Virtusa PNC Team

PNC Bank | Virtusa Corp

PNC Java Developer Training Manual

Version 0.2

Contents

[Introduction 3](#_Toc492990964)

[How it works? 3](#_Toc492990965)

[Time Lines 3](#_Toc492990966)

[Spring Mandatory Knowledge Requirements 3](#_Toc492990967)

[Spring Core 3](#_Toc492990968)

[Spring MVC 3](#_Toc492990969)

[Spring REST 3](#_Toc492990970)

[Exercise 1 4](#_Toc492990971)

[Spring Data JPA 4](#_Toc492990972)

[Hibernate JPA 4](#_Toc492990973)

[Spring AOP 4](#_Toc492990974)

[Exercise 2: 4](#_Toc492990975)

[Exercise 3 6](#_Toc492990976)

[Knowledge Requirements for Micro services Based Projects [Open API & Dev Console] 6](#_Toc492990977)

[Spring Cloud 6](#_Toc492990978)

[OAuth 6](#_Toc492990979)

[Netflix OSS 6](#_Toc492990980)

[ELK Stack 6](#_Toc492990980)

[Existing Project Documentation 7](#_Toc492990981)

[Point of Contact 7](#_Toc492990982)

[Knowledge Requirements for Resiliency Project 8](#_Toc492990983)

[Hystrix 8](#_Toc492990984)

[Point of Contact 8](#_Toc492990985)

[Knowledge Requirements for OLB and PNC Backend Project 9](#_Toc492990986)

[What? 9](#_Toc492990987)

[Point of Contact 9](#_Toc492990988)

[PNC Coding Standards and Guidelines 9](#_Toc492990989)

[References 9](#_Toc492990990)

# Introduction

This is a training manual for any java developer who is joining PNC projects. First section is common for all the Java developers. Later sections focus on the project based areas of focus.

# How it works?

The training manual has references to certain books and also to certain sections of original documentations, which you will have to read, understand and master. The assumption is that you are familiar with other basics that is needed for you to read and understand these sections. If you lack that basic knowledge, please refer required documentation and prepare yourselves for these sections.

Alongside the reading, we have also provided the exercise that you will have to implement using what you have learnt in that section. These exercises are designed in a way that when you finish this training manual, you will have an end to deployable application ready. Your learning will be evaluated for each section using the exercise. And finally on how everything ties together and works as one single application.

# Time Lines

Common sections should be completed in 2 weeks and project related sections should be completed in 1 – 2 weeks depending on the project. These timelines are tested and works well.

# Spring Mandatory Knowledge Requirements

### Spring Core

1. Ref : Spring in action : chap 2 & 4
2. <http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/#spring-core>

### Spring MVC

1. Ref : Spring in action : chap 5 & 6
2. <http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/#spring-web>

### Spring REST

1. REf : Spring in action : chap : 16

### Exercise 1

After completing the above topics now you are capable of creating a spring application. Follow the below steps to create a User application with one RESTful endpoint which provides user information by adhering to below conditions

* 1. Method: should be able to reason-out the choice
  2. Resource endpoint: come up with meaningful endpoint by following REST standard
  3. Produces: application/json
  4. Response Body:200 OK and with proper JSON response structure(please refer JSend for more information)
  5. Handle exception with meaningful JSON structure

### Spring Data JPA

1. <http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/#orm-jpa>
2. Ref : Spring in action chapter : 11.2 & 11.3

### Hibernate JPA

1. <https://docs.jboss.org/hibernate/orm/3.6/quickstart/en-US/html/hibernate-gsg-tutorial-jpa.html>

After completing the Data JPA now you are able to interact with database, create an embedded H2 database and enable console for it.

1. At application startup, load 3 users with dummy values in to H2 DB
2. Expose a GET endpoint to retrieve the details of all the users from DB

### Spring AOP

1. <http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/#aop>
2. <http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/#aop-api>
3. Ref : Spring in action : chap : 4

### Exercise 2:

Now you created a user application, which used to retrieve user details from H2 database from rest endpoint. Follow the same steps to create an account application which retrieves the account information from database for each user. Please read the below steps carefully and make reasonable assumption for anything not explicitly specified.

**Account Application:** Create an application with one RESTful endpoint which provides Account information for specified user from H2 embedded database

a. **Method:** should be able to reason-out the choice

b. **Resource endpoint:** come up with meaningful endpoint by following REST standard

1. **Produces:** application/json
2. At application startup, load 3 set of accounts for 3 users already available in the above user application with dummy values in to H2 DB
3. **Response Body:**200 OK and with proper JSON response structure(please refer JSend for more information)
4. Handle exception with meaningful JSON structure
5. Expose a GET endpoint to retrieve the details of all the accounts tagged to the specified user from DB.

**User Aggregate Application:** Create an application with one RESTful endpoint which consumes endpoint exposed by user application to get the list of users available in H2 DB and iterate through the list of user to get the account that are tagged to each user by consuming the endpoint exposed by account application.

* 1. **Method:** should be able to reason-out the choice
  2. **Resource endpoint:** come up with meaningful endpoint by following REST standard
  3. **Produces:** application/json
  4. **Response body:** 200 OK and with proper JSON response structure(please refser JSend for more information)
  5. Create an aggregate service which produces the list of users and with each user having the list of accounts tagged to that user.(Hint: Consume the user endpoint to get list of users using restTemplate with the list of accounts tagged to each user by consuming the rest endpoint exposed by user and account application)
  6. Create a Post service which consumes Customer and Account information tagged to the user and stores this information. (Hint: Implement Post services in Customer and Account applications which consumes Customer and Account information respectively).
  7. Handle exception with meaningful JSON structure
  8. Expose a GET endpoint to retrieve the details of all users with accounts tagged to each user by consuming the rest endpoints of user and account application.
  9. Write one or two success and failure use case using any of the testing framework specified

Unit Testing:

1. <https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-testing.html>

Write down the unit testing for all the 3 application using spock and mockmvc framework

Integration Testing:

1. <https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-testing.html>

### Exercise 3

Write down the integration test for all the 3 application using springBootTest and mockMvc. As part of integration test cover the below steps

* Controller classes to test their integration with Spring MVC and HTTP infrastructure
* Integration layer classes to test how they successfully pull data from databases and downstream REST services
* Implement security in User Aggregate Application using OAuth2.0.

# Knowledge Requirements for Micro services Based Projects [Open API & Dev Console]

### Spring Cloud

1. <http://cloud.spring.io/spring-cloud-static/spring-cloud-commons/1.2.2.RELEASE/>
2. <http://cloud.spring.io/spring-cloud-static/Dalston.SR1/>

### OAuth

1. <https://oauth.net/2/>

### Netflix OSS

1. <http://cloud.spring.io/spring-cloud-static/spring-cloud-netflix/1.3.1.RELEASE/>
2. <https://github.com/Netflix/Hystrix>

### ELK Stack

1. https://www.elastic.co/learn

### Existing Project Documentation

1. <http://pnc-openapi-doc.mybluemix.net/>

### Point of Contact

Chinthaka Dharmasiri

Palamayuran Surenthiran

# Knowledge Requirements for Resiliency Project

### Hystrix

1. <https://github.com/Netflix/Hystrix>

### Point of Contact

Chakravarthy Panchagnula

# Knowledge Requirements for OLB and PNC Backend Project

### What?

1. What?

### Point of Contact

Chakravarthy Panchagnula

Ramitha Jayasekara

PNC Coding Standards and Guidelines

– <http://pnc-openapi-coding-guidelines.mybluemix.net/> [This is still WIP and yet to be reviewed by Architects]

References

Spring in action 4th edition PDF (<https://github.com/andriyanov-roman/planet/blob/master/Craig%20Walls%20-%20Spring%20in%20Action%204th%20Edition%20-%202014.pdf> )