Source code for phase-5 project Simplilearn—

Developer’s Name – Sachin Kale

CicdAppliedToSpringBootJavaAppApplication.java—

```
package com.cicd.cicdappliedtospringbootjavaapp;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication

public class CicdAppliedToSpringBootJavaAppApplication {



    public static void main(String[] args) {
        SpringApplication.run(CicdAppliedToSpringBootJavaAppApplication.class,
args);
    }

}
```

CicdAppliedToSpringBootJavaAppApplicationTests.java –

```
ackage com.cicd.cicdappliedtospringbootjavaapp;
```

```java
import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;

import org.springframework.test.context.junit4.SpringRunner;

import com.cicd.cicdappliedtospringbootjavaapp.controller.HelloController;

@RunWith(SpringRunner.class)
@SpringBootTest

public class CicdAppliedToSpringBootJavaAppApplicationTests {

    @Autowired
    private HelloController helloController;

    @Test
    public void contextLoads() {

        Assert.assertEquals("Hello World from DZONE",helloController.home() );
    }
```

Hellocontroller.java ---

```java
package com.cicd.cicdappliedtospringbootjavaapp.controller;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloController {

    @GetMapping("/")
    public String home() {
        return "Hello World from DZONE";
    }

}
```

Pom.xml –(description file)=>

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.1.8.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.cicd</groupId>
    <artifactId>cicd-applied-to-spring-boot-java-app</artifactId>
    <packaging>jar</packaging>
    <version>0.0.1-SNAPSHOT</version>
    <name>cicd-applied-to-spring-boot-java-app</name>
    <description>Implementing CI/CD on Spring Boot Java App</description>

    <properties>
        <java.version>1.8</java.version>

        <!-- solving error : Invalid or corrupt jarfile /app.jar -->

        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <project.reporting.outputEncoding>UTF-
8</project.reporting.outputEncoding>

        <!-- property useful for  spotify's dockerfile-maven-plugin -->

        <!-- Instead of "fanjups", please add your Docker Hub username -->

        <docker.image.prefix>fanjups</docker.image.prefix>

        <!--Not adding artifacts to remote repository-->

        <maven.deploy.skip>true</maven.deploy.skip>

        <!-- GitHub OAuth token & server -->
        <github.global.server>github</github.global.server>
        <github.global.oauth2Token>${env.GITHUB_OAUTH_TOKEN}</github.global.oa
uth2Token>

        <!-- Useful for Heroku Deployment -->
```

```
        <full-artifact-name>target/${project.artifactId}-
${project.version}.jar</full-artifact-name>

    </properties>

    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>

            <!-- spotify's dockerfile-maven-plugin -->

            <plugin>
                <groupId>com.spotify</groupId>
                <artifactId>dockerfile-maven-plugin</artifactId>
                <version>1.4.9</version>
                <executions>
                    <execution>
                        <id>default</id>
                        <phase>install</phase>
                        <goals>
                            <goal>build</goal>
                            <goal>push</goal>
                        </goals>
                    </execution>
                </executions>
                <configuration>
                    <repository>${docker.image.prefix}/${project.artifactId}</
repository>

                    <serverId>index.docker.io</serverId>
                <registryUrl>https://index.docker.io:443/v1/</registryUrl>
```

```
            </configuration>
        </plugin>

        <!-- maven-dependency-plugin useful for creating docker image -->

        <plugin>

            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-dependency-plugin</artifactId>
            <executions>
                <execution>
                    <id>unpack</id>
                    <phase>package</phase>
                    <goals>
                        <goal>unpack</goal>
                    </goals>
                    <configuration>
                        <artifactItems>
                            <artifactItem>
                                <groupId>${project.groupId}</groupId>
                                <artifactId>${project.artifactId}</artifac
tId>
                                <version>${project.version}</version>
                            </artifactItem>
                        </artifactItems>
                    </configuration>
                </execution>


            </executions>

        </plugin>

        <!-- jacoco-maven-plugin useful for Codecov -->

        <plugin>
            <groupId>org.jacoco</groupId>
            <artifactId>jacoco-maven-plugin</artifactId>
            <version>0.7.7.201606060606</version>
            <executions>
                <execution>
                    <goals>
                        <goal>prepare-agent</goal>
                    </goals>
                </execution>
                <execution>
                    <id>report</id>
                    <phase>test</phase>
```

```
                        <goals>
                            <goal>report</goal>
                        </goals>
                    </execution>
                </executions>
            </plugin>

            <!-- Generating github  gh pages & maven project documents  -->

            <plugin>

                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-site-plugin</artifactId>
            </plugin>

            <plugin>
                <groupId>com.github.github</groupId>
                <artifactId>site-maven-plugin</artifactId>
                <version>0.12</version>
                <configuration>
                    <message>Building site for ${project.name}
${project.version}</message>
                    <server>github</server>

                </configuration>
                <executions>
                    <execution>
                        <goals>
                            <goal>site</goal>
                        </goals>
                        <phase>site</phase>
                    </execution>
                </executions>
            </plugin>

            <!-- Useful for Heroku Deployment -->

            <plugin>
              <groupId>com.heroku.sdk</groupId>
              <artifactId>heroku-maven-plugin</artifactId>
              <version>2.0.11</version>
              <configuration>
                <appName>cicd-spring-boot-java-app</appName>
                <processTypes>
                  <web>java $JAVA_OPTS -jar -Dserver.port=$PORT ${full-
artifact-name}</web>
                </processTypes>
              </configuration>
```

```
            </plugin>

        </plugins>
    </build>


    <!-- Useful for GitHub site  -->

    <reporting>

        <plugins>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-project-info-reports-plugin</artifactId>
                <version>3.0.0</version>
                <reportSets>
                    <reportSet>
                        <reports>
                            <report>index</report>
                            <report>licenses</report>
                            <report>dependency-info</report>
                            <report>ci-management</report>
                            <report>dependencies</report>
                            <report>dependency-convergence</report>
                            <report>dependency-management</report>
                            <report>distribution-management</report>
                            <report>help</report>
                            <report>issue-management</report>
                            <report>mailing-lists</report>
                            <report>modules</report>
                            <report>plugin-management</report>
                            <report>plugins</report>
                            <report>team</report>
                            <report>scm</report>
                            <report>summary</report>


                        </reports>
                    </reportSet>
                </reportSets>
            </plugin>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-javadoc-plugin</artifactId>
                <version>3.1.1</version>
```

```
            </plugin>


        </plugins>

    </reporting>

    <licenses>
        <license>
            <name>MIT License</name>
            <url>http://www.opensource.org/licenses/mit-license.php</url>
            <distribution>repo</distribution>
        </license>
    </licenses>

    <developers>
        <developer>
            <email>ronakvyas03@gmail.com</email>
            <name>Fanon Jupkwo</name>
            <url>https://github.com/ronvyas</url>
            <id>FanJups</id>
        </developer>

    </developers>

    <organization>
        <name>FAN JUPS TECH</name>
        <url>https://github.com/ronvyas/</url>
    </organization>

    <issueManagement>
        <system>GitHub Issues</system>
        <url>https://github.com/ronvyas/cicd-applied-to-spring-boot-java-app/issues</url>
    </issueManagement>

    <scm>
        <url>https://github.com/FanJups/cicd-applied-to-spring-boot-java-app</url>
        <connection>scm:git:git://github.com/FanJups/cicd-applied-to-spring-boot-java-app.git</connection>
        <developerConnection>scm:git:git://github.com/ronvyas/cicd-applied-to-spring-boot-java-app.git</developerConnection>
    </scm>

</project>
```

Dockerfile—

```
FROM openjdk:8-jdk-alpine

# Add Maintainer Info
LABEL maintainer="ronakvyas03@gmail.com"

# Add a volume pointing to /tmp
VOLUME /tmp

# Make port 8080 available to the world outside this container
EXPOSE 8080

# The application's jar file
ARG JAR_FILE

# Add the application's jar to the container
ADD ${JAR_FILE} app.jar

# Run the jar file
ENTRYPOINT ["java","-Djava.security.egd=file:/dev/./urandom","-
jar","/app.jar"]
```

Sonarproject –

```
sonar.projectKey=FanJups_cicd-applied-to-spring-boot-java-app
```

Travis.yml (source file)---

```
#we use java
language: java

#we add the java development kit (jdk)
jdk:
- openjdk8
services:
#Linking Travis CI and Docker
- docker
```

```
# SonarCloud
addons:
  sonarcloud:
    organization: "cicd-applied-to-spring-boot-java-app"
    token:
      secure: $SONAR_TOKEN # encrypted value of your token

before_install:
- mvn clean
- echo "$DOCKER_PASSWORD" | docker login  -u "$DOCKER_USERNAME" --password-
stdin
- docker pull openjdk:8-jdk-alpine

script:
#We avoid this error :" the job exceeded the maximum log length and has been
terminated "
- echo -e '<?xml version="1.0" encoding="UTF-8"?>\n<settings
xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.1.0
http://maven.apache.org/xsd/settings-1.1.0.xsd"
xmlns="http://maven.apache.org/SETTINGS/1.1.0"\n    xmlns:xsi="http://www.w3.o
rg/2001/XMLSchema-instance">\n  <mirrors>\n    <mirror>\n      <id>mvnsearch-
unavailable</id>\n      <name>mvnsearch-
unavailable</name>\n      <mirrorOf>mvnsearch</mirrorOf>\n      <url>http://re
po1.maven.org/maven2</url>\n    </mirror>\n  </mirrors>\n  <profiles>\n    <pr
ofile>\n      <id>no-
mvnsearch</id>\n      <repositories>\n        <repository>\n          <id>mvns
earch</id>\n          <url>http://www.mvnsearch.org/maven2</url>\n          <r
eleases>\n            <enabled>true</enabled>\n          </releases>\n
 <snapshots>\n            <enabled>true</enabled>\n          </snapshots>\n
     </repository>\n      </repositories>\n    </profile>\n  </profiles>\n  <a
ctiveProfiles>\n    <activeProfile>no-
mvnsearch</activeProfile>\n  </activeProfiles>\n</settings>' >
$HOME/.m2/settings.xml
- cat $HOME/.m2/settings.xml
#deploying the app on Docker & Heroku
- mvn heroku:deploy
after_success:
#Generating Site
- travis_wait mvn site
- bash <(curl -s https://codecov.io/bash)
# SonarCloud
# the following command line builds the project, runs the tests with coverage
and then execute the SonarCloud analysis
- mvn clean org.jacoco:jacoco-maven-plugin:prepare-agent install sonar:sonar
```

Github Link -- https://github.com/sachinskale127/Java_FSD_Phase5_Project