JWT Session-2
Mr. RAGHU (NareshIT)

FB : https://www.facebook.com/groups/thejavatemple/
Email : javabyraghu@gmail.com

JWT - JSON WEB TOKEN
-> Token Based Authentication
-> Stateless Authentication (No Http Session)

JSON FORMAT:
```
{
  key: val,..
}
```

JWT contains 3 parts
Header = JWT Specific information
Payload = Claims (clientId, clientName,provider Name, date, expDate..)
Signature = Encode(Header) + Encode(Payload) <- SecretKey.

a. Generate JWT Token using JAVA
b. READ Token and validate Details using JAVA

JJWT -> JAVA JWT

```
<dependency>
   <groupId>io.jsonwebtoken</groupId>
   <artifactId>jjwt</artifactId>
   <version>0.9.1</version>
</dependency>
```

JWT Token
xxxxxxxxxxxxxxxx.yyyyyyyyyyyyyyyyyy.zzzzzzzzzzzzzzzzzzzzz

eyJhbGciOiJIUzI1NiJ9.

eyJqdGkiOiJBQjEyNTYiLCJzdWIiOiJBSkFZIiwiaXNzIjoiTmFyZXNoSVQtSFlEIiw
iaWF0IjoxNTk4MTIxNTc1LCJleHAiOjE1OTgxMjIxNzV9.

p8fD-oQxMZW_5c-r4fe2W5Lj87wy_GGOqNaR952cjrs

Claims : Get details of a JWT Token using Secret. [Parsing JWT Token]
Input:
-> Token
-> Secret

==========code: JJWT==========

#1. Create one Simple Maven Project

#2. Add below details in pom.xml

```
        <properties>
                <maven.compiler.source>14</maven.compiler.source>
                <maven.compiler.target>14</maven.compiler.target>
        </properties>

        <dependencies>
                <dependency>
                        <groupId>io.jsonwebtoken</groupId>
                        <artifactId>jjwt</artifactId>
                        <version>0.9.1</version>
                </dependency>

                <dependency>
                        <groupId>javax.xml.bind</groupId>
```

```
                        <artifactId>jaxb-api</artifactId>
                        <version>2.3.0</version>
                </dependency>
        </dependencies>
```

#3. Test class - Generating Token

```
package in.nareshit.raghu.test;

import java.util.Base64;
import java.util.Date;
import java.util.concurrent.TimeUnit;

import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.SignatureAlgorithm;

public class JwtTest {

        public static void main(String[] args) {
                String key = "NITRAGHU";

                //Token Generation with details
                String token =
                        Jwts.builder()
                                .setId("AB1256") //client Id
                                .setSubject("AJAY") //clientName
                                .setIssuer("NareshIT-HYD") //provider Name
                                .setIssuedAt(new Date(System.currentTimeMillis())) //token
gen Date
                                .setExpiration(new Date(System.currentTimeMillis()  //exp
date
                                                + TimeUnit.MINUTES.toMillis(10) ))
                                .signWith(SignatureAlgorithm.HS256, //alog
                                                Base64.getEncoder().encode(key.getBytes()))
//secret key
                                .compact(); //convert to JWT STRING

                System.out.println(token);

                //--------------------
                Claims c =
                Jwts.parser() //read token data
                .setSigningKey(Base64.getEncoder().encode(key.getBytes())) //secret key
                .parseClaimsJws(token) //token
                .getBody(); //claims data

                //print / validate ...
                System.out.println(c.getId());
                System.out.println(c.getSubject());
                System.out.println(c.getIssuer());
        }
}
```

-------UTIL class--------------

```
package in.nareshit.raghu.util;

import java.util.Base64;
import java.util.Date;
import java.util.concurrent.TimeUnit;

import io.jsonwebtoken.Claims;
import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.SignatureAlgorithm;

public class JwtUtil {
```

```
	//1. Generate JWT TOKEN
	public String generateToken(
			String id,
			String subject,
			String secret
			)
	{

		return Jwts.builder()
				.setId(id)
				.setSubject(subject)
				.setIssuer("NareshIT")
				.setIssuedAt(new Date(
						System.currentTimeMillis()))
				.setExpiration(new Date(
						System.currentTimeMillis() +
						TimeUnit.MINUTES.toMillis(10)  ))
				.signWith(SignatureAlgorithm.HS256,

Base64.getEncoder().encode(secret.getBytes()))
				.compact()
				;
	}


	//2. GetClaims
	public Claims getClaims(
			String token,
			String secret
			)
	{

		return Jwts.parser()
				.setSigningKey(

Base64.getEncoder().encode(secret.getBytes()))
				.parseClaimsJws(token)
				.getBody()
				;
	}


	public String getSubject(
			String token,
			String secret
			)
	{
		return getClaims(token, secret).getSubject();
	}

	public boolean isValidToken(
			String token,
			String secret
			)
	{

		//  now 12:01 PM < exp: 12:30 PM
		return getClaims(token, secret).getExpiration()
				.after(new Date(System.currentTimeMillis()));
	}


}
```