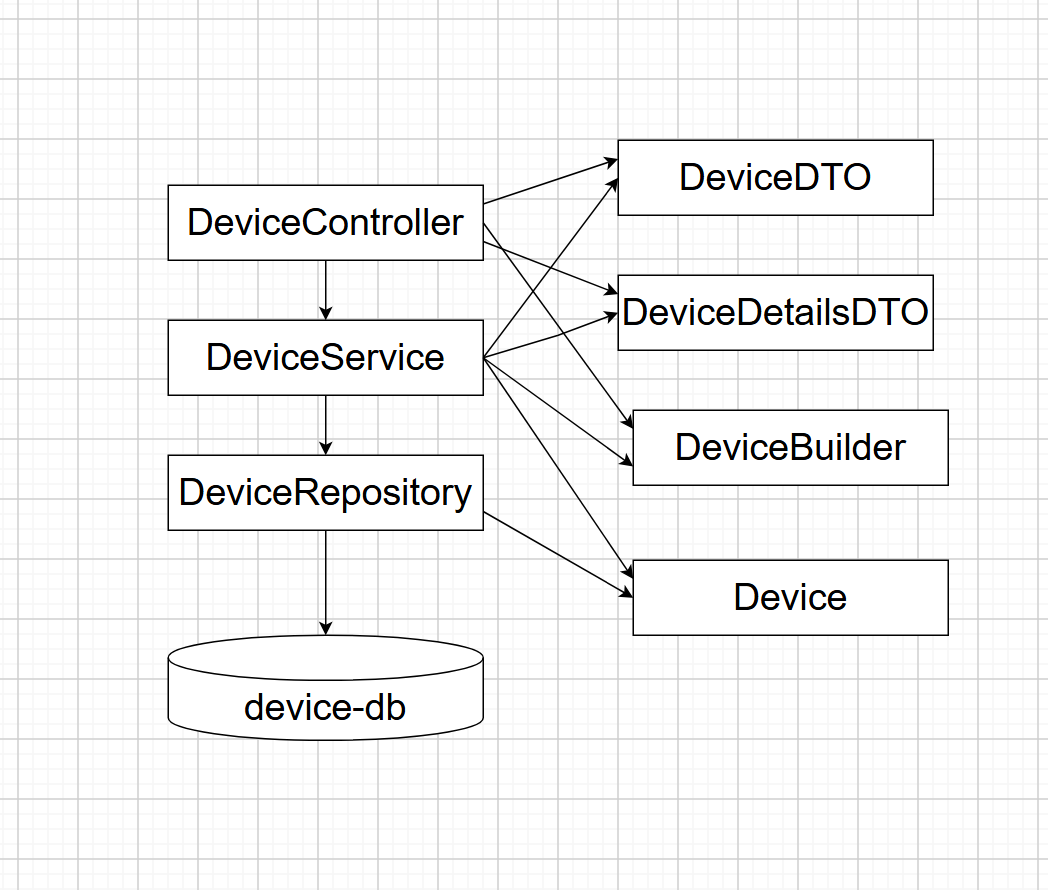
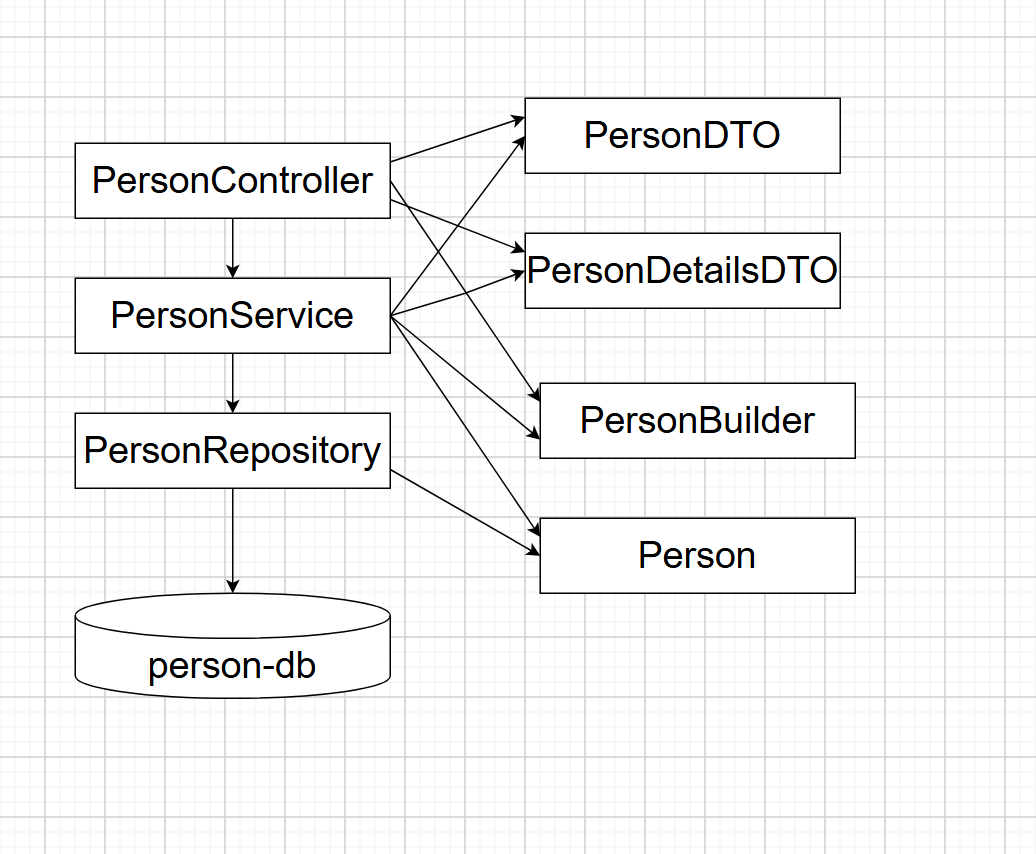
Assignment 2 – distributed systems

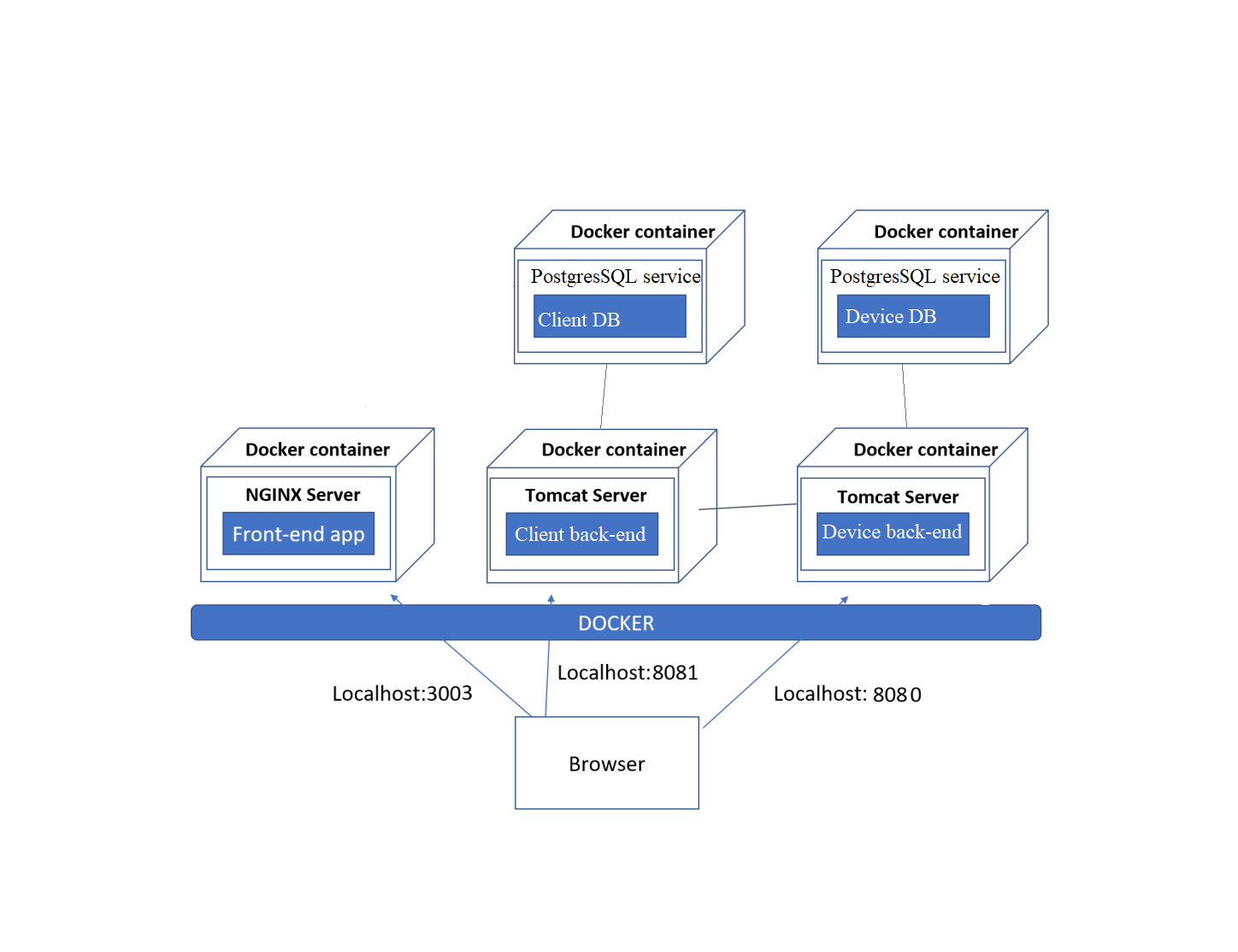
Vaida Diana-Laura, group 3

Conceptual architecture of the distributed system





UML Deployment diagram



My project works for the back end with Spring Boot(in Intellij Idea) and front end with React (in Visual Studio Code). My microservices are Person and Device with their respective databases.

For every microservice I have packages for controllers, dtos, entities, repositories and services.

My microservices communicate with each other through a Rest Template(with urls). In my Device microservice I also have packages for the Person class.

CRUD for Person – I have get, put, delete, update methods.

CRUD for Device - I have get, put, delete, update methods.

Mapping Person with Device –

I can make the mapping for a device to a person (if the idClient and idDevice inserted exist).

If I delete a person, all the connections to any devices are deleted(the idClient for those devices becomes null, they aren’t assigned to anybody).

I can update the idClient for a device, I can change the person the device belongs to.

These functionalities are only available for the admin.

For the user, the only functionality available is to see the devices the user has through a list.

In my Person microservice I also have a method for authentification in PersonService and a login url in PersonController.

In my front end part, I can access the pages to do all these only If i create an account/ log in.

To create an account, the user has to specify the name, password, role, same for the log in. If the person attempts to create an account with a username that already exists, an error will be thrown.

In my PersonService I have urls to communicate with DeviceService from the microservice Device. When a new account is created, it puts itself in the Person microservice and it also communicates through a url with the Device microservice, it adds that specific id of the person in my table Person from that microservice. All of this, I manipulate in my front end part of the project.

At first the user has the option to log in, or create an account.

If it logs in, the admin then it’s redirected to a page to do CRUD for the Devices or Persons, or to do mapping. Users have their id extracted in React and then the url takes it and returns the devices that have that id as idClient in the database.

If a url is searched without a user logged in or having the role necessary, it is redirected automatically to the home page.

For the front end, every action is redirected through a button. The admin can select to view all devices or persons, and then a list is showed with the delete and update buttons, clicking them the admin makes a request to the url for updating and deleting. For the update, after clicking it, a new page is created with all the fields to be edited. If I click the delete button, the object is deleted from the database and it dissapears from the list. A search field is also present in the view page, to insert an id and then try to find it, if that person or device exists it is returned with all the attributes, if not, nothing happens. The ids appear next to the object in the list.

In the mapping section, I assign a device to a person if both exist(their ids). A page is created with a list (containing all the devices, their details and the id of the person they are assigned to). You can search that device by inserting the id, like in the regular CRUD for devices, or you can update the device (change the person it belongs to, change the id, if that person exists).

In my PersonRepository I have a query that searchs if a person with a specific name already exists in the database, to avoid creating multiple accounts with the same name. Every account should have a unique name. In the DeviceRepository I have a query that return the devices that have assigned a specific id for the client (the value of the foreign key between the Device and Person tables from the Device microservice), I use this query in the DeviceService to return the specific devices and then pass them to the DeviceController which maps them into DTOs.

All the functionalities in the mapping sections are done through functions in the PersonService from the person microservice that pass urls that are used by the DeviceController in the device microservice.

My Monitoring and Communication Microservice(consumer microservice) communicates with the Monitoring database(consumer database) in my case.

The information send to the consumer database comes from 2 queue – the client simulator(the producer microservice) and the device database.

The device microservice acesses the device database and sends though a queue an id in the consumer microservice and the maximum hourly consumption for that device. The producer microservice has a config file that has stored an id of that device