

Project

rentalapp

C:\Users\Manolov\Desktop\S3\Code review\felix-r

> .idea

> .mvn

> .vs

> src

> main

> java

> com.fm.rentalapp

> controller

AuthController

ReservationController

VehicleController

> entity

> payload

> repository

> security

> service

RentalappApplication

> resources

> test

.gitignore

mvnw

mvnw.cmd

pom.xml

rentalapp.iml

External Libraries

Scratches and Consoles

AuthController.java

ReservationController.java

VehicleController.java

```
52 JwtUtils jwtUtils;
53
54 // CODE REVIEW: Good authentication method with jwt
55
56 @PostMapping("/signin")
57 @Valid @RequestBody LoginRequest loginRequest) {
58
59     Authentication authentication = authenticationManager.authenticate(
60         new UsernamePasswordAuthenticationToken(loginRequest.getUsername(), loginRequest.getPassword()));
61
62     SecurityContextHolder.getContext().setAuthentication(authentication);
63     String jwt = jwtUtils.generateJwtToken(authentication);
64
65     UserDetailsImpl userDetails = (UserDetailsImpl) authentication.getPrincipal();
66     List<String> roles = userDetails.getAuthorities().stream()
67         .map(item -> item.getAuthority())
68         .collect(Collectors.toList());
69
70     return ResponseEntity.ok(new JwtResponse(jwt,
71         userDetails.getId(),
72         userDetails.getUsername(),
73         userDetails.getEmail(),
74         roles));
75 }
76
77 @PostMapping("/signup")
78 @Valid @RequestBody SignupRequest signUpRequest) {
79     if (userRepository.existsByUsername(signUpRequest.getUsername())) {
80         return ResponseEntity
81             .badRequest()
82             .body(new MessageResponse("Error: Username is already taken!"));
83     }
84 }
```

Project

rentalapp C:\Users\Manolov\Desktop\S3\Code review\felix-m

> .idea

> .mvn

> .vs

> src

main

java

com.fm.rentalapp

controller

AuthController

ReservationController

VehicleController

entity

payload

repository

security

service

RentalappApplication

resources

test

.gitignore

mvnw

mvnw.cmd

pom.xml

rentalapp.iml

External Libraries

Scratches and Consoles

AuthController.java

ReservationController.java

VehicleController.java

```
10
11 @CrossOrigin(origins = "http://localhost:3000")
12 @RestController
13 public class VehicleController {
14
15     @Autowired
16     private VehicleService service;
17
18     // CODE REVIEW: Authorization added to each controller
19
20     @PostMapping("/addVehicle")
21     @PreAuthorize("hasRole('STAFF') or hasRole('OWNER')")
22     public Vehicle addVehicle(@RequestBody Vehicle vehicle) { return service.saveVehicle(vehicle); }
23
24
25
26
27     @PostMapping("/addVehicles")
28     @PreAuthorize("hasRole('STAFF') or hasRole('OWNER')")
29     public List<Vehicle> addVehicles(@RequestBody List<Vehicle> vehicles) { return service.saveVehicles(vehicles); }
30
31
32
33
34     @GetMapping("/vehicles")
35     @PreAuthorize("hasRole('USER') or hasRole('STAFF') or hasRole('OWNER')")
36     public List<Vehicle> findAllVehicles() { return service.getVehicles(); }
37
38
39
40
41     @GetMapping("/vehicles/{id}")
42     @PreAuthorize("hasRole('USER') or hasRole('STAFF') or hasRole('OWNER')")
43     public Vehicle findVehicleById(@PathVariable int id) { return service.getVehicleById(id); }
44
45
46
47
48     @GetMapping("/vehicle/{name}")
49     @PreAuthorize("hasRole('USER') or hasRole('STAFF') or hasRole('OWNER')")
50     public Vehicle findVehicleByName(@PathVariable String name) { return service.getVehicleByName(name); }
51
52
53
54
55     @PutMapping("/updateVehicle")
56     @PreAuthorize("hasRole('STAFF') or hasRole('OWNER')")
57     public Vehicle updateVehicle(@RequestBody Vehicle vehicle) { return service.updateVehicle(vehicle); }
```

Project ▾

+

−

⚙

▮

▼ **rentalapp** C:\Users\Manolov\Desktop\S3\Code review\felix-r  
    > .idea  
    > .mvn  
    > .vs  
    ▼ src  
        ▼ main  
            ▼ java  
                ▼ com.fm.rentalapp  
                    ▼ controller  
                        AuthController  
                        ReservationController  
                        VehicleController  
                ▼ entity  
                    **ERole**  
                    Reservation  
                    Role  
                    User  
                    Vehicle  
                > payload  
                > repository  
                > security  
                > service  
                RentalappApplication  
            > resources  
            > test  
            .gitignore  
            mvnw  
            mvnw.cmd  
            pom.xml  
            rentalapp.iml  
    > External Libraries  
    > Scratches and Consoles

AuthController.java × ReservationController.java × Reservation.java × **ERole.java ×** Role.java × V

1 package com.fm.rentalapp.entity;  
2 //CODE REVIEW: Good practice to make the roles enums  
3 public enum ERole {  
4     ROLE\_USER,  
5     ROLE\_STAFF,  
6     ROLE\_OWNER  
7  
8  
9 }  
10

rentalapp C:\Users\Manolov\Desktop\S3\Code review\felix-r  
> .idea  
> .mvn  
> .vs  
src  
 main  
 java  
 com.fm.rentalapp  
 controller  
 AuthController  
 ReservationController  
 VehicleController  
 entity  
 ERole  
 Reservation  
 Role  
 User  
 Vehicle  
 payload  
 repository  
 security  
 service  
 RentalappApplication  
 resources  
 test  
 .gitignore  
 mvnw  
 mvnw.cmd  
 pom.xml  
 rentalapp.iml  
External Libraries  
Scratches and Consoles

```
1 package com.fm.rentalapp.entity;  
2  
3 import ...  
4  
5  
6  
7  
8  
9  
10  
11 //CODE REVIEW: Properties to fields are set good, also the relation between the tables is made good  
12  
13 @Entity  
14 @Table( name = "users",  
15         uniqueConstraints = {  
16             @UniqueConstraint(columnNames = "username"),  
17             @UniqueConstraint(columnNames = "email")  
18         })  
19 public class User {  
20     @Id  
21     @GeneratedValue(strategy = GenerationType.IDENTITY)  
22     private Long id;  
23  
24     @NotBlank  
25     @Size(max = 20)  
26     private String username;  
27  
28     @NotBlank  
29     @Size(max = 50)  
30     @Email  
31     private String email;  
32  
33     @NotBlank  
34     @Size(max = 120)  
35     private String password;  
36  
37     @ManyToMany(fetch = FetchType.LAZY)  
38     @JoinTable( name = "user_roles",  
39                 joinColumns = @JoinColumn(name = "user_id"),
```

Project

rentalapp

C:\Users\Manolov\Desktop\S3\Code review\felix-m

.idea

.mvn

.vs

src

main

java

com.fm.rentalapp

controller

AuthController

ReservationController

VehicleController

entity

ERole

Reservation

Role

User

Vehicle

payload

request

LoginRequest

SignupRequest

response

JwtResponse

MessageResponse

repository

security

service

RentalappApplication

resources

test

.gitignore

mvnw

mvnw.cmd

pom.xml

tion.java

ERole.java

Role.java

User.java

Vehicle.java

LoginRequest.java

SignupRequest.java

J

1

2

3

4

5

6

7

8

9

10

11

12

13

14

17

18

21

22

25

26

29

```
package com.fm.rentalapp.payload.request;

import javax.validation.constraints.NotBlank;

//CODE REVIEW: Good practice to make requests and responses for login and registration

public class LoginRequest {
    @NotBlank
    private String username;

    @NotBlank
    private String password;

    public String getUsername() { return username; }

    public void setUsername(String username) { this.username = username; }

    public String getPassword() { return password; }

    public void setPassword(String password) { this.password = password; }
}
```



Project

java

com.fm.rentalapp

controller

AuthController

ReservationController

VehicleController

entity

ERole

Reservation

Role

User

Vehicle

payload

request

LoginRequest

SignupRequest

response

JwtResponse

MessageResponse

repository

security

jwt

AuthEntryPointJwt

AuthTokenFilter

JwtUtils

services

UserDetailsImpl

UserDetailsServiceImpl

WebSecurityConfig

service

RentalappApplication

resources

test

.gitignore

myenv

Role.java

User.java

Vehicle.java

LoginRequest.java

SignupRequest.java

AuthTokenFilter.java

JwtResponse.java

MessageResponse.java

1

2

3

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

```
package com.fm.rentalapp.security.jwt;

import ...

public class AuthTokenFilter extends OncePerRequestFilter {
    @Autowired
    private JwtUtils jwtUtils;

    //CODE REVIEW: Nice filter but lack of comments

    @Autowired
    private UserDetailsServiceImpl userDetailsService;

    private static final Logger logger = LoggerFactory.getLogger(AuthTokenFilter.class);

    @Override
    protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain filterChain)
        throws ServletException, IOException {
        try {
            String jwt = parseJwt(request);
            if (jwt != null && jwtUtils.validateJwtToken(jwt)) {
                String username = jwtUtils.getUserNameFromJwtToken(jwt);

                UserDetails userDetails = userDetailsService.loadUserByUsername(username);
                UsernamePasswordAuthenticationToken authentication = new UsernamePasswordAuthenticationToken(
                    userDetails, credentials: null, userDetails.getAuthorities());
                authentication.setDetails(new WebAuthenticationDetailsSource().buildDetails(request));

                SecurityContextHolder.getContext().setAuthentication(authentication);
            }
        } catch (Exception e) {
            logger.error("Cannot set user authentication: {}", e);
        }
    }
}
```

Project

entity

ERole

Reservation

Role

User

Vehicle

payload

request

LoginRequest

SignupRequest

response

JwtResponse

MessageResponse

repository

security

jwt

AuthEntryPointJwt

AuthTokenFilter

JwtUtils

services

UserDetailsImpl

UserDetailsServiceImpl

WebSecurityConfig

service

RentalappApplication

resources

test

java

com.fm.rentalapp

service

ReservationServiceUnitTest

VehicleServiceUnitTest

RentalappApplicationTests

.gitignore

mvnw

SignupRequest.java

AuthTokenFilter.java

UserDetailsImpl.java

UserDetailsServiceImpl.java

WebSecurityConfig.java

ReservationServiceUnitTest.java

```
1 package com.fm.rentalapp.service;
2
3 import ...
4
19
20 @ExtendWith(MockitoExtension.class)
21 @ExtendWith(SpringExtension.class)
22 class ReservationServiceUnitTest {
23
24     //CODE REVIEW: All service test passes
25
26     @MockBean
27     private ReservationRepository reservationRepository;
28
29     @InjectMocks
30     private ReservationService reservationService;
31
32     @Test
33     void saveReservation() {
34         Reservation reservation = new Reservation();
35         reservation.setUserId(Long.valueOf(0));
36         reservation.setVehicleId(0);
37         reservation.setVehicleName("Audi");
38         reservation.setPrice(200);
39         Mockito.when(reservationRepository.save(reservation)).thenReturn(reservation);
40         Reservation savedReservation = reservationService.saveReservation(reservation);
41         assertEquals(reservation, savedReservation);
42
43     }
44
45     @Test
46     void saveReservations() {
47         Reservation reservation = new Reservation();
48         reservation.setUserId(Long.valueOf(0));
```

- Reservation
- Role
- User
- Vehicle
- payload
  - request
    - LoginRequest
    - SignupRequest
  - response
    - JwtResponse
    - MessageResponse
- repository
- security
  - jwt
    - AuthEntryPointJwt
    - AuthTokenFilter
    - JwtUtils
  - services
    - UserDetailsImpl
    - UserDetailsServiceImpl
  - WebSecurityConfig
- service
- RentalappApplication
- resources
- test
  - java
    - com.fm.rentalapp
      - service
        - ReservationServiceUnitTest
        - VehicleServiceUnitTest
        - RentalappApplicationTests

```
1 package com.fm.rentalapp;
2
3 import ...
4
5
6 @SpringBootTest
7 class RentalappApplicationTests {
8
9     //CODE REVIEW: Lack of controller test and unit test
10
11     @Test
12     void contextLoads() {
13     }
14
15 }
16
```

- .gitignore
- mvnw
- mvnw.cmd



EXPLORER

...

JS auth-header.jsJS auth.service.jsJS reservation.service.jsJS user.service.jsJS vehicle.service.js

> OPEN EDITORS1 UNSAVED

EINDHOVEN-RENT-FRONT-E...

> node\_modules

> public

> src

components

JS AddVehicle.js

JS BoardAdmin.js

JS Home.js

JS Login.js

JS ManageVehicles.js

JS Profile.js

JS Register.js

JS ReservationList.js

JS UpdateVehicle.js

JS VehicleList.js

JS ViewVehicle.js

services

JS auth-header.js

JS auth.service.js

JS reservation.service.js

JS user.service.js

JS vehicle.service.js

# App.css

JS App.js

JS App.test.js

# index.css

JS index.js

logo.svg

JS reportWebVitals.js

JS setupTests.js

OUTLINE

TIMELINE

src > services > JS auth.service.js > ...

1import axios from "axios";

2

3const Backend\_URL = "http://localhost:8080/api/auth/";

4

5// CODE REVIEW: Good practice to have separate services for performing requests

6

7const register = (username, email, password) =>

8{

9| return axios.post(Backend\_URL + "signup", {username, email, password});

10};

11

12const login = (username, password) =>

13{

14| return axios

15| .post(Backend\_URL + "signin", {username,password,})

16| .then((response) => {

17| | if (response.data.accessToken)

18| | {

19| | | localStorage.setItem("user", JSON.stringify(response.data));

20| | }

21| | return response.data;

22| }

23});

24

25const logout = () => {

26| localStorage.removeItem("user");

27

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS C:\Users\Manolov\Desktop\S3\Code review\felix-morenc-s3-web-based-car-rental-app\reactjs\eindhoven-rent-front-end>

EXPLORER

> OPEN EDITORS 2 UNSAVED

EINDHOVEN-RENT-FRONT-E...

> node\_modules

> public

> src

> components

JS AddVehicle.js

JS BoardAdmin.js

JS Home.js

JS Login.js

JS ManageVehicles.js

JS Profile.js

JS Register.js

JS ReservationList.js

JS UpdateVehicle.js

JS VehicleList.js

JS ViewVehicle.js

> services

JS auth-header.js

JS auth.service.js

JS reservation.service.js

JS user.service.js

JS vehicle.service.js

# App.css

JS App.js

JS App.test.js

# index.css

JS index.js

logo.svg

JS reportWebVitals.js

JS setupTests.js

JS auth-header.js

JS auth.service.js

JS VehicleList.js

JS reservation.service.js

JS user.service.js

JS vehicle.service.js

src > components > JS VehicleList.js > VehicleComponent > constructor

1 import React, { useState, useEffect } from "react";

2

3 import VehicleService from "../services/vehicle.service";

4

5 //CODE REVIEW: Lifecycle methods are properly used

6

7 class VehicleComponent extends React.Component

8 {

9 constructor(props)

10 {

11 super(props)

12 this.state = {vehicles: []}

13 }

14

15 componentDidMount()

16 {

17 VehicleService.getVehicles().then((response) =>

18 {

19 this.setState({vehicles: response.data})

20 })

21 }

22

23 viewVehicle(id)

24 {

25 this.props.history.push(`/view-vehicle/\${id}`);

26 }

27 }

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

1: powershell

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS C:\Users\Manolov\Desktop\S3\Code review\felix-morenc-s3-web-based-car-rental-app\reactjs\eindhoven-rent-front-end>