

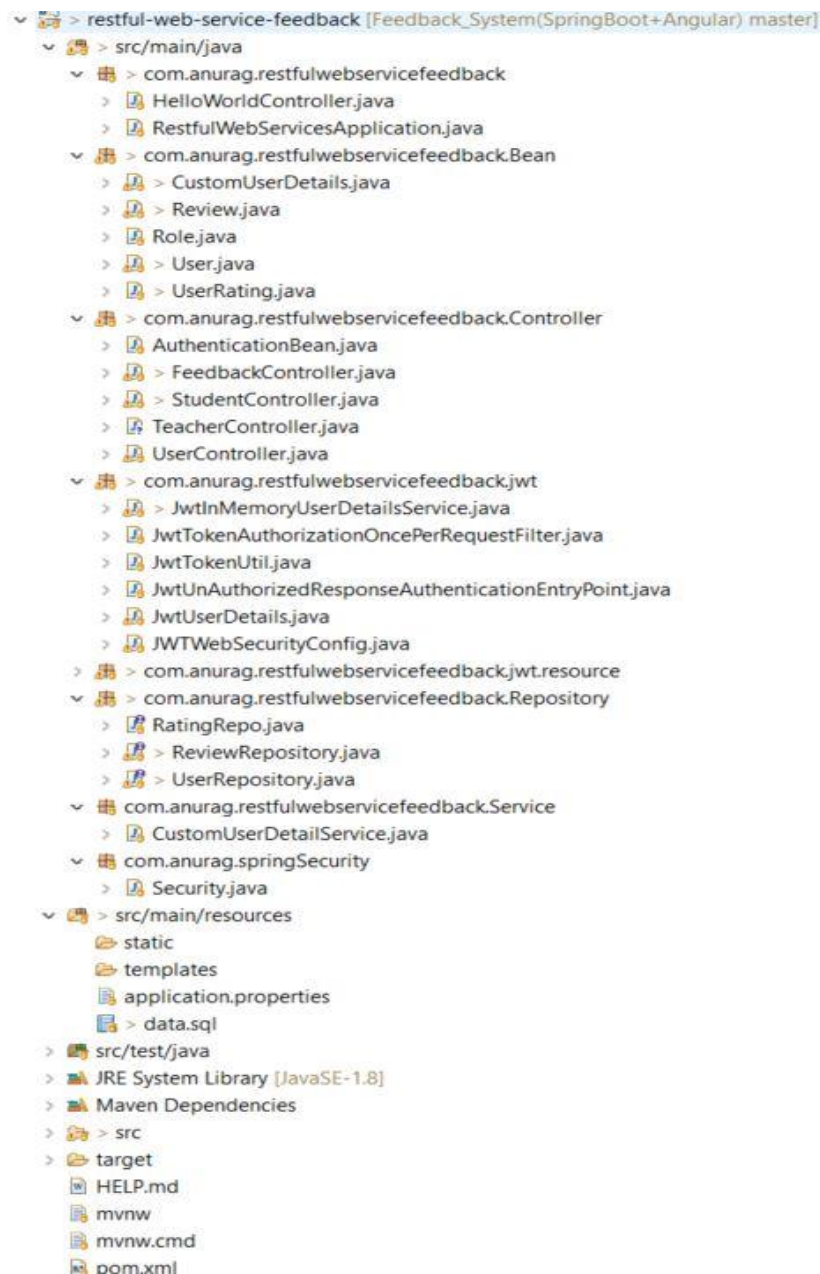
Feedback System

Basic Functionality:

This is the feedback system helps to maintain feedback given by students to their respective teachers of their particular departments. This application is built using AngularJS and Spring Boot using JWT for Authentication Token and H2 Database.

There are three categories that are defined in this system and these are Admin, Student and Teacher. Firstly, we will talk about admin part and as the name suggest Admin is the one who will manage all the users in this system. Admin will be able to Add, modify or even delete user details. He will be the one who will assign credentials to the users after they get approval to use this software. Secondly, there is student category and they will be able to give reviews to their peers by using this system. They can also rate their performance by selecting their star rating. Thirdly, it comes to the teacher part. Here, they will be able to see the feedback given to them by their students. They will be able to see the chart which is generated by the ratings of their students.

Spring Boot Structure Walkthrough:

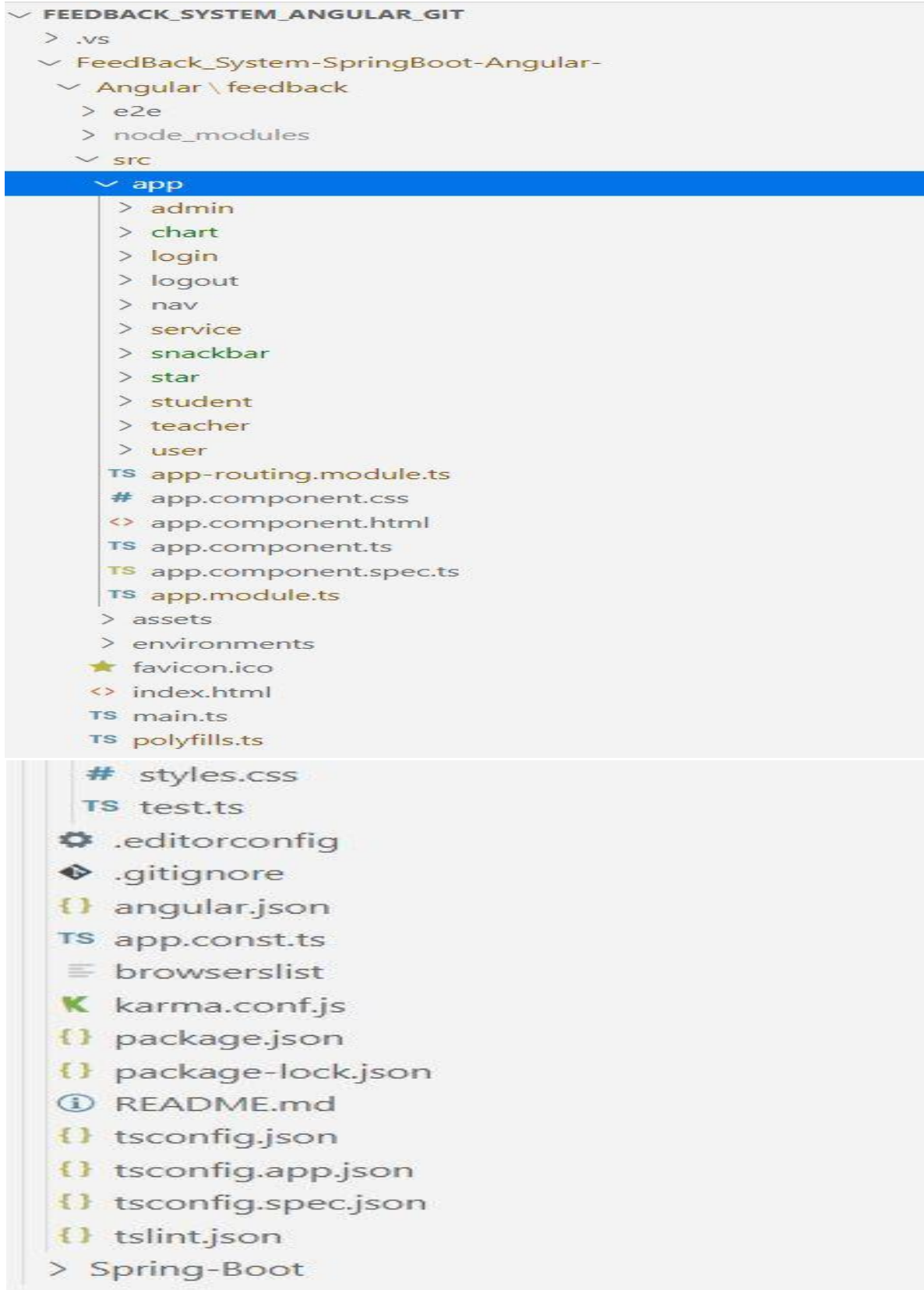


The above given image shows the code structure of the Spring Boot Application. Let's talk about this structure one by one.

src/main/java: It is the source package which contains all the packages and files which are necessary for this project to run and execute. We will try to understand the package and classes which are included in this source package.

1. **com.anurag.restfulwebservicefeedback**: This is the package which includes RestfulWebServicesApplication.java file. This file is very important for the startup of project and this is autogenerated file in Spring Boot application.
2. **com.anurag.restfulwebservicefeedback.Bean**: This is the package which can be called as the Model package for this project. It contains CustomUserDetails.java, Review.java, Role.java, User.java, UserRating.java files which are the Model classes for their respective work. Model classes are the classes which are used to interact with database in a Java application.
3. **com.anurag.restfulwebservicefeedback.Controller**: As the name suggests, this is the Controller package for this project. It contains AuthenticationBean.java, FeedbackController.java, TeacherController.java, UserController.java classes which work as controllers and they all have different functionality as per their job requirement.
4. **com.anurag.restfulwebservicefeedback.jwt**: This package contains all the necessary files to get the JWT authentication task done. It contains JWTInMemoryUserDetailsService.java, JwtTokenAuthorizationOncePerRequestFilter.java, JwtTokenUtil.java, JwtUnAuthorizedResponseAuthenticationEntryPoint.java, JwtUserDetails.java and JWTWebSecurityConfig.java files which are used to get Data from database and match the credentials with the user given information.
5. **com.anurag.restfulwebservicefeedback.jwt.resource**: This package contains files such as AuthenticationException.java, JwtAuthenticationRestController.java, JwtTokenRequest.java, JwtTokenResponse.java. This package basically works as a controller in this project for the JWTAuthenticationToken. Controller initially invoke method to request a JWT token and then it goes to the other method where response for JWT token is received.
6. **com.anurag.restfulwebservicefeedback.Repository**: This package contains all the repository's which are used to invoke the database query's to update things in backend.
7. **com.anurag.restfulwebservicefeedback.Service**: This package contains all the service classes which is used to get username and password and match with the user credentials. This is a custom build class where these credentials are matched.
8. **Data.sql**: This is the file where H2 database is populated using database. This is populated while runtime.
9. **Pom.xml**: This file contains all the dependencies which are used in this project. This is a maven project in which spring boot project automatically download all the required jar files used in this project.

Angular Structure Walkthrough:



The above given diagram shows about Angular code structure, which is also referred as the front-end code. As we can see there are number of components generated in order to get everything work together. We can see these components in the source folder, which are: admin, chart, login, logout, nav, service, snackbar, star, student, teacher, user. These all are components which were generated in this angular application. Let's discuss them one by one:

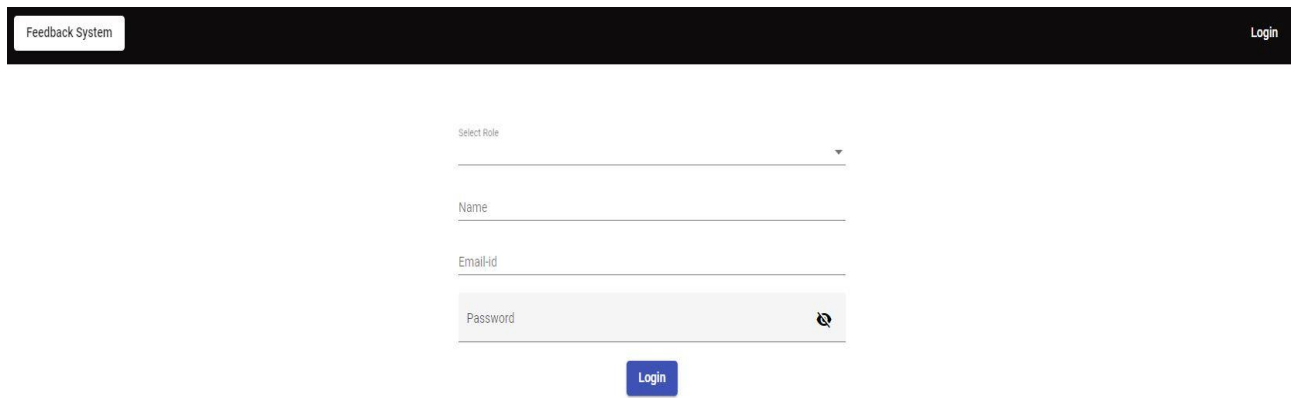
1. Admin: this component is generated to handle all the requests coming for Admin category at front end, for instance: to show list of users/to add a new user or to update a user.
2. Chart: The sole purpose of this component is to generate chart for this application. We have installed chart.js library in this project to achieve the motive to generate a new chart. This component is embedded in the teacher module to show the chart at front-end.
3. Login: This component is generated to match the credentials of the user and after confirming all the user-provided details user is navigated to the selected category (Admin/student/teacher).
4. Logout: This component is used to logout from the dashboard or we can say that to destroy the session method and navigate user to the login component.
5. Nav: this is the navigation bar generated for a better UI to navigate through the whole dashboard.
6. Star: This component is generated to show the star rating review, which is embedded in the student component for the reviews given to the teacher by student.
7. Student: This component is generated to handle all the requests for the student, where they can give reviews to their teacher by selecting their names as specified in their sections.
8. Teacher: This component is generated to handle all the requests for the teacher, where they can see the reviews given by their students.
9. User: This component was generated for the update and add a new user in the system and this component is embedded in the admin component.
10. Service: This is basically a service layer which is generated to handle all the http requests made to handle the data requested by the front-end. The benefit of the service layer is that it can be injected in any other component and we can get data using service component and embed data in the other component.

There are other standard files also present in this angular project and some of them are app-routing.module.ts, app.module.ts and in these files we have set the routing paths of this project and we can add new API in the app.module.ts file, just to add the new features in our application.

Technical Functionality:

Now we will discuss about the technical functionality of this project, which are given below:

1. At front-end user selects its category (i.e. Admin, Student or Teacher) and enter their username, password and email address.



The image shows a web application interface for a 'Feedback System'. At the top, there is a dark header bar with 'Feedback System' on the left and 'Login' on the right. Below the header, the login form is centered. It consists of a dropdown menu labeled 'Select Role' with a downward arrow. Below this are three text input fields: 'Name', 'Email-Id', and 'Password'. The 'Password' field has a small eye icon to its right. At the bottom of the form is a blue button labeled 'Login'.

2. At back end it goes into UserController.java where it calls the function in JWTAuthenticationRestController.java file and where user detail is matched at backend and after verifying all those details, a JWT authentication token is released.
3. After getting the authentication token by the JWTAuthenticationRestController.java, from front end the Http URL is fired which returns the data corresponds to the given user Details (i.e. JWT token and User Details from database) and if the given information is invalid it will give away an invalid response from the backend.
4. After getting all the desired information from back end, using 'subscribe' method in Angular we can get all the necessary data and navigate the user to their respective page (For instance if user is Admin, they will be redirected to the Admin interface or student Interface etc.)
5. Now, we will talk about Admin Side initially. In this category, User is logged in as an admin and they have authority to manage all the users. Initially admin will be able to see all the list of the users which are enrolled in the program. So, to display all the users at front end, first of all we need to get list of users from backend, which is done by using 'findAll()' method in spring boot. In this case we created a Repository in Spring Boot project so that we can talk with database.

6. After showing list on the Front-end, admin can add new user and update user Info.

Admin Login

Username	Email	Password	Section	Category	Update	Delete
anurag1	anurag@gmail	d		admin	Update	
anurag2	anuragdhiman@gmail	d	C	teacher	Update	
anurag3	anuragdhiman@gmail	d	C	teacher	Update	
anurag4	anuragdhiman@gmail	d	D	student	Update	
anurag5	anuragdhiman@gmail	d	D	student	Update	
sunny1	anuragdhiman@gmail	d	C	student	Update	
anurag6	anuragdhiman@gmail	d	A	student	Update	
anurag7	anuragdhiman@gmail	d	B	student	Update	
anurag8	anuragdhiman@gmail	d	A	student	Update	
anurag9	anuragdhiman@gmail	d	D	teacher	Update	

Add

7. When admin clicked on update user, that request goes to Http URL for update request in

User Data

Name

anurag2

Email-id

anuragdhiman@gmail

Password

Section

C

Category

teacher

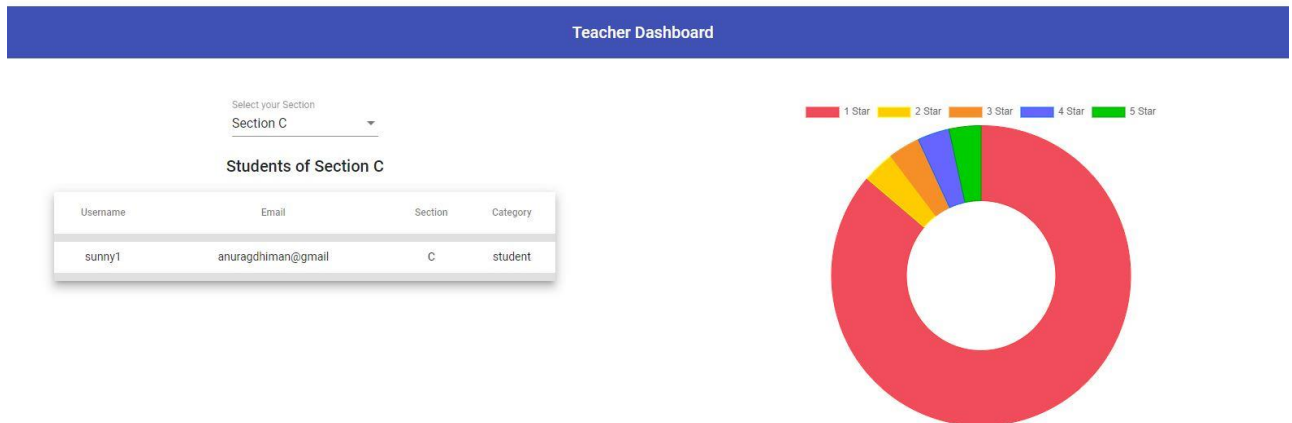
Submit

UserController.java. At Backend data is fetched from database and returned by the controller.

8. After getting data from backend, it is filled in the angular form by using subscribe () method and ngModel in the frontend.
9. Secondly, we will talk about Student side. In this category user is logged in as a student where they need to answer some questions which are hard coded in front end and they give star rating to those questions and when they submit their ratings these are saved in backend by controller. We have created a bean/model_class named FeedbackController.java which takes all the requests coming from front-end and show the output on screen.

The screenshot displays a web interface for a student dashboard. At the top, there is a blue header bar with the text "Student Dashboard". Below this, the section is titled "Teachers of Section D". Underneath the title, there is a dropdown menu labeled "Select Teacher For Review" with "anurag9" selected. The main heading for the form is "Please give ratings to complete Review". The form contains five numbered questions, each with a star rating system (five stars) and a "Rate: 1" label. The questions are: 1. Your Class Teacher is caring and respects the students; 2. Your Class Teacher has a sense of ownership and responsibility; 3. Your Class Teacher treats each with respect; 4. Your Class Teacher collectively brainstorms on resolutions to provide effective learning; 5. Your Class Teacher and students are committed to school values. At the bottom of the form, there is a text input field labeled "Write your Review Here" and a blue button labeled "Submit Survey".

10. In this category, Student selects its teacher according to their section and after selecting section a list of questions is shown at the front end and student is supposed to select rating according to their review. This star rating is built using bootstrap and the response is stored at the backend using Feedback controller.
11. This response from student will automatically be saved at backend because the user is already logged in and the token is valid.
12. Thirdly, we will talk about Teacher category. In this category teacher is able to see the review submitted by the students according to their sections. They can select their section and see the list of students. There is a graph also which can be seen on the teacher dashboard and by looking at that teacher can see the average ratings given to the by students.



13. The graph is coming from chart.js library and the data is fed into this library by getting it from backend i.e. from Feedback controller.
14. There are sessions also generated at frontend according to the credentials and these are saved once authenticated token is generated and it is saved in this session storage and used to identify the name of the username.
15. By hitting the logout button user can destroy session and remove the stored username from the browser. After removing the destroying the session the authentication token is also destroyed and it needs to be generated again to get the authorization of the database from the backend.
16. In the teacher front end design, we have used a split screen from bootstrap which is useful in this case to design the front-end according to the need of the task.
17. There are many other small functionalities are added in the graph using chart.js library, for instance the hover affects when the cursor is hovered on the graph it shows the rating given to the teacher.