

Bank Management – Case Study

Various SpringBoot dependencies are injected in **pom.xml** to perform spring boot functions

```
BankManagementApplication.java  BankManagement/pom.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
4    <modelVersion>4.0.0</modelVersion>
5    <parent>
6      <groupId>org.springframework.boot</groupId>
7      <artifactId>spring-boot-starter-parent</artifactId>
8      <version>2.5.0</version>
9      <relativePath/> <!-- lookup parent from repository -->
10   </parent>
11   <groupId>com.wipro</groupId>
12   <artifactId>BankManagement</artifactId>
13   <version>0.0.1-SNAPSHOT</version>
14   <name>BankManagement</name>
15   <description>Demo project for Spring Boot</description>
16   <properties>
17     <java.version>1.8</java.version>
18   </properties>
19   <dependencies>
20     <dependency>
21       <groupId>org.springframework.boot</groupId>
22       <artifactId>spring-boot-starter-web</artifactId>
23     </dependency>
24
25     <dependency>
26       <groupId>org.springframework.boot</groupId>
27       <artifactId>spring-boot-starter-test</artifactId>
28       <scope>test</scope>
29     </dependency>
30   </dependencies>
31
32   <build>
33     <plugins>
34       <plugin>
35         <groupId>org.springframework.boot</groupId>
36         <artifactId>spring-boot-maven-plugin</artifactId>
37       </plugin>
38     </plugins>
39   </build>
40
41 </project>
42
```

BankManagementApplication main class is created to initialize **@SpringBootApplication** Annotation.

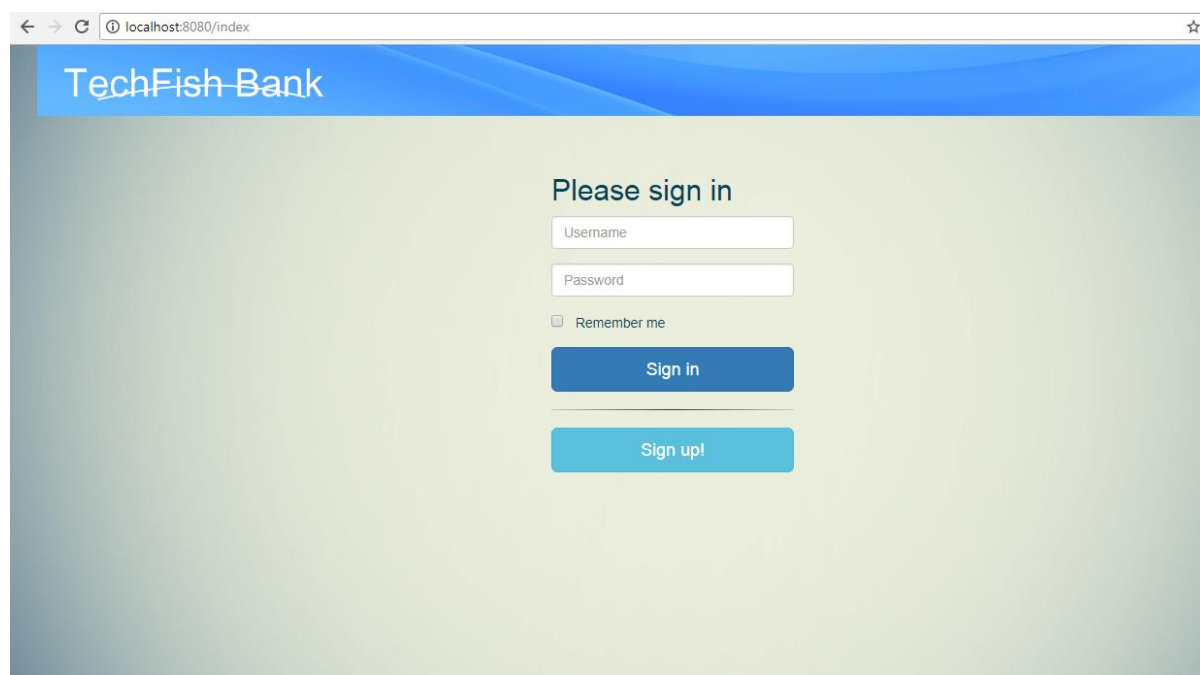
```
BankManagementApplication.java  BankManagement/pom.xml
1  package com.wipro.BankManagement;
2
3  import org.springframework.boot.SpringApplication;
4
5
6  @SpringBootApplication
7  public class BankManagementApplication {
8
9      public static void main(String[] args) {
10         SpringApplication.run(BankManagementApplication.class, args);
11     }
12
13 }
14
```

`HomeController` is used to get the page URL <http://localhost:8080/index> to get the login page.

The URL is mapped using `@RequestMapping` Annotation

```
BankManagementApplication.java  BankManagement/pom.xml
1  package com.wipro.BankManagement.controller;
2
3  import org.springframework.stereotype.Controller;
4
5
6  @Controller
7  public class HomeController {
8      @RequestMapping("/")
9      public String home() {
10         return "redirect:/index";
11     }
12
13     @RequestMapping("/index")
14     public String index() {
15         return "index";
16     }
17
18
19
20 }
21
```

The first login page of the project



localhost:8080/index

TechFish Bank

Please sign in

Username

Password

☐ Remember me

Sign in

Sign up!

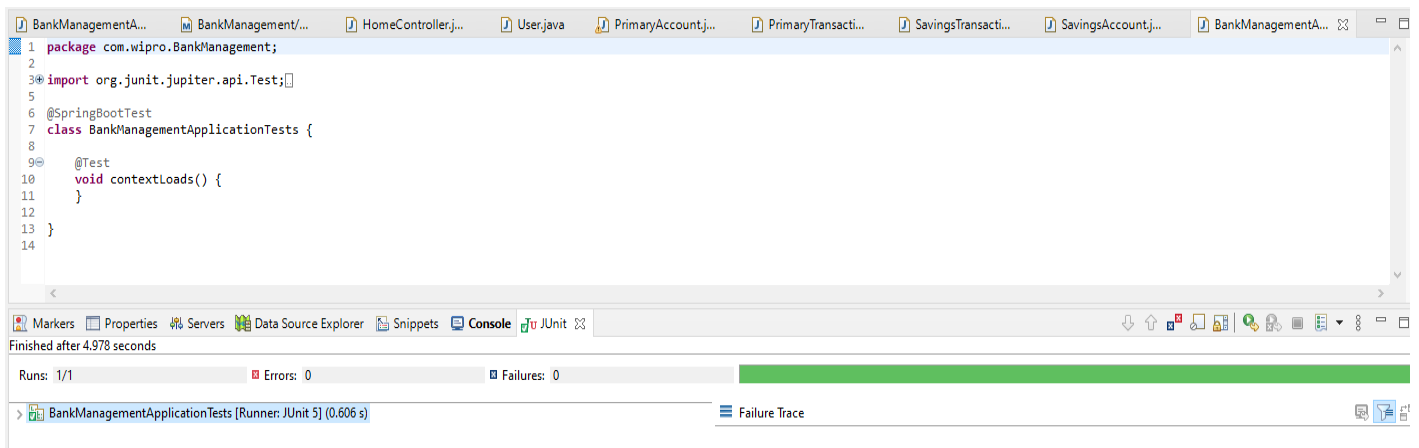
User class in the project is used to store the details of the customer in the Lists

Such as userID, username, password, firstName, LastName, email, phone

```
1 package com.wipro.BankManagement.domain;
2
3 import java.util.List;
4
5 public class User {
6     private Long userId;
7     private String username;
8     private String password;
9     private String firstName;
10    private String lastName;
11    private String email;
12    private String phone;
13
14    private boolean enabled = true;
15
16
17    private PrimaryAccount primaryAccount;
18    private SavingsAccount savingsAccount;
19    private List<Appointment> appointmentList;
20    private List<Recipient> recipientList;
21
22    public Long getUserId() {
23        return userId;
24    }
25
26    public void setUserId(Long userId) {
27        this.userId = userId;
28    }
29
30    public String getUsername() {
31        return username;
32    }
33
34    public void setUsername(String username) {
35        this.username = username;
36    }
37
38    public String getPassword() {
39        return password;
40    }
41
42    public void setPassword(String password) {
43        this.password = password;
44    }
45
46    public String getFirstName() {
47        return firstName;
48    }
49
50    public void setFirstName(String firstName) {
51        this.firstName = firstName;
52    }
53
54    public String getLastName() {
55        return lastName;
56    }
57
58    public void setLastName(String lastName) {
59        this.lastName = lastName;
60    }
61
62    public String getEmail() {
63        return email;
64    }
65
66    public void setEmail(String email) {
67        this.email = email;
68    }
69
70    public String getPhone() {
71        return phone;
72    }
73
74    public void setPhone(String phone) {
75        this.phone = phone;
76    }
77
78    public boolean isEnabled() {
79        return enabled;
80    }
81
82    public void setEnabled(boolean enabled) {
83        this.enabled = enabled;
84    }
85
86    public PrimaryAccount getPrimaryAccount() {
87        return primaryAccount;
88    }
89
90    public void setPrimaryAccount(PrimaryAccount primaryAccount) {
91        this.primaryAccount = primaryAccount;
92    }
93
94    public SavingsAccount getSavingsAccount() {
95        return savingsAccount;
96    }
97
98    public void setSavingsAccount(SavingsAccount savingsAccount) {
99        this.savingsAccount = savingsAccount;
100    }
101
102    public List<Appointment> getAppointmentList() {
103        return appointmentList;
104    }
105
106    public void setAppointmentList(List<Appointment> appointmentList) {
107        this.appointmentList = appointmentList;
108    }
109
110    public List<Recipient> getRecipientList() {
111        return recipientList;
112    }
113
114    public void setRecipientList(List<Recipient> recipientList) {
115        this.recipientList = recipientList;
116    }
117
118    @Override
119    public String toString() {
120        return "User [userId=" + userId + ", username=" + username + ", password=" + password + ", firstName="
121            + firstName + ", lastName=" + lastName + ", email=" + email + ", phone=" + phone + ", enabled="
122            + enabled + "]\n";
123    }
124
125 }
```

Various classes **SavingsAccount**, **SavingsTransaction**, **Receipt** are used to perform the transaction among the accounts.

BankManagementApplicationTest is the testing class which is used to run the Junit testing using **@test** Annotation



The screenshot displays an IDE window with a Java test class named `BankManagementApplicationTests`. The code is as follows:

```
1 package com.wipro.BankManagement;
2
3 import org.junit.jupiter.api.Test;
4
5
6 @SpringBootTest
7 class BankManagementApplicationTests {
8
9     @Test
10     void contextLoads() {
11     }
12 }
13
14
```

Below the code editor, the JUnit console shows the test results:

- Finished after 4.978 seconds
- Runs: 1/1
- Errors: 0
- Failures: 0

The bottom of the console shows the test class name and runner: `BankManagementApplicationTests [Runner: JUnit 5] (0.606 s)`. A "Failure Trace" tab is also visible on the right.