2023

Technology Justification

STOCKBROOD

Techstack – Stockbrood

Inhoud

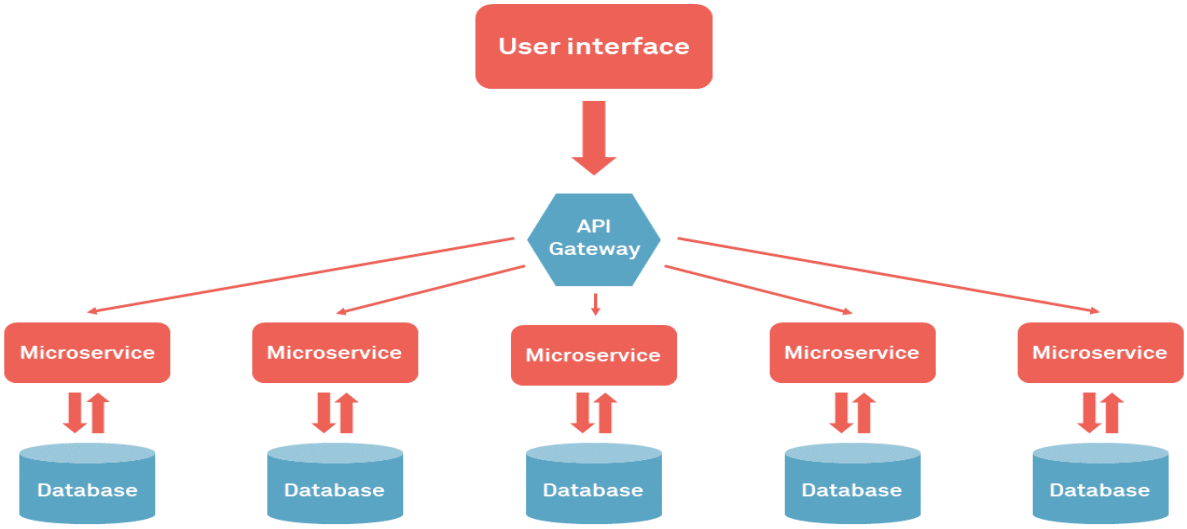
Techstack – Stockbrood

Microservices architecture

We chose to use a microservice architecture for our project over a monolithic architecture because it has several advantages. For one, microservices are flexible and can be easily scaled up or down based on the specific needs of each service. This means we can handle sudden surges in traffic or resource demands more efficiently.

Another big benefit of microservices is their reliability. If one service fails, the rest of the system can continue to function normally whilst we restore the failed microservice.

Lastly, microservices promote modularity and flexibility, making it easier to update and maintain individual services without disrupting the whole system. This is particularly useful due to the fact that our project is expected to be handed over to different development teams.



Angular

Angular is a framework that allows us to build responsive and reactive web applications. Angular is particularly useful because it also allows us to easily scale our web application. Another benefit is that Angular has a large following of developers and therefore has a lot of available libraries that we can use to quickly implement functionalities. Finally Angular has a large amount of online resources that can be used for troubleshooting.

Considering we're students, the choice for Angular was also made because we wanted to learn a new framework. Our group had previous experience with React and wanted to learn something new.

Techstack – Stockbrood

Go

Go is high-level language developed by Google that first appeared in 2009. Go is known for being easy to learn, high performance and for supporting concurrency. Go has become quite popular over the years and has become well integrated with popular tools used in software development such as Docker and Kubernetes.
As we are developing a program that we believe will require us to develop a scaling architecture that is high-performance and can run concurrently, we think that Go is a good choice for us.
Finally, just like our argument for using Angular: we want to gain experience with new technologies. That is why despite having experience with Java, we decided on Go.

gRPC & Protobuf

We decided to use gRPC & protobuf for our project because it allows for fast messaging between services. This is because protobuf is a much more efficient format than JSON, the format used by standard REST services. Because of this our backend systems can spend less time encoding and decoding messages, which is a large portion of the cpu load in a backend system. We expect a lot of traffic for our backend systems so speed is very important. Whilst REST is much simpler to implement as nearly everyone is familiar with it, gRPC & protobuf are quickly becoming more common. We again, also wanted to learn something new.

Docker

We're using Docker for our project because it offers a lot of benefits. Docker allows us to package our application and its dependencies into a single container that can be easily deployed across different environments. This means that we don't have to worry about compatibility issues and can distribute our program with more ease. Another benefit of Docker is that it makes it easier to manage and scale our application. With Docker, we can easily spin up new instances of our application as needed, without having to worry about configuring each one individually.

Finally, Docker is easy to use with Kubernetes.

Kubernetes

Kubernetes is used in our project because it provides a lot of benefits for managing containerized applications. Kubernetes allows us to quickly and easily deploy and manage containers across multiple machines, making it easier to scale our application as needed. Kubernetes provides features such as load balancing and automatic failover, this makes it so that we ensure a reliable application.

RabbitMQ

RabbitMQ’s purpose in our project is to provide service to service communication. The main benefit of RabbitMQ is that it is a reliable and scalable messaging technology. Even when a service goes down RabbitMQ can ensure that this service receives its messages. Finally RabbitMQ provides load balancing. All of this makes RabbitMQ a good choice to handle communication between our services.