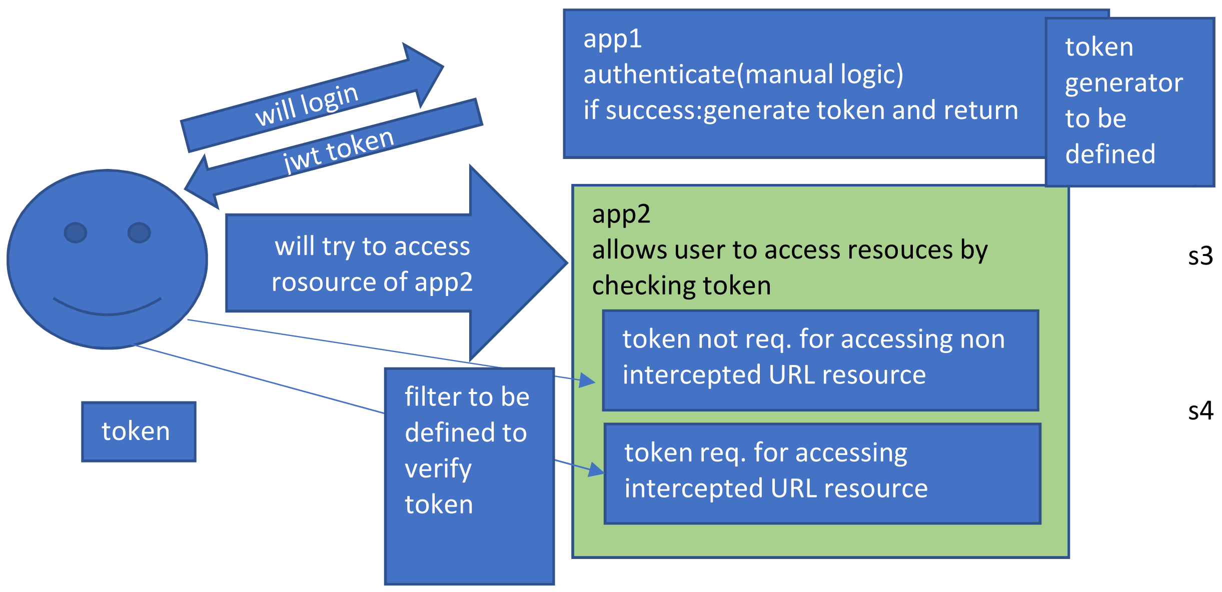

will login
jwt token
will try to access rosource of app2
token
filter to be defined to verify token
app1
authenticate(manual logic)
if success:generate token and return
token generator to be defined
app2
allows user to access resouces by checking token
token not req. for accessing non intercepted URL resource
token req. for accessing intercepted URL resource

user tries to login
with id+pwd

gets token (auth is success)

s3 user tries to access
app2 resources
by adding token

s4 app2 checks token
and gives reponse

Authentication

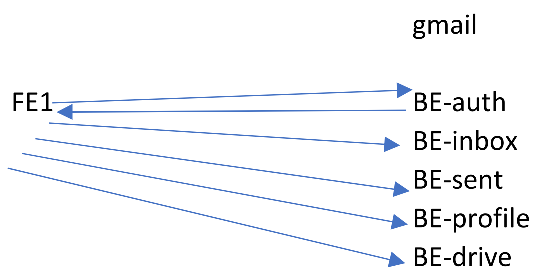
checking user details before entering to premises

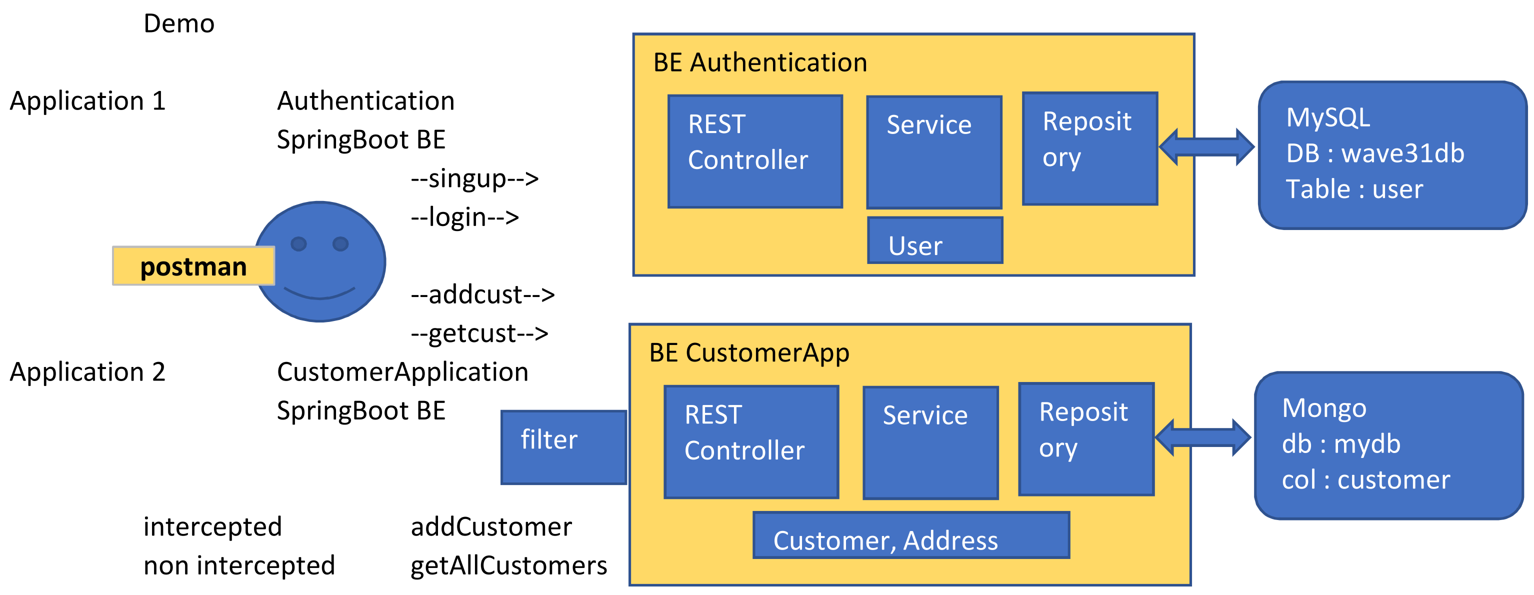
id+pwd / smart card / biometric / otp / URL click / .. . .

Authorization

after login,

allowing / not allowing to access particular resources as per user ROLE





What Application1 has to do

new user can signup/register

existing user signin/login

jwt token to be returned as response if login authentication is success

What Application2 has to do

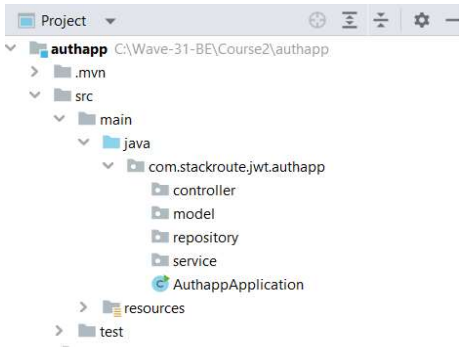
request to be filterd for jwt if request comes for intercepted url, then provide the resource

Steps to make Application 1 (authentcation)

1 create new spring boot application

web
jpa
mysql lombok

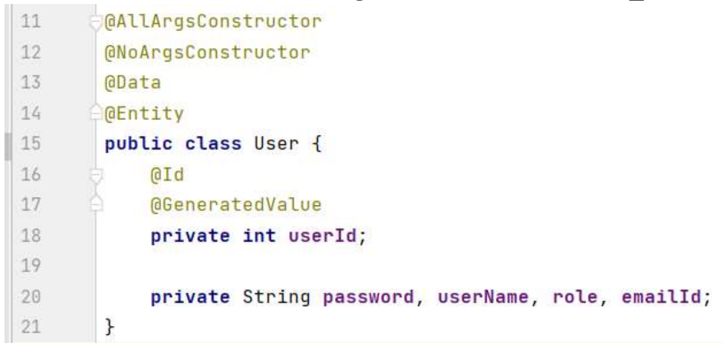
download application, extract to workspace and open in intellij
create required packages



```
Project
authapp  C:\Wave-31-BE\Course2\authapp
  .mvn
  src
    main
      java
        com.stackroute.jwt.authapp
          controller
          model
          repository
          service
          AuthappApplication
      resources
    test
```

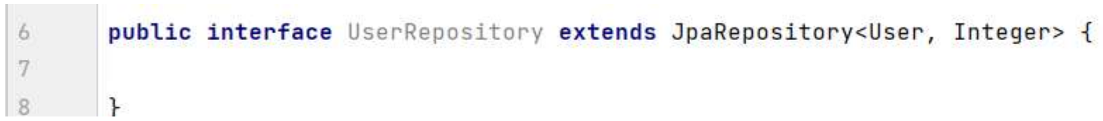
2 create user model class

*userid	int	autogen
username	string	
emailid	string	
password	string	
role	string	ROLE_USER



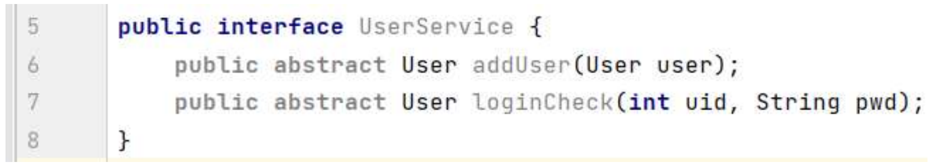
```
@AllArgsConstructor
@NoArgsConstructor
@Data
@Entity
public class User {
    @Id
    @GeneratedValue
    private int userId;

    private String password, userName, role, emailId;
}
```

3 create User repository

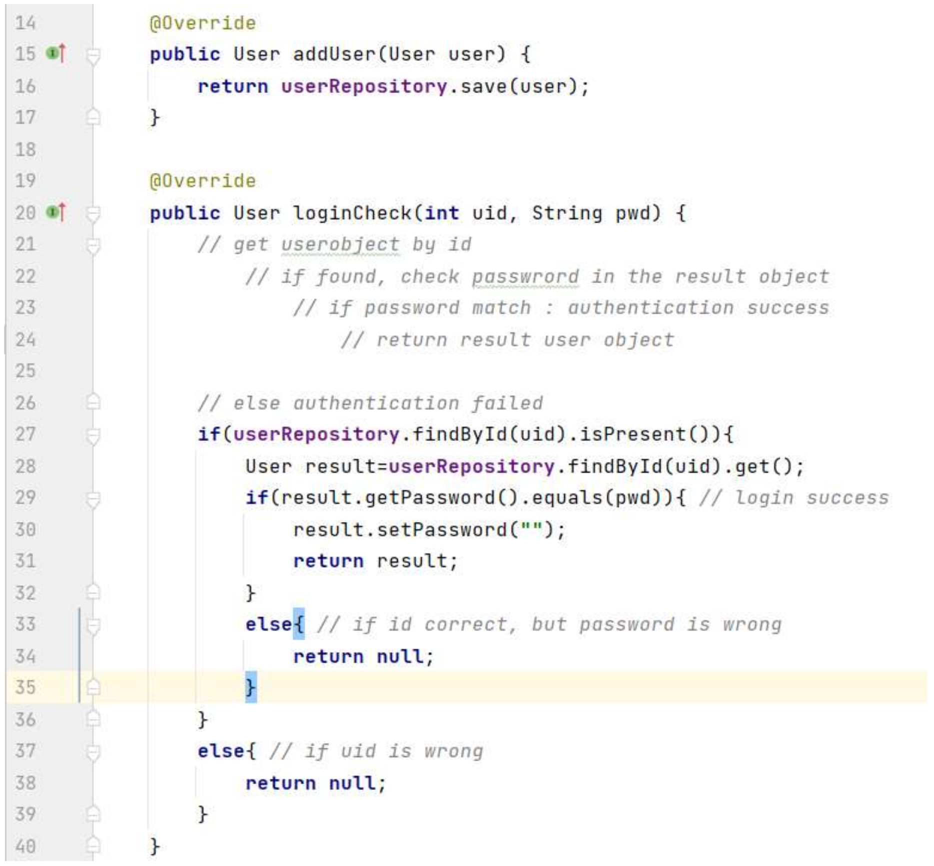


```java
public interface UserRepository extends JpaRepository<User, Integer> {

}
```

4 create service layer



```java
public interface UserService {
    public abstract User addUser(User user);
    public abstract User loginCheck(int uid, String pwd);
}
```



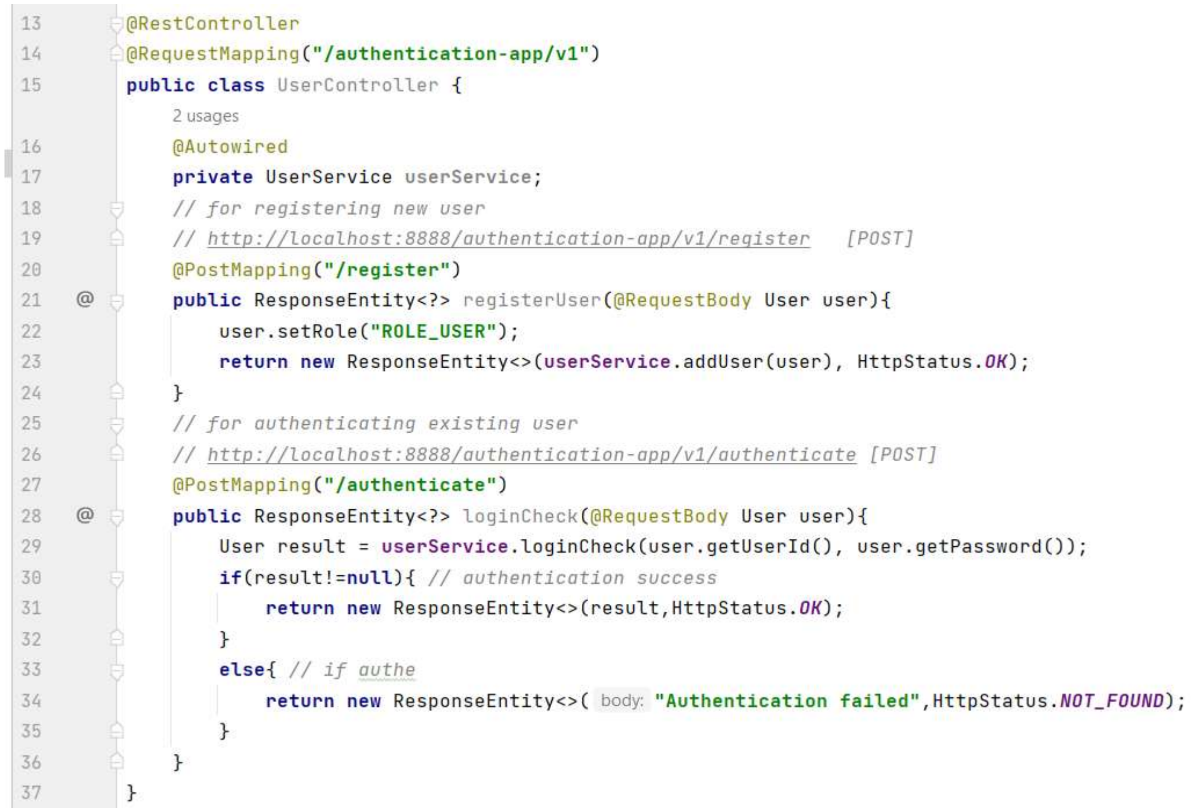
```java
    @Override
    public User addUser(User user) {
        return userRepository.save(user);
    }

    @Override
    public User loginCheck(int uid, String pwd) {
        // get userobject by id
            // if found, check passwrord in the result object
                // if password match : authentication success
                    // return result user object

        // else authentication failed
        if(userRepository.findById(uid).isPresent()){
            User result=userRepository.findById(uid).get();
            if(result.getPassword().equals(pwd)){ // login success
                result.setPassword("");
                return result;
            }
            else{ // if id correct, but password is wrong
                return null;
            }
        }
        else{ // if uid is wrong
            return null;
        }
    }
```

5 create controller

define request handler metods for two tasks

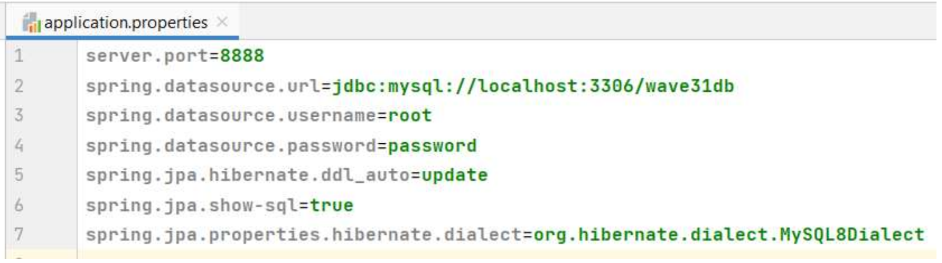
register/signup for new user

logincheck for existing user



```java
@RestController
@RequestMapping("/authentication-app/v1")
public class UserController {
    2 usages
    @Autowired
    private UserService userService;
    // for registering new user
    // http://localhost:8888/authentication-app/v1/register   [POST]
    @PostMapping("/register")
    public ResponseEntity<?> registerUser(@RequestBody User user){
        user.setRole("ROLE_USER");
        return new ResponseEntity<>(userService.addUser(user), HttpStatus.OK);
    }
    // for authenticating existing user
    // http://localhost:8888/authentication-app/v1/authenticate [POST]
    @PostMapping("/authenticate")
    public ResponseEntity<?> loginCheck(@RequestBody User user){
        User result = userService.loginCheck(user.getUserId(), user.getPassword());
        if(result!=null){ // authentication success
            return new ResponseEntity<>(result,HttpStatus.OK);
        }
        else{ // if authe
            return new ResponseEntity<>( body: "Authentication failed",HttpStatus.NOT_FOUND);
        }
    }
}
```

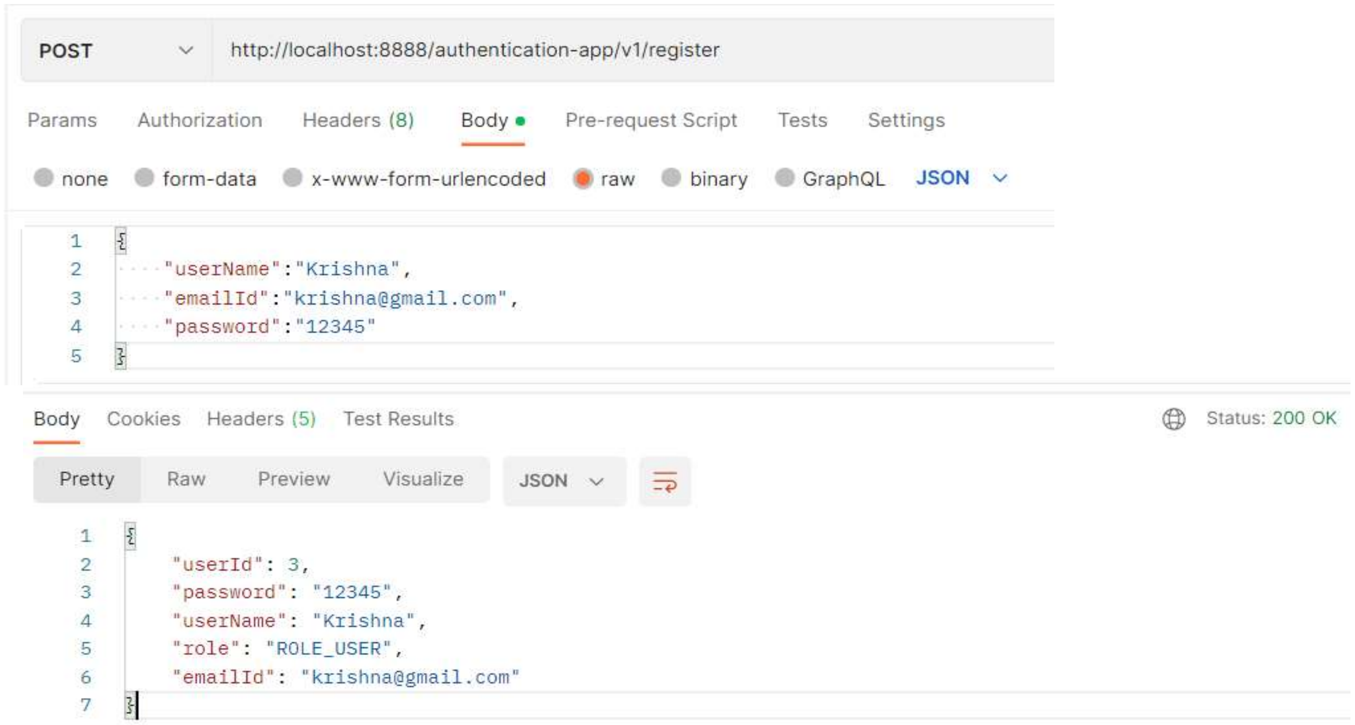
6 edit application properties



application.properties

```
server.port=8888
spring.datasource.url=jdbc:mysql://localhost:3306/wave31db
spring.datasource.username=root
spring.datasource.password=password
spring.jpa.hibernate.ddl_auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
```

Register process in postman

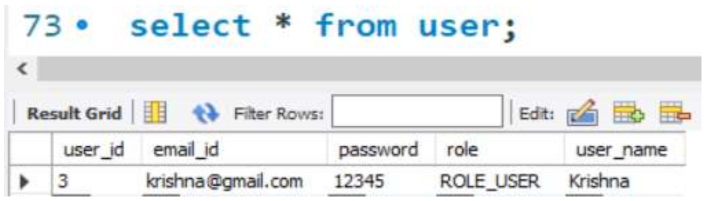

POST http://localhost:8888/authentication-app/v1/register

Body: raw, JSON

```
{
    "userName":"Krishna",
    "emailId":"krishna@gmail.com",
    "password":"12345"
}
```

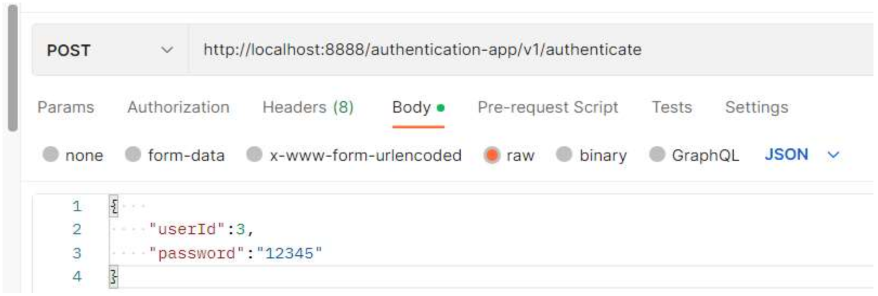
Status: 200 OK

```
{
    "userId": 3,
    "password": "12345",
    "userName": "Krishna",
    "role": "ROLE_USER",
    "emailId": "krishna@gmail.com"
}
```


```
73 •  select * from user;
```

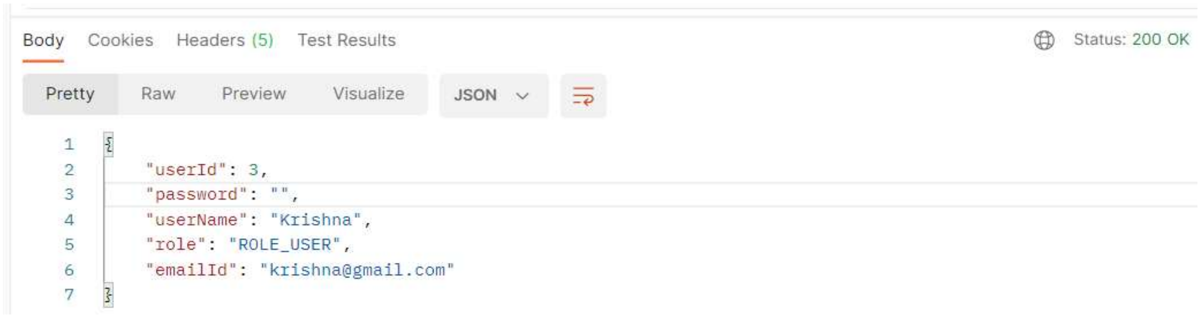
user_id	email_id	password	role	user_name
3	krishna@gmail.com	12345	ROLE_USER	Krishna

login check process in postman

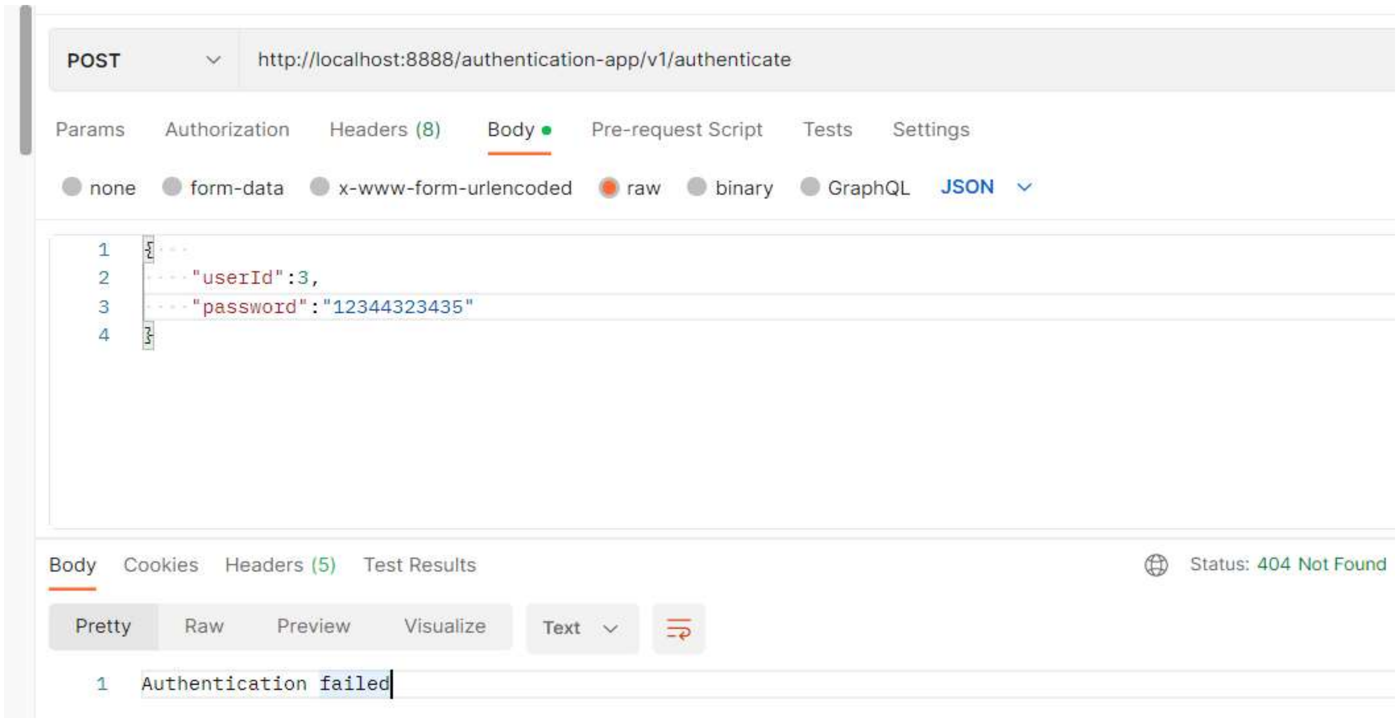

POST http://localhost:8888/authentication-app/v1/authenticate

Body: raw, JSON

```
{
    "userId":3,
    "password":"12345"
}
```



```
{
    "userId": 3,
    "password": "",
    "userName": "Krishna",
    "role": "ROLE_USER",
    "emailId": "krishna@gmail.com"
}
```



```
POST http://localhost:8888/authentication-app/v1/authenticate
```

```
{
    "userId":3,
    "password":"12344323435"
}
```

```
Authentication failed
```

add JWT generating logic if login success case

return response as JWT token

controller
login check () {

authenticating by using
userservice.loginCheck();

generate token by using
generatejwtservice.generate()

}

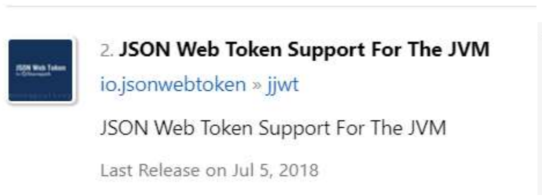
services package
UserService (interface + class)
GenerateJwtService(interface + class)

7 Define Generate JWT token service

define login to generate token based on received user object

add dependency in pom

io.jsonwebtoken

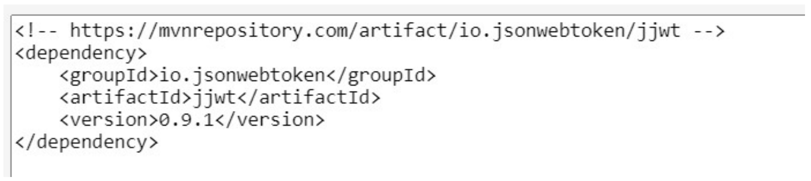


2. **JSON Web Token Support For The JVM**

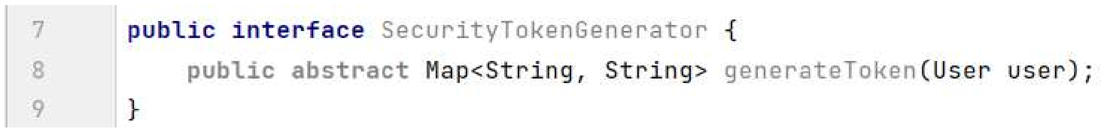
io.jsonwebtoken » jjwt

JSON Web Token Support For The JVM

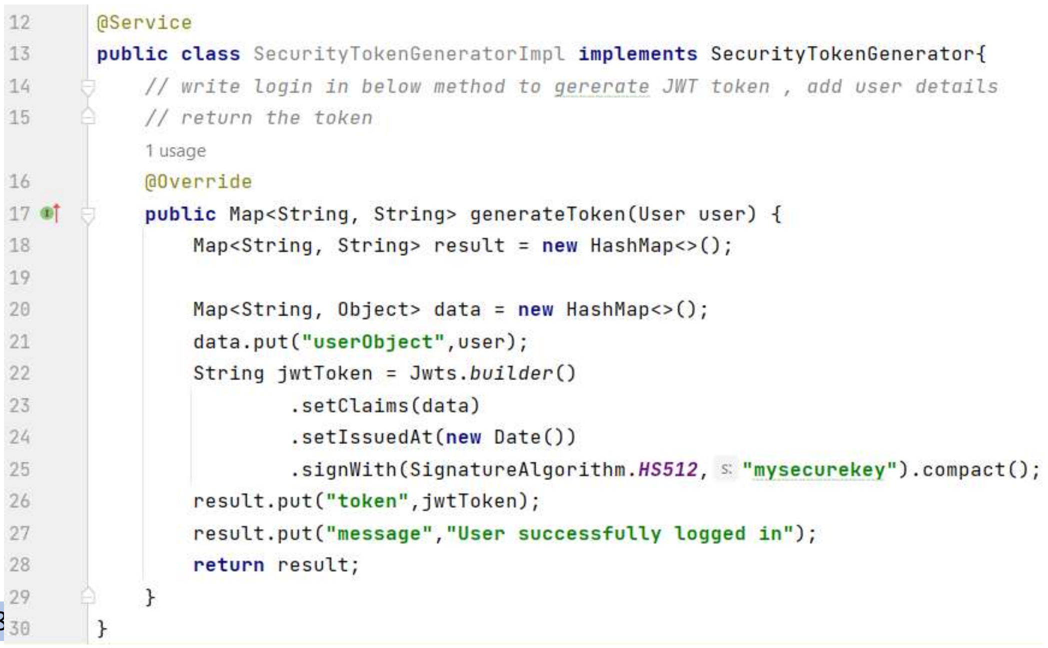
Last Release on Jul 5, 2018



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

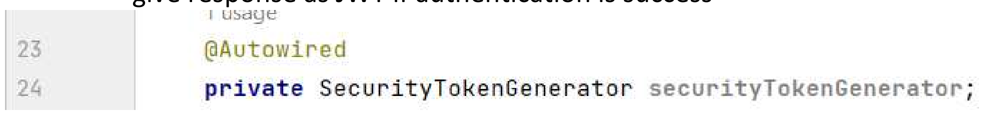


```
public interface SecurityTokenGenerator {
    public abstract Map<String, String> generateToken(User user);
}
```

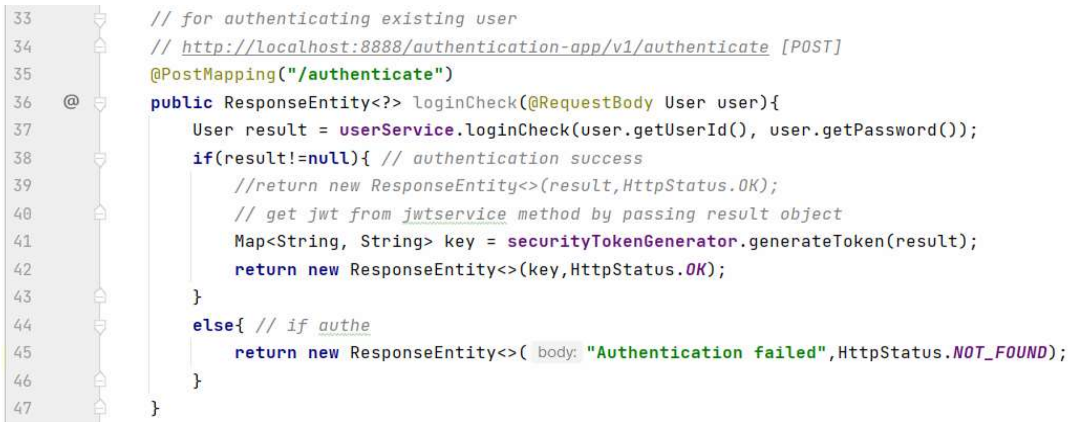


```java
@Service
public class SecurityTokenGeneratorImpl implements SecurityTokenGenerator{
    // write login in below method to gererate JWT token , add user details
    // return the token
    1 usage
    @Override
    public Map<String, String> generateToken(User user) {
        Map<String, String> result = new HashMap<>();

        Map<String, Object> data = new HashMap<>();
        data.put("userObject",user);
        String jwtToken = Jwts.builder()
                .setClaims(data)
                .setIssuedAt(new Date())
                .signWith(SignatureAlgorithm.HS512, s: "mysecurekey").compact();
        result.put("token",jwtToken);
        result.put("message","User successfully logged in");
        return result;
    }
}
```

8 give response as JWT if authentication is success



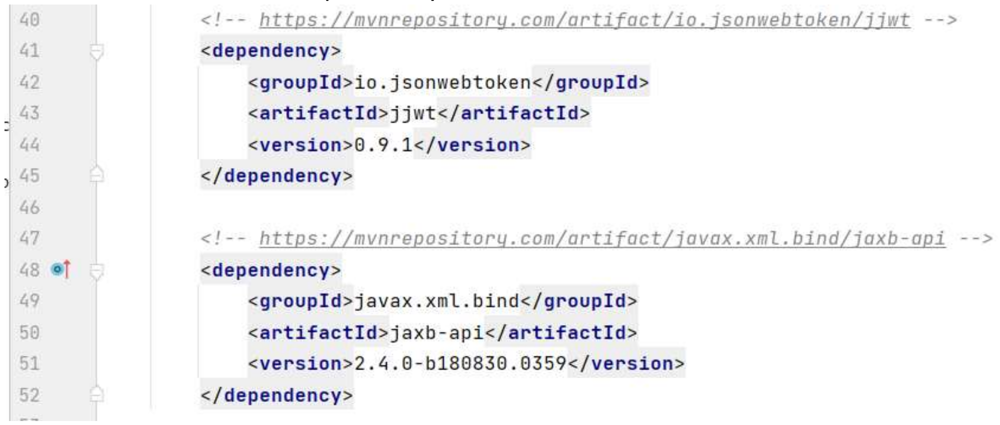
```java
    @Autowired
    private SecurityTokenGenerator securityTokenGenerator;
```



```java
    // for authenticating existing user
    // http://localhost:8888/authentication-app/v1/authenticate [POST]
    @PostMapping("/authenticate")
    public ResponseEntity<?> loginCheck(@RequestBody User user){
        User result = userService.loginCheck(user.getUserId(), user.getPassword());
        if(result!=null){ // authentication success
            //return new ResponseEntity<>(result,HttpStatus.OK);
            // get jwt from jwtservice method by passing result object
            Map<String, String> key = securityTokenGenerator.generateToken(result);
            return new ResponseEntity<>(key,HttpStatus.OK);
        }
        else{ // if authe
            return new ResponseEntity<>( body: "Authentication failed",HttpStatus.NOT_FOUND);
        }
    }
```

Steps to enable intercepting in app 2

step 1 make sure existing mongo application is working
demo1
add dependency

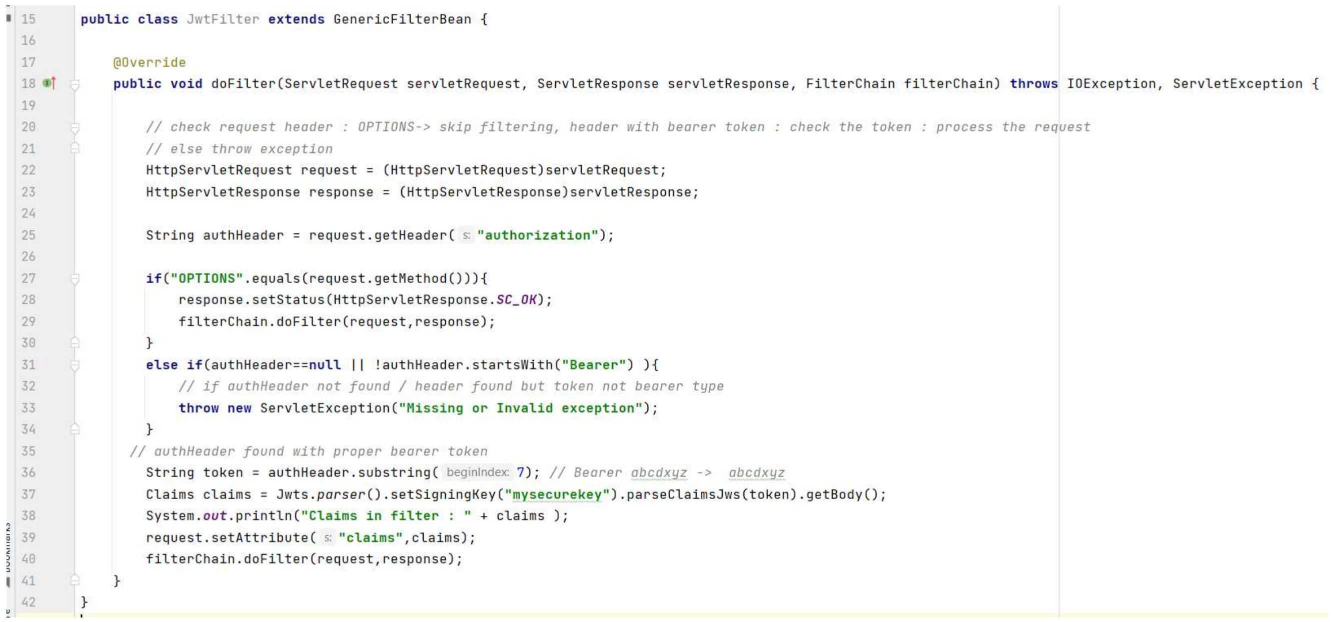


```
<!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->
<dependency>
    <groupId>io.jsonwebtoken</groupId>
    <artifactId>jjwt</artifactId>
    <version>0.9.1</version>
</dependency>

<!-- 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>
```

step 2 identify which URLs to be intercepted
intercepted
http://localhost:9999/customerapp/v1/register-customer

not intercepted
other all urls
http://localhost:9999/customerapp/v1/get-customers
http://localhost:9999/customerapp/v1/get-customer-by-id/CUST00001
http://localhost:9999/customerapp/v1/get-customers-by-city/Chennai

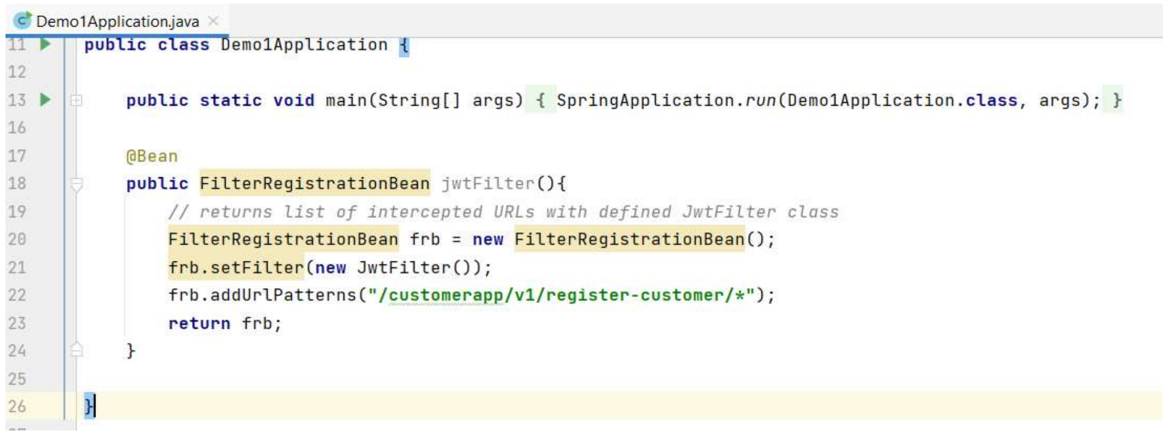
step 3   Define filter



```java
public class JwtFilter extends GenericFilterBean {

    @Override
    public void doFilter(ServletRequest servletRequest, ServletResponse servletResponse, FilterChain filterChain) throws IOException, ServletException {

        // check request header : OPTIONS-> skip filtering, header with bearer token : check the token : process the request
        // else throw exception
        HttpServletRequest request = (HttpServletRequest)servletRequest;
        HttpServletResponse response = (HttpServletResponse)servletResponse;

        String authHeader = request.getHeader( s: "authorization");

        if("OPTIONS".equals(request.getMethod())){
            response.setStatus(HttpServletResponse.SC_OK);
            filterChain.doFilter(request,response);
        }
        else if(authHeader==null || !authHeader.startsWith("Bearer") ){
            // if authHeader not found / header found but token not bearer type
            throw new ServletException("Missing or Invalid exception");
        }
      // authHeader found with proper bearer token
        String token = authHeader.substring( beginIndex: 7); // Bearer abcdxyz ->  abcdxyz
        Claims claims = Jwts.parser().setSigningKey("mysecurekey").parseClaimsJws(token).getBody();
        System.out.println("Claims in filter : " + claims );
        request.setAttribute( s: "claims",claims);
        filterChain.doFilter(request,response);
    }
}
```

step 4   Load filter as bean by adding intercepted URLs info
in main()

intercepted
http://localhost:9999/customerapp/v1/register-customer



Demo1Application.java

```java
public class Demo1Application {

    public static void main(String[] args) { SpringApplication.run(Demo1Application.class, args); }

    @Bean
    public FilterRegistrationBean jwtFilter(){
        // returns list of intercepted URLs with defined JwtFilter class
        FilterRegistrationBean frb = new FilterRegistrationBean();
        frb.setFilter(new JwtFilter());
        frb.addUrlPatterns("/customerapp/v1/register-customer/*");
        return frb;
    }

}
```

How to check

App2
http://localhost:9999/customerapp/v1/register-customer
above request not allowed without valid token

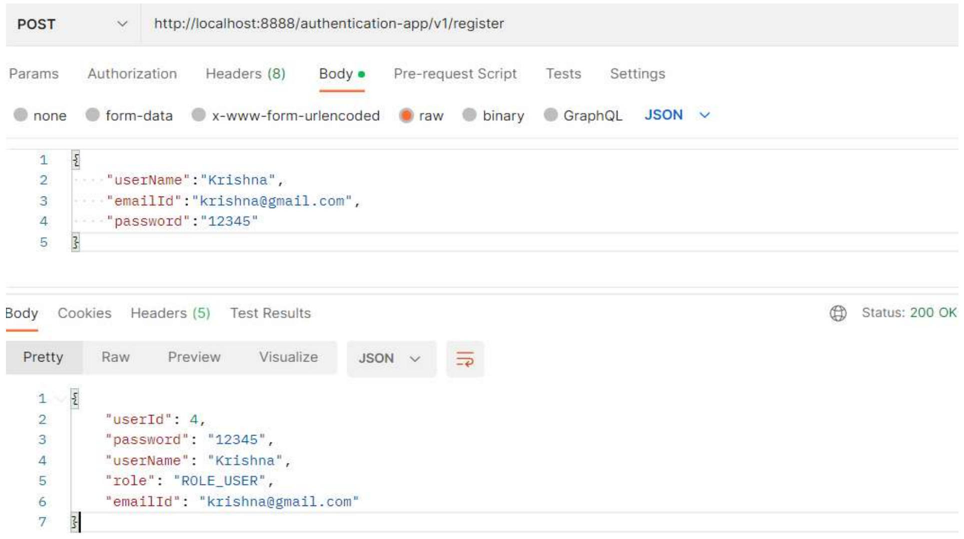
http://localhost:9999/customerapp/v1/get-customers
http://localhost:9999/customerapp/v1/get-customer-by-id/CUST00001
http://localhost:9999/customerapp/v1/get-customers-by-city/Chennai

How to check complete flow

step 1 App1
singup as new user
POST
http://localhost:8888/authentication-app/v1/register

```
{
   "userName":"Krishna",
   "emailId":"krishna@gmail.com",
   "password":"12345"
}
```

make sure account created in mysql



POST http://localhost:8888/authentication-app/v1/register

raw JSON

```
{
    "userName":"Krishna",
    "emailId":"krishna@gmail.com",
    "password":"12345"
}
```

Status: 200 OK

```
{
    "userId": 4,
    "password": "12345",
    "userName": "Krishna",
    "role": "ROLE_USER",
    "emailId": "krishna@gmail.com"
}
```

step 2 App1

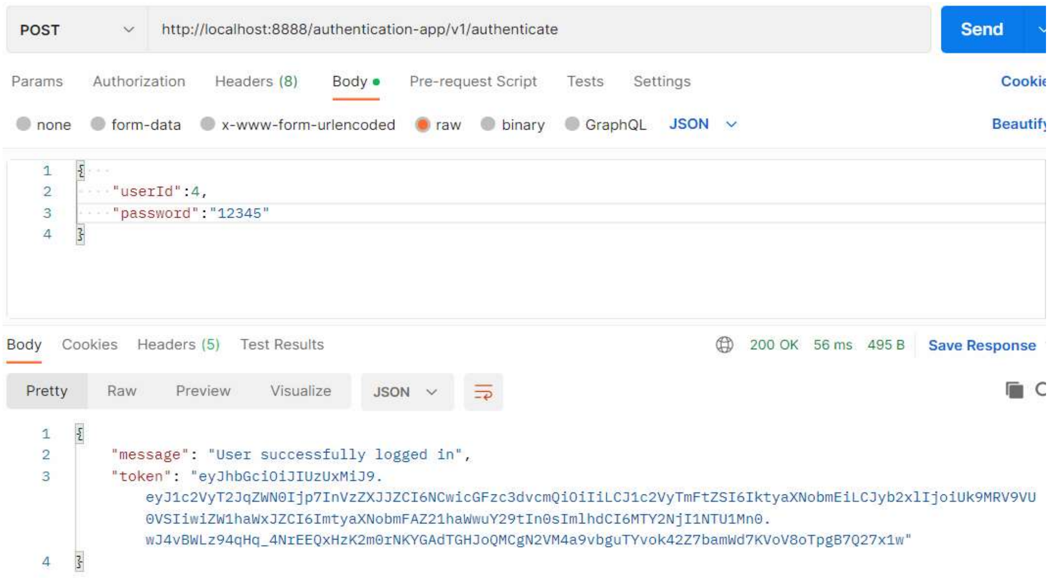
login as existing user

userId	4
password	12345

POST

http://localhost:8888/authentication-app/v1/authenticate

```
{
   "userId":4,
   "password":"12344323435"
}
```



POST http://localhost:8888/authentication-app/v1/authenticate

Body — raw — JSON

```
{
    "userId":4,
    "password":"12345"
}
```

200 OK 56 ms 495 B

```
{
    "message": "User successfully logged in",
    "token": "eyJhbGciOiJIUzUxMiJ9.
        eyJ1c2VyT2JqZWN0Ijp7InVzZXJJZCI6NCwicGFzc3dvcmQiOiIiLCJ1c2VyTmFtZSI6IktyaXNobmEiLCJyb2xlIjoiUk9MRV9VU
        0VSIiwiZW1haWxJZCI6ImtyaXNobmFAZ21haWwuY29tIn0sImlhdCI6MTY2NjI1NTU1Mn0.
        wJ4vBWLz94qHq_4NrEEQxHzK2m0rNKYGAdTGHJoQMCgN2VM4a9vbguTYvok42Z7bamWd7KVoV8oTpgB7Q27x1w"
}
```

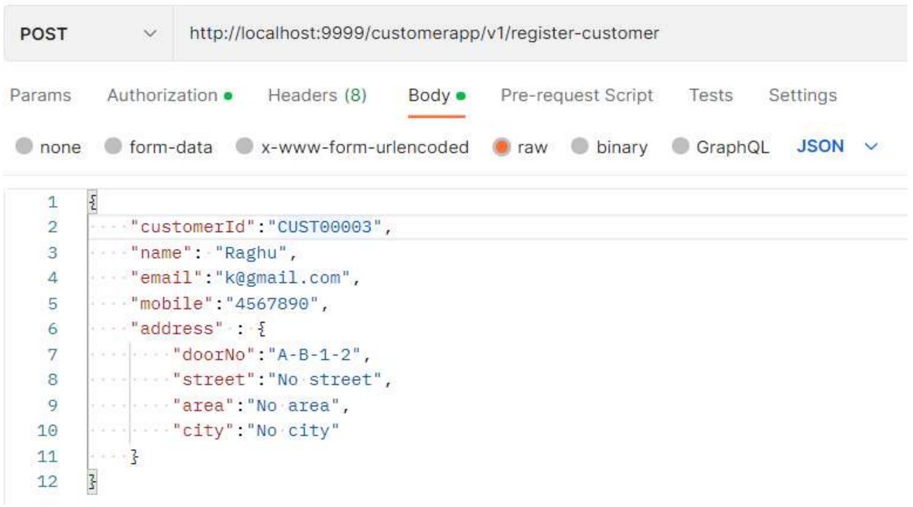
copy the token paste in notepad

step 3 App2 check intercepted URL by adding token in request auth header

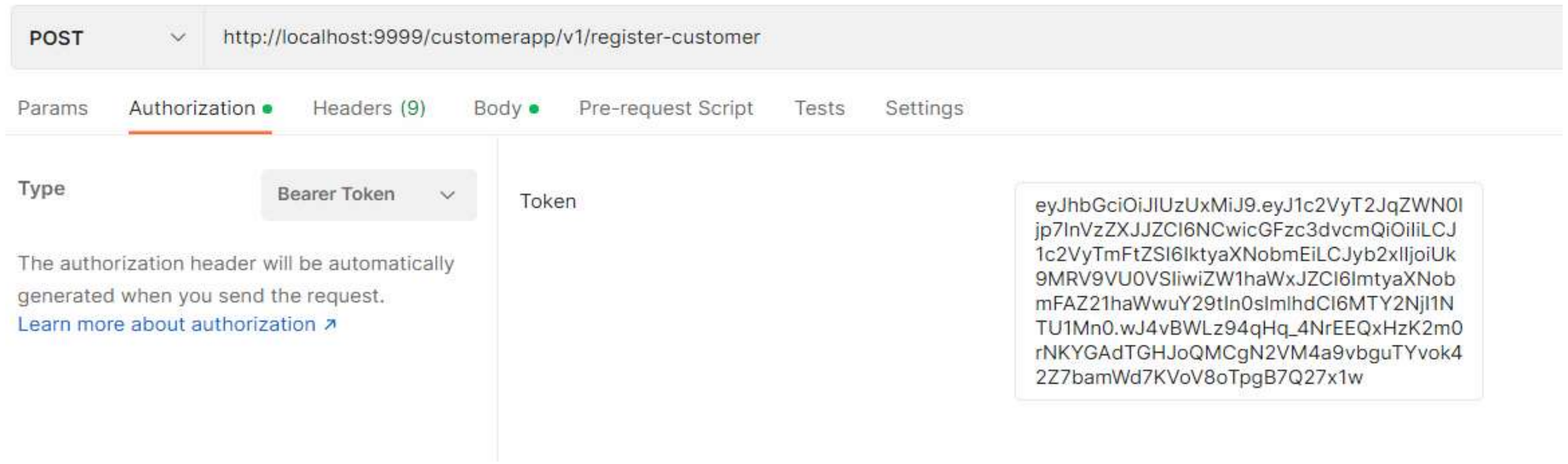
POST

http://localhost:9999/customerapp/v1/register-customer

Intercepted



```
{
    "customerId":"CUST00003",
    "name": "Raghu",
    "email":"k@gmail.com",
    "mobile":"4567890",
    "address" : {
        "doorNo":"A-B-1-2",
        "street":"No street",
        "area":"No area",
        "city":"No city"
    }
}
```



Type: Bearer Token

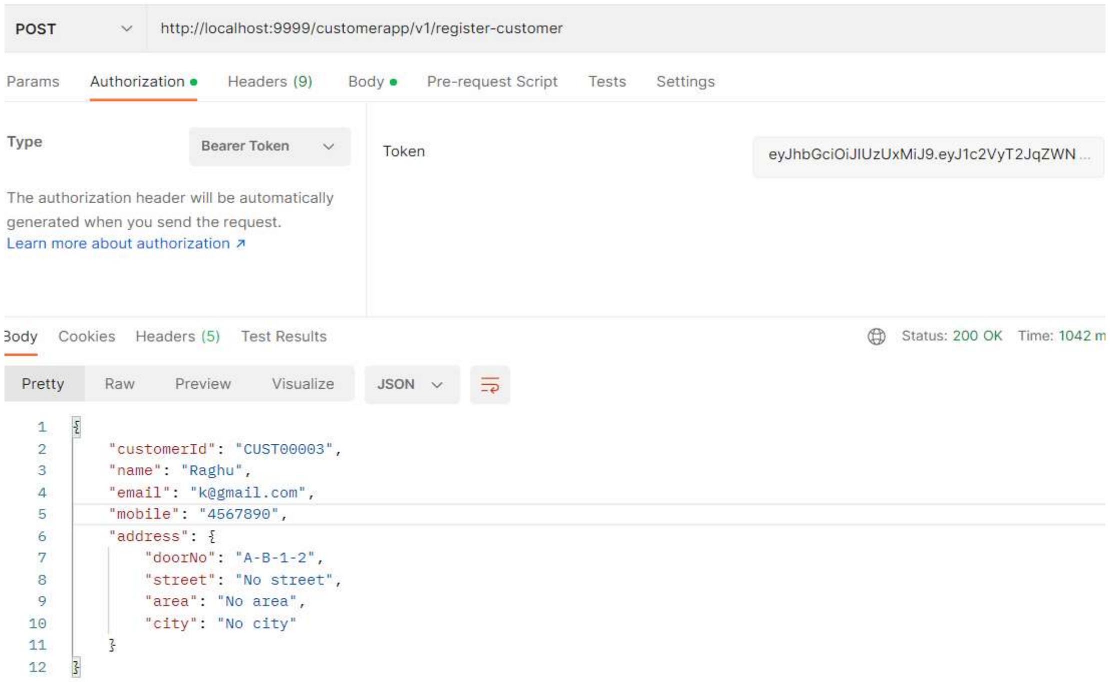
The authorization header will be automatically generated when you send the request.

Learn more about authorization ↗

Token

```
eyJhbGciOiJIUzUxMiJ9.eyJ1c2VyT2JqZWN0I
jp7InVzZXJJZCI6NCwicGFzc3dvcmQiOiIiLCJ
1c2VyTmFtZSI6IktyaXNobmEiLCJyb2xlIjoiUk
9MRV9VU0VSIiwiZW1haWxJZCI6ImtyaXNob
mFAZ21haWwuY29tIn0sImlhdCI6MTY2NjI1N
TU1Mn0.wJ4vBWLz94qHq_4NrEEQxHzK2m0
rNKYGAdTGHJoQMCgN2VM4a9vbguTYvok4
2Z7bamWd7KVoV8oTpgB7Q27x1w
```

Make sure request processed


POST
http://localhost:9999/customerapp/v1/register-customer
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Type
Bearer Token
Token
eyJhbGciOiJIUzUxMiJ9.eyJ1c2VyT2JqZWN ...
The authorization header will be automatically generated when you send the request.
Learn more about authorization ↗
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 1042 m
Pretty
Raw
Preview
Visualize
JSON

```
{
    "customerId": "CUST00003",
    "name": "Raghu",
    "email": "k@gmail.com",
    "mobile": "4567890",
    "address": {
        "doorNo": "A-B-1-2",
        "street": "No street",
        "area": "No area",
        "city": "No city"
    }
}
```