IADI project report

1. Architecture description

To implement our server side application we used multitier architectural style. We used Kotlin with Spring Boot framework, for more effective and stable performance.
We divided our backend app into 4 layers.

Data layer was used to specify entities, create repositories, and provide the database. We used object oriented database. Entity classes helped us to easily manage and operate on the data objects. The challenge we faced was how to implement actors of the project in such a way, that every of the actors could have different functionalities and characteristics, but be considered still as a user - an entity with assigned role, username, email and password which are used for app security. Solution for this, was creating an abstract class AppUser from which the actors inherit fields mentioned above, and still have information and functionalities that are specific to their role only.

In the **service layer** we created services for each main endpoints, that used the functions defined in the data layer and helped to communicate between data repositories which store data access objects and their representatives in the front end. In the services we allow only given roles to access the functions to match the application logic and requirements.

```
@Service
class HubService (val hubs:HubRepository){

    Antoni Lasik +1 *
    @RolesAllowed("HUBWORKER", "MANAGER")
    fun getAll()=hubs.findAll()
```

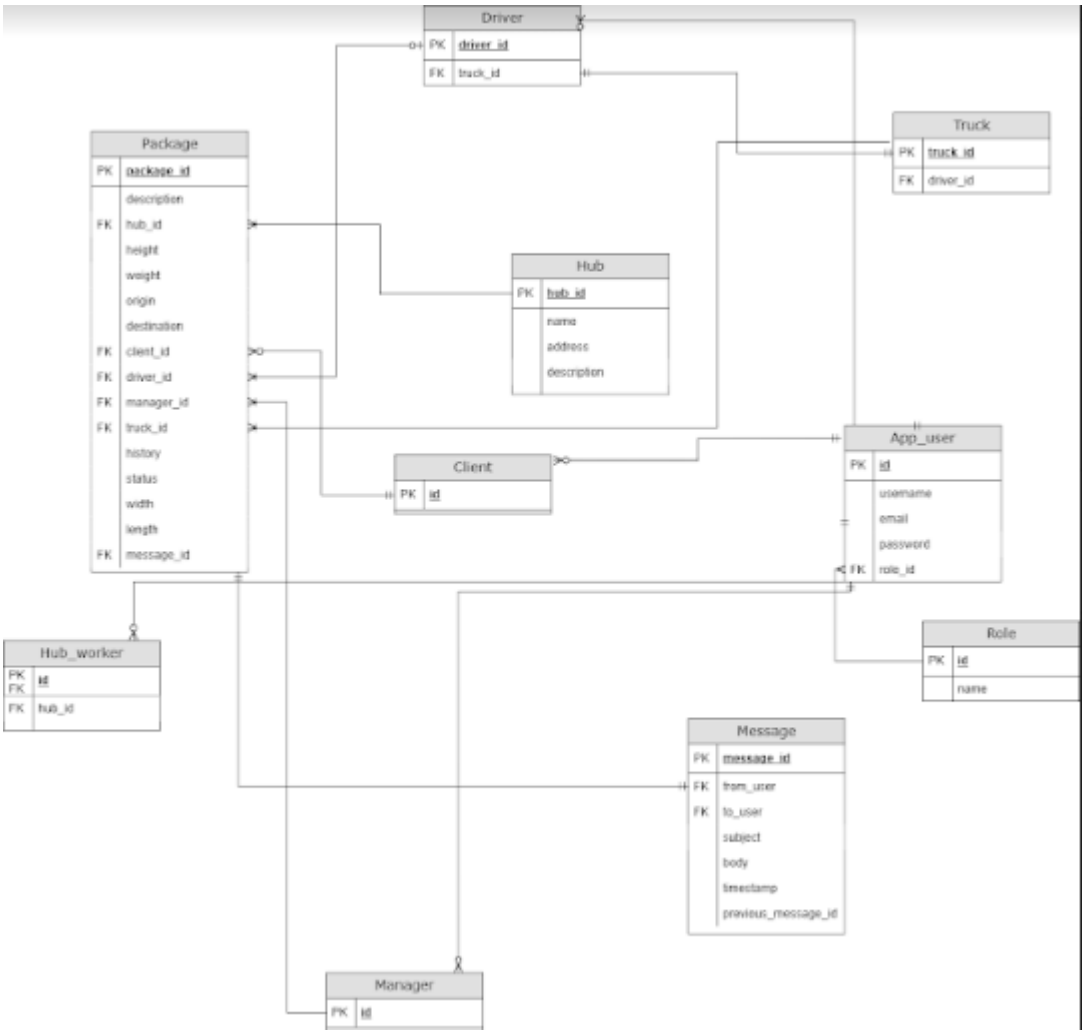
In the **presentation layer** we have defined the controllers for Clients, Trucks, Hubs, Packages and Messages. In each of the API we used functions from corresponding services to map the DAO into DTO.

This layer was also responsible for controlling and obeying security rules, such as allowing given user to see his, and only his packages, or massages. Example bellow.

```
@PreAuthorize(CanGetClientPackages.condition)
annotation class CanGetClientPackages {
    Mirson99 +1
    companion object {
        @Language("SpEL")
        const val condition = "(hasRole('CLIENT') AND @mySecurityService.myPackage(principal,#id))" +
                "OR hasRole('MANAGER')"
    }
}
```

User interface layer is implemented using typescript with redux (for easy state storage) and react (for quick and efficient user interface creation). This layer provides functionalities specified in the user stories. All data needed for interface is provided by presentation layer obtained on requests from client side through API to which client side is connected by proxy.

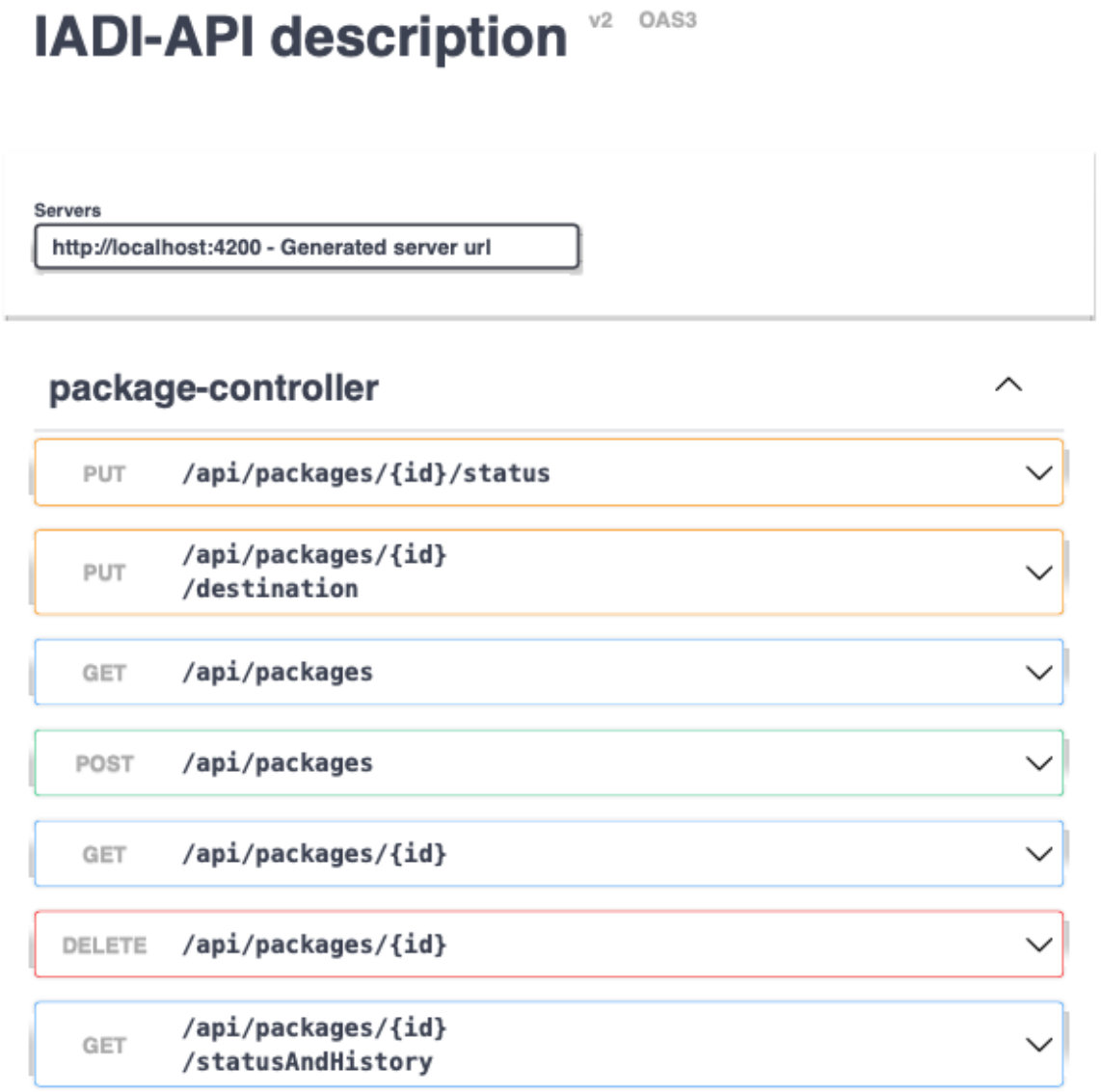
2. Database schema



link to jpg with diagram: new_database_schema.drawio.png

3. **OpenAPI summary**

Full file
https://github.com/IADI-FCT-NOVA/iadi-project-assignment-65355_65337_65324_65343/blob/c7c7b2435eacb01a636ce84098bad312aab56e4d/Full%20API.pdf



message-controller

- GET /api/messages
- POST /api/messages
- GET /api/messages/{id}

client-controller

- GET /api/user/{username}/truck
- GET /api/user/{username}/role
- GET /api/user/{username}/packages
- GET /api/user/{username}/messages
- GET /api/user/{username}/hub

truck-controller

- GET /api/trucks
- GET /api/trucks/{id}/packages

hub-controller

- GET /api/hubs
- GET /api/hubs/{id}/packages

4. Security rules implemented

- CanCancelMyPackage - client can cancel a shipment if it is under the status Awaiting entry and it's his shipment
- CanGetClientPackages - client can see list of packages which belongs to him
- CanUpdatePackage (only working on backend) - driver can only change the status of a shipment to in transit or delivered, hub worker can only change the status of a shipment to in storage, manager can change the status of a shipment to any of the possible states
- ClientCanGetHisMessages - user can see his messages
- ClientCanGetPackagesInfo - client can see details about his packages

5. User stories implemented

Link to youtube video with functionalities:
https://youtu.be/jEOxYddHNrY

- As an anonymous user, I want access the application and see the main page of the application.
- As an anonymous user, I want to type in my username and password in order to sign in
- As a signed-in client, I want to access the homepage and see a list with my shipments present and past, with dates, identification, and status
- As a signed-in client, I want to access the homepage and see a list with my messages
- As a signed-in client, from list of my shipments i can go to create shipment page and see how to create a new shipment
- As a signed-in client, I want to introduce the details of a package to create a new shipment and seeing it in the list of shipments in the status "Awaiting entry".
- As a signed-in client, I want to access the list of shipments to select a shipment and seeing the details of a shipment including history and status, and messages
- As a signed-in client, I want to access the details of a shipment to go to send message page and send a message related to the package and seeing the message in the message list
- As a signed-in warehouse worker, I want to see the list of packages in my warehouse to filter the list by status destination, and date and see the filtered list of packages
- As a signed-in warehouse worker, I want to see the filtered list of packages in my warehouse to select a package and see the details of the package
- As a signed-in truck driver, I want to I want to see the list of packages in my truck to filter the list by destination, and date and see the filtered list of packages
- As a signed-in user, I want to sign-out and see the main page of the application

6. Group's dynamic

Work in our group was well distributed with some of us feeling better in some topics and developing specific parts of an app and some treating the whole project holistically. With Łuasz experience in front-end development and Mateusz, Mateusz and Antoni will to learn and work hard we managed to coordinate our work that it ran smoothly and was well distributed.

7. Links to commits

First part of project commit:
https://github.com/IADI-FCT-NOVA/iadi-project-assignment-65355_65337_65324_65343/commit/ebdc573635e5235f9097601c3651c3b21ae34ee4

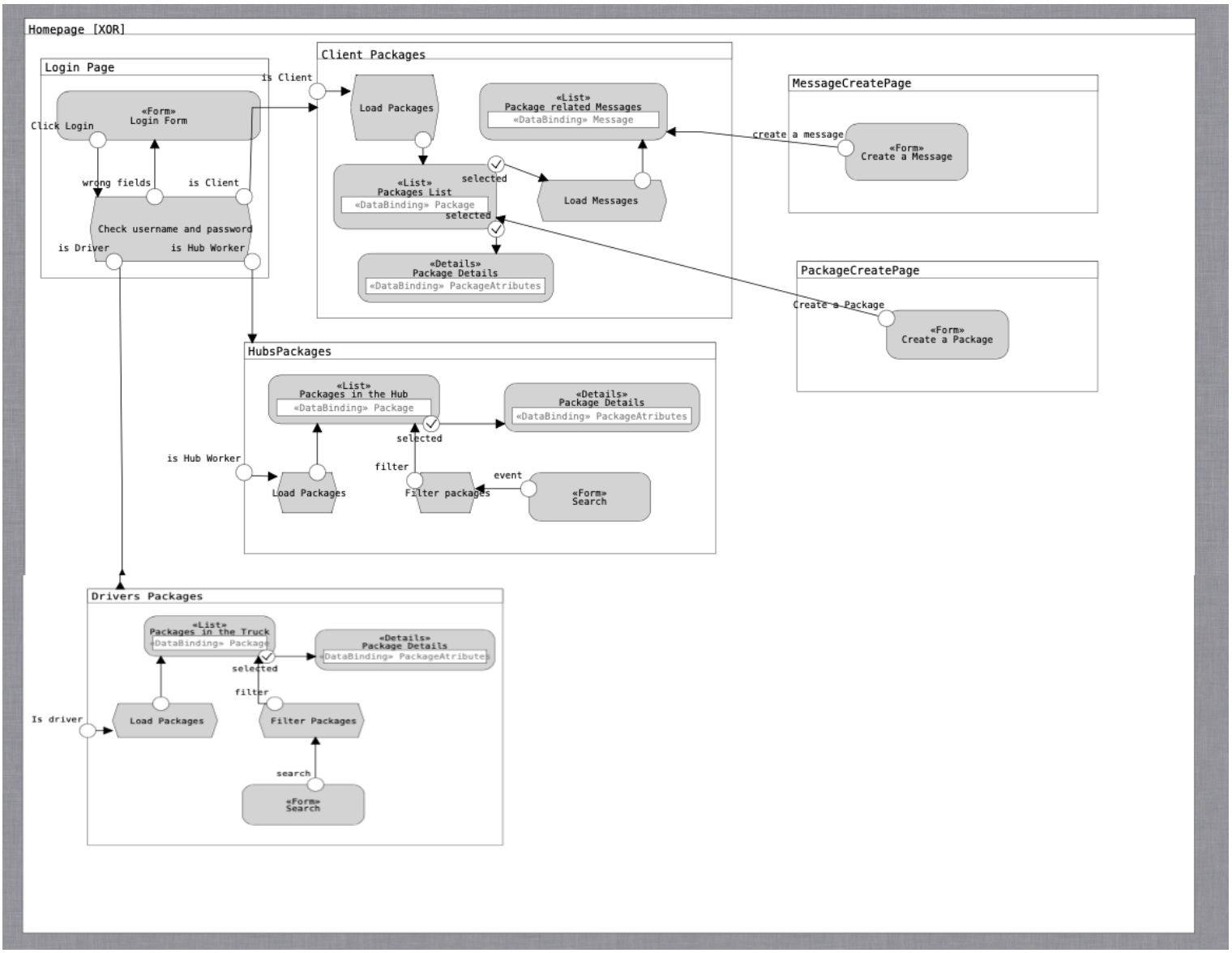
Final commit:
https://github.com/IADI-FCT-NOVA/iadi-project-assignment-65355_65337_65324_65343/commit/68feba18a2ca10a38e4844c69f29a81ee334b801

Final commit after fixes (after deadline):
https://github.com/IADI-FCT-NOVA/iadi-project-assignment-65355_65337_65324_65343/commit/370cac18264e145585315a7cb1835b62d5b583c2?fbclid=IwAR25EJ9hLQLObJ6UuE3GFf3o7QMHfUG-f5S9qwK81HZ6CgMzot4fsarHZj0

8. Lessons learned: Highs and lows in the project development

We had a hard time with learning all of the new things, most of us were not fluent in java, typescript and html, also configuring the security and dealing with circular dependencies was quite hard. Most of our frustration was with little mistakes that we didn't understand because our lack of experience. One of the lessons that we have learned is that work should be well distributed in time and with a time comes better understanding of the subject. Also that its hard to create front end of the app if you have only one week to learn it.

9. IFML diagram



10. Auto evaluation

We think that our work deserves a final grade equal to 75% of maximum points, because in the first phase we have fulfilled almost every requirement and the second phase didn’t go so well. But on the other hand we don’t have as much experience as other attending this course. For three of us the internet app development is not in the main scope of our home courses. But we have tried to provide the best of us. Given above mentioned facts we think that we deserve 75%.