Farkalit Usman

vXebia@usmanuat.com

|  |  |  |
| --- | --- | --- |
| **Version** | **Name** | **Description** |
| 1.0 | Farkalit Usman | Initial document for Usman bank MS Startup and configuration |
|  |  |  |
|  |  |  |
|  |  |  |

Abstract

Startup and Configuration of the Micro Services application

Usman bank MS Startup and Configuration

Micro Services Startup and Configuration

#### Usman Bank Micro Services Startup and configuration

Table of Contents

[Tools and Technology: 2](#_Toc19611588)

[Start IDE (STS4) 2](#_Toc19611589)

[Start Usman Gateway 2](#_Toc19611590)

[Start Customer Micro Service 5](#_Toc19611591)

[Eureka Server with Registered Instances 6](#_Toc19611592)

[User (Customer) Roles Mapping 7](#_Toc19611593)

[REST API Path Param and Variables 8](#_Toc19611594)

[Spring boot Actuator 8](#_Toc19611595)

[Spring boot Dev tools 8](#_Toc19611596)

[Log4j2 Configuration Details 8](#_Toc19611597)

## Tools and Technology:

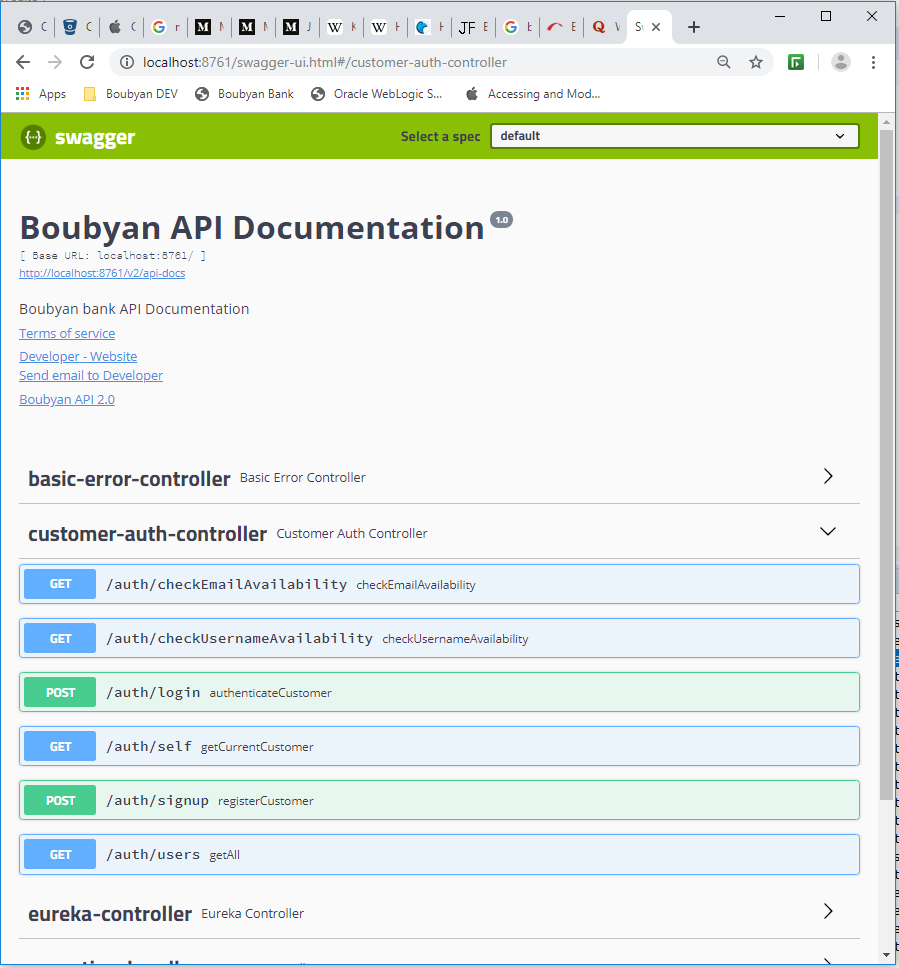
1. Install and Configure Java 1.8
2. Install and Configure Maven 3.x version
3. Install and Configure Spring Tool Suite 4
4. Configure the Oracle Database (URL: jdbc:oracle:thin:@10.1.13.182:1521:CEBS).
5. Get/Clone the Following Usman Micro services application project from bit-bucket
   1. usman-gateway-mgmt (User registration with role and login. Login will return JWT )
   2. usman-customer-mgmt
   3. usman-utility-services (AUDIT, EMAIL & SMS )
   4. usman-common-utils (constants and validations)

## Start IDE (STS4)

1. Open STS4
2. Import the all the code which is cloned from bit-bucket.

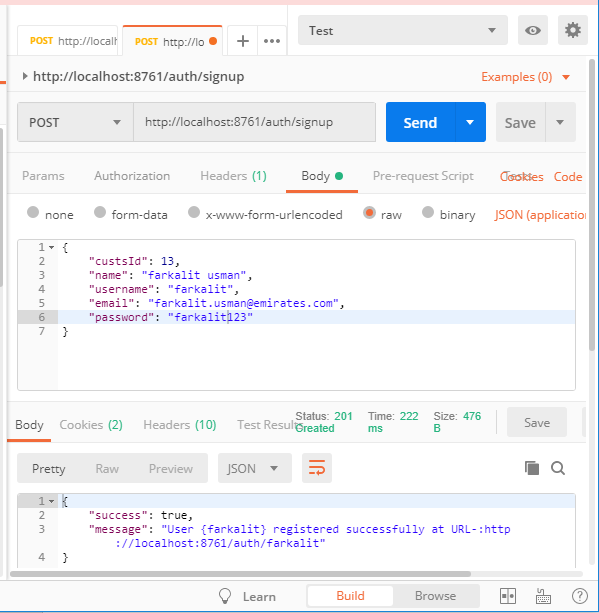
## Start Usman Gateway

1. Form STS4 run the “usman-gateway” service as Spring boot App. If it is started properly; open the browser and paste this URL: <http://localhost:8761/swagger-ui.html#/>. You will see the following details in the page:



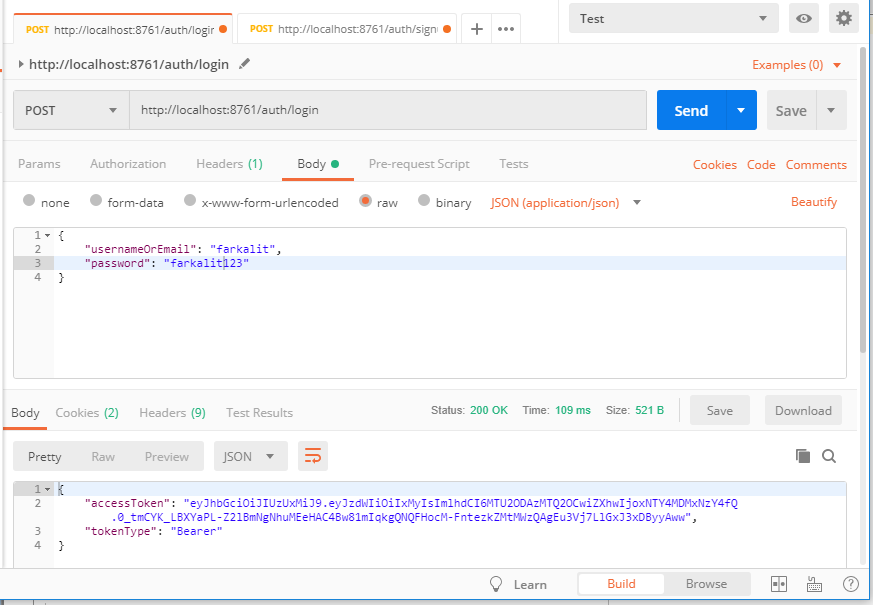
1. Create/Signup for a new User. It will create the new user in the CUSTOMER table. Use the following request and response

POST URL: http://localhost:8761/auth/signup



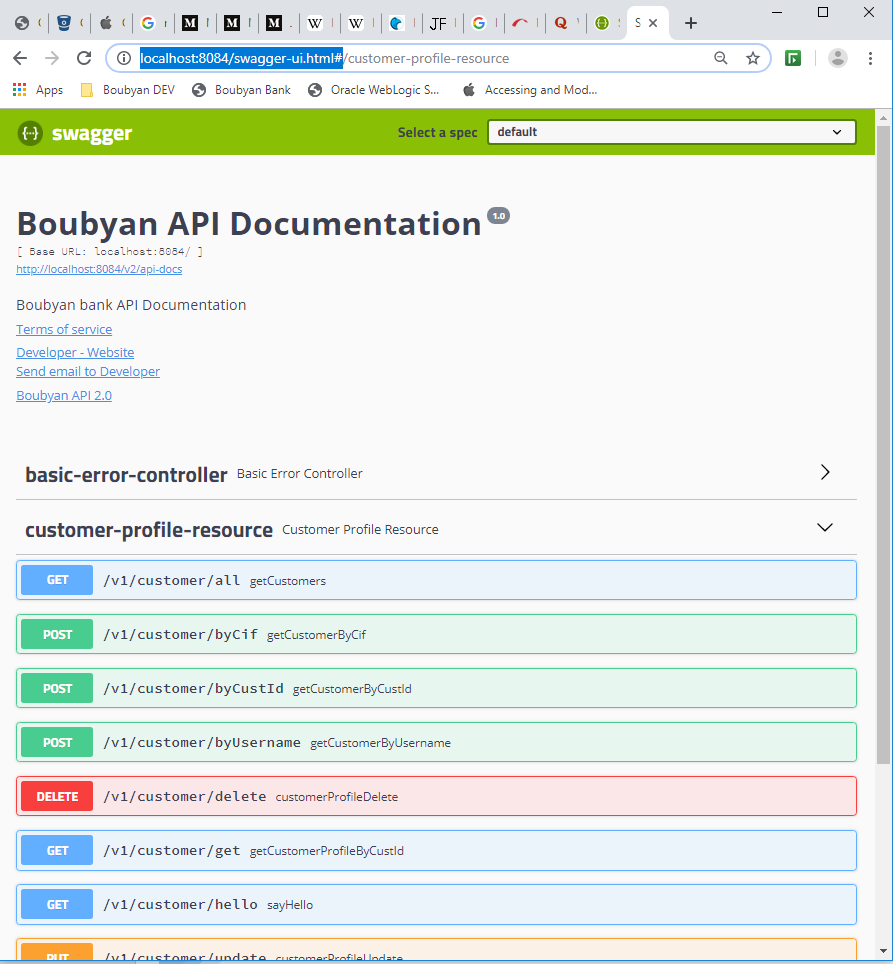
1. Login with the created new User as follow; it will generate the JSON Web Token; this token will be used for all the rest services API which is used afterwards.

POST URL: http://localhost:8761/auth/login

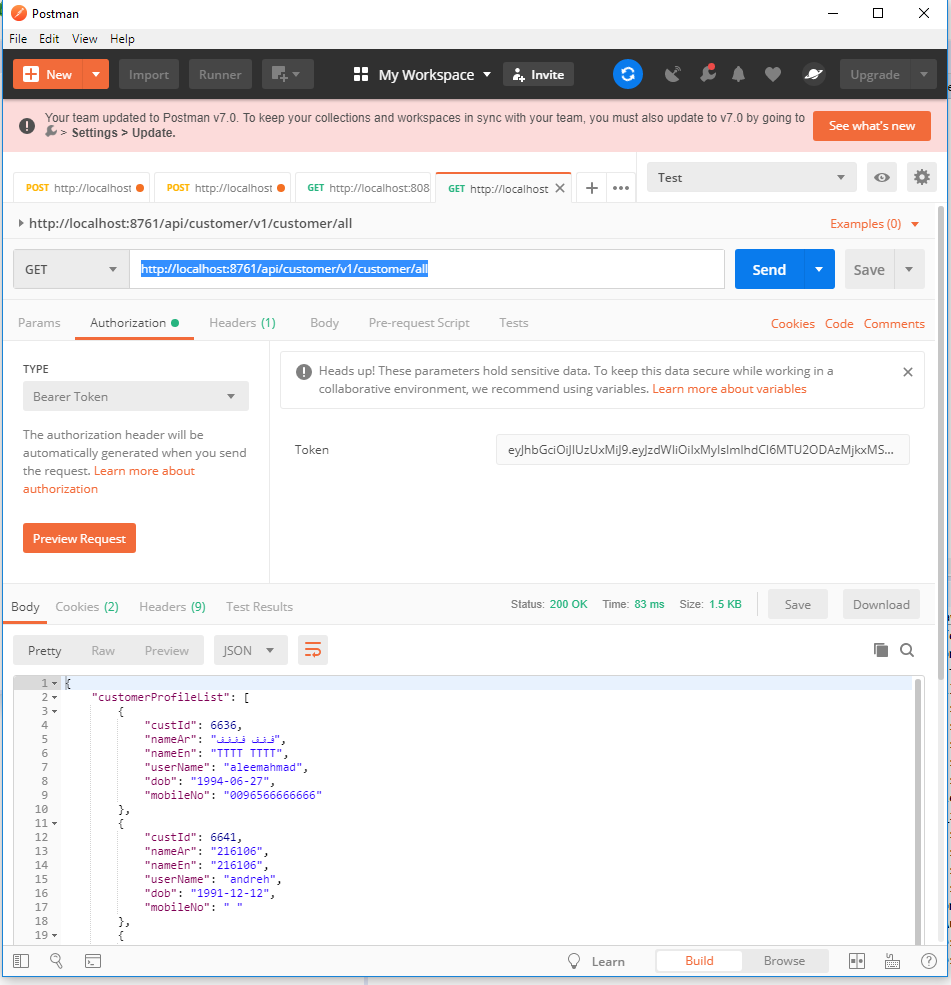


## Start Customer Micro Service

1. From STS4 right click to run the “usman-customer” as Spring Boot App.
2. If it starts successfully then open browser and type URL: [http://localhost:8084/swagger-ui.html#](http://localhost:8084/swagger-ui.html). You will see the swagger details of service.



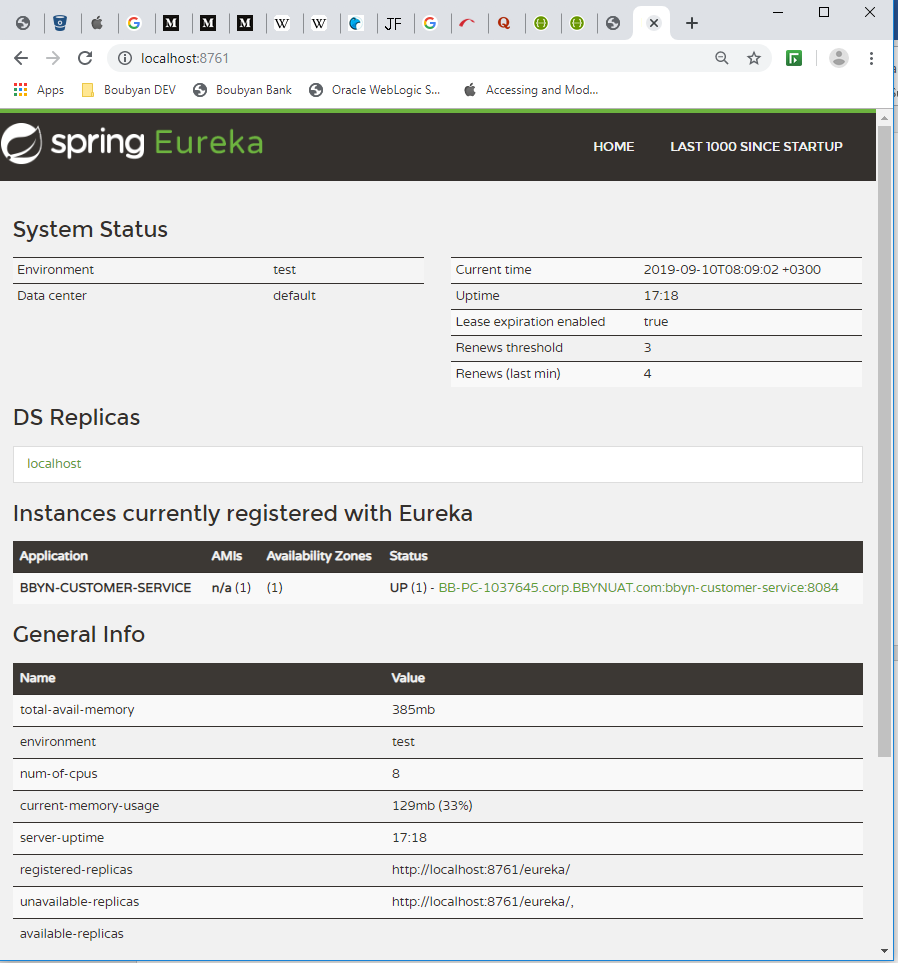
1. Get the bearer JWS token which generated by LOGIN, use this token to view the customer details via GATEWAY with the GET URL: http://localhost:8761/api/customer/v1/customer/all



## Eureka Server with Registered Instances

If you want to view the Eureka server with its registered instances you may hit this

URL: [http://localhost:8761](http://localhost:8761/)



## User (Customer) Roles Mapping

1. First create some roles is the ROLESNAME table.
2. Use the following PL/SQL to view the customer and its role.

Select \* from CUSTOMERS;

Select \* from ROLESNAME;

Select \* from CUSTOMER\_ROLESNAME;

## REST API Path Param and Variables

/\*\*

\* URL http://localhost:8761/USMANMS/rs1/customer-mgmt/test?message=ddddd

\* **@param** message

\* **@return** String

\*/

@GetMapping(value = "/test", produces = { MediaType.***APPLICATION\_JSON\_VALUE*** })

**public** ResponseEntity<String> getTest(@RequestParam("message") String msg) {

String messages= "Message requested : " + msg;

***LOG***.info(messages);

**return** ResponseEntity.*ok*(messages);

}

/\*\*

\* URL http://localhost:8761/USMANMS/rs1/customer-mgmt/msg/messagetest

\* **@param** message

\* **@return** String

\*/

@GetMapping(value = "/msg/{message}", produces = { MediaType.***APPLICATION\_JSON\_VALUE*** })

**public** ResponseEntity<String> getMsg(@PathVariable("message") String msg) {

String messages= "Message requested : " + msg;

***LOG***.info(messages);

**return** ResponseEntity.*ok*(messages);

}

## Spring boot Actuator

It provides health, audit, bean and many more information related to the services.

1. <http://localhost:8091/actuator/>
2. <http://localhost:8091/actuator/health>
3. <http://localhost:8091/actuator/info>

## Spring boot Dev tools

It is good to use this tool for run time deployment while developing the application. No need to rebuild and redeploy again and again.

## Log4j2 Configuration Details

<https://www.journaldev.com/7128/log4j2-example-tutorial-configuration-levels-appenders>