

**Weekly Report 6-7-8 / 28.11.2022**

**Footballer**

**Group 13**

****

**2022-2023 FALL SEMESTER**

**BIM423 – SOFTWARE ENGINEERING**

**Members**

***1. Ali Han ÖZDOĞAN***

***2. Koray KAHRAMAN***

***3. Mechmet Chotzoglou CHALIL***

***4. Sıddık Can DAĞDEVİREN***

***5. Onur Eren Hanife***

**Links**

***• Github Link:*** [***https://github.com/Footballer-SE***](https://github.com/Footballer-SE)

***• Website Link:*** [***https://footballer-se.github.io/Footballer-WS/***](https://footballer-se.github.io/Footballer-WS/)

***• Repo Link:*** [***https://github.com/Footballer-SE/Footballer***](https://github.com/Footballer-SE/Footballer)

***• API Link:*** [***https://footballerapi.herokuapp.com/***](https://footballerapi.herokuapp.com/)

**Works Done**

* Database has been updated. A table to hold the user's login information was created in the database and linked to the "footballer" table with one-to-many relationship.

Text

Description automatically generated

* There are four Java classes in the Payload package: APIRequest, AuthResponse, LoginRequest and SignUpRequest. LoginRequest takes the mail and password that the user used during login and uses the @NotBlank footnote to make sure they are not empty, and the @Email footnote is used to ensure that the mail is written in the appropriate format.

Text

Description automatically generated

* The SigninRequest code works similarly to the LoginRequest code. @NotBlank and @Email notations are used to ensure that the user is registered in the system with correct information. The only difference between them is that the name is taken as an extra. You can see the SigninRequest code on the next page.

Text

Description automatically generated

* APIResponse indicates whether the request from our user was successful and its code is as follows

Text

Description automatically generated

* AuthResponse, the last Java file of this package, generates an access tocken, which will be used later in the Security package to check the user's login.

A screenshot of a computer

Description automatically generated

* The codes under the Exceptions package are simple methods that allow us to detect errors more easily, and the task of these methods is to print the error to the console. These methods create a template for us to use in other parts of our program. In total, seven exceptions have been identified and they are mentioned below.

- BadRequestException: If the request made by the user is invalid.

- DepartmentNotFound, DirectorateNotFound, EmployeeNotFound, and

MunicNotFound: Unit not found for Department, Employee, and Municipality.

* OAuth2AuthenticationProcessingException: Authentication failed.
* RecourcesNotFoundException : Not found error.

Text

Description automatically generated

* At the bottom of the Util package is a Java Class named CookiesUtils and this class defines the cookie handling. Cookies are a type of identification file left on the computer. In cookie files, session information and similar data are stored. This class allows us to organize the cookies of our program with methods such as adding and deleting.

Text

Description automatically generated

* getCookie method allows us to obtain cookies, the addCookie method allows us to add cookies and the deleteCookie method allows us to delete cookies.
* The class named AppProperties under the Configuration package is coded to assist security codes that authenticate and generate authentication tokens. It receives and sets the token created for authentication by using the getter and setter codes inside. There is also a function that maintains a list of authorized forwarding URIs (Uniform Resource Identifier). The code is as follows.

Text

Description automatically generated

* The task of the Java class named WebMvcConfig under the same package allows all resources, all headers and HTTP methods (GET, POST, PUT, PATCH, DELETE, OPTIONS) and its code is given below.

Text

Description automatically generated

* Finally, the SecurityConfig class has been created to perform security activations of our application and to define addresses that can be accessed without authentication.

Text

Description automatically generated

* After the configuration package was completed, the Java classes in the Security package were defined. First, the class named UserPrincipal started to be coded. This code helps to identify the user, get user’s credentials, email address and password, and also defines the user role.

Text

Description automatically generated

* The TokenProvider class included in this package contains the authentication token generation for users and the flaw detection method that will be used to validate the token.

Text

Description automatically generated

* Another Java class is TokenAuthenticationFilter. This code receives the user authentication token after the user's request to the backend, and after filtering, it detects which user made the request. After these operations, the method gives a response.

Text

Description automatically generated

* The CustomUserDetailsService class checks whether the user who wants to log in to our system exists in our database, based on e-mail or credentials. If it exists, it transfers it to the UserPrincipal class to perform other checks.

Text

Description automatically generated

* The cookie settings of the program are encoded in the file named HttpCookieOAuth2AuthorizationRequestRepository. This code has four methods.

Text

Description automatically generated

* The authorization request save method checks whether this request is empty. If it is empty, it deletes the cookie and adds a new cookie if it is full.

Text

Description automatically generated

* The method that removes the authorization request sends an empty http request to the first written method, making it null, that is, it is deleted.

Text

Description automatically generated

* The last method of this file undertakes the task of deleting the cookies that occur in the authorization request.

Text

Description automatically generated

* Another Java file available under the security suite is CustomOAuth2UserService. The codes in this file are used to load user information, detect typos that may occur during login, add new users and update existing users.

The method to load user information is as follows.

Text

Description automatically generated

* The method named processOAuth2User performs error checking by taking the user's request and information and gives us the user's attributes.

Text

Description automatically generated

* registerNewUser method creates a new user and transfers the user's information to our system.

Text

Description automatically generated

* The last method of this file undertakes the task of updating existing users. After getting the user and the user's current identity information into the method, he can change the name and mail.

Text

Description automatically generated

* The Java file that checks the authentication status under the security package allows us to create different behavior for different authentication errors and returns a 401 error message with information about the error. For any unknown authentication error, it prints the error message

Text

Description automatically generated

* The most important of the methods in the Java file that checks the authentication successful status under the security package is the code block that checks and directs the authorization to the targeted URL address after the user login.

Text

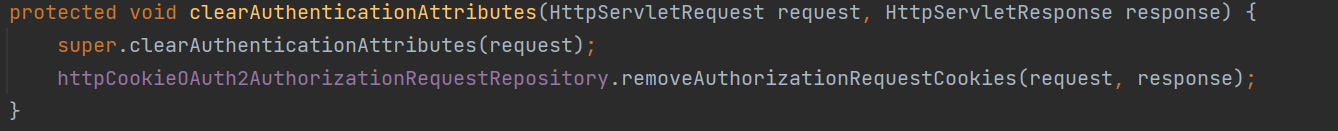
Description automatically generated

* The method that checks whether the user is authorized to go to that URL address is as follows.

Text

Description automatically generated

* The method that creates a cookie to track the user's status is as follows.



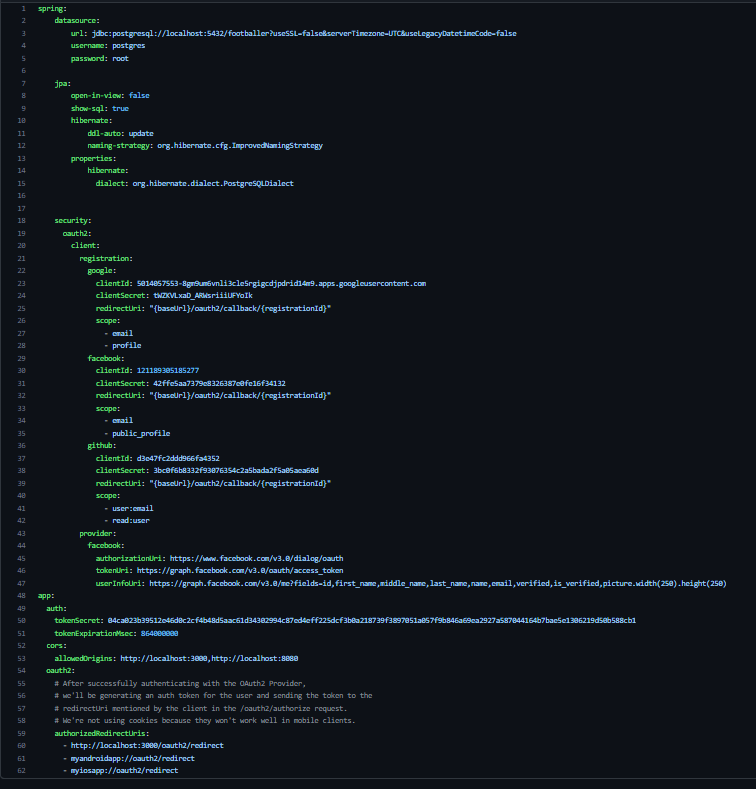
* Finally, a method named onAuthenticationSuccess, in which all three methods are used, is coded to assign cookies when all of these cases must be checked at the same time and there is no error from the URL authorization control methods.

Text

Description automatically generated

Coding of security packages is complete. The security measures taken have ensured that the program will work as it should and that unauthorized persons will be restricted in our web application.

Here is our configuration file : application.yml



* We provide Social login authentication on Google,Facebook,Github.We have classes for all of them.



* Same codes for Facebook and Github classes.Each class extends OAuth2UserInfo abstract class and overrides its abstract methods.

* The design and implementation of the login page is as follows.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

* Finally, the design of the register page is as follows.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Task Distribution**

***1. Ali Han ÖZDOĞAN: Backend Development(Spring Security and Firebase Connection)***

***2. Koray KAHRAMAN: Backend Development(Spring Security and Firebase Connection)***

***3. Mechmet Chotzoglou CHALIL: Backend Development(Spring Security and Firebase Connection)***

***4. Sıddık Can DAĞDEVİREN: UI design(UI design)***

***5. Onur Eren Hanife: Frontend Development***

**Individual Contribution**

o  ***Ali Han ÖZDOĞAN:*** ***20%***

o  ***Koray KAHRAMAN: 20%***

o ***Mechmet Chotzoglou CHALIL: 20%***

o ***Sıddık Can DAĞDEVİREN: 20%***

o ***Onur Eren HANİFE : 20%***

**Plans for the Following Week**

* **We will continue to front-end development.**
* **We will check if there any bug or error in back-end and if there is we will fix it.**