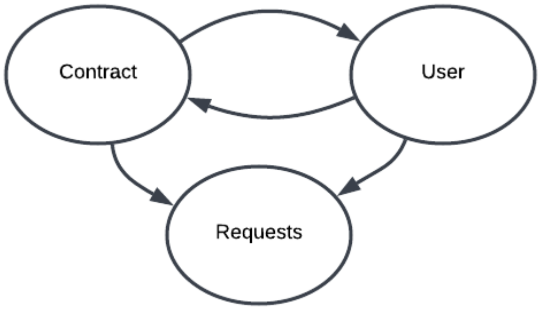
Software Architecture
Task 1

Group 27a: Ata Ağırkaya, Alex Bolfă, Lauren de Hoop, Ioan Hedea, Paris Loizides, Rok Štular

Bounded Contexts



The HR system comprises 3 different domains, the Contract domain, the User domain and the Request domain. Using these domains we can group similar functionality as they will share the same domain model. Within the contract domain, the contract microservice will be implemented. The user domain will be responsible for all user and authentication implementation. There is a constant flow of data between the user and contract microservice, as each contract is bound to a user. The flow of data in the other directions is in regards to the user information part of the contract, such as the users address.
There is independent functionality that will be responsible for handling the requests users can make to the system. This request domain will retrieve a data flow from both the user and contract domain to satisfy requests made by the users. This domain does not store any of the user or contract information, it only relays information.

Architecture Overview

It is important to ensure that each bounded context is well- defined and fully focused on a specific area of functionality. In our software architecture, we have the authentication microservice which gets an http request and gets the user netId and the password from the body of this http request. Security is covered in the authentication microservice, given that it uses Spring Security.

The User microservice deals with storing the user netId and important information about them in the User database, and also getting the registered users from the authentication microservice, both of which determine a subsystem. It deals with requests of querying the User database, and has to give the role for each user, to determine the authorization for each individual user.

The contract microservice deals with storing the contracts of each user, and with editing the contract of them. For a certain user to edit a contract, that user has to be authorized with regard to their role. The contract microservice relies on the user information provided by the user microservice, such as the netId and role, which determines the salary scale.

The requests microservice has to deal with these kinds of operations, such as the request of editing or terminating a contract. Both the contract and the user microservices are in a determined subsystem to ensure encapsulation on the given operations. The API gateway connects all of these given subsystems. Once a user makes a query, such as seeing their contract, the given subsystems are connected to post the given query response. The role of the API gateway in the system would be to act as a proxy for requests from users, forwarding the requests to the appropriate microservice and returning the response to the user.

Microservices

The role of each microservice in the overall HR application would depend on its specific functionality. Here is a description of each microservice and its role in the system:

Contract:
This microservice will be responsible for managing and storing all the necessary information about the contracts, such as the terms of the contract, contract type, and the duration of the contract.

The microservice will include services for creating new contract records, updating existing records, and deleting records that are no longer needed. These services will also allow for querying and retrieving contract information, such as the terms of a specific contract or the type of the contract. This also gives the ability to edit draft contracts in the negotiation phase of hiring new employees.

Request:
The management of requests microservice would be an essential part of any system that involves the handling of requests from users or employers. This microservice will be responsible for managing and storing all the necessary information about the requests, such as the type of request, the date and time it was made, the details of the request, and the current status of the request.

The microservice would provide a range of services that will allow other parts of the system to interact with the request data. These services will include functions for creating new request records, updating existing records, and deleting records that are no longer needed. They will also allow for querying and retrieving request information, such as the status of a particular request or the details of a specific request.

By handling the management of requests in a dedicated microservice, the system will be able to scale more easily and provide better performance for users. The microservice will be able to handle a large number of requests concurrently, and it will be able to store the request data in a scalable and efficient manner. The services provided by the microservice would also make it easy for other parts of the system to access the request data and integrate it into their own processes.

Notifications:

The notifications microservice will be an important part of any system that needs to keep users informed of important changes or updates. This microservice would be responsible for sending notifications to users when certain events occur, such as the termination of a contract or the update of a request that they have made.

The notifications microservice will include functions for specifying the type of notification to be sent, the recipients of the notification, and the content of the notification:

User:
This microservice will be responsible for managing and storing all the necessary information about the users, such as their personal details, contact information, and roles within the system.

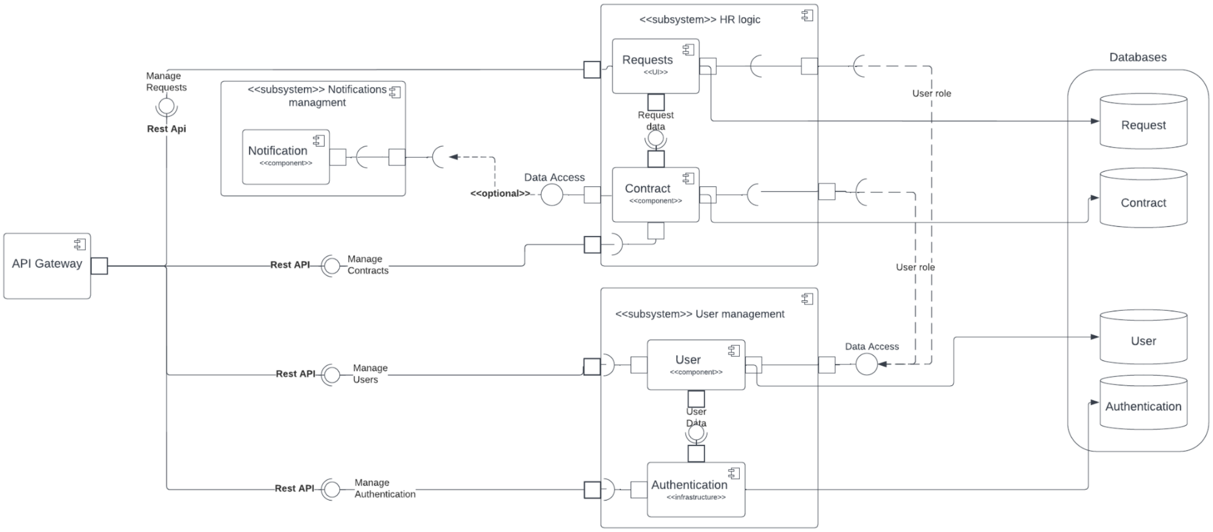
The user management microservice will also provide tools for managing the roles of users within the system. This could include features for assigning and revoking user roles, as well as checking the roles of users when they attempt to access certain parts of the system. This will help to ensure that users are only able to access the parts of the system that they are authorized to use.

Authentication:
The authentication microservice handles the security of our system and is part of the user domain. This microservice provides functionality for checking authentication through Spring Security as well as safely storing the authentication data, e.g. ids and passwords. Furthermore through the authentication service a new user can be registered by HR and added to the system. As no other microservice can access the authentication data, it is not possible to gain access to this data through any other part of our system.

API Gateway:
The API gateway will be an important component of any system that uses microservices. The role of the API gateway would be to act as a single entry point for requests from users, forwarding the requests to the appropriate microservice and returning the response to the user.

The API gateway can also make it easier to manage and monitor the traffic flowing through the system. The API gateway can provide a single entry point for all requests, allowing the system administrators to monitor and analyze the traffic at a high level without needing to look at the individual microservices. This can help to identify patterns and trends in the traffic, and it can provide valuable insights into how the system is being used and where improvements can be made.

In summary, the HR application will use a set of microservices to handle various tasks and functions. Each microservice will have a specific role to play in the overall system, such as managing contracts, requests, notifications, or user data. These microservices will provide services that would allow other parts of the system to interact with the data and perform tasks such as creating, updating, or deleting records, or querying and retrieving information. The API gateway would provide a single entry point for all requests, improving the security and performance of the system, as well as making it easier to manage and monitor the traffic flowing through the system.



<<subsystem>> HR logic
Requests
<<UI>>
Request data
Contract
<<component>>
Manage Requests
Rest Api
<<subsystem>> Notifications managment
Notification
<<component>>
Data Access
<<optional>>
User role
Databases
Request
Contract
API Gateway
Rest API
Manage Contracts
User role
<<subsystem>> User management
Rest API
Manage Users
User
<<component>>
Data Access
User
User Data
Authentication
Rest API
Manage Authentication
Authentication
<<infrastructure>>