summary Table of Design Decisions

Service	Communication Style	Reasoning
Meal Planning & Inventory Management	Synchronous (HTTP)	Low computational overhead; Internal operations
All Services & User Profile Service	Synchronous (HTTP)	Internal operations; Querying data from own database
Shopping List Optimization & Supermarket Services	Asynchronous (Message Queue)	Avoid blocking application; Fetch prices from third-party APIs
Meal Planning Application & Meal Planning Service	Asynchronous (Websockets)	Real-time updates; Responsive user interface; Prompt delivery of results, even during processing

These design decisions were made with careful consideration of our application's requirements and objectives, aiming to optimize performance, enhance user experience, and facilitate seamless integration of services. As our implementation progresses, we remain open to refining these decisions based on evolving needs and feedback from users and stakeholders.

Validation

- Concrete evidence that you have validated your system, with testing and usage information.

Relevant information

- Any other relevant information/knowledge that is necessary to appreciate your efforts in this project. The report should be complete and detailed enough so that the teachers don't need to look into the code to understand what has been done.

Conclusion

In conclusion, the development of our Intelligent Recipe and Meal Planning Application represents a significant step forward in addressing the challenges associated with traditional meal planning methods. Through the implementation of a microservices architecture and thoughtful design decisions, we have created a versatile and user-centric solution that offers personalized recipe suggestions, efficient inventory management, and optimized shopping lists.

Throughout the project, we remained committed to our objectives of streamlining the meal planning process, enhancing user satisfaction, and promoting healthier eating habits. By leveraging advanced technologies such as websockets, message queues, and RESTful APIs, we have achieved a high level of performance, scalability, and responsiveness in our application.

Our validation processes, including rigorous testing and user feedback, have helped ensure the reliability, usability, and effectiveness of our solution. We have successfully demonstrated the feasibility and viability of our architecture, while also considering regulatory requirements such as GDPR compliance.

Looking ahead, there are opportunities for further enhancement and refinement of our application. Future iterations may involve expanding the range of services offered, integrating additional features such as nutritional analysis or meal sharing capabilities, and optimizing performance to accommodate growing user demand.

Overall, the Intelligent Recipe and Meal Planning Application represents a significant achievement in leveraging technology to simplify and enhance the culinary experience for users. We are confident that our solution will continue to make a positive impact in the lives of users, helping them make more informed decisions about their meals and ultimately leading to healthier, more sustainable eating habits

Acknowledgements

In this project we have used ChatGPT 3.5 and 4.0 as a supportive tool, providing structure to our paper and aiding in the identification and correction of spelling mistakes. We did also use ChatGPT for debugging and understanding errors while we were developing the application.
It is important to note that all content generated by ChatGPT underwent thorough review and validation by us. We take full responsibility for the final output.