SOA Project

Moldovan Vasilica-Andreea

Group 258-2, SE

This application was designed to be used either by animals' owners or by veterinarians (or other employees of the veterinary clinics) in order to help them having a better experience while adding appointments for their pets. The application can also be used as to do list, being especially useful for veterinary clinics' employees.

The user can login and see a list of appointments, add a new appointment, update or delete an existing appointment, and send the list of appointments to a given email.

The application is composed of several microservices. The backend includes a microservice for handling appointments requests, a microservice for managing authentication, and a microservice that serves as a gateway and delivers client calls to the appropriate microservice. There are two microservices on the frontend, one for the appointments section and one for the login section.

EmailJS is a third-party email sending service that is used by application. After entering a name and an email address, the user can press the 'Send' button to send their list of appointments to a given email address.

The application uses Docker for creating the images and deploying the containers.

Patterns used:

1. Decompose by business capability Pattern

A notion from business architecture modeling is business capability. It is something that a company does to produce value. Every microservice supports a unique business capability. The user data and the appointments are the business objects in our scenario.

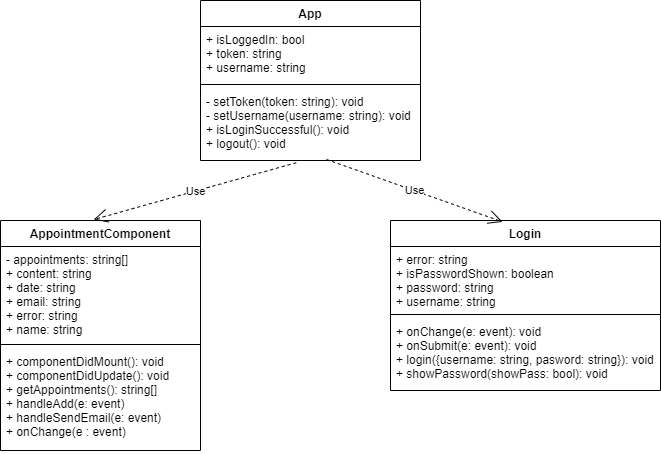
2. Sidecar Pattern

To achieve isolation and encapsulation, we have to deploy an application's component parts into a different process or container. This design pattern can also make it possible for apps to be built using a variety of different components and technologies.

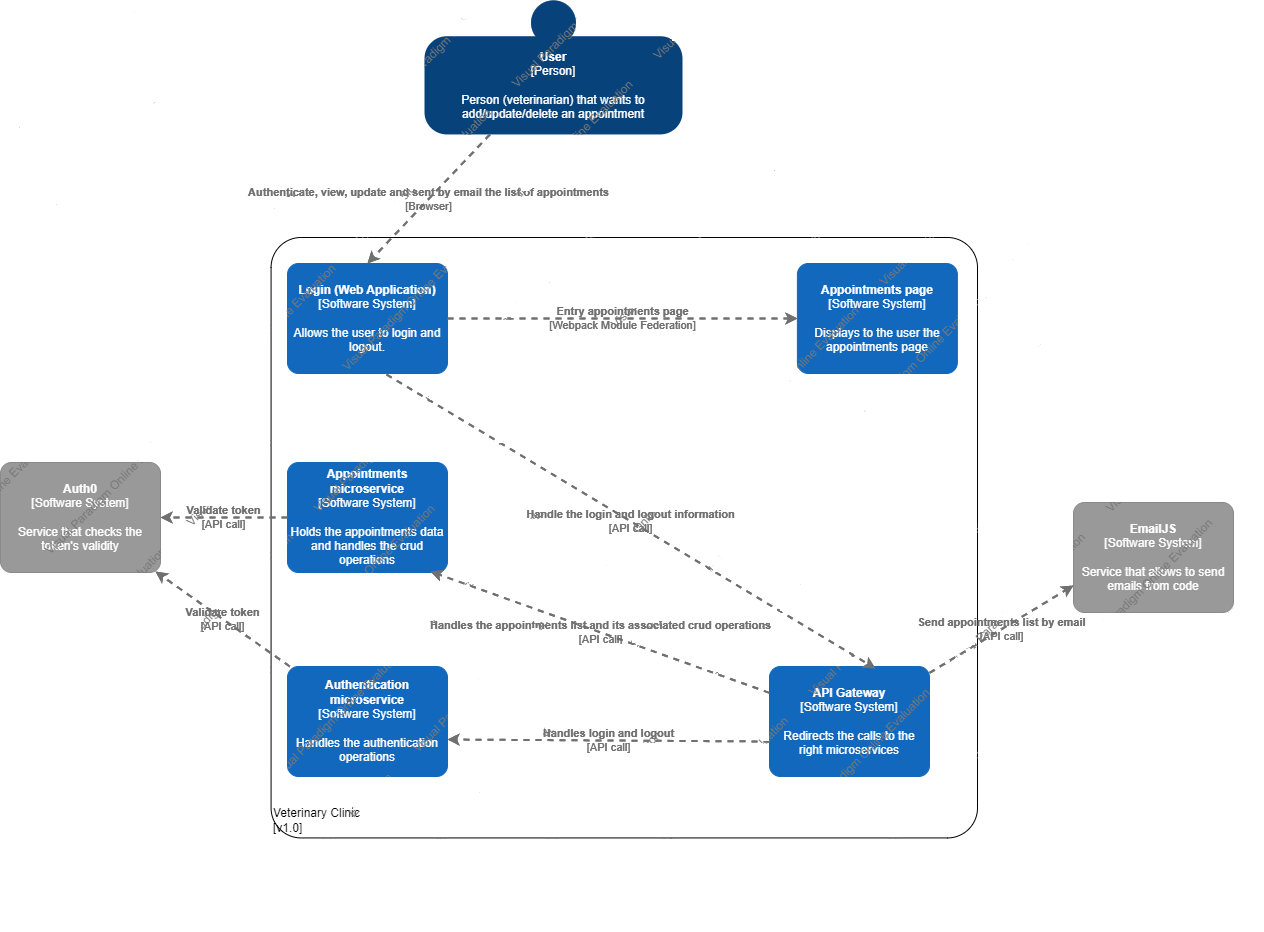
3. API Gateway Pattern

The internal microservices and client apps are connected by an API gateway. It serves as a reverse proxy, forwarding client requests to backend services. Additionally, it offers universal issues like caching, SSL termination, and authentication.

UML Diagram:



c4 Container Diagram:



c4 Context Diagram:

