1. POM.XML

<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/3.6.2 http://maven.apache.org/xsd/maven-3.6.2.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>com.quanlysach</groupId>

<artifactId>bookapi</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- Spring MVC, ORM -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>4.3.10.RELEASE</version>

</dependency>

<!-- Hibernate -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.2.11.Final</version>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-c3p0</artifactId>

<version>5.2.11.Final</version>

</dependency>

<dependency>

<groupId>c3p0</groupId>

<artifactId>c3p0</artifactId>

<version>0.9.1.2</version>

</dependency>

<!-- Postgre connector -->

<dependency>

<groupId>postgresql</groupId>

<artifactId>postgresql</artifactId>

<version>9.1-901-1.jdbc4</version>

</dependency>

<!-- Jackson API for Json -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>2.8.7</version>

</dependency>

<!-- Servlet dependency -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/org.apache.tomcat.maven/tomcat8-maven-plugin -->

<!-- https://mvnrepository.com/artifact/commons-dbutils/commons-dbutils

<dependency> <groupId>commons-dbutils</groupId> <artifactId>commons-dbutils</artifactId>

<version>1.6</version> </dependency> -->

</dependencies>

<build>

<plugins>

<!--

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.5.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.tomcat.maven</groupId>

<artifactId>tomcat8-maven-plugin</artifactId>

<version>2.2</version>

<configuration>

<path>/</path>

</configuration>

</plugin>

-->

<!-- Embedded apche tomcat required for testing web app -->

</plugins>

</build>

</project>

1. AppCònfig.java

package com.quanlysach.spring.config;

import java.util.Properties;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.ComponentScans;

import org.springframework.context.annotation.Configuration;

import org.springframework.context.annotation.PropertySource;

import org.springframework.core.env.Environment;

import org.springframework.orm.hibernate5.HibernateTransactionManager;

import org.springframework.orm.hibernate5.LocalSessionFactoryBean;

import org.springframework.transaction.annotation.EnableTransactionManagement;

import static org.hibernate.cfg.Environment.\*;

@Configuration

@PropertySource("classpath:db.properties")

@EnableTransactionManagement

@ComponentScans(value = {

@ComponentScan("com.quanlysach.spring.DAO"),

@ComponentScan("com.quanlysach.spring.service")

})

public class AppConfig {

@Autowired

private Environment env;

@Bean

public LocalSessionFactoryBean getSessionFactory() {

LocalSessionFactoryBean factoryBean = new LocalSessionFactoryBean();

Properties props = new Properties();

//setting JDBC properties

props.put(DRIVER, env.getProperty("datasource.driver"));

props.put(URL, env.getProperty("datasource.url"));

props.put(USER, env.getProperty("datasource.username"));

props.put(PASS, env.getProperty("datasource.password"));

//settinh hibernate prop

props.put(SHOW\_SQL, env.getProperty("hibernate.show\_sql"));

props.put(HBM2DDL\_AUTO, env.getProperty("hibernate.hbm2ddl.auto"));

//setting c3p0 prop

props.put(C3P0\_MIN\_SIZE, env.getProperty("hibernate.c3p0.min\_size"));

props.put(C3P0\_MAX\_SIZE, env.getProperty("hibernate.c3p0.max\_size"));

props.put(C3P0\_ACQUIRE\_INCREMENT, env.getProperty("hibernate.c3p0.acquire\_increment"));

props.put(C3P0\_TIMEOUT, env.getProperty("hibernate.c3p0.timeout"));

props.put(C3P0\_MAX\_STATEMENTS, env.getProperty("hibernate.c3p0.max\_statements"));

factoryBean.setHibernateProperties(props);

factoryBean.setPackagesToScan("com.quanlysach.spring.model");

return factoryBean;

}

@Bean

public HibernateTransactionManager getTransactionManager() {

HibernateTransactionManager transactionManager = new HibernateTransactionManager();

transactionManager.setSessionFactory(getSessionFactory().getObject());

return transactionManager;

}

}

1. Mywebapp

package com.quanlysach.spring.config;

import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;

public class MyWebAppInitializer extends AbstractAnnotationConfigDispatcherServletInitializer{

@Override

protected Class<?>[] getRootConfigClasses() {

return new Class[] {AppConfig.class};

}

@Override

protected Class<?>[] getServletConfigClasses() {

return new Class[] {WebConfig.class};

}

@Override

protected String[] getServletMappings() {

return new String[] {"/"};

}

}

1. WenConfig

package com.quanlysach.spring.config;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.Configuration;

import org.springframework.web.servlet.config.annotation.EnableWebMvc;

import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;

@Configuration

@EnableWebMvc

@ComponentScan(basePackages = {"com.quanlysach.spring.controller"})

public class WebConfig extends WebMvcConfigurerAdapter {

}

1. Book.java

package com.quanlysach.spring.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity(name = "Book")

public class Book {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String title;

private String author;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public String getAuthor() {

return author;

}

public void setAuthor(String author) {

this.author = author;

}

@Override

public String toString() {

return "Book [id=" + id + ", title=" + title + ", author=" + author + "]";

}

}