Pension Management System-Mini Project

By

Sesha Gayathri Garimella

Guided By

Mr. Kumar

Abstract

Given the discouraging record of the traditional pension system and the limited coverage of the private-sector pension schemes, there is need for robust web-based pension fund management system that can handle the pension process efficiently. This work focus on the design and implementation of a web-based pension fund management application to replace the manual system, thus getting rid of the hurdles involved in the traditional pension management process.

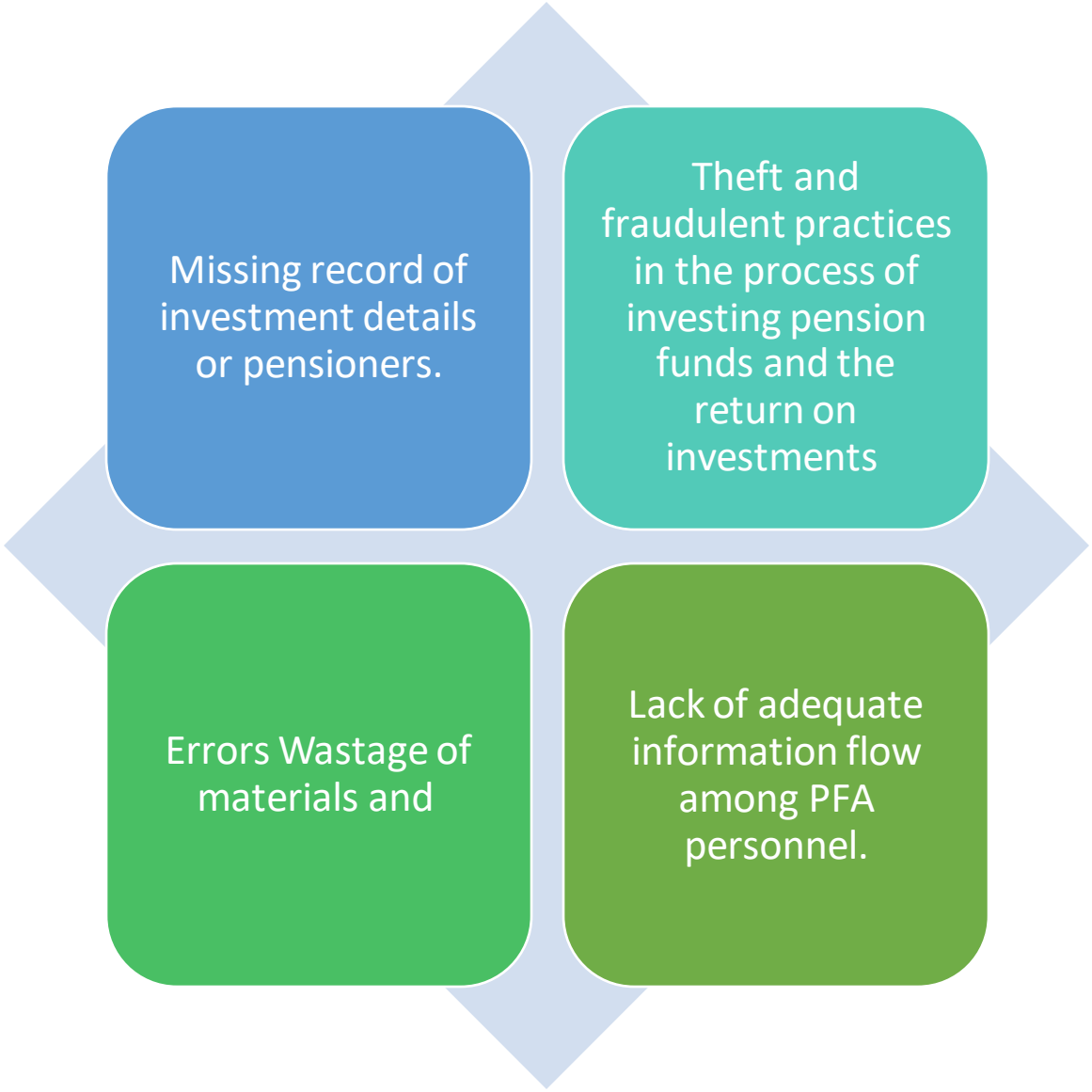
Introduction

Pension is an arrangement of providing people with an income when they are no longer earning a regular income from employment. However, pensions should not be confused with severance pay; the former is paid in regular installments, while the latter is paid in one lump sum. In general, the common use of the term pension is to describe the payments a person receives upon retirement, usually under predetermined legal and/or contractual terms.

Existing System

The manual method of pension fund management maintains or keeps records in a filing cabinet about each employee/pensioner that is registered with a Pension Fund Administrator (PFA). When an employee decides to open an account with a PFA for the purpose of retirement savings, the former has to go to the physical location of the PFA to obtain and fill a registration form. The file is then passed on to the next PFA personnel to verify the employee's registration which may take some days if not weeks ,as the case may be.

Disadvantages of Existing



Proposed System

- As a result of the problems encountered in the existing system of operations in pension fund administration which include too many people involved in the administrative process and unnecessary paper work, an improved web-based-Spring Boot pension fund administration developed eradicates the shortcomings of the existing system. To have this pension management system I have used Spring Boot. As this is a spring boot project , there is no need to add items to tomcat. I have used Maven in this project.

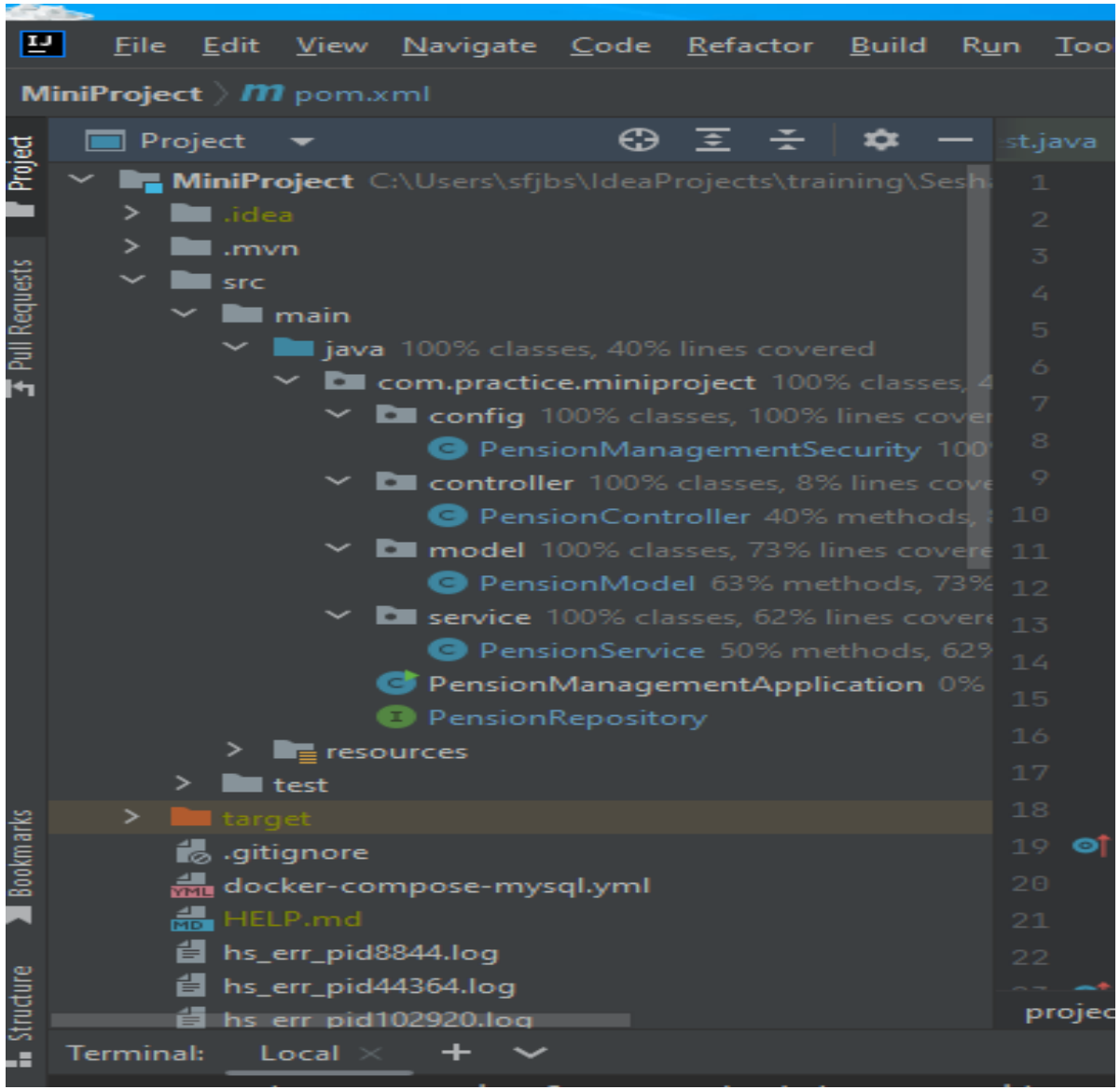
Spring Boot

- Spring Boot is **an open source Java-based framework used to create a micro Service**. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications.

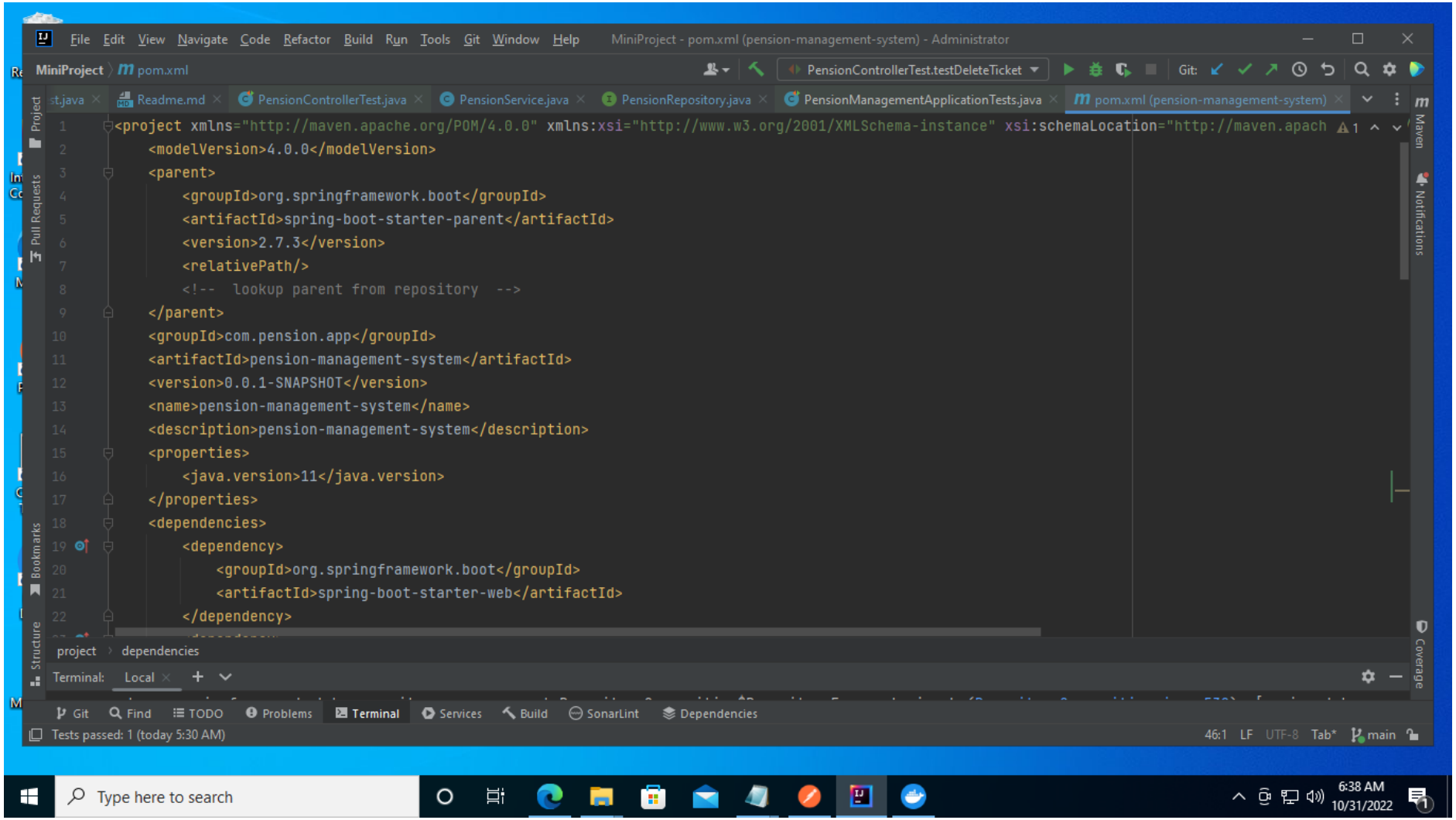
Advantages of Spring Boot

- Easy to start with
- Simple setup and management
- More accessible and much faster development Spring-based applications
- No need for the deployment of war files as which is a collection of JAR-files, XML-files, Java server pages, tag libraries, etc.
- It has embeded servers like tomcat,jetty etc.
- Its distinct ability to create standalone applications.
- It offers the number of **plug-ins**.

Project Structure



pom.xml


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
MiniProject - pom.xml (pension-management-system) - Administrator
MiniProject pom.xml
PensionControllerTest.testDeleteTicket
Git:
Readme.md
PensionControllerTest.java
PensionService.java
PensionRepository.java
PensionManagementApplicationTests.java
pom.xml (pension-management-system)
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apach
<modelVersion>4.0.0</modelVersion>
<parent>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>2.7.3</version>
<relativePath/>
<!-- lookup parent from repository -->
</parent>
<groupId>com.pension.app</groupId>
<artifactId>pension-management-system</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>pension-management-system</name>
<description>pension-management-system</description>
<properties>
<java.version>11</java.version>
</properties>
<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>
project dependencies
Terminal: Local
Git Find TODO Problems Terminal Services Build SonarLint Dependencies
Tests passed: 1 (today 5:30 AM)
46:1 LF UTF-8 Tab* main
Type here to search
6:38 AM
10/31/2022

Implementation

REST stands for REpresentational State Transfer.REST is an architectural style not a protocol. **A Spring Boot RESTful service is typically divided into three layers: Repository, Service, and Controller**. This layering helps to segregate the RESTful application responsibilities and enabling loose coupling between the objects. Can use postman to perform CRUD operations.
I provided security by using config and issued user role and password, so that only the intended user can access the appropriate method/functionality.

CRUD

- CRUD stands for Create, Read/Retrieve, Update and Delete and these are the four basic operations that we perform on persistence storage.
  So, standard CRUD Operations are as follows:
- POST: Creates a new resource.
- GET: Reads/Retrieve a resource.
- PUT: Updates an existing resource.
- DELETE: Deletes a resource.

Methods

- In Service

Get All Pension

Get Pension By Id

Delete Applicant

In Controller

Create_Applicant

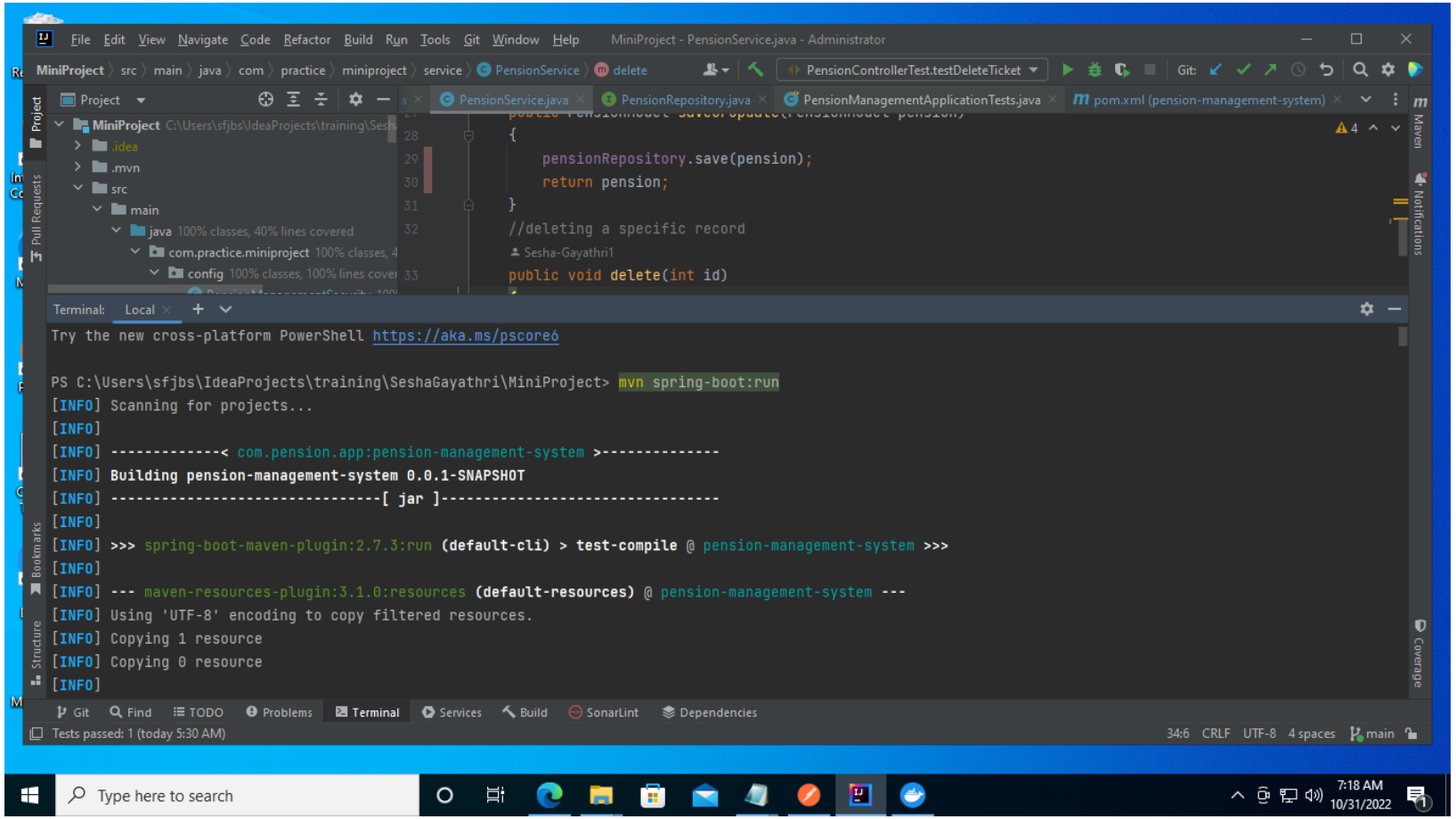
Edit Applicant

Get Status

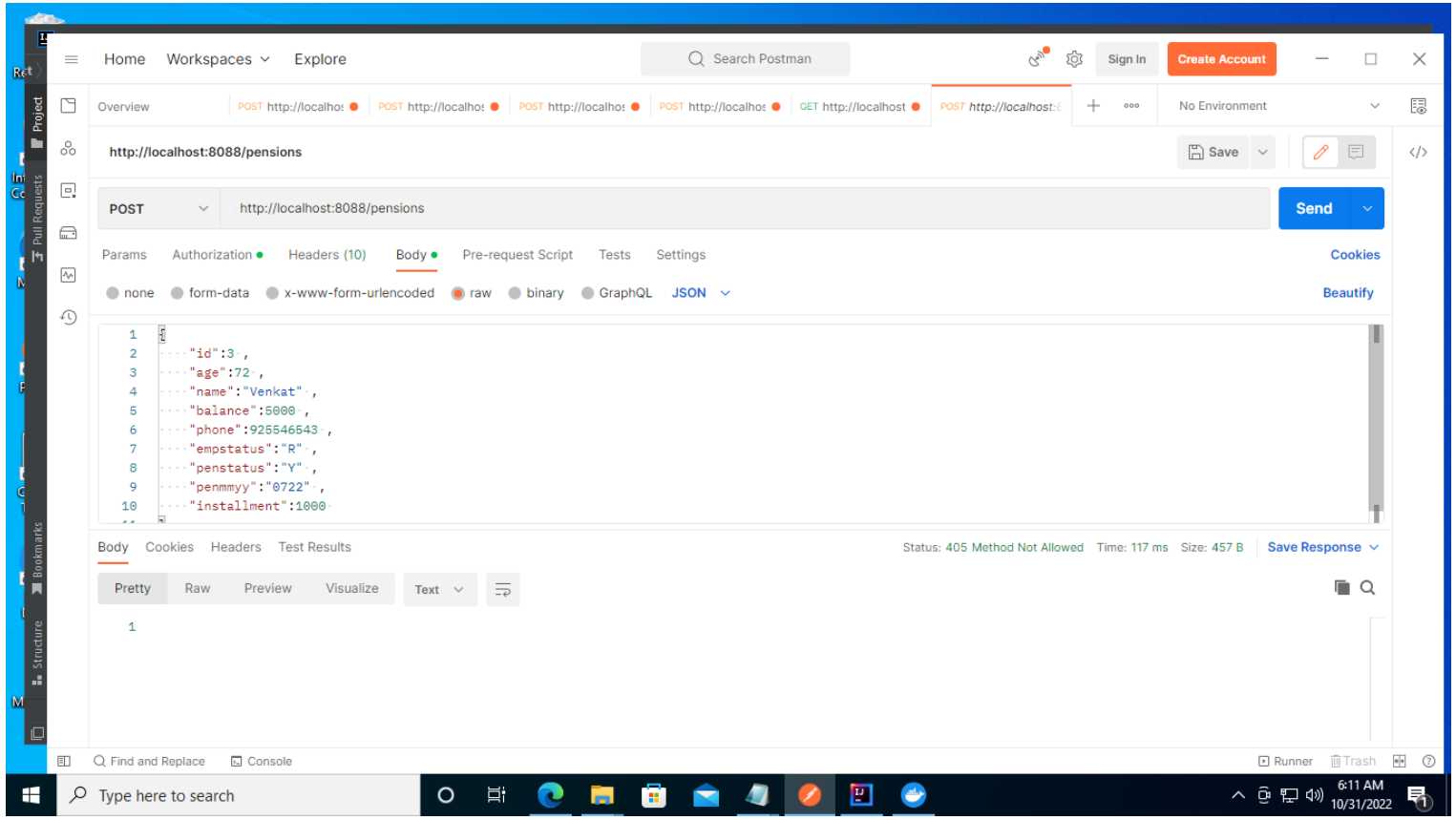
And many more....

Screenshots

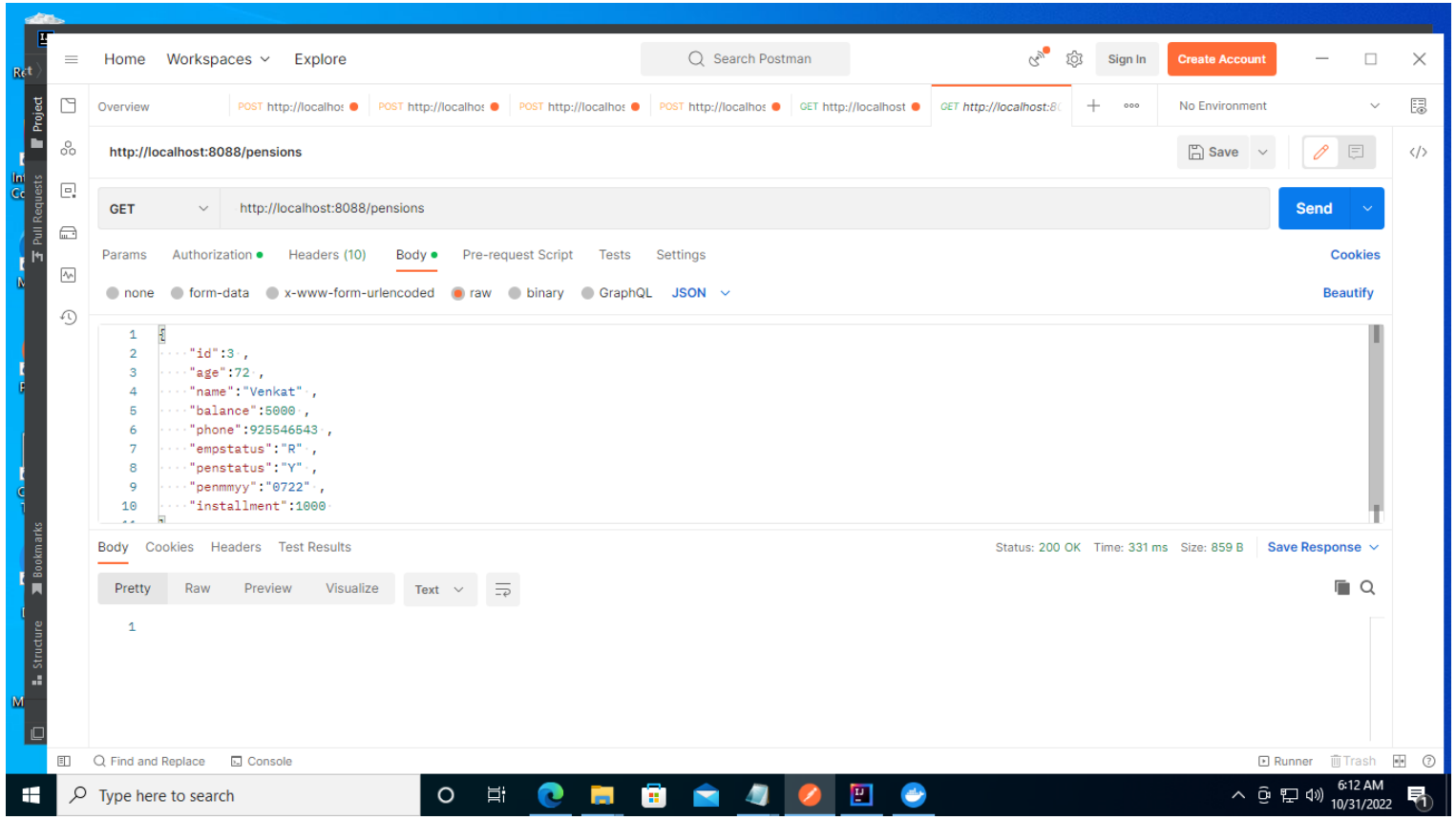
Run



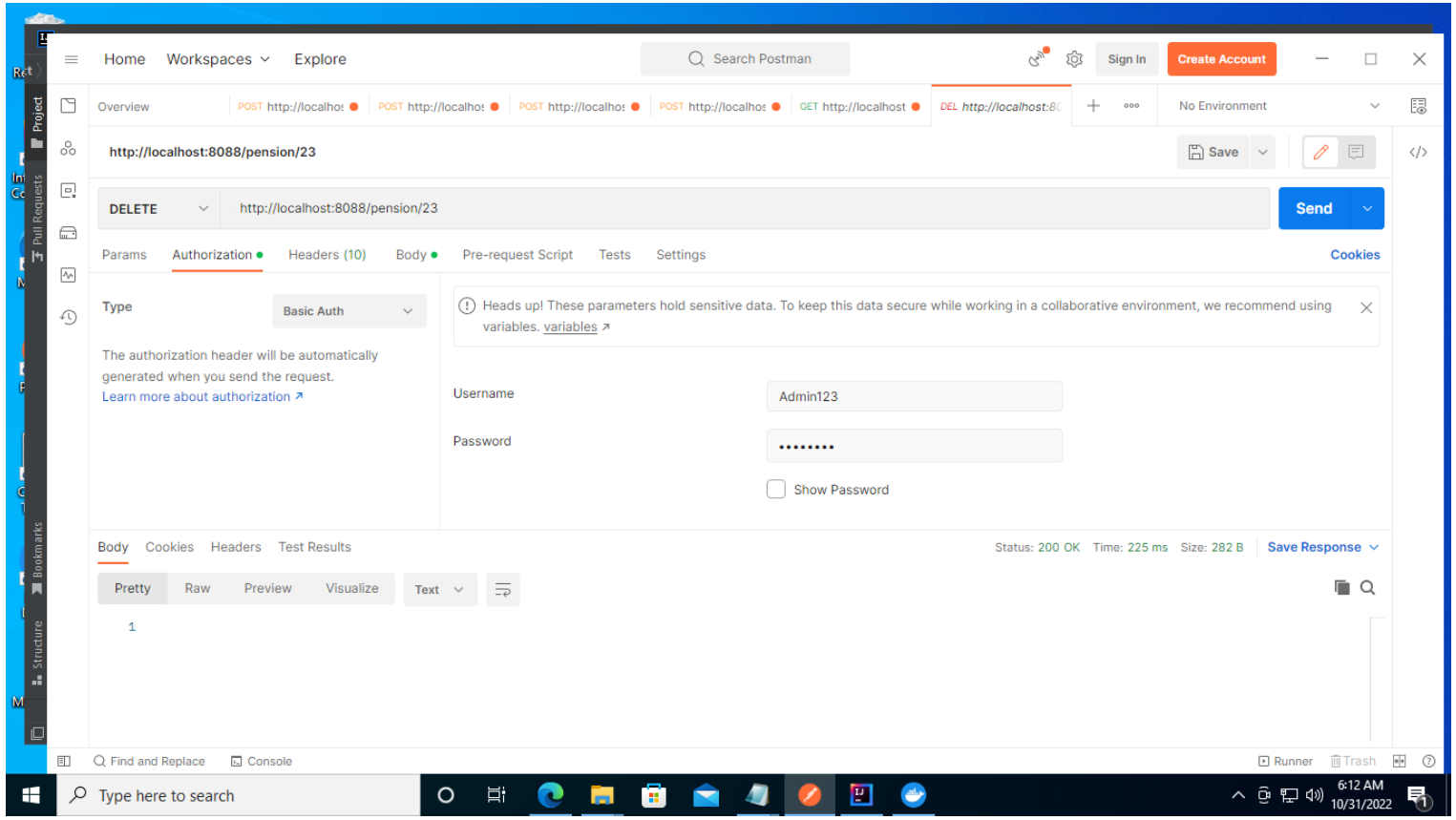
POST


Home
Workspaces
Explore
Search Postman
Sign In
Create Account
Overview
No Environment
http://localhost:8088/pensions
Save
POST
http://localhost:8088/pensions
Send
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id":3,
"age":72,
"name":"Venkat",
"balance":5000,
"phone":925546543,
"empstatus":"R",
"penstatus":"Y",
"penmmyy":"0722",
"installment":1000
Body
Cookies
Headers
Test Results
Status: 405 Method Not Allowed
Time: 117 ms
Size: 457 B
Save Response
Pretty
Raw
Preview
Visualize
Text
Find and Replace
Console
Runner
Trash
Type here to search
6:11 AM
10/31/2022

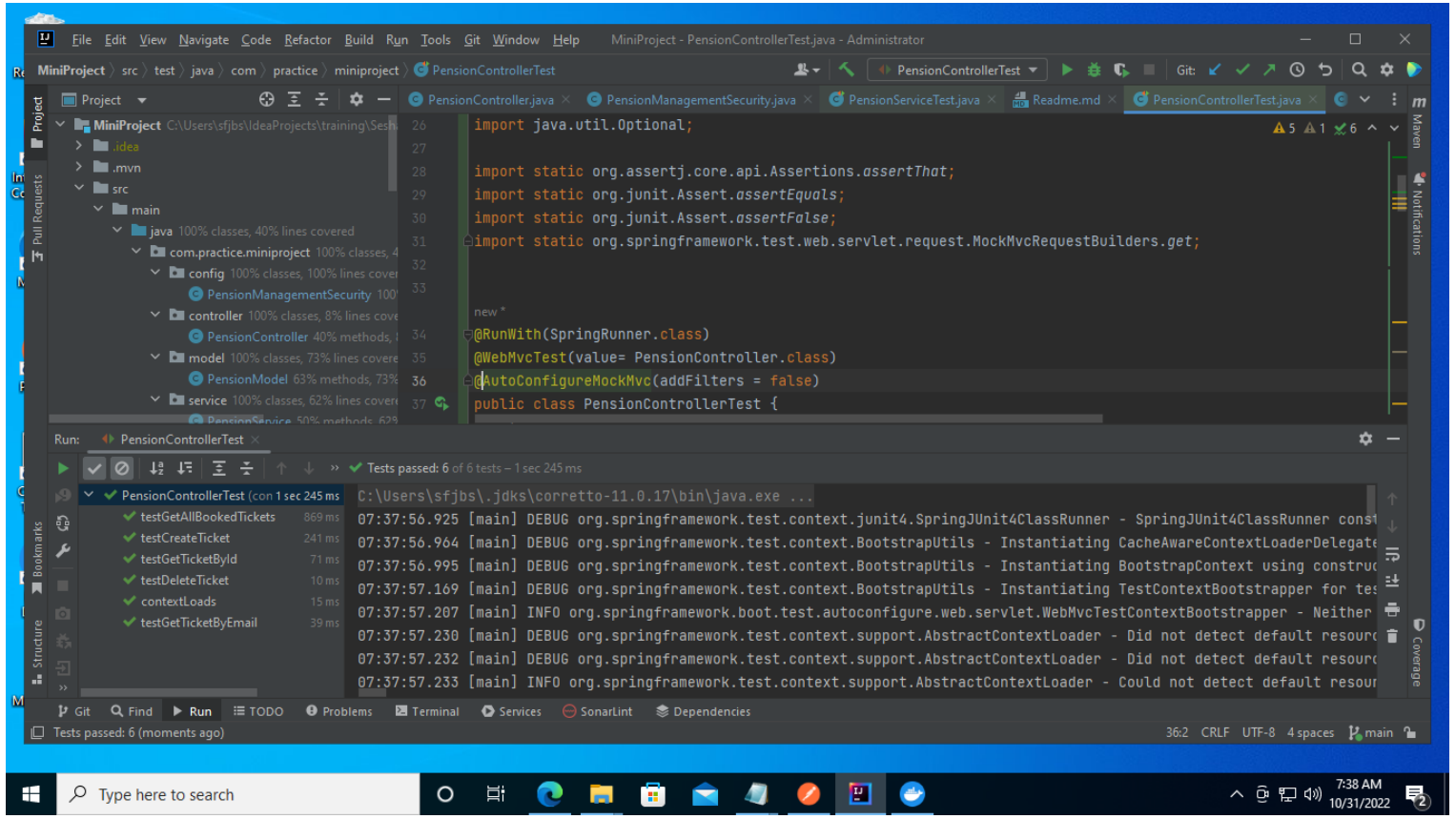
GET

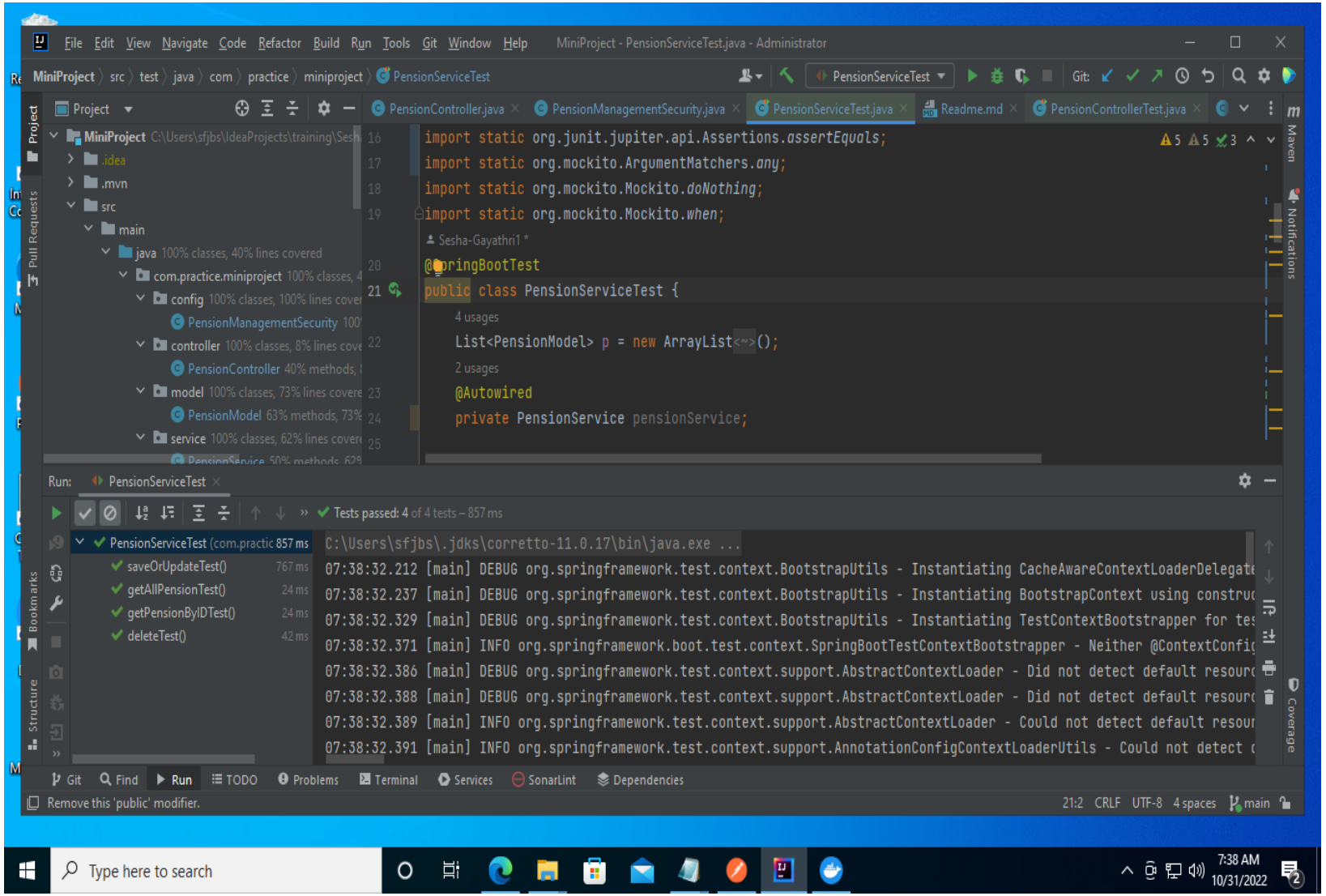

Home
Workspaces
Explore
Search Postman
Sign In
Create Account
Overview
No Environment
http://localhost:8088/pensions
Save
GET
http://localhost:8088/pensions
Send
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id":3,
"age":72,
"name":"Venkat",
"balance":5000,
"phone":925546543,
"empstatus":"R",
"penstatus":"Y",
"penmmyy":"0722",
"installment":1000
Body
Cookies
Headers
Test Results
Status: 200 OK
Time: 331 ms
Size: 859 B
Save Response
Pretty
Raw
Preview
Visualize
Text
Find and Replace
Console
Runner
Trash
Type here to search
6:12 AM
10/31/2022

DELETE


Home
Workspaces
Explore
Search Postman
Sign In
Create Account
Overview
No Environment
http://localhost:8088/pension/23
Save
DELETE
http://localhost:8088/pension/23
Send
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Cookies
Type
Basic Auth
The authorization header will be automatically generated when you send the request.
Learn more about authorization
Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. variables
Username
Admin123
Password
Show Password
Body
Cookies
Headers
Test Results
Status: 200 OK
Time: 225 ms
Size: 282 B
Save Response
Pretty
Raw
Preview
Visualize
Text
1
Find and Replace
Console
Runner
Trash
Type here to search
6:12 AM
10/31/2022

Test Cases


MiniProject - PensionControllerTest.java - Administrator
import java.util.Optional;
import static org.assertj.core.api.Assertions.assertThat;
import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertFalse;
import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.get;
@RunWith(SpringRunner.class)
@WebMvcTest(value= PensionController.class)
@AutoConfigureMockMvc(addFilters = false)
public class PensionControllerTest {
Tests passed: 6 of 6 tests – 1 sec 245 ms
testGetAllBookedTickets
testCreateTicket
testGetTicketById
testDeleteTicket
contextLoads
testGetTicketByEmail
07:37:56.925 [main] DEBUG org.springframework.test.context.junit4.SpringJUnit4ClassRunner - SpringJUnit4ClassRunner
07:37:56.964 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating CacheAwareContextLoaderDelegate
07:37:56.995 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating BootstrapContext using
07:37:57.169 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating TestContextBootstrapper for
07:37:57.207 [main] INFO org.springframework.boot.test.autoconfigure.web.servlet.WebMvcTestContextBootstrapper - Neither
07:37:57.230 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default
07:37:57.232 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default
07:37:57.233 [main] INFO org.springframework.test.context.support.AbstractContextLoader - Could not detect default
Tests passed: 6 (moments ago)
Type here to search
7:38 AM
10/31/2022

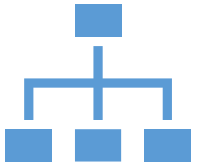

File Edit View Navigate Code Refactor Build Run Tools Git Window Help
MiniProject - PensionServiceTest.java - Administrator
MiniProject src test java com practice miniproject PensionServiceTest
PensionServiceTest
Git:
Project
PensionController.java
PensionManagementSecurity.java
PensionServiceTest.java
Readme.md
PensionControllerTest.java
MiniProject C:\Users\sfjbs\IdeaProjects\training\Sesh
.idea
.mvn
src
main
java 100% classes, 40% lines covered
com.practice.miniproject 100% classes,
config 100% classes, 100% lines cover
PensionManagementSecurity 100
controller 100% classes, 8% lines cove
PensionController 40% methods,
model 100% classes, 73% lines covere
PensionModel 63% methods, 73%
service 100% classes, 62% lines covere
16 import static org.junit.jupiter.api.Assertions.assertEquals;
17 import static org.mockito.ArgumentMatchers.any;
18 import static org.mockito.Mockito.doNothing;
19 import static org.mockito.Mockito.when;
Sesha-Gayathri1 *
20 @SpringBootTest
21 public class PensionServiceTest {
4 usages
22 List<PensionModel> p = new ArrayList<>();
2 usages
23 @Autowired
24 private PensionService pensionService;
25
Run: PensionServiceTest
Tests passed: 4 of 4 tests – 857 ms
PensionServiceTest (com.practic 857 ms
saveOrUpdateTest() 767 ms
getAllPensionTest() 24 ms
getPensionByIDTest() 24 ms
deleteTest() 42 ms
C:\Users\sfjbs\.jdks\corretto-11.0.17\bin\java.exe ...
07:38:32.212 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating CacheAwareContextLoaderDelegat
07:38:32.237 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating BootstrapContext using constru
07:38:32.329 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating TestContextBootstrapper for tes
07:38:32.371 [main] INFO org.springframework.boot.test.context.SpringBootTestContextBootstrapper - Neither @ContextConfig
07:38:32.386 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default resourc
07:38:32.388 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default resourc
07:38:32.389 [main] INFO org.springframework.test.context.support.AbstractContextLoader - Could not detect default resour
07:38:32.391 [main] INFO org.springframework.test.context.support.AnnotationConfigContextLoaderUtils - Could not detect
Git Find Run TODO Problems Terminal Services SonarLint Dependencies
Remove this 'public' modifier.
21:2 CRLF UTF-8 4 spaces main
Type here to search
7:38 AM
10/31/2022

Conclusion

In conclusion, the introduction of the new system to replace the existing one is definitely a way of maintaining a good management practice. On successful implementation, the system will correct the limitations identified in the existing system as previously stated. It is important to know that for the web-based system to work effectively.

Good environment.

Good maintenance practices.



Good administration should be provided.

Thank You