**User Authentication module :**

Problem Statement:

Need to Build a User Authentication Module.

**Tools and Technologies Used**

* Spring Boot 2.3.1.RELEASE
* JDK 1.8
* Maven - 3.2+
* IDE - Eclipse or Spring Tool Suite (STS)
* MYSQL
* PostMan
* JAVA 8 Features
* Spring Security

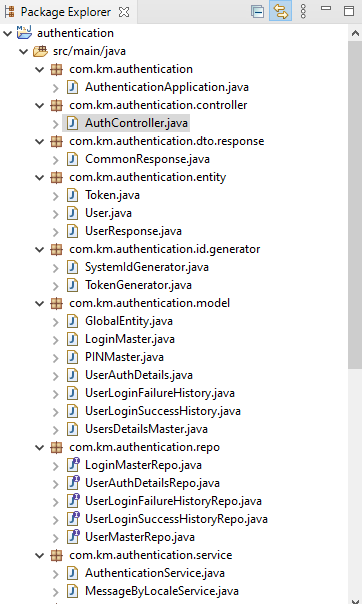
## Creating and Importing a Project

There are many ways to create a Spring Boot application. The simplest way is to use Spring Initializr at <http://start.spring.io/>, which is an online Spring Boot application generator.

we have specified the following details:

* Generate: Maven Project
* Java Version: 1.8 (Default)
* Spring Boot:2.3.1 Release
* Package Name: com.km.authentication
* Packaging: jar (This is the default value)
* Dependencies: Web, JPA, MySQL, Security

## Project Structure



## Database Create SQL Scripts

**Table 1:**

Main Table: **authentication.users\_details\_master**

This table is used to store the all Users information like Username, encrypted pass word etc.

CREATE TABLE users\_details\_master (

system\_id varchar(30) NOT NULL,

created\_by varchar(255) DEFAULT NULL,

created\_on datetime NOT NULL,

modified\_by varchar(80) NOT NULL,

modified\_on datetime NOT NULL,

incorrect\_count int NOT NULL,

last\_updated\_date datetime DEFAULT NULL,

password varchar(150) NOT NULL,

previous\_password varchar(150) NOT NULL,

status int NOT NULL,

user\_name varchar(60) NOT NULL,

PRIMARY KEY (system\_id)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

Output:

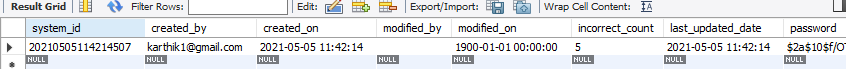


Table 2 :

authentication.login\_master

This table is used to maintain the login information (UserName info as Email/Phone No).Its join with the table authentication.users\_details\_master. Key issystem\_id.

CREATE TABLE login\_master (

login\_id varchar(80) NOT NULL,

created\_by varchar(255) DEFAULT NULL,

created\_on datetime NOT NULL,

modified\_by varchar(80) NOT NULL,

modified\_on datetime NOT NULL,

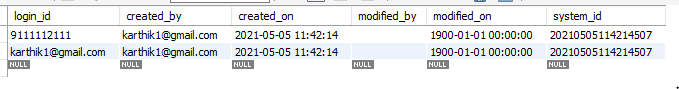
system\_id varchar(30) DEFAULT NULL,

PRIMARY KEY (login\_id),

KEY `FKm1t62dds8sqodwd6u6kknxji1` (system\_id),

CONSTRAINT `FKm1t62dds8sqodwd6u6kknxji1` FOREIGN KEY (`system\_id`) REFERENCES `users\_details\_master` (`system\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



Note : We can able to use username as MobileNo/Email.

Table 3: authentication.user\_login\_failure\_history.

**This is table is used to stored failure log information after failure authentication.**

CREATE TABLE **user\_login\_failure\_history**(

id bigint NOT NULL,

login\_date datetime NOT NULL,

system\_id varchar(30) NOT NULL,

user\_name varchar(60) NOT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

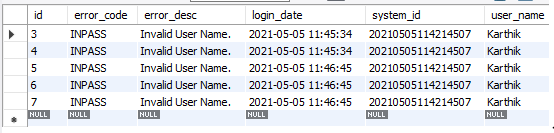


Table 4: **authentication.user\_login\_success\_history.**

**This is table is used to stored successful log information after successful authentication.**

CREATE TABLE user\_login\_success\_history(

id bigint NOT NULL,

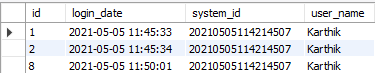
login\_date datetime NOT NULL,

system\_id varchar(30) NOT NULL,

user\_name varchar(60) NOT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;



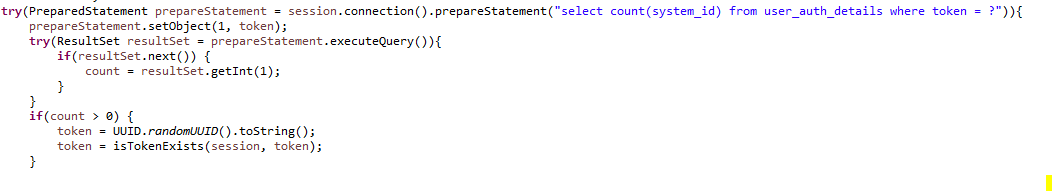
**Token Generation Details:**

**Generate UUID token:**

Class : TokenGenerator.class



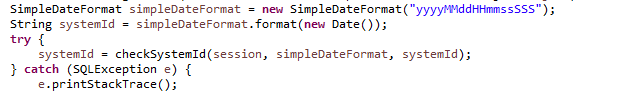
**Validate Token exists:**



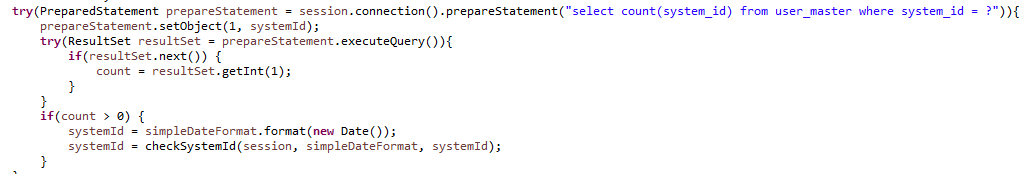
**System Id Generation:**

Class : SystemIdGenerator.class

Unique Id generation for Primary Key.

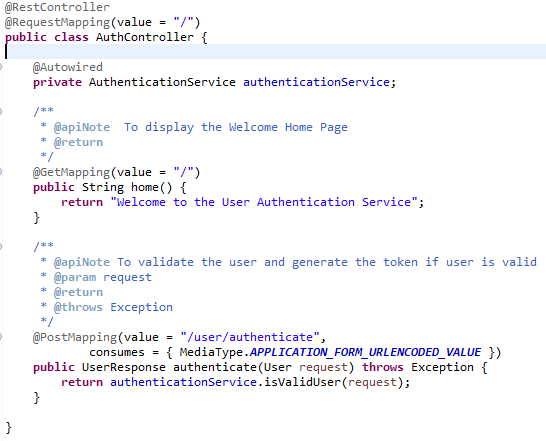


**Validate Id exists:**



## Spring Rest Controller - AuthController.java

Class : AuthController.java



## Spring Rest Service - AuthenticationService.java



**Testing :**

Step 1 :

Need to do Maven build after import the Projects or run the class (UserMasterRepoTests.class) which has contains user details, Same details will be inserted into respective tables(users\_details\_master,login\_master).

Text

Description automatically generated

**Step 2: Execute the Tests.**

**Home Page**

Graphical user interface, text, application

Description automatically generated

**Test 1: Using Post Man**

Valid User Authentication

Graphical user interface, text, application, email

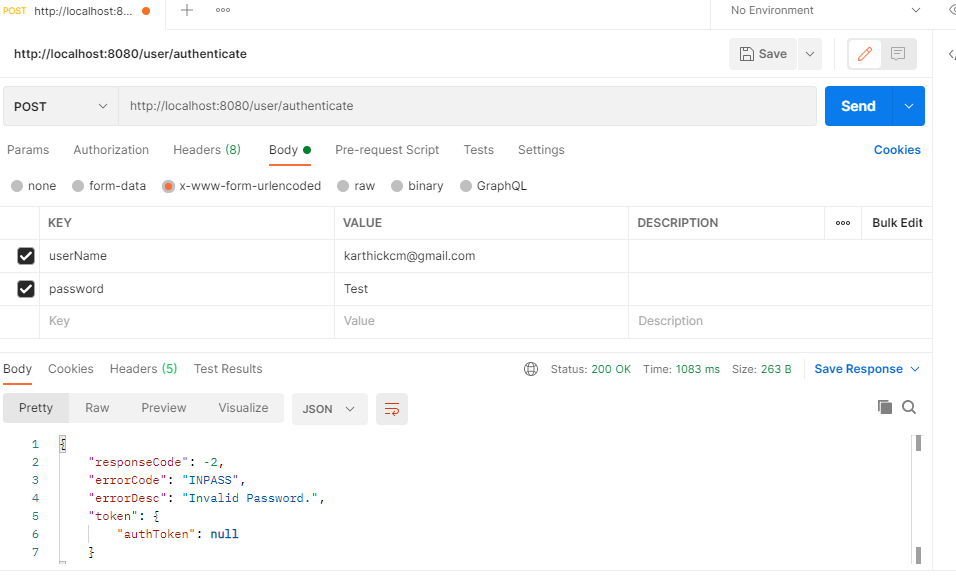
Description automatically generated

Invalid UserName

Graphical user interface, text, application, email

Description automatically generated

Invalid Password



**Test 2: JUNIT Test Cases:**

**Class :** AuthenticationServiceTest.class

**Success:**

@Before

**public** **void** setUpData() {

request = **new** UserAuthRequestDto();

request.setUserName("karthickcm@gmail.com");

request.setPassword("Test@123");

}

@Test

**public** **void** verifyValidUser() **throws** Exception {

*assertEquals*(0, authentication.isValidUser(request).getResponseCode());

*assertEquals*("000", authentication.isValidUser(request).getErrorCode());

}

**INUSNM=Invalid User Name.**

@Before

**public** **void** setUpData() {

request = **new** UserAuthRequestDto();

request.setUserName("karthickcm@gmail.com");

request.setPassword("Test@123");

}

@Test

**public** **void** verifyInvalidUser() **throws** Exception {

request.setUserName("Test");

*assertEquals*(-2, authentication.isValidUser(request).getResponseCode());

*assertEquals*("INUSER", authentication.isValidUser(request).getErrorCode());

}

**INPASS=Invalid Password.**

@Before

**public** **void** setUpData() {

request = **new** UserAuthRequestDto();

request.setUserName("karthickcm@gmail.com");

request.setPassword("Test@123");

}

@Test

**public** **void** verifyInvalidPassword() **throws** Exception {

request.setPassword("test");

*assertEquals*(-2, authentication.isValidUser(request).getResponseCode());

*assertEquals*("INPASS", authentication.isValidUser(request).getErrorCode());

}