**web – java - Spring Boot projects**

Contents

[ **springboot-mvc-hibernate-github** 1](#_Toc154859414)

[ **springboot-mvc-hibernate\_appLogin** 1](#_Toc154859415)

[ **Uploading Files: gs-uploading-files-main** 2](#_Toc154859416)

[ **Validating Form Input: gs-validating-form-input-main** 2](#_Toc154859417)

[ **Handling Form Submission: gs-handling-form-submission-main** 3](#_Toc154859418)

[ **youtube clone** 4](#_Toc154859419)

[ **reddit clone** 9](#_Toc154859420)

[ **Blog with Spring and Angular** 11](#_Toc154859421)

* **springboot-mvc-hibernate-github**

Java 8, Spring Boot, Spring Security, Hibernate, Spring MVC, MySQL (motor) y MySQL Workbench (Cliente), IDE Spring Tool Suite.

* **springboot-mvc-hibernate\_appLogin**

Java JDK 1.8 (Java 8)

Java Frameworks: Spring Web MVC v4.0.1, Hibernate v4.3.1.

Base de Datos: MySQL (Motor), MySQL Workbench (Cliente Interfaz Gráfica)

IDE: Netbeans 8.0

Servlets

* **Videos**

1\_Intro\_Java Web MVC Interfaz login

<https://www.youtube.com/watch?v=Pgw1lfIHReg&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx&index=8>

2\_Creando proyecto\_Spring\_Hibernate

<https://www.youtube.com/watch?v=uSPQTdERN08&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx>

3\_Proyecto Login Java Web MVC

<https://www.youtube.com/watch?v=Tplm3nep2oM&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx&index=2>

4\_Primer Controlador

<https://www.youtube.com/watch?v=SWlB0vePx1c&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx&index=3>

5\_Obteniendo datos de la URL

<https://www.youtube.com/watch?v=X86HhrFL3yE&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx&index=4>

6\_Controladores Home y Error

<https://www.youtube.com/watch?v=EM0L-Q9fHu4&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx&index=5>

7\_Proporcionando estilo a nuestro proyecto

<https://www.youtube.com/watch?v=O8yNxL_f8jQ&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx&index=6>

8\_Proyecto final

<https://www.youtube.com/watch?v=Jy4-UYYUODY&list=PLfkODrpjGnhmQu_Uj3WHmX2PVY7kl6EGx&index=7>

* **Uploading Files: gs-uploading-files-main**

<https://spring.io/guides/gs/uploading-files/>

creating a server application that can **receive HTTP multi-part file uploads**.

a Spring Boot web application that accepts file uploads. You will also build a simple HTML interface to upload a test file

repository: <https://github.com/spring-guides/gs-uploading-files>

**In a production scenario, you more likely would store the files in a temporary location, a database, or perhaps a NoSQL store (such as Mongo’s GridFS). It is best to NOT load up the file system of your application with content.**

* Dependencies:

Spring Web

Thymeleaf

* **Validating Form Input: gs-validating-form-input-main**

<https://spring.io/guides/gs/validating-form-input/>

configuring a web application form to support validation.

repository: <https://github.com/spring-guides/gs-validating-form-input>

A screenshot of a computer

Description automatically generated with low confidence

* Dependencies:

Spring Web

Thymeleaf

Validation

* **Handling Form Submission: gs-handling-form-submission-main**

<https://spring.io/guides/gs/handling-form-submission/>

create and submit a web form.

repository: <https://github.com/spring-guides/gs-handling-form-submission>

* Dependencies:

Spring Web

Thymeleaf

* Test:

<http://localhost:8080/greeting>

A screenshot of a computer

Description automatically generated with medium confidence

To submit:

A screenshot of a computer

Description automatically generated with medium confidence

* **youtube clone**

Date: 2021-07-10

video streaming application like Youtube

Backend: Java, Spring Boot, AWS S3 (to store Videos and Thumbnails)

Frontend: Angular

Database: MongoDB

* repository: <https://github.com/SaiUpadhyayula/youtube-clone-project>

youtube-clone-project-master

my repository: <https://github.com/Zarbio2019/youtube-clone-project>

* **Demo**

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A screenshot of a video

Description automatically generated

* **Topics**
* Upload Videos to AWS S3
* Angular Upload Videos to AWS S3
* Angular Create Header Component
* Save Video Details
* Save Video Details – Angular
* Save Video Details - Angular Contd.
* Upload Thumbnails – Angular
* Get Video Details
* Save Video Details and Auth0
* Secure Angular App using Auth0
* Secure Spring Boot using Auth0
* Implement Video Details Page
* Save Users to Database
* Implement Like/Dislike API
* Complete the Backend part
* Implement Home Page Angular
* Final Part
* Video: **video.mp4**

Spring Boot Angular Project in 8 Hours - Build a Youtube Clone

<https://www.youtube.com/watch?v=DW1nQ4o3sCI>

Playlist: <https://www.youtube.com/playlist?list=PLSVW22jAG8pC-4yRGXgbuSIhvaohXSDPw>

* **Part 1: Spring Boot Angular Project – Build a Youtube Clone – Part 1**

<https://programmingtechie.com/2021/07/10/spring-boot-angular-project-build-a-youtube-clone-part-1/>

<https://www.youtube.com/watch?v=rMJMGWb1zBk>

part1.pdf

* **Functional Requirements:**

User can Upload new Videos

User can Upload Thumbnails for the Videos

User can View Videos

User can Like/Dislike a Video

User can Subscribe to another User, to receive updates about future videos

User can Login/Logout using Single Sign On

User can comment on Videos

User can view the History of Videos he/she watched

User can view the List of Videos he/she Liked

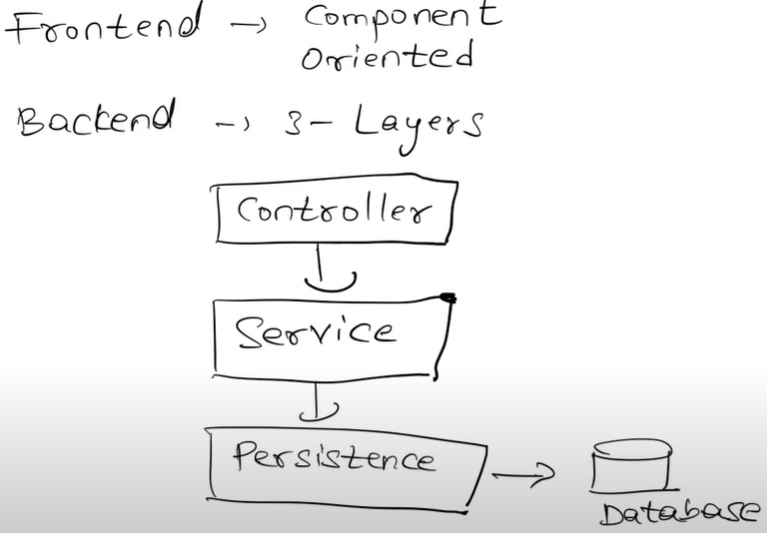
* **Application Architecture:**

Follow a **3 Tier Architecture:**

Client Application (Frontend): Angular Framework

Backend Application (running on a server): Spring Boot

Database: MongoDB



A picture containing text, handwriting, diagram, sketch

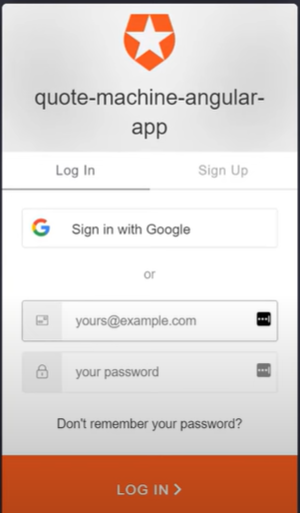
Description automatically generated

1. **Storing Media in AWS S3:**

Is a video streaming application to store and retrieve our Media Files (Video and Image (thumbnails)), storage and streaming of media content to AWS.

1. **Outsourcing Authentication to Auth0:**

When user login, we are using External Authorization Server like **Auth0.**

****

others Authorization Server: AWS Cognito, Microsoft Azure AD, and **Okta**, or your own Authorization Server by using Keycloak.

how to implement Authentication and Authorization using Spring Boot and Keycloak

<https://www.youtube.com/playlist?list=PLSVW22jAG8pAXU0th247M7xPCekzeNdrH>

**Architecture Diagram:**

A picture containing diagram, screenshot, line, text

Description automatically generated

1. **Frontend Architecture:**

implement a **Component** Oriented Architecture in our Angular application, means into a different component to **re-use** the components across different places.

Angular

Angular Material (Google): styling framework using in Youtube

frontend maven plugin: <https://github.com/eirslett/frontend-maven-plugin>

1. **Backend Architecture:**

Presentation/Controller Layer: receive REST calls from the clients and delegating the request to the **Service Layer.**

Service Layer: business logic

Persistency Layer: database

* **MongoDB Schema:**

To maintain 2 Collections inside our Database:

Video: to store the meta data related to videos

User: to store the meta data related to users

A screenshot of a computer

Description automatically generated

* Backend – Setting up Spring Boot Project:

<https://start.spring.io/>

Project – Maven Project

Group: com.programming.techie

Artifact: youtube-clone

Package-name: com.programming.techie

Dependencies:

Spring Web

Spring Data MongoDB

Lombok

AWS Cloud Support

Spring Security Resource Server Support for OAuth2

* Converting to a Multi-Module Maven Project:

two modules: for backend and frontend

Multi Module System works: <https://www.youtube.com/watch?v=JhSBS2OpGdU>

* Frontend – Setting up Angular Project:
* Frontend – Installing Angular Material:

Angular Material (Google): styling framework to build the frontend like Youtube.

* Frontend – Build Angular Project using Maven:

we can bundle the backend and frontend modules into a single JAR.

* Backend – Configuring MongoDB in our Project:

For MongoDB in your machine you can use Docker Image or download Community edition of MongoDB:

Docker Image for Mongo: <https://hub.docker.com/_/mongo>

Community Edition of MongoDB: <https://www.mongodb.com/try/download/community>

Mongo DB configurations: application.properties

Backend – Create Data Model in Spring Boot Project:

com.programming.techie.youtubeclone.model: Video.java, VideoStatus.java, Comment.java, User.java.

* **Part 2: Spring Boot Angular Project – Build a Youtube Clone – Part 2**

<https://programmingtechie.com/2021/08/05/spring-boot-angular-project-build-a-youtube-clone/>

[https://programmingtechie.com/2021/08/05/spring-boot-angular-project-build-a-youtube-clone/#](https://programmingtechie.com/2021/08/05/spring-boot-angular-project-build-a-youtube-clone/)

part2.pdf

building our Spring Boot backend API to upload the videos to AWS S3.

* Feature 1: Upload Video to AWS S3:

A diagram of a service

Description automatically generated

1. Create an AWS S3 Account

<https://aws.amazon.com/free/>

they provide a free tier for one year.

2. Create Access Key and Secret Key

to connect our Spring Boot Application to AWS S3 we need to create an Access Key and a Secret Key for your AWS account.

3. Create S3 bucket

bucket: is a container that holds all your objects stored inside S3.

4. Configure S3 inside Spring Boot Project

dependencies:

<dependency>

<groupId>io.awspring.cloud</groupId>

<artifactId>spring-cloud-aws-context</artifactId>

<version>{spring-cloud-aws-version}</version>

</dependency>

<dependency>

<groupId>io.awspring.cloud</groupId>

<artifactId>spring-cloud-aws-autoconfigure</artifactId>

<version>{spring-cloud-version}</version>

</dependency>

latest version of spring-cloud-aws: <https://github.com/awspring/spring-cloud-aws/releases>

add the Access Key ID and Secret Access Key to the application.properties

5. Implement API to upload videos to S3

com.programming.techie.youtubeclone.controller: VideoController.java

com.programming.techie.youtubeclone.service: VideoService.java, FileService.java, S3Service.java

com.programming.techie.youtubeclone.repository: VideoRepository.java

* **Part 3: Spring Boot Angular Full Stack Project - Youtube Clone - Part 3**

<https://www.youtube.com/watch?v=_VbmA1F7a-0>

Test the implementation

Implement frontend Angular for drag and drop interface to upload the videos to AWS S3.

* **Part 4: Spring Boot Angular Full Stack Project - Youtube Clone - Part 4**

<https://www.youtube.com/watch?v=l_sxmjSJnaM>

Create a header bar section on top of each page of our Angular application.

Angular Material

* **Part 5: Spring Boot Angular Full Stack Project - Youtube Clone - Part 5**

<https://www.youtube.com/watch?v=6opzyIBmW7g>

API backend to update the video metadata.

* **Part 6: Spring Boot Angular Full Stack Project - Youtube Clone - Part 6**

<https://www.youtube.com/watch?v=RuPrS2Jgdfs>

Angular flex layout: provides a HTML ui layout for Angular applications.

create the form for the video to save the media metadata in Angular.

* **Part 7: Spring Boot Angular Full Stack Project - Youtube Clone - Part 7**

<https://www.youtube.com/watch?v=G_tO5TdAivM>

continue fixes for the form for the video to save the media metadata in Angular.

* **Part 8: Spring Boot Angular Full Stack Project - Youtube Clone - Part 8**

<https://www.youtube.com/watch?v=7XXplNzKrs4>

finish up the save video details functionality in Angular.

* **Part 9: Spring Boot Angular Full Stack Project - Youtube Clone - Part 9**

<https://www.youtube.com/watch?v=ypgN8TLUyu4>

* **Part 10: Spring Boot Angular Full Stack Project - Youtube Clone - Part 10**

<https://www.youtube.com/watch?v=Z1daPTK1ptY>

secure our angular and spring boot application using Oauth2 (External Authorization server).

In this case we use auth0: <https://auth0.com/>

* **Part 11: Spring Boot Angular Full Stack Project - Youtube Clone - Part 11**

<https://www.youtube.com/watch?v=EVgUiKJ2b_A>

* **Part 12: Spring Boot Angular Full Stack Project - Youtube Clone - Part 12**

<https://www.youtube.com/watch?v=6A23TTBQK34>

* **Part 13. Spring Boot Angular Full Stack Project - Youtube Clone - Part 13**

<https://www.youtube.com/watch?v=4oCzPYhpMuc>

* **Part 14: Spring Boot Angular Full Stack Project - Youtube Clone - Part 14**

<https://www.youtube.com/watch?v=0p3LWDdiXzY>

* **Part 15: Spring Boot Angular Full Stack Project - Youtube Clone - Part 15**

<https://www.youtube.com/watch?v=OP_WsEn0W4M>

* **Part 16: Spring Boot Angular Full Stack Project - Youtube Clone - Part 16**

<https://www.youtube.com/watch?v=vvcgVXrjF8o>

* **Part 17: Spring Boot Angular Full Stack Project - Youtube Clone - Part 17**

<https://www.youtube.com/watch?v=h1liYyyoChU>

* **[FINAL PART]Spring Boot Angular Full Stack Project - Youtube Clone - Part 18**

<https://www.youtube.com/watch?v=ScUOKe-QP8E>

* **reddit clone**

Date: 2020-May

Spring Boot And Angular Full Stack Development | Reddit Clone

<https://www.youtube.com/watch?v=LfZ7xdMPytI>

* **Frontend**

Angular 9 and Bootstrap 4

Repository: <https://github.com/SaiUpadhyayula/angular-reddit-clone>

* **Backend**

Java 8, Spring Boot, Spring MVC, Spring Security, Spring Data JPA, MySQL database, Hibernate JPA, Java Mail Sender

Token Based Authentication in the form of JSON Web Tokens (JWT)

Repository: <https://github.com/SaiUpadhyayula/spring-reddit-clone>

* **resources**

Reddit clone with Spring boot and Angular

<https://programmingtechie.com/2020/05/14/building-a-reddit-clone-with-spring-boot-and-angular/>

See explication of development

<https://programmingtechie.com/2019/03/17/how-to-build-a-simple-blog-application-using-spring-boot-and-angular/>

* **videos**

Full Stack Reddit Clone with Springboot & Angular: Part 1

<https://www.youtube.com/watch?v=7PsjGI-88sc>

Full Stack Reddit Clone with Springboot & Angular: Part 2

<https://www.youtube.com/watch?v=kpKUMmAmcj0>

Full Stack Reddit Clone with Springboot & Angular: Part 3

<https://www.youtube.com/watch?v=PMr2Mh9xzm4>

Full Stack Reddit Clone with Springboot & Angular: Part 4

<https://www.youtube.com/watch?v=1ojKQxVssPQ>

Full Stack Reddit Clone with Springboot & Angular: Part 5

<https://www.youtube.com/watch?v=PN3gxkub4Ew>

Full Stack Reddit Clone with Springboot & Angular: Part 6

<https://www.youtube.com/watch?v=sNpQ_72VySc>

Full Stack Reddit Clone with Springboot & Angular: Part 7

<https://www.youtube.com/watch?v=2hsS6JJwa4M>

Full Stack Reddit Clone with Springboot & Angular : Part 8

<https://www.youtube.com/watch?v=e3Zbic5Jl18>

Full Stack Reddit Clone with Springboot & Angular : Part 9|Spring Boot JWT Refresh token

<https://www.youtube.com/watch?v=OdXojEhQcOU>

Full Stack Reddit Clone with Springboot & Angular : Part 10

<https://www.youtube.com/watch?v=euB-o-GJ1Ig>

Full Stack Reddit Clone with Springboot & Angular : Part 11

<https://www.youtube.com/watch?v=MmVu0sGF8vE>

Full Stack Reddit Clone with Springboot & Angular : Part 12

<https://www.youtube.com/watch?v=a5dE9f8HO1I>

Full Stack Reddit Clone with Springboot & Angular : Part 13

<https://www.youtube.com/watch?v=-pcwrmsd4u0>

Full Stack Reddit Clone with Springboot & Angular : Part 14

<https://www.youtube.com/watch?v=gXs3ZOcXjcQ>

Full Stack Reddit Clone with Springboot & Angular : Part 15

<https://www.youtube.com/watch?v=c-q-rd0TU3g>

Full Stack Reddit Clone with Springboot & Angular : Part 16

<https://www.youtube.com/watch?v=XP_pg9Tukps>

Full Stack Reddit Clone with Springboot & Angular : Part 17

<https://www.youtube.com/watch?v=OLT3fNcHQh4>

Full Stack Reddit Clone with Springboot & Angular : Part 18

<https://www.youtube.com/watch?v=28Yl7nF9Zdg>

* **Topics**

Part 1: Intro and Demo, Project Setup

Part 2: User Registration. User Verification & Async Processing

Part 3: User Authentication with JWT

Part 4: JWT Validation & Subreddit API

Part 5: Intro to Mapstruct & Implement Post API

Part 6: Implement API to Manage Comments

Part 7: Implement API to Manage Votes

Part 8: Implement Logout using Refresh Tokens

Part 9: Getting started with Frontend Application

Part 10: Document REST API using Swagger & Springfox

Part 11: Implement User Registration in Angular application

Part 12: Implement Login in Angular Application

Part 13: Using Refresh Tokens in Angular Application

Part 14: Refactoring Home Page Component

Part 15: Create Subreddits and Posts in Angular Application

Part 16: Post Comments & Implement User Profile Page

Part 17: Final part - Implement Voting in Angular App

Testing Spring Boot App

Deploy Spring Boot app to Heroku

Deploy Angular app to Heroku

* **Part 1: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 1**

<https://programmingtechie.com/2019/09/30/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-1/>

Spring Initialize: <https://start.spring.io/>

dependencies:

Lombok, Spring Web, Spring Security, Spring Data JPA, MySQL Java Driver, Java Mail Sender

JWT

For dynamically displaying the relative duration (like “Posted 1 day ago”): timeago

Configure Database (mysql), Hibernate and Java Mail (mailtrap) Properties:

src/main/resources/application.properties

For sending Account Activation Emails and Comment Notification Emails to the users:

Use SMTP server to send the emails, we can use a Fake SMTP Server called as MailTrap.

Database Schema Diagram:

A computer screen shot of a computer

Description automatically generated

Creating Domain Entities:

Users can create Subreddits and Posts.

other users can add Comments on the Posts, and can Vote.

**Lombok** Annotations (@Data, @AllArgsConstructor, @NoArgsConstructor) generate Getters/Setters/Equals and HashCode/toString methods and Constructors at compile time. To be able to use these annotations, you have to enable Annotation Processing in your IDE: [**https://www.baeldung.com/lombok-ide**](https://www.baeldung.com/lombok-ide)

com.programming.techie.springredditclone.model:

User.java

Post.java

Subreddit.java

Vote.java

Comment.java

VoteType.java

VerificationToken.java

NotificationEmail.java

Configure Repositories: to store these entities in the database.

com.programming.techie.springredditclone.repository:

UserRepository.java

PostRepository.java

CommentRepository.java

SubredditRepository.java

VoteRepository.java

VerificationTokenRepository.java

Run application: ./mvnw spring-boot:run

It shows:

A close-up of a computer screen

Description automatically generated

* **Part 2: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 2**

[https://programmingtechie.com/2019/10/09/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-2/#](https://programmingtechie.com/2019/10/09/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-2/)

Create Endpoint to Verify Users

* Configure Spring Security:

SecurityConfig.java

@EnableWebSecurity: enables the Web Security module

Bcrypt Algorithm (BCryptPasswordEncoder): to encode passwords.

* Implement API to register users in our application:

com.example.springredditclone.controller

AuthController.java: API to register users

com.programming.techie.springredditclone.dto:

RegisterRequest.java

* encode the password of the user, before storing them in the database:

com.programming.techie.springredditclone.services:

AuthService.java

* send out emails to the user for Account Activation:

Let us enhance the registration process by only allowing the user to log in after they verify their email.

We will generate a verification token, right after we save the user to the database and send that token as part of the verification email. Once the user is verified, then we enable the user to login to our application.

com.programming.techie.springredditclone.services:

AuthService.java, generateVerificationToken()

As we have the token, now its time to send an email that contains this verification token.

to send HTML emails, use Thymeleaf:

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

com.programming.techie.springredditclone.service:

MailContentBuilder.java: create our email message using the HTML template

src/main/resources/templates:

mailTemplate.html

com.programming.techie.springredditclone.util:

Constants.java

@UtilityClass: is a Lombok annotation, at compile time

Marks the class as final.

It generates a private no-arg constructor.

It only allows the methods or fields to be static.

Using MailTrap to send emails:

Fake SMTP server called MailTrap

Create an account in Mailtrap and after registration you get details to configure in application.properties (Mail Properties).

com.programming.techie.springredditclone.service:

MailService.java: to send out the emails

com.programming.techie.springredditclone.services:

AuthService.java, signup()

* API to verify the users and enable them:

Create Endpoint to Verify Users

com.programming.techie.springredditclone.controller: AuthController.java

com.programming.techie.springredditclone.service: AuthService.java

* **Part 3: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 3**

<https://programmingtechie.com/2019/11/08/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-3/>

API for Login functionality using the Token-Based Authentication System with the help of JWT (JSON Web Tokens).

* Authentication Flow:

A diagram of a customer service

Description automatically generated

1. So it starts with the Client sending a login request to the server.

2. The server checks the credentials provided by the user, if the credentials are right, it creates a JSON Web Token (JWT).

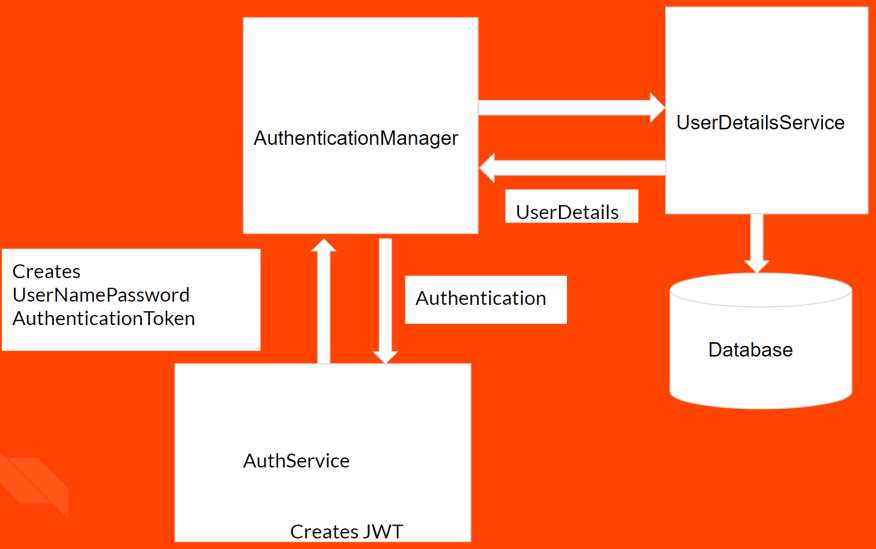
3. It responds with a success message (HTTP Status 200) and the JWT.

4. The client uses this JWT in all the subsequent requests to the user, it provides this JWT as an Authorization header with Bearer authentication scheme.

5. When the server, receives a request against a secured endpoint, it checks the JWT and validates whether the token is generated and signed by the server or not.

6. If the validation is successful, the server responds accordingly to the client.

* Spring Security Authentication Flow: authentication mechanism inside our backend (Server)



1. The login request is received by **AuthenticationController** and is passed on to the **AuthService** class.
2. This class creates an object of type **UserNamePasswordAuthenticationToken** which encapsulates the username and password provided by the user as part of the login request.
3. Then this is passed on to **AuthenticationManager** which takes care of the authentication part when using Spring Security. It implements lot of functionality in the background and provides us nice API we can use.
4. The **AuthenticationManager** further interacts with an interface called **UserDetailsService**, this interface as the name suggests deals with user data. There are several implementations that can be used depending on the kind of authentication we want. There is support for in-memory authentication, database-authentication, LDAP based authentication.
5. As we store our user information inside the Database, we used Database authentication, so the implementation access the database and retrieves the user details and passes **UserDetails** back to **AuthenticationManager**.
6. The **AuthenticationManager** now checks the credentials, and if they match it creates an object of type **Authentication** and passes it back to the **AuthService** class.
7. Then we create the JWT and respond back to the user

* Configure SecurityConfig with AuthenticationManager:

com.example.springredditclone.config: SecurityConfig.java

* Defining custom UserDetailsService class:

com.programming.techie.springredditclone.service: UserDetailsServiceImpl.java

* Define REST Endpoint for Login:

com.programming.techie.springredditclone.controller: AuthController.java

com.programming.techie.springredditclone.service: AuthService.java

* Creating JWT:

service class JWTProvider to create our JWT.

com.example.springredditclone.security: JWTProvider.java

using AsymmetricEncryption to sign our JWT’s using Java Keystore (using Public-Private Key).

created a Java Keystore and used the Private Key inside this Keystore to sign the JWT.

* **Part 4: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 4**

<https://programmingtechie.com/2019/11/30/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-4/>

validate the JWT and we will implement the API to create Subreddits.

* Implement JWT Validation:

when the client has authenticated successfully, on each subsequent request, the client provides the JWT to the server and the server should validate it.

See Authentication Flow diagram from Part 3.

A diagram of a computer

Description automatically generated

The client makes a REST call to our API, the token is sent as part of the Authorization header by following the Bearer Scheme.

The request is intercepted by the **JWTAuthenticationFilter**, which is a custom component, this filter class validates JWT, and if the token is valid, the request is forwarded to the corresponding Controller.

com.example.springredditclone.security: JwtAuthenticationFilter.java

jwtProvider.validateToken(jwt):

in Part 3 we created our JWT by signing it with the Private Key. Now we can use the corresponding Public Key, to validate the token. **Create token with Private Key, Validate token with Public Key**.

com.example.springredditclone.security: JwtProvider.java

com.example.springredditclone.config: SecurityConfig.java

* Create API for Subreddits:

to test the above JWT validation, we need a secured API: API to Create and Read Subreddits

com.example.springredditclone.controller: SubredditController.java

com.example.springredditclone.service: SubredditService.java

com.example.springredditclone.dto: SubredditDto.java

com.example.springredditclone.service: AuthService.java

* Testing the Subreddit API and JWT Validation:

Log in and get Authentication Token

Test Subreddit API without Access Token:

response: 403 Forbidden

Test Subreddit API with the access token

response: HTTP Status 200

A screenshot of a computer

Description automatically generated

* **Part 5: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 5**

<https://programmingtechie.com/2019/12/14/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-5/>

write API’s to read and create Posts

Mapstruct library: generates Java Bean Mappings

<https://mapstruct.org/>

install: <https://mapstruct.org/documentation/installation/>

Mapstruct **expressions**: <https://mapstruct.org/documentation/stable/reference/html/#expressions>

* Create Mappings:

com.example.springredditclone.mapper: SubredditMapper.java

* Use Mappers inside Spring Components:

com.example.springredditclone.service: SubredditService.java

* Implement API’s to Create and Read Posts:

create APIs to create Posts inside the Subreddits and Read those posts

com.example.springredditclone.controller: PostController.java

com.example.springredditclone.service: PostService.java

com.programming.techie.springredditclone.mapper: PostMapper.java (use Mapstruct)

* **Part 6: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 6**

<https://programmingtechie.com/2020/02/18/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-6/>

APIs to comment on the Posts created by a user.

send Comment Notification emails to the creator of the Post.

* Implementing API for Managing Comments:

APIs for posting Comments and Votes on the posts created by the user.

com.example.springredditclone.controller: CommentsController.java

com.example.springredditclone.service: CommentService.java

com.example.springredditclone.service: MailService.java

* **Part 7: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 7**

<https://programmingtechie.com/2020/02/24/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-7/>

API to submit a vote on the Posts created by the user.

refactor the post API to retrieve the vote information along with the Post Details.

* API Design for Votes:

com.example.springredditclone.controller: VoteController.java

com.example.springredditclone.service: VoteService.java

**Builder Design Pattern**

* Update Post API with Vote:

When the user queries the Post API to get a single post or all posts. We have to also provide the vote information along with post details.

com.example.springredditclone.dto: PostResponse.java

com.example.springredditclone.mapper: PostMapper.java

getDuration(): use TimeAgo library

TimeAgo library: is a java library that shows us the dates in the relative Time Ago format. <https://github.com/marlonlom/timeago>

is also a Kotlin library, dependencies: kotlin-stdlib-jdk8, kotlin-test-junit and a plugin – kotlin-maven-plugin

* **Part 8: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 8**

<https://programmingtechie.com/2020/03/10/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-8/>

API to logout/invalidation of our JSON Web Tokens (JWT)

* **How to implement logout?**

**JWT's** (**JSON Web Tokens**) are mainly used in the **authorization** flows in the web application.

The **user** first **provides** the **credentials** to the **server** and the server **responds** back to the client with a **JWT** if the credentials are correct.

So the **client** **uses** this **token** to **authorize** itself for all the subsequent **requests**. Usually, if the client is a web browser, the token is stored in a form of **browser storage**.

So the obvious thing to do is to delete this token from the browser storage as part of the Logout implementation.

**ways to implement Logout/JWT Invalidation** in our application:

1. Introduce **expiration of JWT**

we introduce expiry time for our tokens and after this time, the tokens are no longer valid. We ideally keep this expiration time short (15 minutes).

2. Store **JWT** inside **Database**

it defeats the purpose of using JWT. **JWT is by definition stateless and the advantage of using JWT is to bypass the database lookup when authorizing the client**.

3. Implement **Token Blacklisting**

store the tokens inside an in-memory database like Redis <https://redis.io/> .

When the user log’s out from the browser we delete the token and store this token inside Redis. On each user request, we perform a lookup against Redis and if the token is found inside, we throw an exception.

We can also improve the performance even more by removing the expired tokens from the Redis database.

4. Introduce **Refresh Tokens**

When the client first **authenticates**, the server provides an additional token called a **Refresh Token** (stored inside our **database**) additional to the **short-lived JWT**.

When our **JWT** is **expired** or about to be expired, we will use the **refresh token** to request a new JWT from the server.

In this way, we can keep on rotating the token until the user decides to logout from the application. Once the user logs out, we will also **delete** the **refresh token** from the **database**.

This leaves us with a very short window where the user logs out and the token is still valid.

I **recommend** using **OAuth** to implement **authentication** and **authorization** in your application. In this way, you need **not worry about how to handle user login and logout functionality**.

How to log out when using JWT

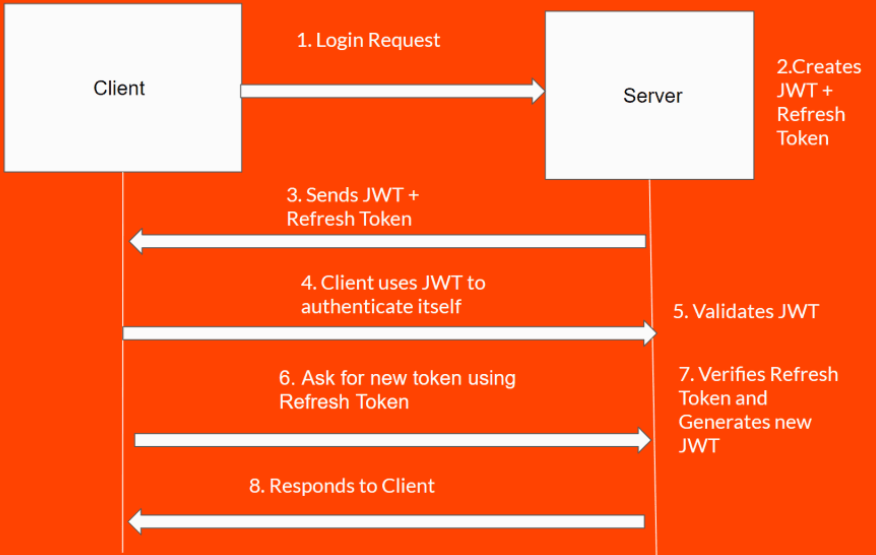
<https://medium.com/devgorilla/how-to-log-out-when-using-jwt-a8c7823e8a6>

JWT (JSON Web Tokens)

<https://jwt.io/#libraries>

* Updated Authentication/Authorization Flow:

using approach Refresh Tokens



1. In Step 2, after the authentication is successful, we send a Refresh Token along with the JWT back to the client.

2. In Step 6, the client understands that the JWT is expired or about to be expired and request the server to provide a new JWT by including the Refresh Token inside the request.

3. In Step 7, the server then verifies the Refresh Token by looking it up in the database and if it matches, generates a new JWT and responds back to the client (Step 8).

* Steps to implement:

Step 1: Introducing expiration times for our JWT

com.example.springredditclone.security: JwtProvider.java

Step 2: Implement logic to generate Refresh Tokens

com.example.springredditclone.service: RefreshTokenService.java

Step 3: login functionality includes generated Refresh Token

com.example.springredditclone.dto: AuthenticationResponse.java

com.example.springredditclone.service: AuthService.java, login(), refreshToken()

Step 4: Implement endpoints for Refresh Token and Logout.

com.example.springredditclone.controller: AuthController.java

* **Part 9: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 9**

<https://programmingtechie.com/2020/03/16/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-9/>

building our Front end application

Visual Studio code: <https://code.visualstudio.com/download>

* **Part 10: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 10**

<https://programmingtechie.com/2020/03/23/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-10/>

document the REST API using Swagger and Springfox.

* Part 11: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 11

<https://programmingtechie.com/2020/03/28/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-11/>

implement Signup functionality in our Angular application

* **Part 12: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 12**

<https://programmingtechie.com/2020/04/03/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-12/>

implement Login

handle JWT and Refresh Tokens on the client-side

* **Part 13: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 13**

<https://programmingtechie.com/2020/04/05/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-13/>

how to use refresh tokens to rotate our JWT Authentication Tokens.

* **Part 14: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 14**

<https://programmingtechie.com/2020/04/08/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-14/>

we will improve the aesthetics of our home page, and start creating Subreddits and Posts.

* **Part 15: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 15**

<https://programmingtechie.com/2020/04/16/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-15/>

to create Subreddit and Posts

* **Part 16: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 16**

<https://programmingtechie.com/2020/04/20/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-16/>

Implement Page to View Posts

Implement User Profile Page to check Posts and Comments submitted by the user.

* **Part 17: Build a Full Stack Reddit Clone with – Spring boot and Angular – Part 17**

<https://programmingtechie.com/2020/04/24/build-a-full-stack-reddit-clone-with-spring-boot-and-angular-part-17/>

Implement Voting Mechanism

Implement Logout

Protect our application routes using AuthGuards

* **Blog with Spring and Angular**

<https://programmingtechie.com/2019/03/17/how-to-build-a-simple-blog-application-using-spring-boot-and-angular/>

* **Backend**

Java 8, Spring Boot, Spring MVC, Spring Security, Spring Data JPA, MySQL

Token Based Authentication in the form of JSON Web Tokens (JWT)

* **Frontend**

Angular 7 and Bootstrap 4

* **Demo**

**A screenshot of a login page

Description automatically generated with low confidence**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

* **repository**

backend:

<https://github.com/SaiUpadhyayula/spring-ng-blog>

frontend:

<https://github.com/SaiUpadhyayula/ng-spring-blog-frontend>