

Authentication

To check identity of user

Validating user

userid+pwd

emailid+pwd

otp

emailid+otp

dynamic link

JPA Repo

Mongo Repo

authentication

op1

implement auth
mechanism
in all applications
individually

op2

implement one
centralized auth
mechanism, let all apps
use the same

Authorization

After authentication

Checks ROLE/Design/Permission

Allows to access a particular resource

id+pwd

RESTAPI
server

Authentication

auth

DB

raj@gmail.com
abc@gmail.com

12345

12345

xyz

RESTAPI
cleint

inbox

DB

raj@gmail.com
abc@gmail.com

m1,m2,m3

m4,m5,m6

xyz

RESTAPI
cleint

sent

DB

raj@gmail.com
abc@gmail.com

s1,s2,s3

s4,s5,s6

xyz

RESTAPI
cleint

profile

DB

raj@gmail.com
abc@gmail.com

Rajeswar theam1

Anand theam2

xyz

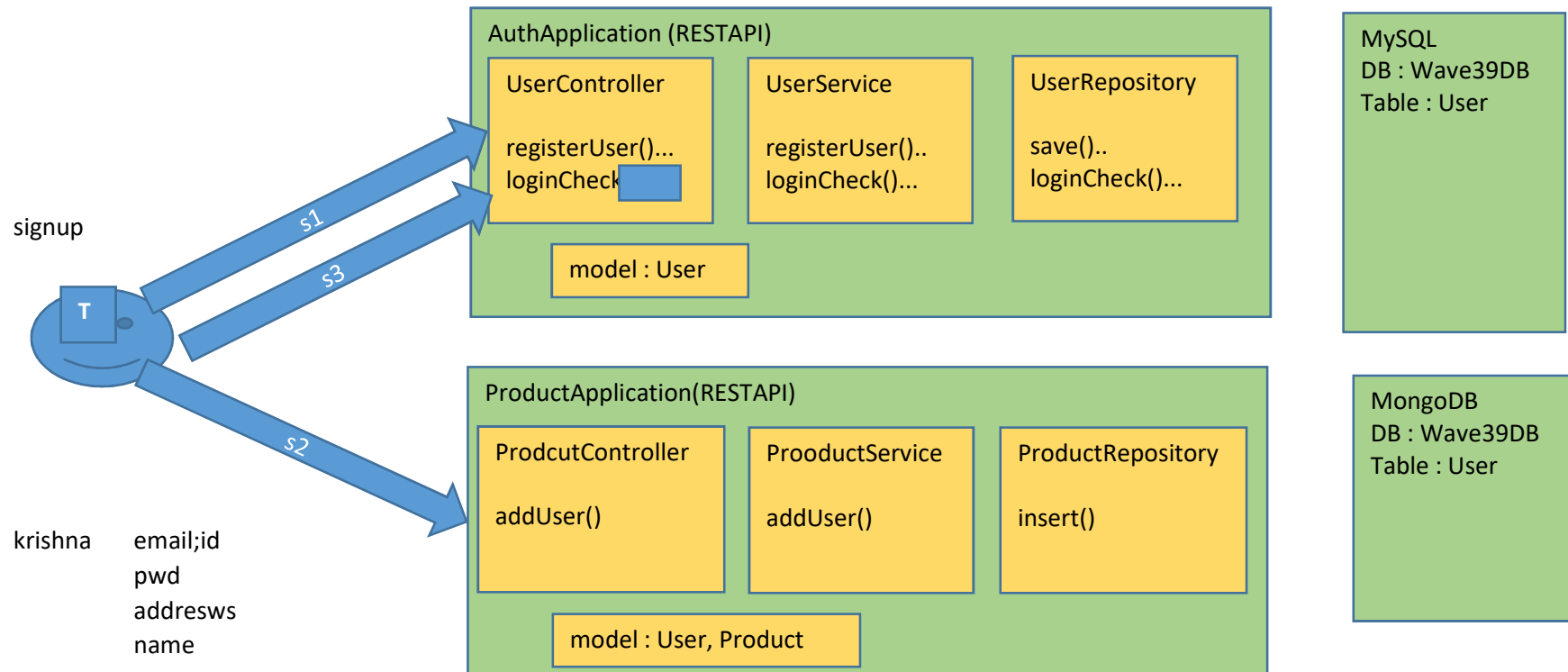


client
app

How to implement manual logincheck mechanism

capture userid+pwd
 send the same to BE
 BE checks whether the data matching with DB record or not
 if matching : login is success
 else : login failed

emailid *	password	role
rajeswar@gmail.com	12345	ROLE_USER
anand@gmail.com	12345	ROLE_USER
krsishna@gmail.com	12345	ROLE_USER



signup data

	emailid	pwd	name	address	product	name	desc	price
authapp	emailid	pwd						
productapp	emailid	name	address					

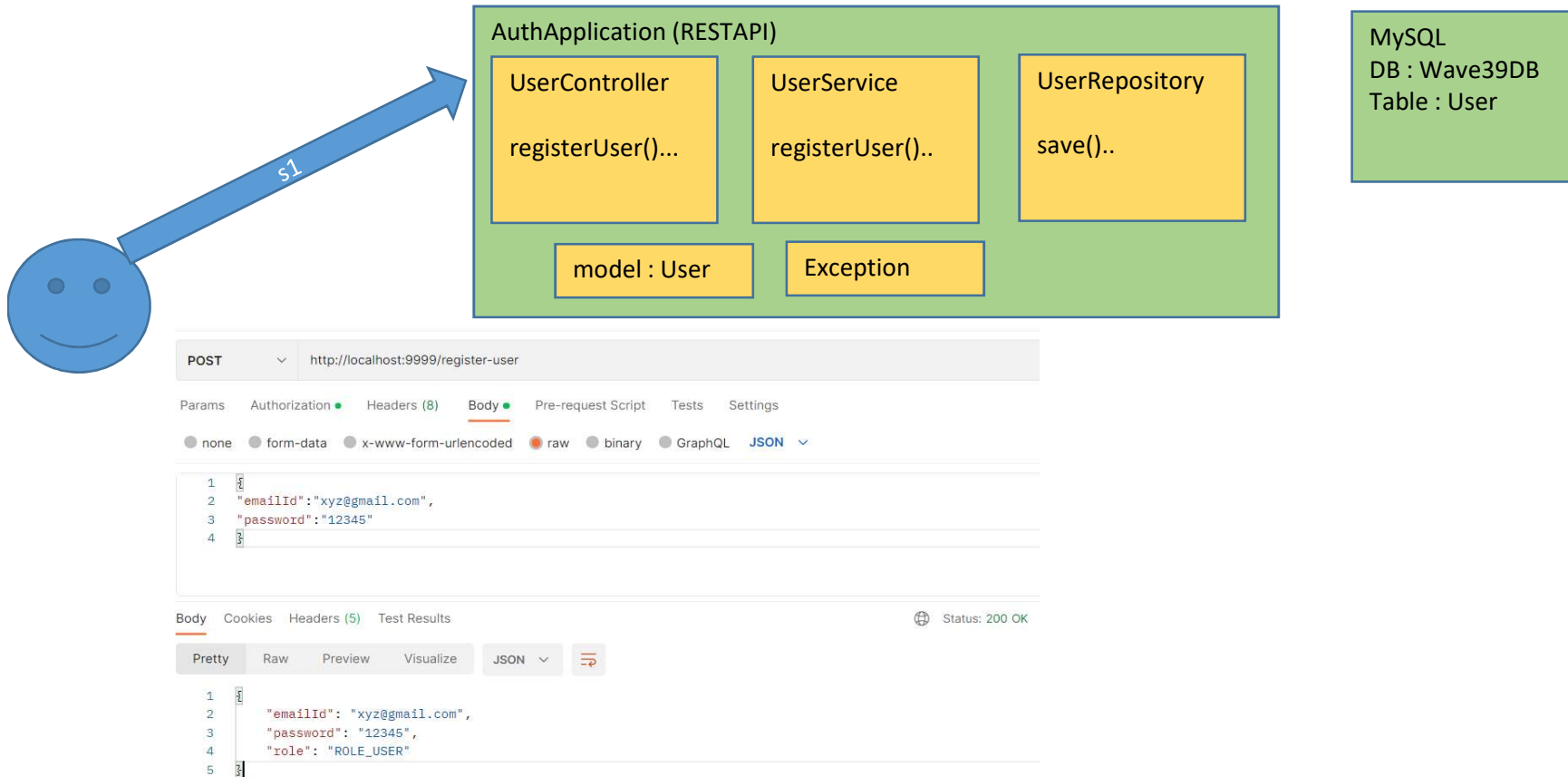
emailid	name	products	s1	creating user a/c in auth app	auth app
rajeswar@gmail.com	rajeswar	[]	s2	creating user a/c in product app	product app
anand@gmail.com	anand	[]	s3	login check	auth app
krishna@gmail.com	krishna	[]	s4	access product data with token	

S1 creating user account in auth app

create new springboot application (web,mysql,jpa,lombok)

model : emailid, pwd, role

create flow for creating user a/c in db



S2 creating user account in product app

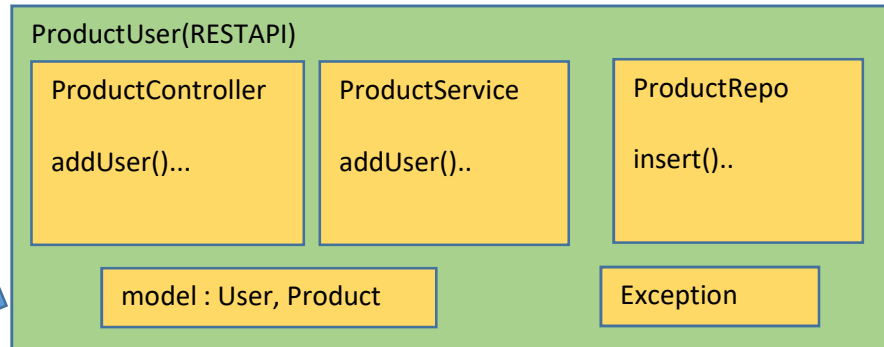
create new springboot application (web,mongo data,lombok)

model : emailid, name, address,product[]

create flow for creating user a/c in db



S2



MongoDB
DB : Wave39DB
Table : User

POST http://localhost:8888/add-user

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

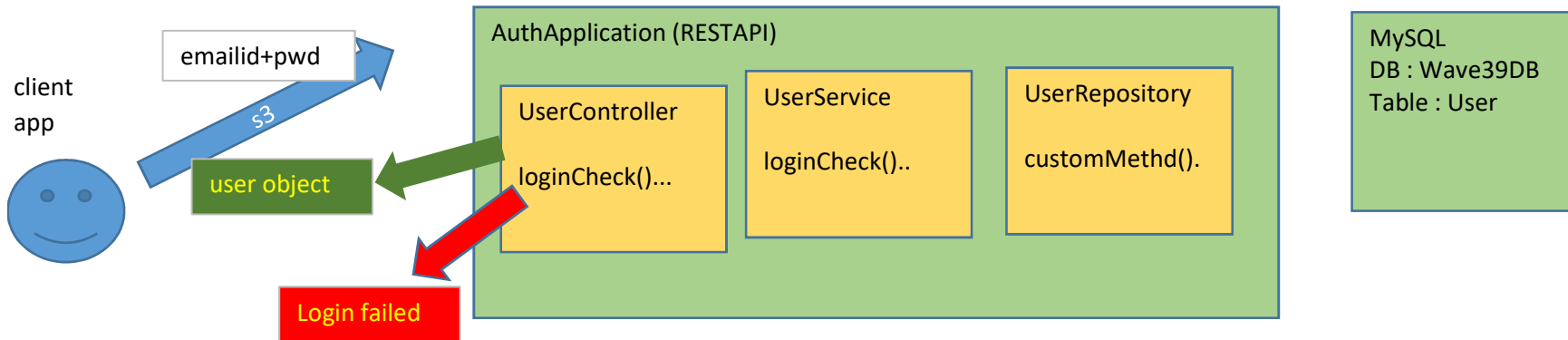
```
1 {
2   "emailId": "xyz@gmail.com",
3   "name": "xyz",
4   "address": "chennai"
5 }
```

Body Cookies Headers (5) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON

```
1 {
2   "emailId": "xyz@gmail.com",
3   "name": "xyz",
4   "address": "chennai",
5   "products": []
6 }
```

S3 logincheck in auth app



MySQL
DB : Wave39DB
Table : User

POST http://localhost:9999/login-check

Params Authorization Headers (8) Body Pre-request Scri

none form-data x-www-form-urlencoded raw binary

```
1 [
2   "emailId": "xyz@gmail.com",
3   "password": "12345"
4 ]
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   "emailId": "xyz@gmail.com",
3   "password": "",
4   "role": "ROLE_USER"
5 ]
```

POST http://localhost:9999/login-check

Params Authorization Headers (8) Body Pre-request S

none form-data x-www-form-urlencoded raw binary

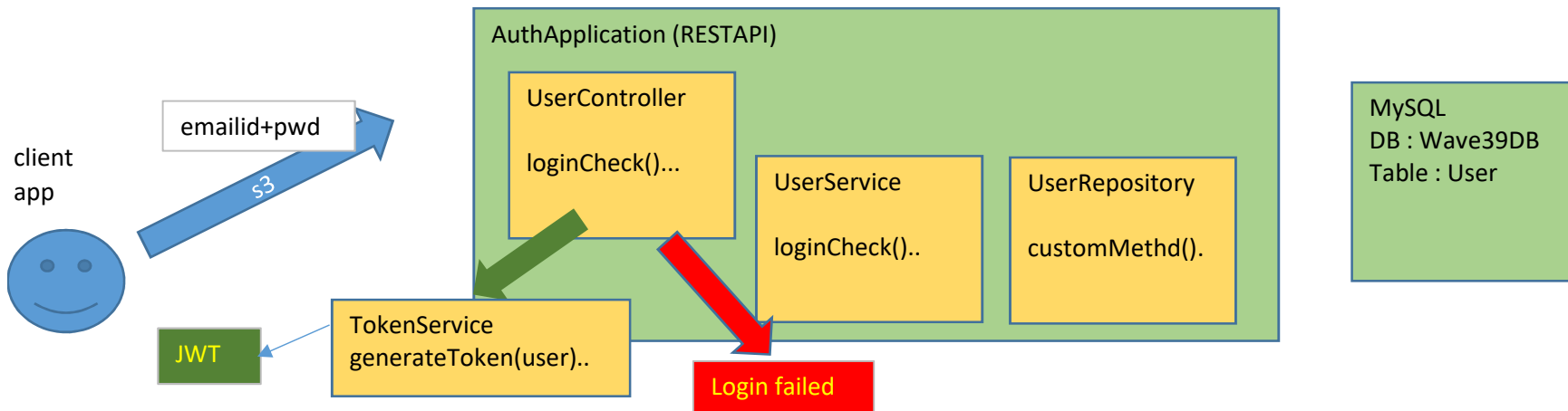
```
1 [
2   "emailId": "xyz11@gmail.com",
3   "password": "12345"
4 ]
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize Text

```
1 Login failed
```

User object is returned as response if login is success



JWT to be given as response if login is success

step 1 add required dependency

Search results for **jjwt** on Maven Central:

Found 31 results

Sort: **relevance** | popular | newest

- 1. JJWT :: API**
 io.jsonwebtoken » jjwt-api
 JJWT :: API
 Last Release on Apr 28, 2022
- 2. JSON Web Token Support For The JVM**
 io.jsonwebtoken » jjwt
 JSON Web Token Support For The JVM
 Last Release on Jul 5, 2018

```
<!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt</artifactId>
  <version>0.9.1</version>
</dependency>
```

```
<!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt</artifactId>
  <version>0.9.1</version>
</dependency>
```

Step 2 Define service layer

```
7 public interface JwtTokenGenerator {
8     public abstract Map<String,String> generateJwt(User user);
9 }

12 @Service
13 public class JwtTokenGeneratorImpl implements JwtTokenGenerator {
14
15     @Override
16     @
17     public Map<String, String> generateJwt(User user) {
18         Map<String, String> result = new HashMap<String, String>();
19         // build token ( header,payload, by compacting)
20         // user has emailid, password, role
21         Map<String,Object> userdata = new HashMap<>();
22         userdata.put("emailid",user.getEmailId());
23         userdata.put("role",user.getRole());
24
25         String jwt = Jwts.builder()
26             .setClaims(userdata) // user data to be filled as claims
27             .setIssuedAt(new Date())
28             .setIssuer("MyCompany")
29             .signWith(SignatureAlgorithm.HS512, s: "idontsay")
30             .compact();
31         result.put("token",jwt);
32         result.put("message","Login success, Token generated");
33         return result;
34         // return map : token(key,val), message(key,val)
35     }
}
```

Step 3 in controller

add tokengenerator service dependency
logincheck()
call token service
get token
give response as token

```
19 @Autowired
20 private JwtTokenGenerator jwtTokenGenerator;
```




Algorithm HS512

Encoded PASTE A TOKEN HERE

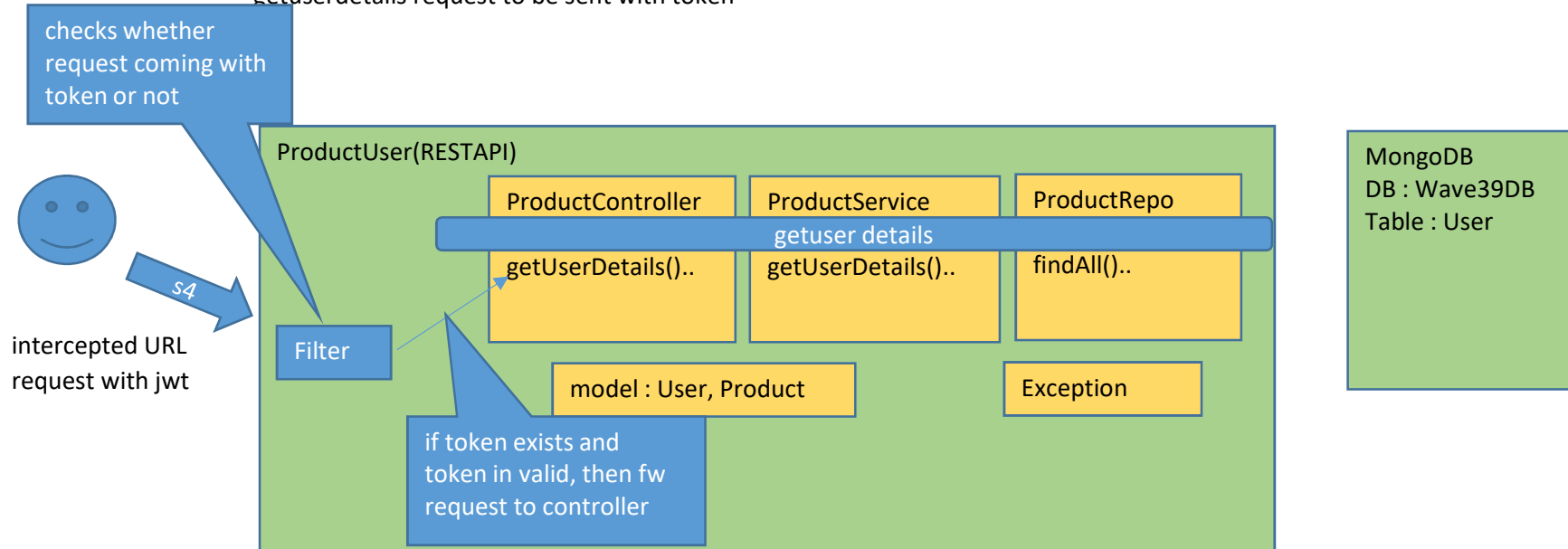
```
eyJhbGciOiJIUzUxMiJ9.eyJyb2xlIjoiUk9MRV
9VU0VSIiwiaXNzIjoib21wYW55IiwiaW1ha
WxpZCI6Inh5ekBnbWFpbC5jb20iLCJpYXQiOiE2
NzI2NTk5MDd9.ARWpGu2htUEtxUIg0Lq0AFEN8_
fKQH2rhG3YJdjIYV5DvZkSbzkbJ1F0qgx2bfB_
ooj5AD09MofojhZfX-XQ
```

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE
<pre>{ "alg": "HS512" }</pre>
PAYLOAD: DATA
<pre>{ "role": "ROLE_USER", "iss": "MyCompany", "emailid": "xyz@gmail.com", "iat": 1672659907 }</pre>
VERIFY SIGNATURE
<pre>HMACSHA512(base64UrlEncode(header) + "." + base64UrlEncode(payload), your-256-bit-secret) <input type="checkbox"/> secret base64 encoded</pre>

S4 accessing intercepted URL resource with header included request

productuser app
getuserdetails request to be sent with token



Stage 1

check for token, forward request to controller if token exists and token is valid

step a define flow for get user details
 service layer
 controller layer

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:8888/get-user-details
- Authorization:** No Auth
- Body:** JSON

The response body is displayed in the 'Body' tab, showing a JSON array of three user objects:

```
1  [
2    {
3      "emailId": "xyz@gmail.com",
4      "name": "xyz",
5      "address": "chennai",
6      "products": []
7    },
8    {
9      "emailId": "raju@gmail.com",
10     "name": "raju",
11     "address": "pune",
12     "products": []
13   },
14   {
15     "emailId": "krishna@gmail.com",
16     "name": "krishna",
17     "address": "Delhi",
18     "products": []
19   }
20 ]
```

http://localhost:8888/get-user-details

the above request should be forwarded to controller only after checking for JWT in request

above URL to be made as intercepted URL

step b define filter class

check whether request carrying authheader or not

if carrying, check whether token is valid or not

if valid:forward to controller

else: throw exception

add required dependencies

```
<!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt</artifactId>
  <version>0.9.1</version>
</dependency>

<!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->
<dependency>
  <groupId>javax.xml.bind</groupId>
  <artifactId>jaxb-api</artifactId>
  <version>2.4.0-b180830.0359</version>
</dependency>
```

```
40 <!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->
41 <dependency>
42   <groupId>io.jsonwebtoken</groupId>
43   <artifactId>jjwt</artifactId>
44   <version>0.9.1</version>
45 </dependency>
46 <!-- https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api -->
47 <dependency>
48   <groupId>javax.xml.bind</groupId>
49   <artifactId>jaxb-api</artifactId>
50   <version>2.4.0-b180830.0359</version>
51 </dependency>
```

```

13 import java.io.IOException;
14 public class JwtFilter extends GenericFilterBean {
15     @Override
16     public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain)
17         throws IOException, ServletException {
18         // downcast req, response
19         // check authheader in req
20         // get token from authheader
21         // try to parse token : checking whether token valid or not based on shared key (idontsay)
22         // parsing gets failed if token is invalid(tampered/expired) / key mismatch
23         // if not throwing any exception (parsing is success): forward request to next level
24         HttpServletRequest request = (HttpServletRequest)servletRequest;
25         HttpServletResponse response = (HttpServletResponse)servletResponse;
26
27         String authHeader = request.getHeader("authorization");
28         // if authheader is not existing OR not carrying Bearer type of token : throw excep saying token missing
29         if(authHeader==null || !authHeader.startsWith("Bearer") ){
30             throw new ServletException("Token is missing");
31         }
32         else{ // authHeader existing and carrying the Bearer token
33             // authHeader : Bearer sdfsdgsdgsd.fsdgsdgsd.fsdgsdgsd
34             String token = authHeader.substring(7);
35             System.out.println("\nIn Filter class, Token : " + token);
36             Claims claims=Jwts.parser().setSigningKey("idontsay").parseClaimsJws(token).getBody();
37             // above parsing fails if key is wrong/token expires/ token tampered : throw exception
38             // if above parsing success, claims variable gets user details
39             System.out.println("\nIn Filter, claims : " + claims);
40             filterChain.doFilter(request,response); // forwarding
41         }
42     }
43 }

```

step c Enable filtering on URLs (then URLs become intercepted URLs)
in main class

```

16 @Bean
17 public FilterRegistrationBean filterBean(){
18     // which URLs to be Intercepted, by using which Filter definition
19     FilterRegistrationBean frb = new FilterRegistrationBean();
20     frb.setFilter(new JwtFilter());
21     frb.addUrlPatterns("/get-user-details");
22     return frb;
23 }

```

run applicatoin

how to test

try to access intercepted url

without authorization header

header with other than bearer data

bearer with wrong token

GET http://localhost:8888/get-user-details

Params Authorization Headers (6) Body Pre-request Script Tests

Type No Auth

This request will be

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2023-01-03T06:03:30.028+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "Token is missing",
6   "path": "/get-user-details"
7 }
```

GET http://localhost:8888/get-user-details

Params Authorization Headers (7) Body Pre-request Script Tests

Type Basic Auth Username Password

The authorization header will be

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2023-01-03T06:03:57.089+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "Token is missing",
6   "path": "/get-user-details"
7 }
```

GET http://localhost:8888/get-user-details

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Type Bearer Token this is my token

The authorization header will be

Body Cookies Headers (4) Test Results Status: 500 Internal Server Error Time

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2023-01-03T06:04:30.884+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "message": "JWT strings must contain exactly 2 period characters. Found: 0",
6   "path": "/get-user-details"
7 }
```

above screens showing exception as this URL request needs valid token
perform S3, login check
get token
perform S4 with token

S3

POST ▼ http://localhost:9999/login-check Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Coo

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼ Beau

```
1 {
2   "emailId": "krishna@gmail.com",
3   "password": "12345"
4 }
```

Body Cookies Headers (5) Test Results 🌐 Status: 200 OK Time: 24 ms Size: 441 B Save Response

Pretty Raw Preview Visualize **JSON** ▼ ☰

```
1 {
2   "message": "Login success, Token generated",
3   "token": "eyJhbGciOiJIUzUxMiJ9.eyJyY2xiIjoiUk9MRV9VU0VSIlwiaXNzIjoiTXlDb21wYW55IiwiaWZwIjoiYXNobmFAZ21haWwY29tIiwiaWF0IjoxNjc5NzI2MDkxZjQ.
4   jnuZwkyNordgVOnld7ESw1BZl1HNb9vI2ia3qZcl4aazcMPe-6sNiJfk9lgB1Ehpx-QdpsFb4u5BjfCh6wQtA"
}
```

copy token

and paste in S4, authorization header, bearer

GET ▼ http://localhost:8888/get-user-details

Params **Authorization** Headers (7) Body Pre-request Script Tests Settings

Type Bearer Token ▼ Token

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

```
eyJhbGciOiJIUzUxMiJ9.eyJyY2xiIjoiUk9MRV9VU0VSIlwiaXNzIjoiTXlDb21wYW55IiwiaWZwIjoiYXNobmFAZ21haWwY29tIiwiaWF0IjoxNjc5NzI2MDkxZjQ.
jnuZwkyNordgVOnld7ESw1BZl1HNb9vI2ia3qZcl4aazcMPe-6sNiJfk9lgB1Ehpx-QdpsFb4u5BjfCh6wQtA
```

GET

http://localhost:8888/get-user-details

Params

Authorization

Headers (7)

Body

Pre-request Script

Tests

Settings

Type

Bearer To...

Token

eyJhbGciOiJIUzUxMiJ9.eyJyYb2xlljoiUk9MR\...

The authorization header will be

Body

Cookies

Headers (5)

Test Results

Status: 200 OK Time: 27 ms Size: 395 E

Pretty

Raw

Preview

Visualize

JSON

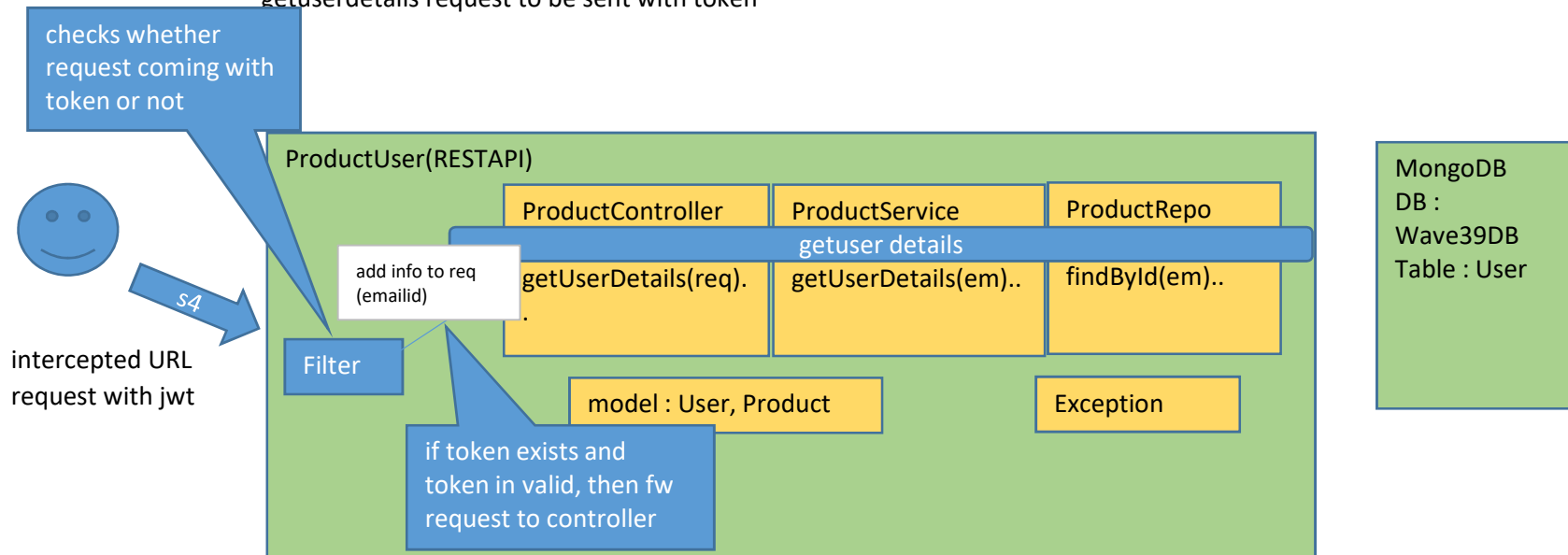
```
1  [
2    {
3      "emailId": "xyz@gmail.com",
4      "name": "xyz",
5      "address": "chennai",
6      "products": []
7    },
8    {
9      "emailId": "raju@gmail.com",
10     "name": "raju",
11     "address": "pune",
12     "products": []
13   },
14   {
15     "emailId": "krishna@gmail.com",
16     "name": "krishna",
17     "address": "Delhi",
18     "products": []
19   }
20 ]
```


Stage 2

Get only currently logged in user associated data

productuser app

getuserdetails request to be sent with token



Modify JWT filter

get emailid from claims, add as attribute to request before forwarding to controller

```
39 System.out.println("\nIn Filter, claims : " + claims);
40 request.setAttribute("curr_user_emailid", claims.get("emailid"));
41 filterChain.doFilter(request, response); // forwarding
```

edit service layer

```
//public abstract List<User> getUserDetails();
1 usage 1 implementation 1 related problem
public abstract User getUserDetails(String eid);
```

```

26 // @Override
27 // public List<User> getUserDetails() {
28 //     return productRepository.findAll();
29 // }
30
31 // 1 usage 1 related problem
32 @Override
33 public User getUserDetails(String eid) {
34     return productRepository.findById(eid).get();
35 }

```

edit controller

```

42 @GetMapping("/get-user-details")
43 @ public ResponseEntity<> getUserDetails(HttpServletRequest request){
44     //return new ResponseEntity<>(userService.getUserDetails(),HttpStatus.OK);
45     String current_user_emailId=(String)request.getAttribute("curr_user_emailid");
46     return new ResponseEntity<>(userService.getUserDetails(current_user_emailId),HttpStatus.OK);
47 }

```

GET
http://localhost:8888/get-user-details

Params
Authorization
Headers (7)
Body
Pre-request Script
Test

Type
Bearer Token
Token

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Body
Cookies
Headers (5)
Test Results

Pretty
Raw
Preview
Visualize
JSON

1
2
3
4
5
6
{
 "emailId": "raju@gmail.com",
 "name": "raju",
 "address": "pune",
 "products": []
}