EE8205: PRINCIPLES OF SOFTWARE ARCHITECTURE

ASSIGNMENT NO. 02

NAME :SENEVIRATHNA M.G.T.T

REG NO. :EG/2015/2766

SEMESTER :08

DATE :17/02/2020

CHANGES DONE TO THE PROPOSED ARCHITECTURE AND IMPLEMENTATION DETAILS

The initial architecture had several components as shown in the below figure 1. However the implementation was done only for the happy path. Therefore all the other components are kept aside for future developments.

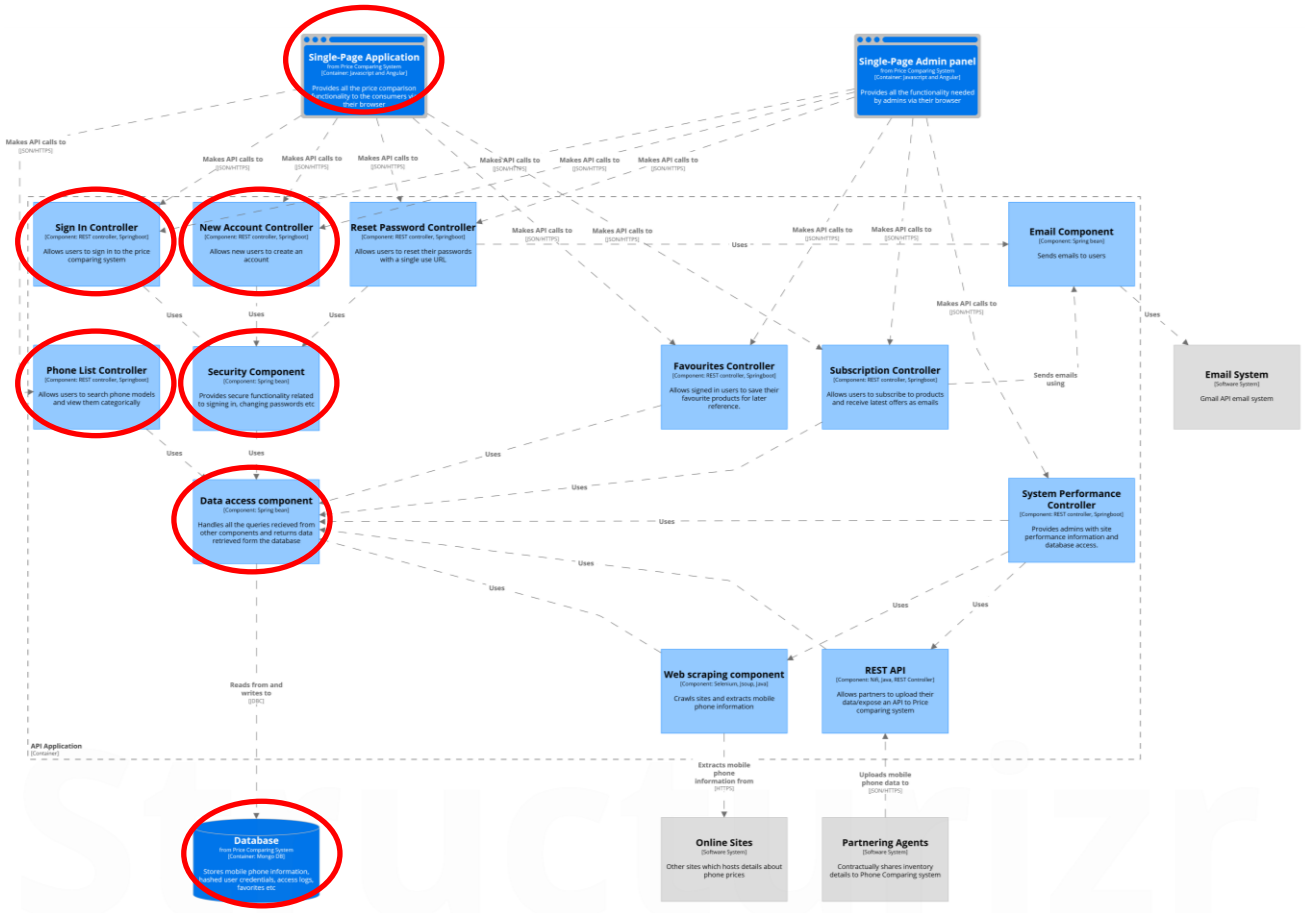

Figure 1: Initial plan for the components. The red circles mark the components which were implemented.

From figure 1, the red circles show the implemented components. I developed the whole project using services and repositories. Models were used to transfer data across layers. The frontend of the project was designed as a separate project using Angular. The backend is another separate project using spring boot. Then the Database was implemented using Mongo DB. The database was connected using the "mongo-java-driver". Then the database querying was done with the help of "spring-boot-starter-data-mongodb" dependency where all the implementations for basic crud operations are ready made and available. The security component of the backend was accomplished using "spring-security-jwt" tokenization. There were several problems with regard to this which are discussed in the next section.

CHANGES

Since the implementation is not of large scale the following changes were done to the initial design.

- The sign in component and new account components were not implemented separately. Since two controllers were enough for the two tasks, the two components were designed together.
- Searching of the phones is done in the front end. Since all the phones are loaded in the initial step, the items can be search much faster in the front end. A real time search result can be shown with this kind of implementation.
- Since the happy path is our main concern, I did not implement the components which are not circles form figure 1. However the data obtained from the outside is assumed to be stored in the database after web tasks such as web scraping and uploads by the clients. A separate button for favourites was added. Subscription controller and the rest of the components can be added in the future as need be. Reset password controller was added since it stays out of the happy path.
- The part where data is collected and stored is assumed to be completed before hand by the crawlers, web scraper and the provided REST API for clients. Then this data is stored in the databases after processing. Then it is used in the parts I have created.
- I created the roles and the classes need to implement it but could not complete due to limited time we had.

However the room for development is available for all the other components if an addition needs to be done in the future.

PROBLEMS FACED AND SOLUTIONS TAKEN

Some of the major problems that happened to me while doing the project are mentioned below.

1. Starting the project in Angular was problematic since my Angular version had a problem in the beginning. Then the problem was solved by upgrading all the packages to latest angular stable version. Before doing that the Node.js and NPM also had to be upgraded for better reliance. A similar problem happened for Mongo DB. This was solved by reinstalling the Mongo DB for windows.
2. Angular was not a much familiar language to me. Thus I had to learn most of the parts completely new.
3. Understanding the routing in Angular took some time. Then finally after intensive studying I was able to fix the routing problems I had in the project.
4. Problem with login was the addition of tokenization. This took ages to complete. Using the interceptors in the frontend was a concept that took a lot of effort to understand. Then after several attempts I was able to get it done.
5. The security in the backend was also not an easy task. There was one major error that I faced during the development of backend with JWT. Creation of tokens was clear to me at a glance. But the Usage of token filtration took some time to digest.

6. There was the continuous 401 error that occurred without any errors in the code. So I had to spend a lot of time trying to figure out what problem it was. Then I found out that I had not configured the cross origin URL in the PhoneController class. The concept was not familiar for me at the beginning. But then after some thorough internet searching, I was able to find out the reason.
7. Hard coding for database queries was not needed in the Mongo DB. This I found out the hard way. However, I learned that there are already implemented functions in the spring dependency "spring-boot-starter-data-mongodb". Therefore using them through an interface one of the time savers.
8. Finally when I tried to bring my code from my home PC to the Laptop, I faced the problem where versioning of components failed to match. I had to go through the process of reinstalling all the necessary dependencies. Databases had to be created with necessary mock data.
9. Creating the documentation for REST APIs is a tedious task. Thus I used Swagger to make it easier. However, implementing this using the "springfox-swagger-ui" took a considerable amount of time. Numerous problems occurred during this phase.
10. In addition to these problems I came across a several other problems such as minor syntax errors and mismatching of Mongo credentials. However those were corrected in relatively lower time periods.

SETUP GUIDELINES FOR THE PROJECT

GitHub Link for the project's backend, DB and class diagram –
https://github.com/t-T-s/phone-price-comparer
Google Drive link for Project's Frontend -
https://drive.google.com/file/d/1oV8MY7alnQcbLlnWjcSuZLQvd6dTlA_A/view?usp=sharing
(This is because Angular frontend has a lot of files and that exceeds the maximum number of files (100) that can be uploaded to the GitHub repo at a time.)

1. You need to install Mongo DB first. Then Start the mongo daemon beforehand. If it is running as a service, check for the port numbers. There shouldn't be any conflicts.
2. Then download the DB folder in repository and using command line or Mongo DB GUI, add the phones JSON file as a collection to the database "phoneList". (No necessary to make it manually. By running the backend once and singing in also creates this database.)
3. Then download the two projects. You can open the projects in Intellij Idea. The configurations for running are already there in the .idea folder. So no need to separately create those configurations. Then run the backend first. After that you can run the frontend. (Prerequisites: You need to alredy have installed java 8 and Angular 8 or better. Also the Node.js version should be 10.x or above.)
4. Finally you can visit the link http://localhost:4200/ in your browser and select register. There onwards the system is very intuitive. Once the registration is complete, enter the login credentials. There after you'd enter the search home page. In the top right corner you'll see the sign out button to sign out once the searching is done.

Thank you for reading