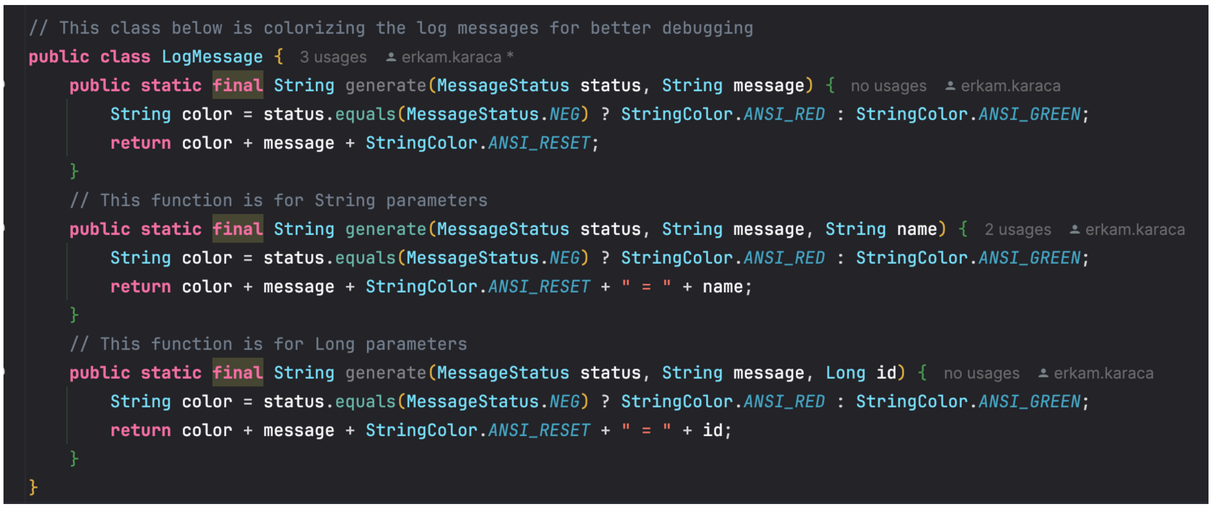
Final Project Development Journal

(This document is a journal of the development process that I made day by day.)

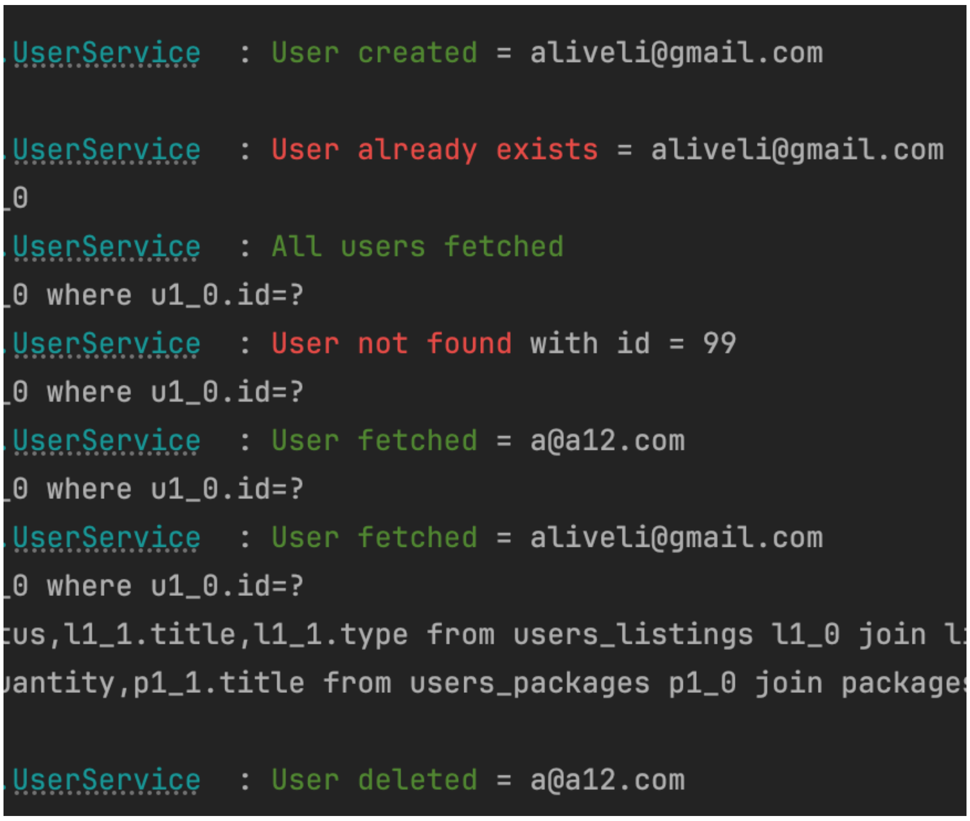
First Week

04.07.2024

LogMessage class implemented for logging and debugging purposes. This class is colorizing the errors and exceptions as red, and success messages or info as green.

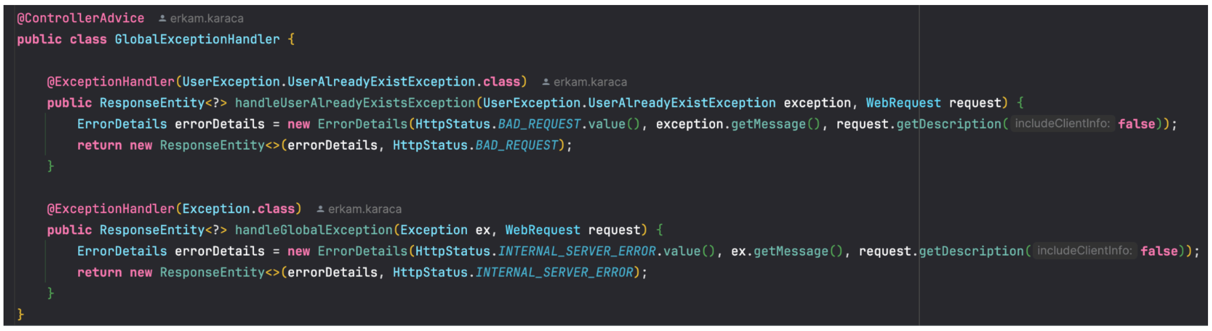


```java
// This class below is colorizing the log messages for better debugging
public class LogMessage {  3 usages  erkam.karaca *
    public static final String generate(MessageStatus status, String message) {  no usages  erkam.karaca
        String color = status.equals(MessageStatus.NEG) ? StringColor.ANSI_RED : StringColor.ANSI_GREEN;
        return color + message + StringColor.ANSI_RESET;
    }
    // This function is for String parameters
    public static final String generate(MessageStatus status, String message, String name) {  2 usages  erkam.karaca
        String color = status.equals(MessageStatus.NEG) ? StringColor.ANSI_RED : StringColor.ANSI_GREEN;
        return color + message + StringColor.ANSI_RESET + " = " + name;
    }
    // This function is for Long parameters
    public static final String generate(MessageStatus status, String message, Long id) {  no usages  erkam.karaca
        String color = status.equals(MessageStatus.NEG) ? StringColor.ANSI_RED : StringColor.ANSI_GREEN;
        return color + message + StringColor.ANSI_RESET + " = " + id;
    }
}
```



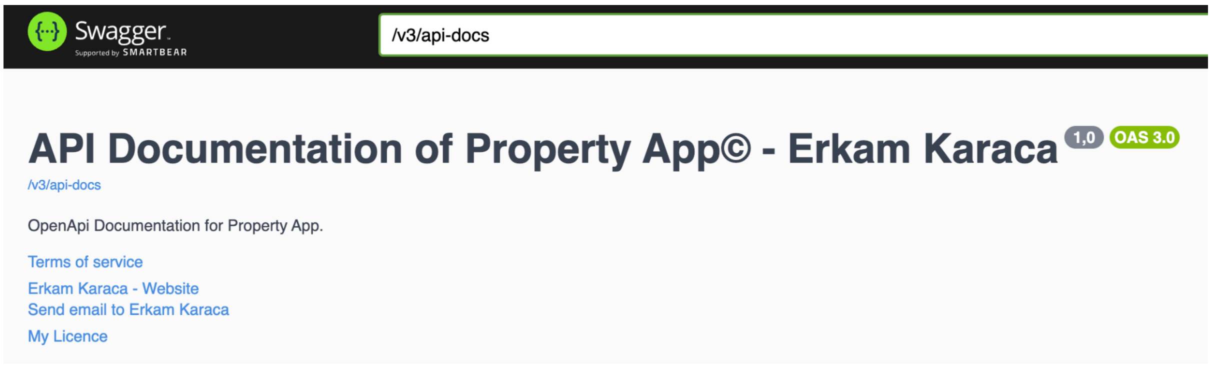
```
.UserService   : User created = aliveli@gmail.com

.UserService   : User already exists = aliveli@gmail.com
_0
.UserService   : All users fetched
_0 where u1_0.id=?
.UserService   : User not found with id = 99
_0 where u1_0.id=?
.UserService   : User fetched = a@a12.com
_0 where u1_0.id=?
.UserService   : User fetched = aliveli@gmail.com
_0 where u1_0.id=?
tus,l1_1.title,l1_1.type from users_listings l1_0 join l
antity,p1_1.title from users_packages p1_0 join package

.UserService   : User deleted = a@a12.com
```

Global Exception Handler and Error Details classes implemented.



```
@ControllerAdvice  erkam.karaca
public class GlobalExceptionHandler {

    @ExceptionHandler(UserException.UserAlreadyExistException.class)  erkam.karaca
    public ResponseEntity<?> handleUserAlreadyExistsException(UserException.UserAlreadyExistException exception, WebRequest request) {
        ErrorDetails errorDetails = new ErrorDetails(HttpStatus.BAD_REQUEST.value(), exception.getMessage(), request.getDescription( includeClientInfo: false));
        return new ResponseEntity<>(errorDetails, HttpStatus.BAD_REQUEST);
    }

    @ExceptionHandler(Exception.class)  erkam.karaca
    public ResponseEntity<?> handleGlobalException(Exception ex, WebRequest request) {
        ErrorDetails errorDetails = new ErrorDetails(HttpStatus.INTERNAL_SERVER_ERROR.value(), ex.getMessage(), request.getDescription( includeClientInfo: false));
        return new ResponseEntity<>(errorDetails, HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

SwaggerUI implemented.



API Documentation of Property App© - Erkam Karaca 1,0 OAS 3.0

/v3/api-docs

OpenApi Documentation for Property App.

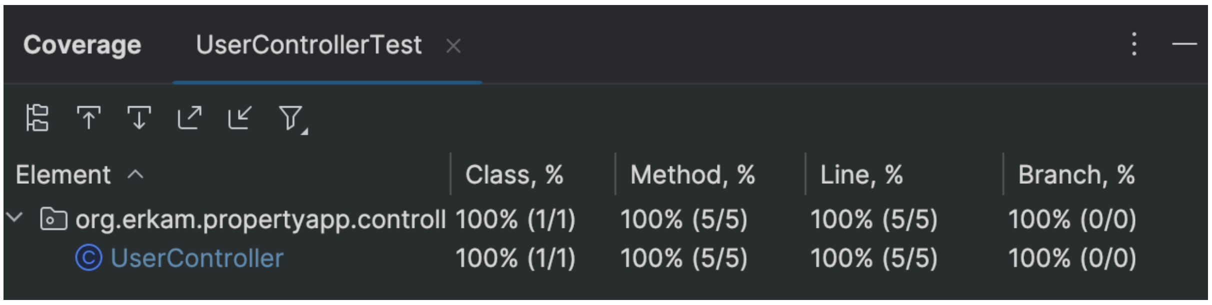
Terms of service

Erkam Karaca - Website

Send email to Erkam Karaca

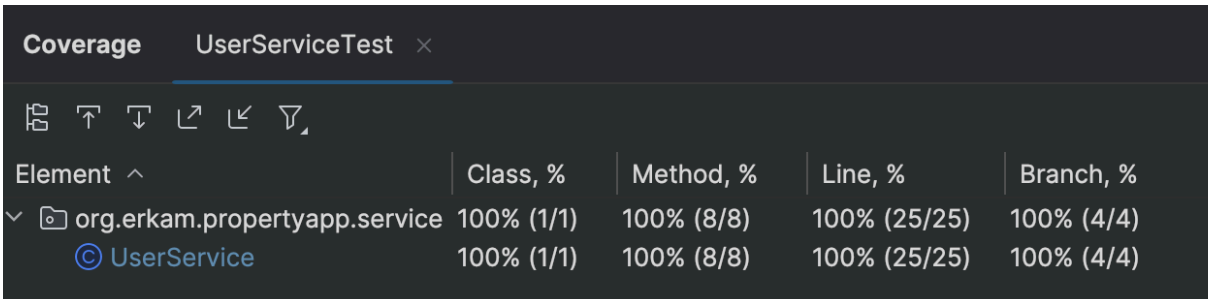
My Licence

Unit tests of User Service and User Controller added at the beginnings of the project.



Coverage UserControllerTest

Element	Class, %	Method, %	Line, %	Branch, %
org.erkam.propertyapp.controll	100% (1/1)	100% (5/5)	100% (5/5)	100% (0/0)
UserController	100% (1/1)	100% (5/5)	100% (5/5)	100% (0/0)

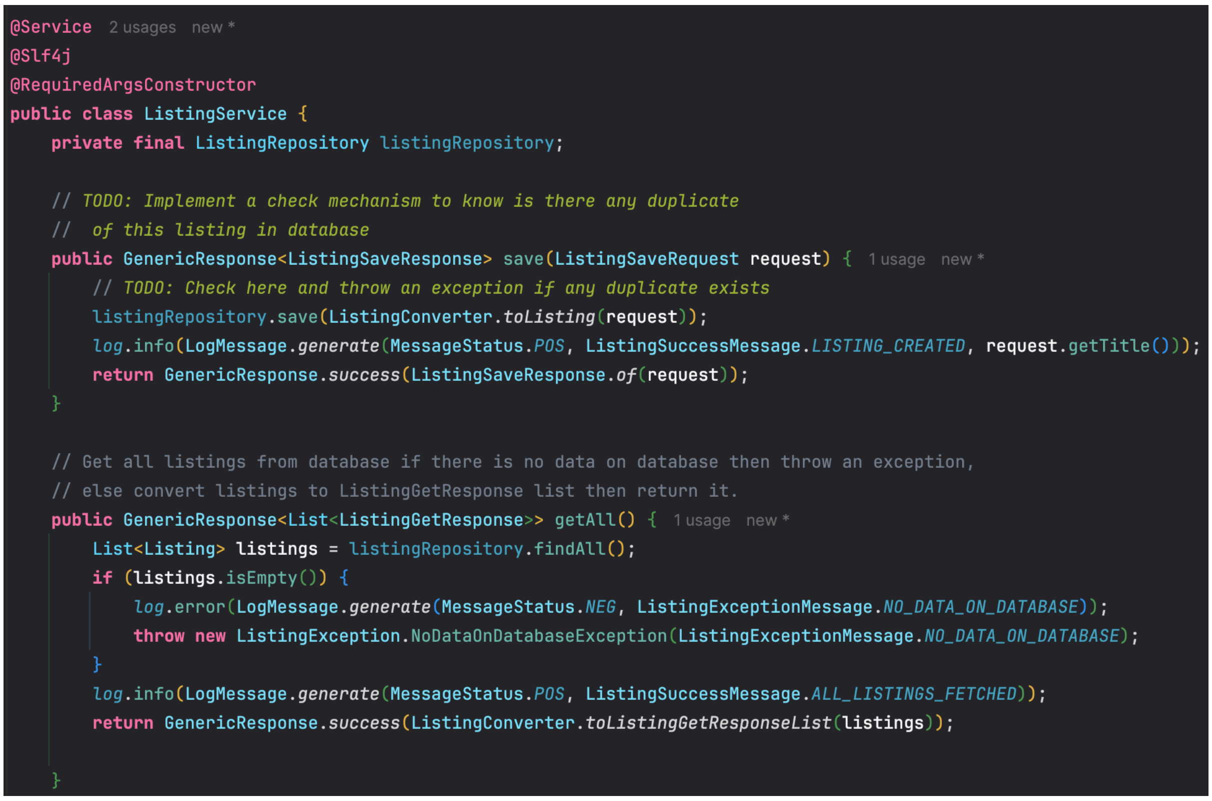


Coverage UserServiceTest

Element	Class, %	Method, %	Line, %	Branch, %
org.erkam.propertyapp.service	100% (1/1)	100% (8/8)	100% (25/25)	100% (4/4)
UserService	100% (1/1)	100% (8/8)	100% (25/25)	100% (4/4)

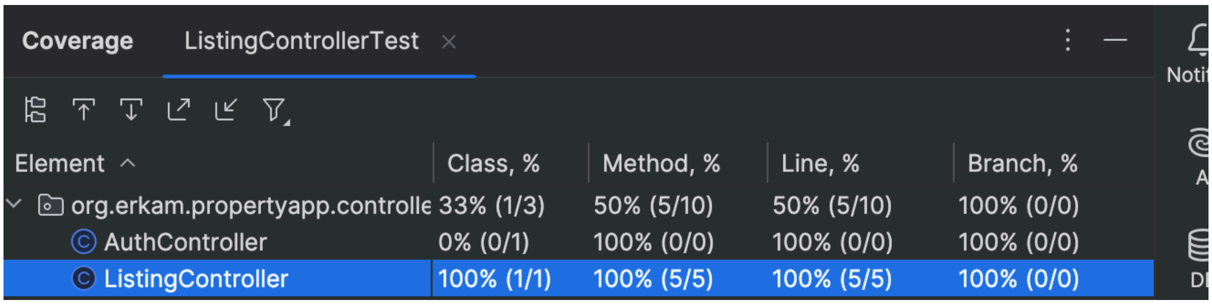
07.07.2024

Listing controller, service and repository added.

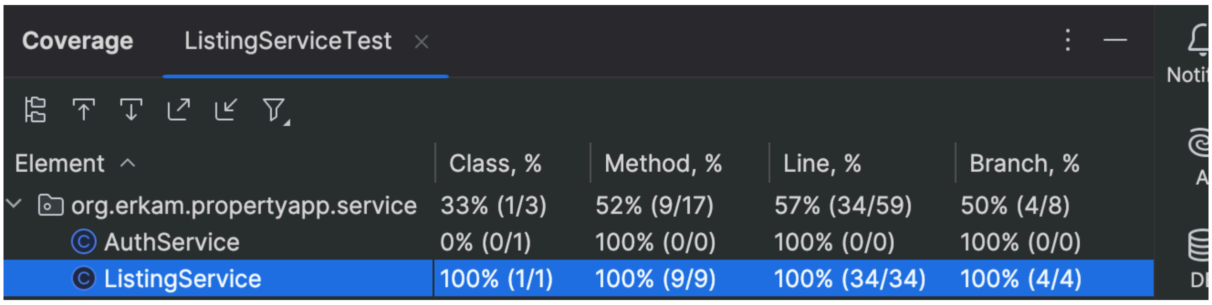


```
@Service  2 usages  new *
@Slf4j
@RequiredArgsConstructor
public class ListingService {
    private final ListingRepository listingRepository;

    // TODO: Implement a check mechanism to know is there any duplicate
    //  of this listing in database
    public GenericResponse<ListingSaveResponse> save(ListingSaveRequest request) {  1 usage  new *
        // TODO: Check here and throw an exception if any duplicate exists
        listingRepository.save(ListingConverter.toListing(request));
        log.info(LogMessage.generate(MessageStatus.POS, ListingSuccessMessage.LISTING_CREATED, request.getTitle()));
        return GenericResponse.success(ListingSaveResponse.of(request));
    }

    // Get all listings from database if there is no data on database then throw an exception,
    // else convert listings to ListingGetResponse list then return it.
    public GenericResponse<List<ListingGetResponse>> getAll() {  1 usage  new *
        List<Listing> listings = listingRepository.findAll();
        if (listings.isEmpty()) {
            log.error(LogMessage.generate(MessageStatus.NEG, ListingExceptionMessage.NO_DATA_ON_DATABASE));
            throw new ListingException.NoDataOnDatabaseException(ListingExceptionMessage.NO_DATA_ON_DATABASE);
        }
        log.info(LogMessage.generate(MessageStatus.POS, ListingSuccessMessage.ALL_LISTINGS_FETCHED));
        return GenericResponse.success(ListingConverter.toListingGetResponseList(listings));

    }
```

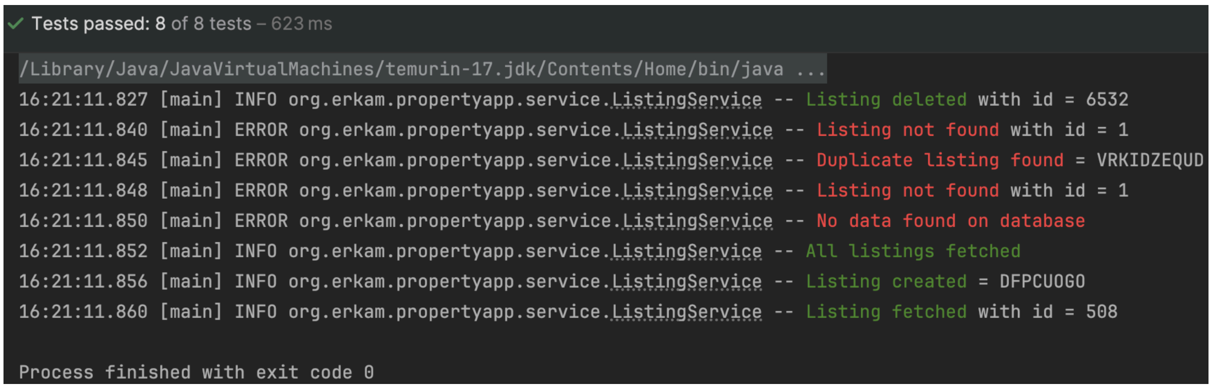
Listing controller and service unit tests added.



Element	Class, %	Method, %	Line, %	Branch, %
org.erkam.propertyapp.controlle	33% (1/3)	50% (5/10)	50% (5/10)	100% (0/0)
AuthController	0% (0/1)	100% (0/0)	100% (0/0)	100% (0/0)
ListingController	100% (1/1)	100% (5/5)	100% (5/5)	100% (0/0)



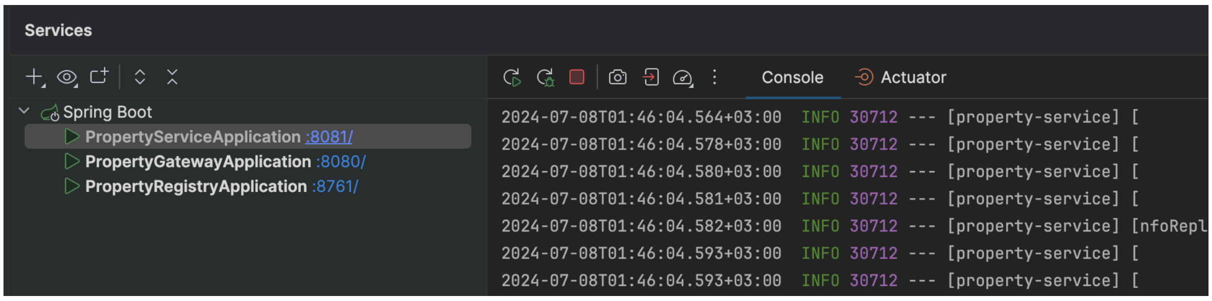
Element	Class, %	Method, %	Line, %	Branch, %
org.erkam.propertyapp.service	33% (1/3)	52% (9/17)	57% (34/59)	50% (4/8)
AuthService	0% (0/1)	100% (0/0)	100% (0/0)	100% (0/0)
ListingService	100% (1/1)	100% (9/9)	100% (34/34)	100% (4/4)



Tests passed: 8 of 8 tests – 623 ms

```
/Library/Java/JavaVirtualMachines/temurin-17.jdk/Contents/Home/bin/java ...
16:21:11.827 [main] INFO org.erkam.propertyapp.service.ListingService -- Listing deleted with id = 6532
16:21:11.840 [main] ERROR org.erkam.propertyapp.service.ListingService -- Listing not found with id = 1
16:21:11.845 [main] ERROR org.erkam.propertyapp.service.ListingService -- Duplicate listing found = VRKIDZEQUD
16:21:11.848 [main] ERROR org.erkam.propertyapp.service.ListingService -- Listing not found with id = 1
16:21:11.850 [main] ERROR org.erkam.propertyapp.service.ListingService -- No data found on database
16:21:11.852 [main] INFO org.erkam.propertyapp.service.ListingService -- All listings fetched
16:21:11.856 [main] INFO org.erkam.propertyapp.service.ListingService -- Listing created = DFPCUOGO
16:21:11.860 [main] INFO org.erkam.propertyapp.service.ListingService -- Listing fetched with id = 508

Process finished with exit code 0
```

Gateway and Registry added.

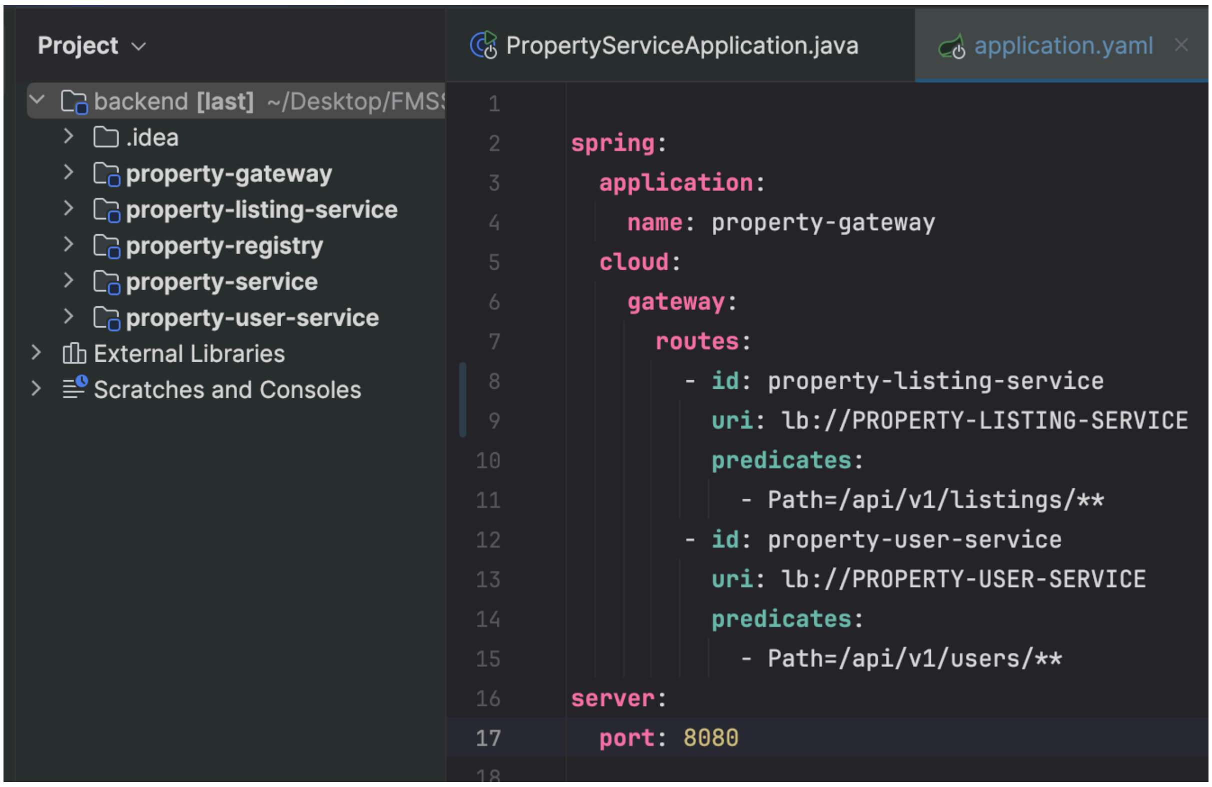


```
Spring Boot
  PropertyServiceApplication :8081/
  PropertyGatewayApplication :8080/
  PropertyRegistryApplication :8761/
```

```
2024-07-08T01:46:04.564+03:00  INFO 30712 --- [property-service] [
2024-07-08T01:46:04.578+03:00  INFO 30712 --- [property-service] [
2024-07-08T01:46:04.580+03:00  INFO 30712 --- [property-service] [
2024-07-08T01:46:04.581+03:00  INFO 30712 --- [property-service] [
2024-07-08T01:46:04.582+03:00  INFO 30712 --- [property-service] [nfoRepl
2024-07-08T01:46:04.593+03:00  INFO 30712 --- [property-service] [
2024-07-08T01:46:04.593+03:00  INFO 30712 --- [property-service] [
```

<u>08.07.2024</u>

Listing Service and User Service turned into microservices.

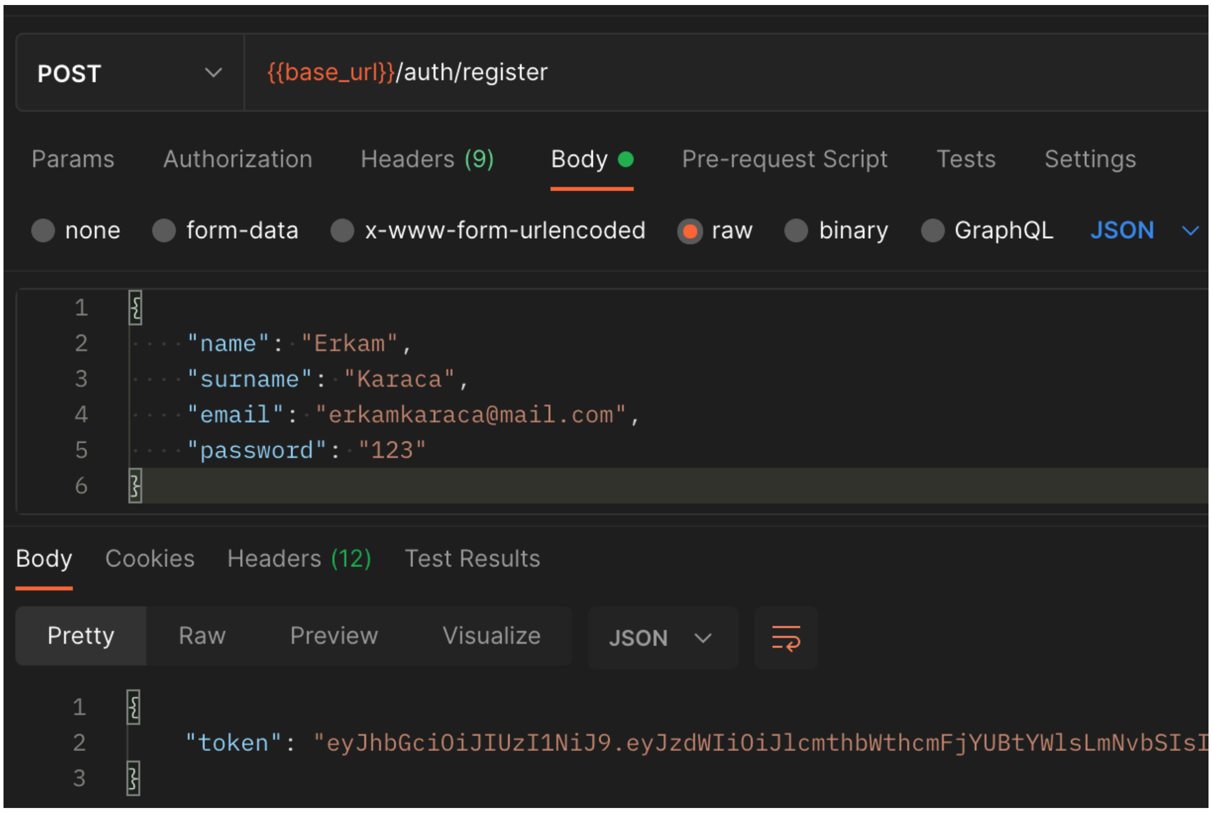


```
backend [last] ~/Desktop/FMS
  .idea
  property-gateway
  property-listing-service
  property-registry
  property-service
  property-user-service
External Libraries
Scratches and Consoles
```

```yaml

spring:
  application:
    name: property-gateway
  cloud:
    gateway:
      routes:
        - id: property-listing-service
          uri: lb://PROPERTY-LISTING-SERVICE
          predicates:
            - Path=/api/v1/listings/**
        - id: property-user-service
          uri: lb://PROPERTY-USER-SERVICE
          predicates:
            - Path=/api/v1/users/**
server:
  port: 8080
```

09.07.2024

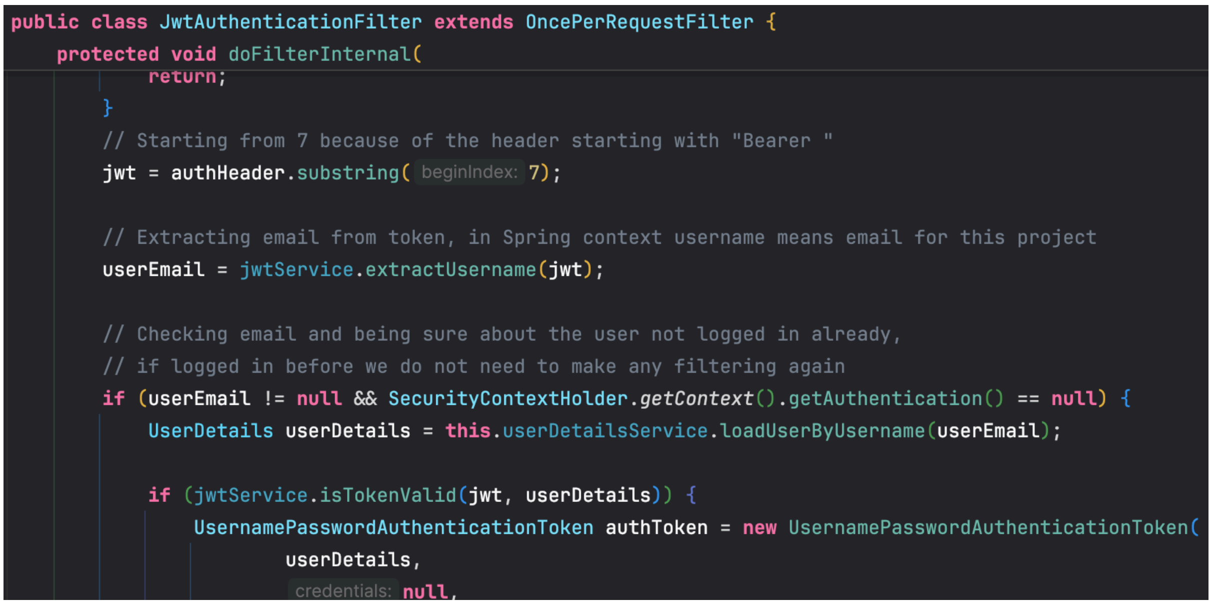
JWT Authentication implemented. Auth Controller and Auth Service added.



```
{{base_url}}/auth/register
```

```
{
    "name": "Erkam",
    "surname": "Karaca",
    "email": "erkamkaraca@mail.com",
    "password": "123"
}
```

```
{
    "token": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJlcmthbWthcmFjYUBtYWlsLmNvbSIsI
}
```



```
public class JwtAuthenticationFilter extends OncePerRequestFilter {
    protected void doFilterInternal(
            return;
        }
        // Starting from 7 because of the header starting with "Bearer "
        jwt = authHeader.substring( beginIndex: 7);

        // Extracting email from token, in Spring context username means email for this project
        userEmail = jwtService.extractUsername(jwt);

        // Checking email and being sure about the user not logged in already,
        // if logged in before we do not need to make any filtering again
        if (userEmail != null && SecurityContextHolder.getContext().getAuthentication() == null) {
            UserDetails userDetails = this.userDetailsService.loadUserByUsername(userEmail);

            if (jwtService.isTokenValid(jwt, userDetails)) {
                UsernamePasswordAuthenticationToken authToken = new UsernamePasswordAuthenticationToken(
                        userDetails,
                        credentials: null,
```