|  |  |
| --- | --- |
| **Content** | **Page** |
| **Setup up mysql database** | **P2** |
| **Improve Existing Maven project in Eclipse.** | **P3** |
| **Run the API** | **P4** |
| **Swagger API** | **P5** |
| **Test Result (By Postman)** | **P6-P9** |
| **Test Result (By Junit Test)** | **P10** |
| **Extra feature: Actuator** | **P11** |
| **Extra feature: Security** | **P12** |

**Setup up mysql database**

CREATE USER 'hbstudent'@'localhost' IDENTIFIED BY 'hbstudent';

GRANT ALL PRIVILEGES ON \* . \* TO 'hbstudent'@'localhost';

https://dev.mysql.com/doc/refman/8.0/en/caching-sha2-pluggable-authentication.html

ALTER USER 'hbstudent'@'localhost' IDENTIFIED WITH mysql\_native\_password BY 'hbstudent';

**Create User and table**

CREATE DATABASE IF NOT EXISTS `employee\_directory`;

USE `employee\_directory`;

--

-- Table structure for table `employee`

--

DROP TABLE IF EXISTS `user\_account`;

CREATE TABLE `user\_account` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`first\_name` varchar(45) DEFAULT NULL,

`last\_name` varchar(45) DEFAULT NULL,

`balance` int DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=1 DEFAULT CHARSET=latin1;

--

-- Data for table `employee`

--

INSERT INTO `user\_account` VALUES

(1,'Sunny','Man',1000)

**Improve Existing Maven project in Eclipse.**

Please follow the following steps to import the Maven project

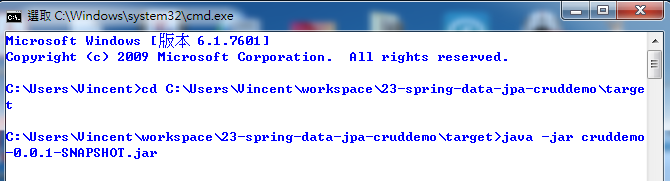
1. Open eclipse
2. Click ****File**** > ****Import****
3. Type ****Maven**** in the search box under ****Select an import source:****
4. Select ****Existing Maven Projects****
5. Click ****Next****
6. Click ****Browse**** and select the folder that is the root of the Maven project (probably contains the pom.xml file)
7. Click ****Next****
8. Click ****Finish****

**Run the API**

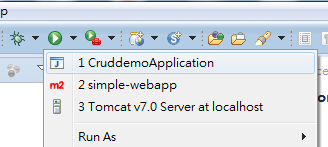
-> One Command line

>cd the target directory contains cruddemo-0.0.1-SNAPSHOT.jar

>java -jar cruddemo-0.0.1-SNAPSHOT.jar the start the application



--> On IDE e.g .eclipse , Press Run button then choose Crudemo.Application

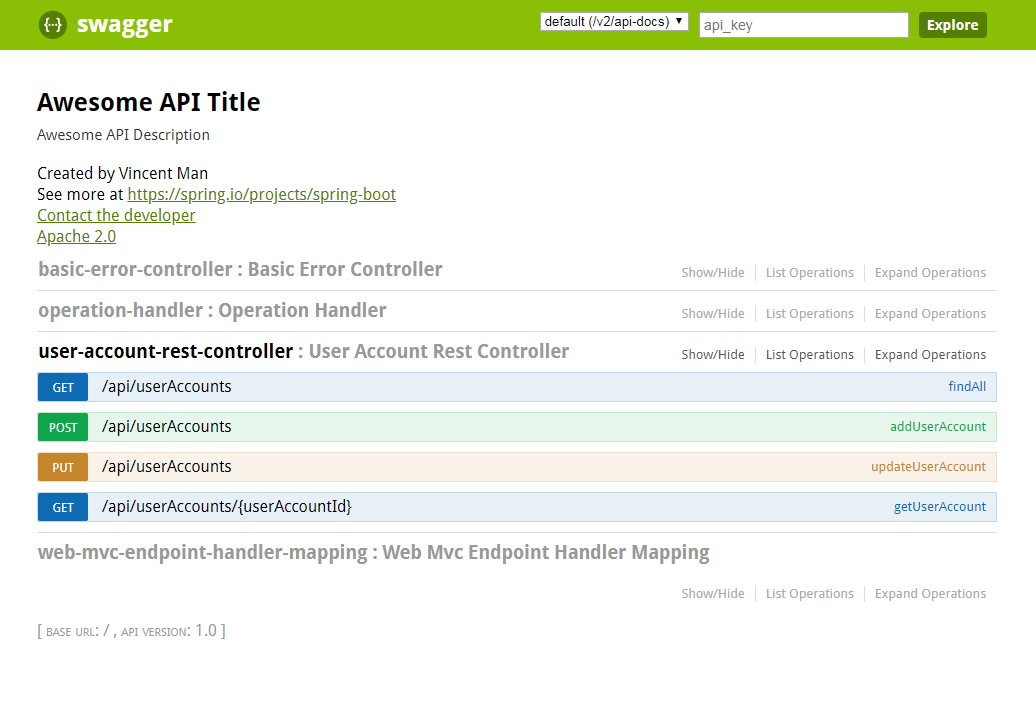


**Swagger API**

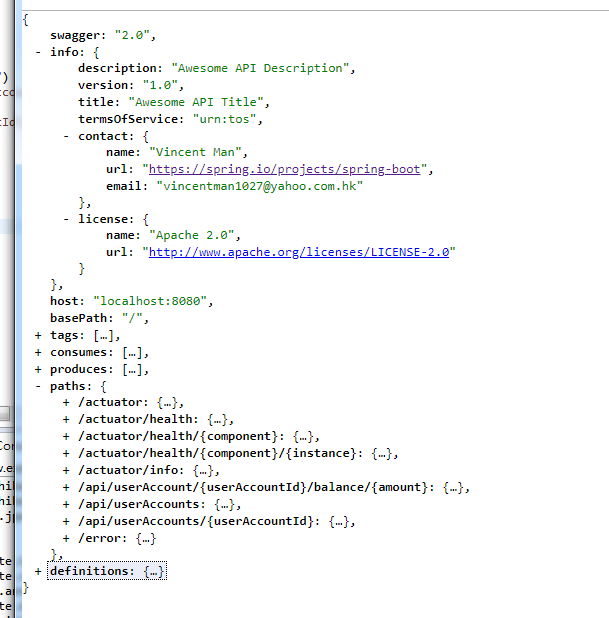
1. When application was run, Check the swagger:

<http://localhost:8080/swagger-ui.html#/>

It shows all the API of the application



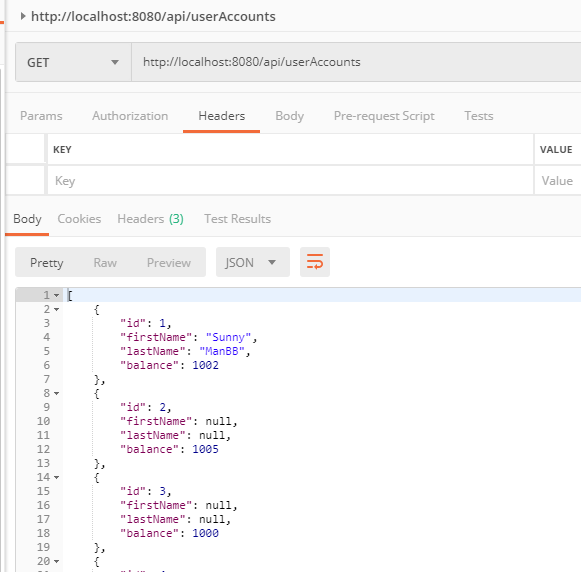
1. <http://localhost:8080/v2/api-docs,> Get the jason file



**Test Result (By Postman)**

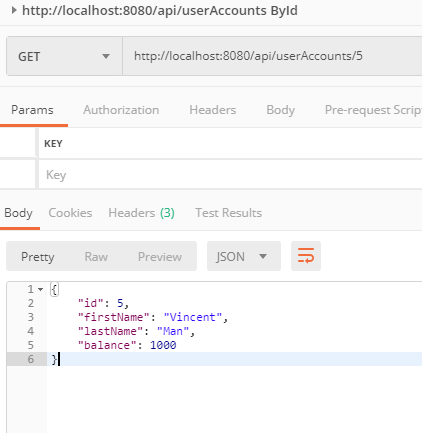
[Get] <http://localhost:8080/api/userAccounts>

Check the output result is same with db



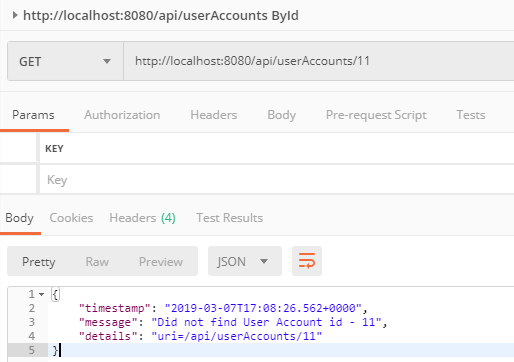
[Get] <http://localhost:8080/api/userAccounts/5>

(Check existing user)

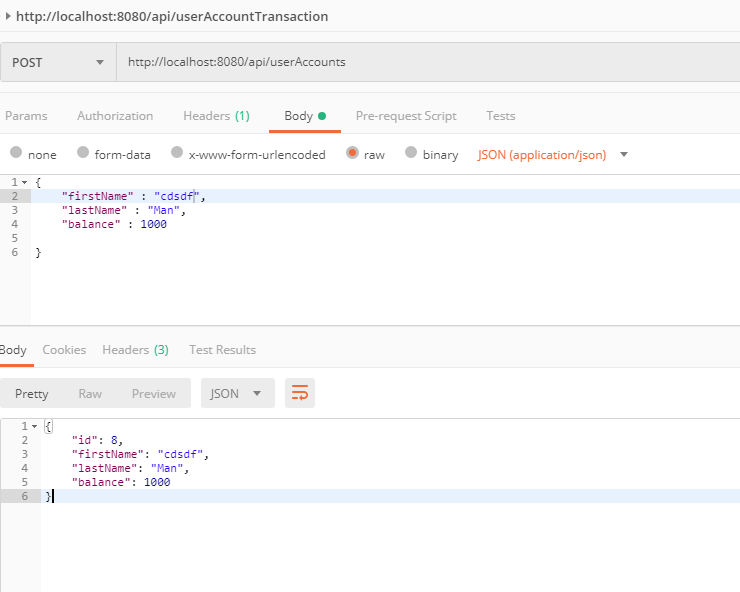


[Get] http://localhost:8080/api/userAccounts/11

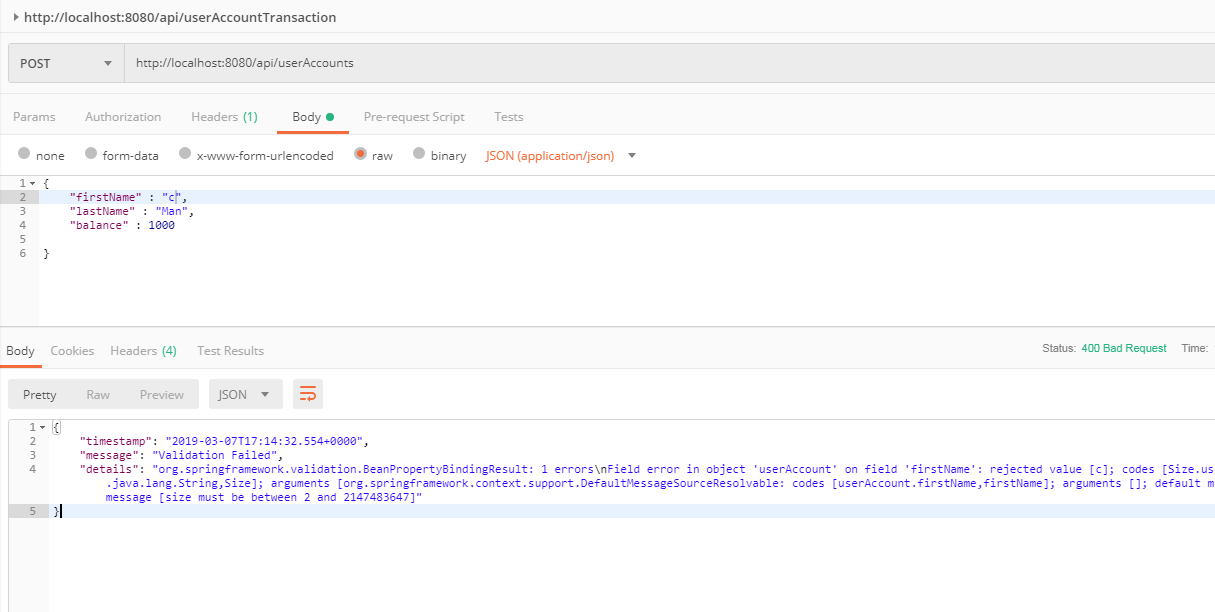
(Check non-existing user)



[POST] <http://localhost:8080/api/userAccounts>



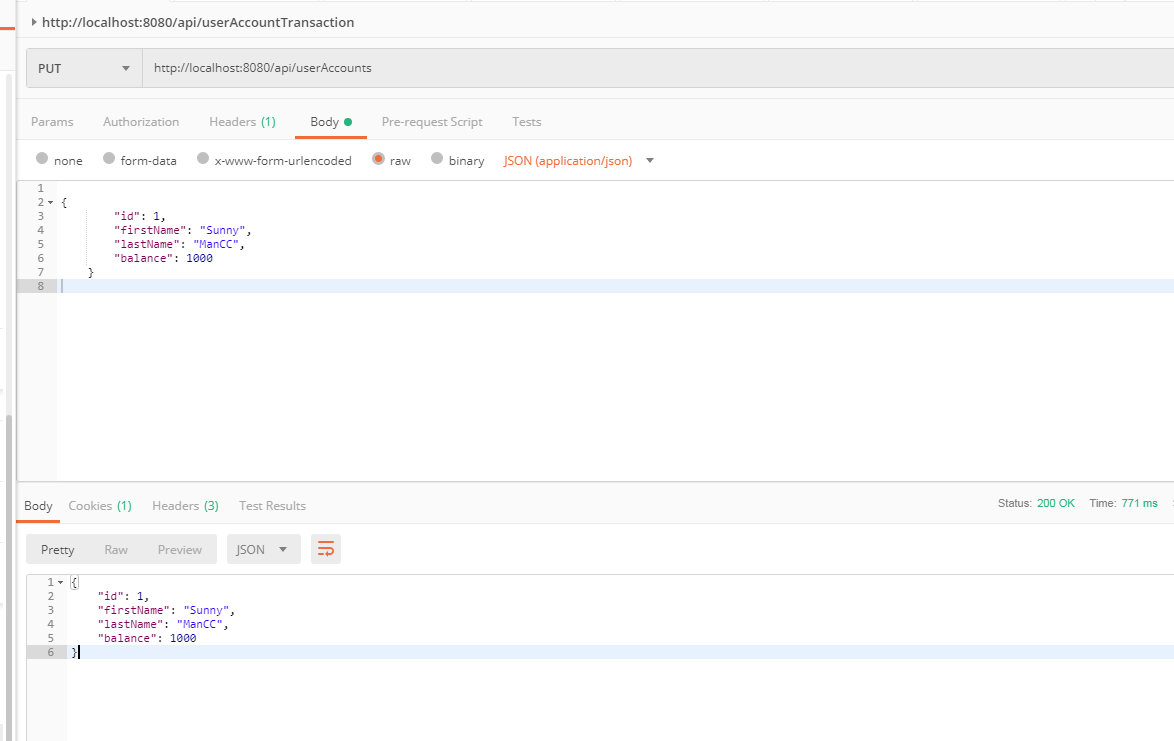
Check that only input one character will show the error as we add in the validation rule (The min length is 2)





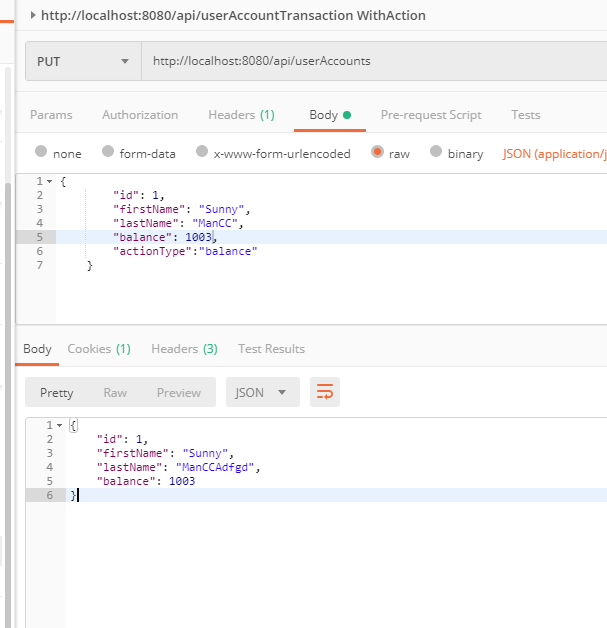
[PUT] <http://localhost:8080/api/userAccount>s

(update whole User Account)



[PUT] <http://localhost:8080/api/userAccount>s

(Update part of accountAccount by ActionType)

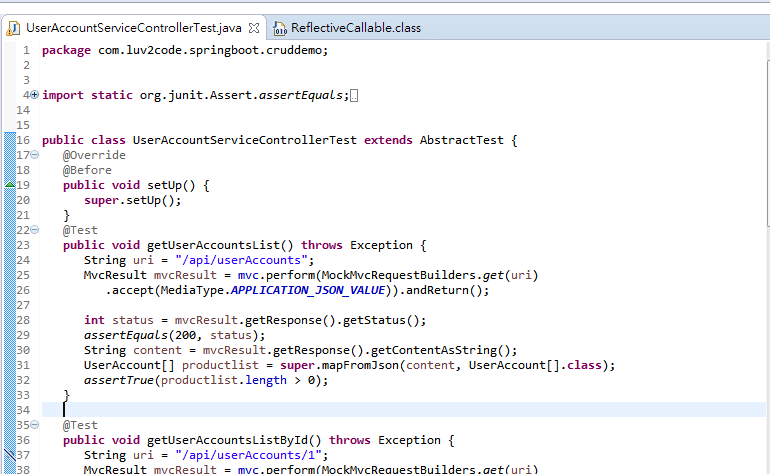


**Test Result (By Unit Test)**

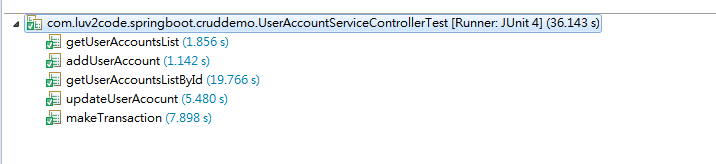
**Run the**

**/cruddemo/src/main/java/com/luv2code/springboot/cruddemo/UserAccountServiceControllerTest.java**

**Remark:** With the help of SpringJUnit4ClassRunner and MockMvc, we can create a web application context to write Unit Test for Rest Controller file

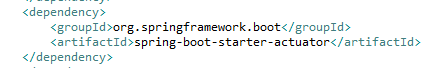


**To make sure all the api result is pass.**

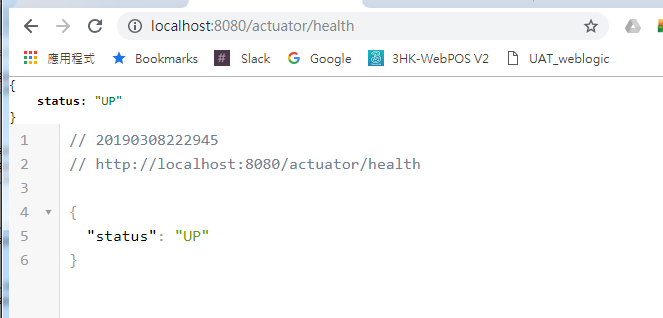


**Extra feature: Actuator**

Add the following in pom.xml to enjoy actuator feature

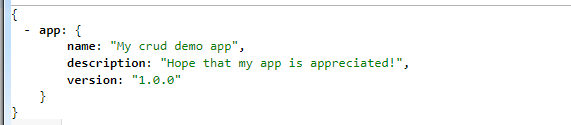


http://localhost:8080/actuator/health



Check the app info

http://localhost:8080/actuator/info

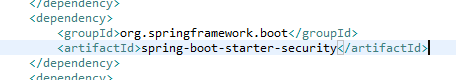


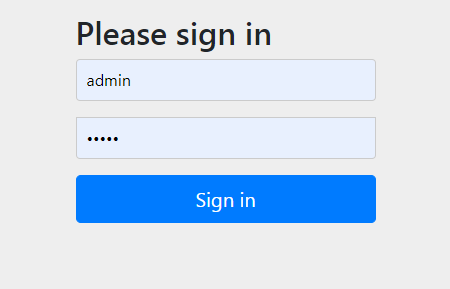
Add the following if open all the endpoints



**Extra feature: Security**

Add the following in pom.xml to enjoy Security feature





The credential information is marked down in application.properties file.

