Contents

[**Authentication** 1](#_Toc21201414)

[**Authorization** 1](#_Toc21201415)

[**JWT (Json Web Token)** 2](#_Toc21201416)

[**JWT Token Structure** 2](#_Toc21201417)

[**Form Based Security** 3](#_Toc21201418)

[**HTTP Basic Authentication** 5](#_Toc21201419)

[**HTTPS & SSL** 6](#_Toc21201420)

[**Unsecure spring boot app** 9](#_Toc21201421)

[**HTTP Basic Security with role-based authorization (http-basic-security)** 9](#_Toc21201422)

[**HTTP Basic Security Permission based authorization(http-basic-security2)** 9](#_Toc21201423)

[**HTTP Basic Security communicating with Database(http-basic-security3)** 10](#_Toc21201424)

[**Form based authentication (form-based-security)** 10](#_Toc21201425)

[**Form based authentication with custom login page (form-based-security2)** 10](#_Toc21201426)

[**JWT Security (jwt-security)** 10](#_Toc21201427)

[**Spring Security OAuth2** 10](#_Toc21201428)

# **Authentication**

Authentication is about, who you are in the context of an application.

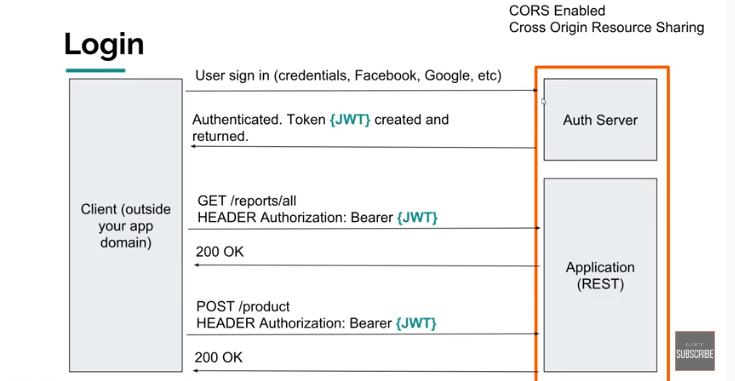
* HTTP Authentication
* Forms Authentication
* Certificates
* Tokens

# **Authorization**

Authorization is all about Privileges/ Authorities

# **JWT (Json Web Token)**

Json Web Token is a compact and safe way to transmit data between two parties. The information can be trusted because it is digitally signed.



## **JWT Token Structure**

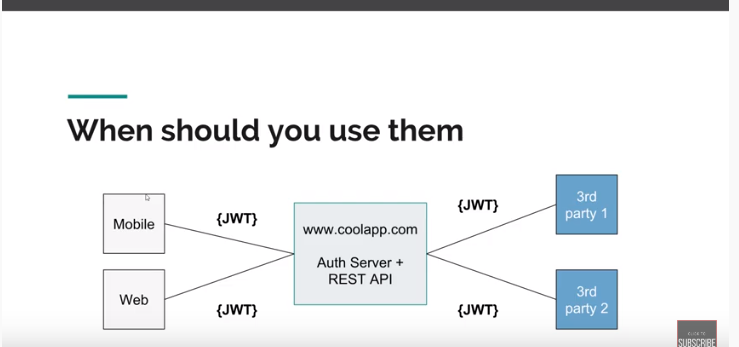
JSON Web Token consists of three parts separated by dots(.).

* Header (which is a json object)
* Payload (Which is a json object)
* Signature (Secret)

Example hhhh.pppp.sssssssssss

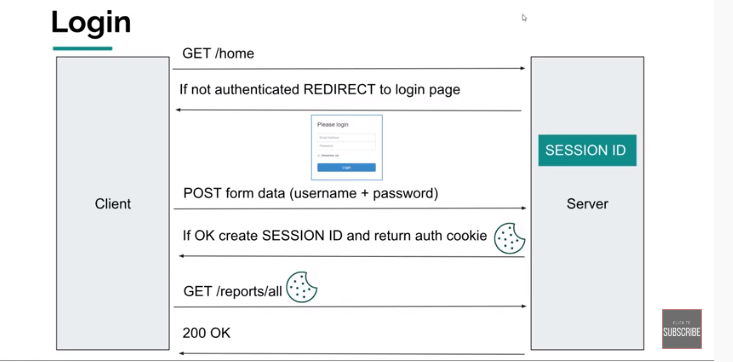
For more details about JWT got to <https://jwt.io/>

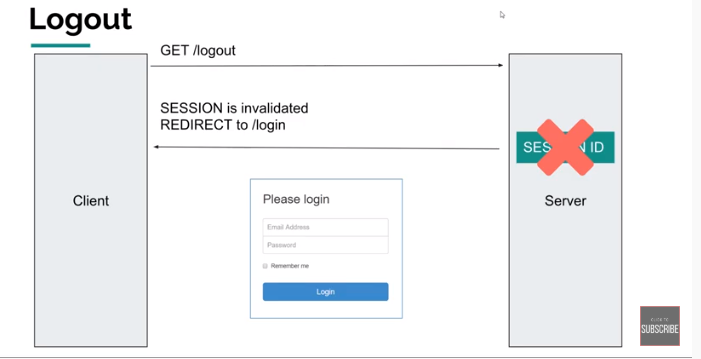
Or visit article written by Mikey Stecky https://medium.com/vandium-software/5-easy-steps-to -understand-json-web-token

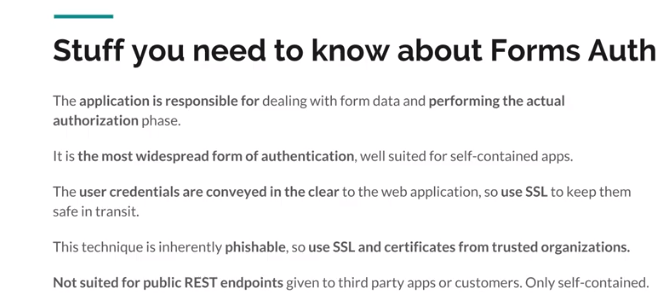


# **Form Based Security**

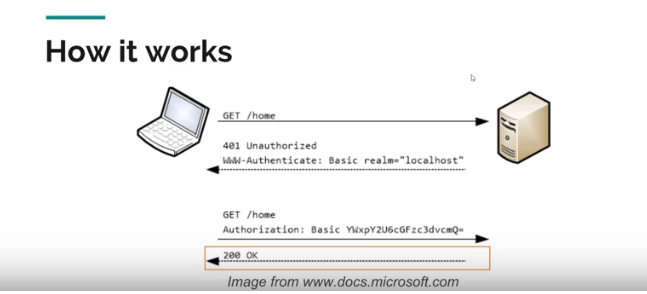
It is a process of authenticating a user by presenting a custom html page that will collect credentials and by directing the authentication responsibility to the web application that collects the form data.

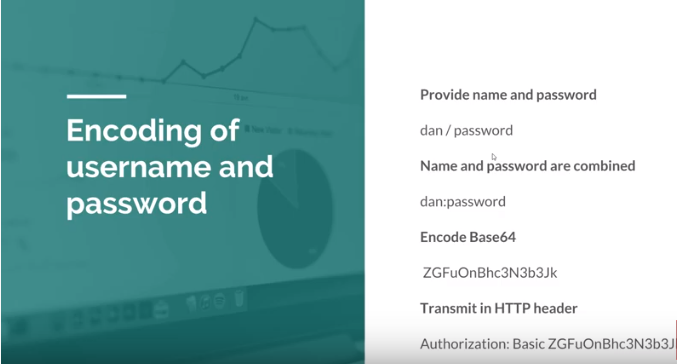


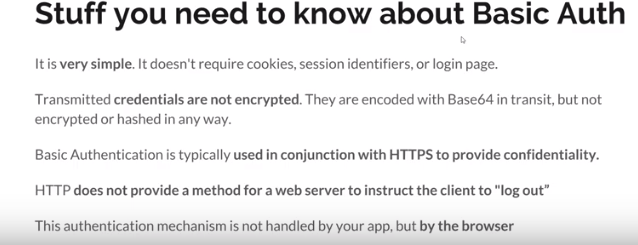


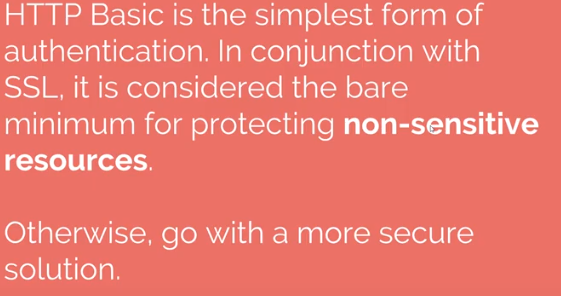


# **HTTP Basic Authentication**



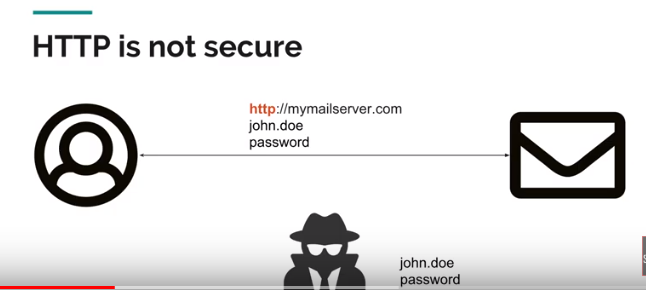


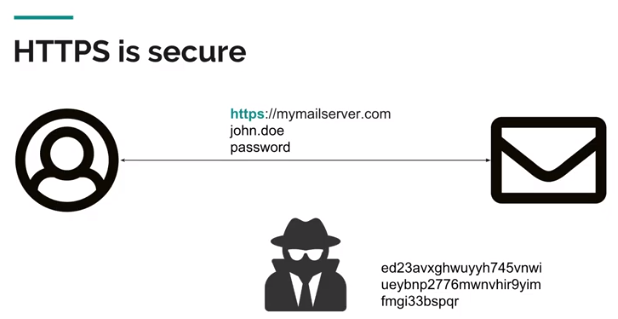


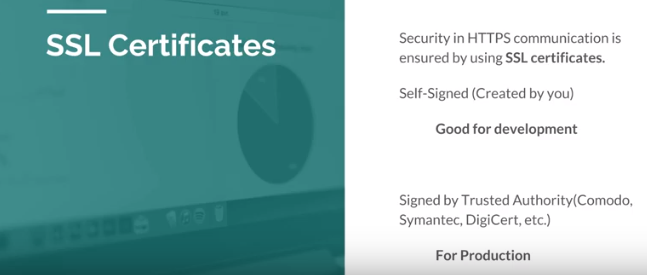


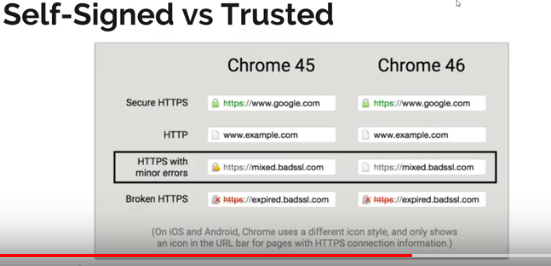
# **HTTPS & SSL**

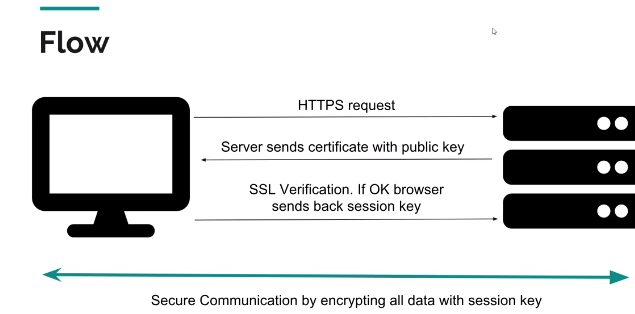
HTTPS is a combination of HTTP plus SSL security layer on top of it. HTTPS is just HTTP that delivers data securely between endpoints.

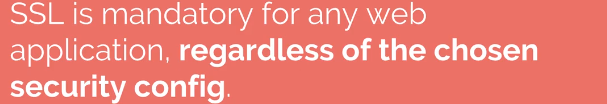












# **Unsecure spring boot app**

# **HTTP Basic Security with role-based authorization (http-basic-security)**

# **HTTP Basic Security Permission based authorization(http-basic-security2)**

1. Configure permission-based security.
2. Enable SSL and HTTPS

* Certificate (can be self-signed or Buy)

To Generate the certificate to to your java installation directory**(/bin)**

Run the below command and provide the required input

.\keytool -genkey -alias bootsecurity -storetype PKCS12 -keyalg RSA -keysize 2048 -keystore bootsecurity.p12 -validity 3650

* Modify application.properties file
* Add @Bean for ServletWebServerFactory (instruct the container that we want to use https and also going to tell him redirect all http traffic to our https port so in that way no matter how user want to access our application the will always get directed to a secure one)

# **HTTP Basic Security communicating with Database(http-basic-security3)**

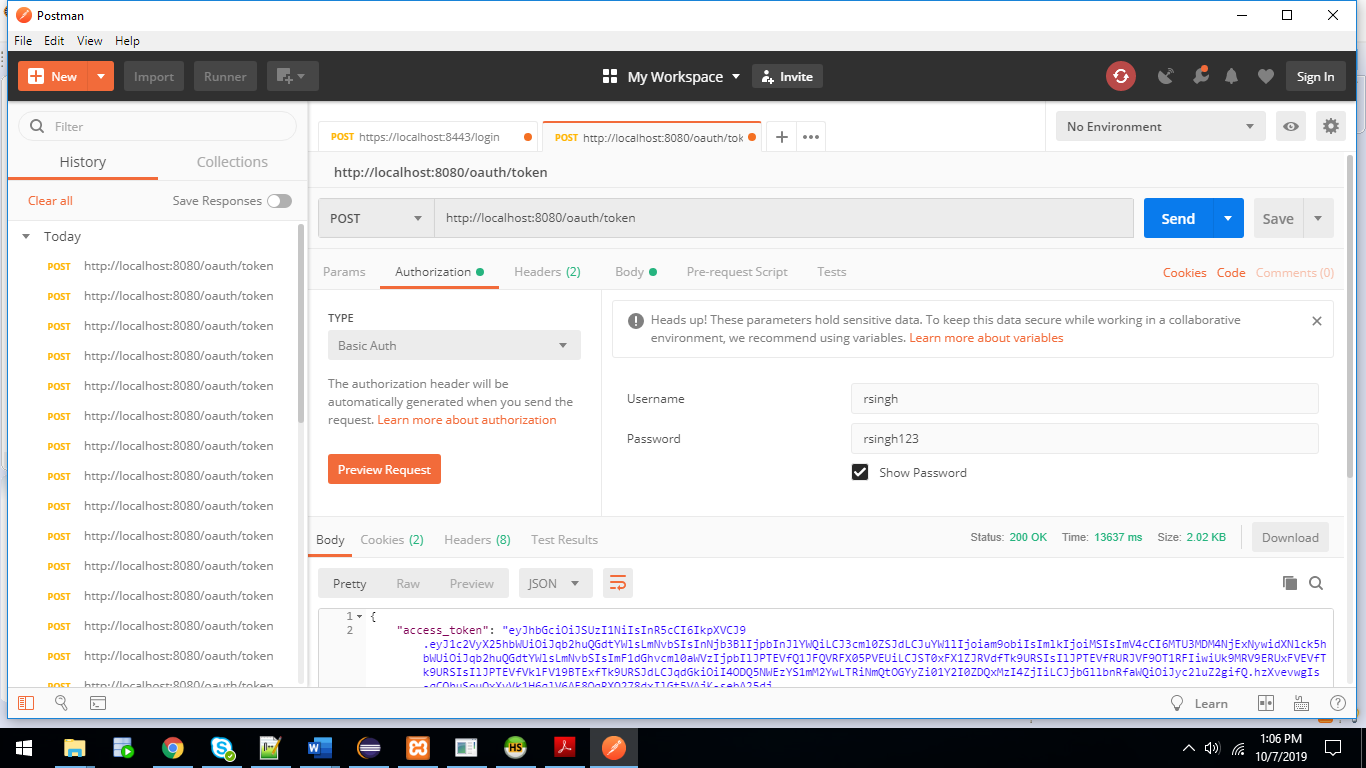
# **Form based authentication (form-based-security)**

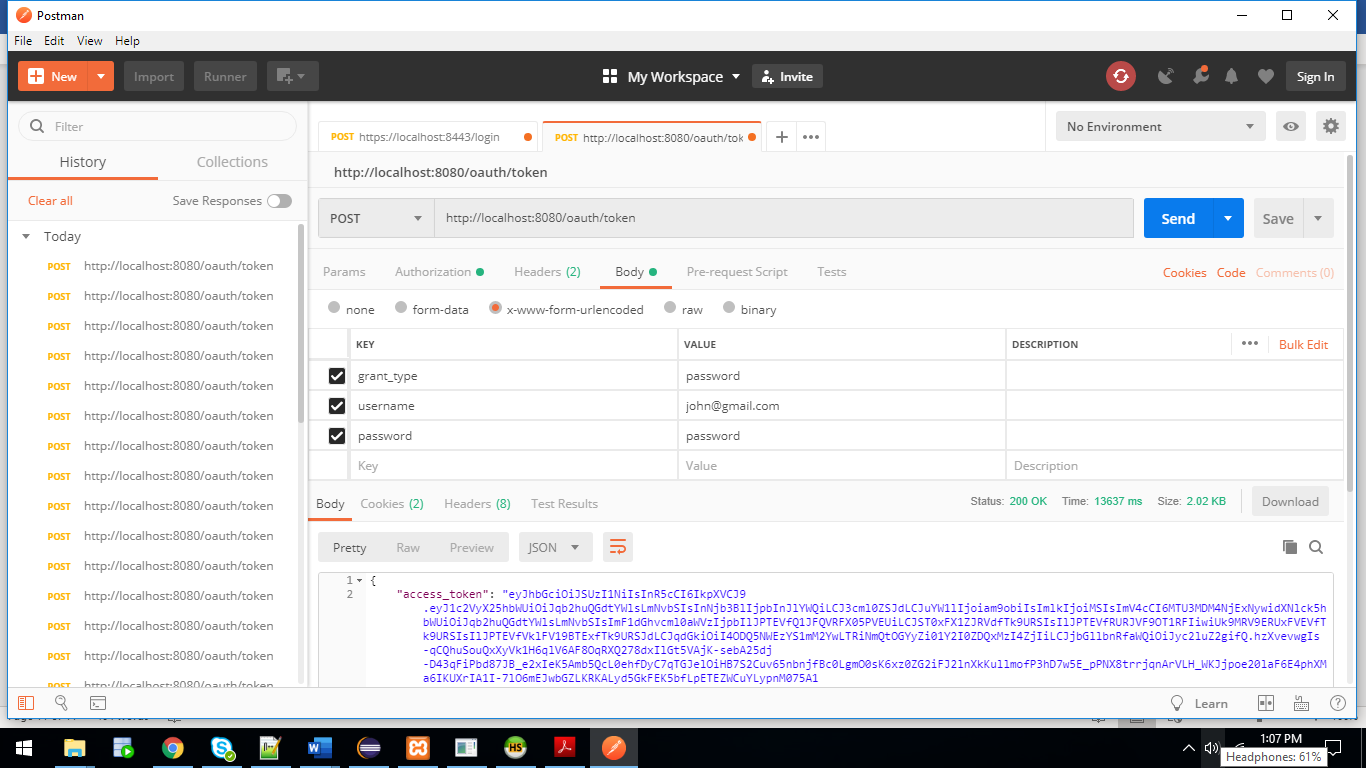
# **Form based authentication with custom login page (form-based-security2)**

# **JWT Security (jwt-security)**

# **Spring Boot Security OAuth2 Authorization server and resource server| JWT | MySql**

* springboot-oauth2-auth-erver





Above details need to be passed in order to get the token, and that jwt token can be verified on jwt.io with the public and private key.

* Springboot-oauth2-resource-server

SQL Queries :-

CREATE TABLE permission (

id int PRIMARY KEY AUTO\_INCREMENT,

permission\_name VARCHAR(20)

)

CREATE TABLE role (

id int PRIMARY KEY AUTO\_INCREMENT,

role\_name VARCHAR(20)

)

CREATE TABLE user (

id int PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(20),

email\_id VARCHAR(255),

PASSWORD VARCHAR(1000)

)

CREATE TABLE assign\_permission\_to\_role (

id int PRIMARY KEY AUTO\_INCREMENT,

permission\_id INT, FOREIGN KEY(permission\_id) REFERENCES permission(id),

role\_id INT, FOREIGN KEY(role\_id) REFERENCES role(id)

)

CREATE TABLE assign\_user\_to\_role (

id int PRIMARY KEY AUTO\_INCREMENT,

user\_id INT, FOREIGN KEY(user\_id) REFERENCES user(id),

role\_id INT, FOREIGN KEY(role\_id) REFERENCES role(id)

)

INSERT INTO permission(id, permission\_name) VALUES(1,'CREATE\_NOTE'),(2,'EDIT\_NOTE'),(3,'DELETE\_NOTE'),(4,'VIEW\_ALL\_NOTE'),(5,'VIEW\_NOTE')

INSERT INTO ROLE(ID,ROLE\_NAME) VALUES(1,'ADMINISTRATOR'),(2,'AUDITOR')

INSERT INTO user(id,NAME,email\_id,PASSWORD) VALUES(1,'john','john@gmail.com','694c5e80-b7b6-4ec8-8fb6-a6b0be583e4b');

INSERT INTO user(id,NAME,email\_id,PASSWORD) VALUES(2,'mike','mike@gmail.com','694c5e80-b7b6-4ec8-8fb6-a6b0be583e4b');

INSERT INTO assign\_permission\_to\_role(permission\_id,role\_id) VALUES(1,1);

INSERT INTO assign\_permission\_to\_role(permission\_id,role\_id) VALUES(2,1);

INSERT INTO assign\_permission\_to\_role(permission\_id,role\_id) VALUES(3,1);

INSERT INTO assign\_permission\_to\_role(permission\_id,role\_id) VALUES(4,1);

INSERT INTO assign\_permission\_to\_role(permission\_id,role\_id) VALUES(5,1);

INSERT INTO assign\_permission\_to\_role(permission\_id,role\_id) VALUES(4,2);

INSERT INTO assign\_permission\_to\_role(permission\_id,role\_id) VALUES(5,2);

INSERT INTO assign\_user\_to\_role(user\_id,role\_id) VALUES(1,1),(2,2);