app.properties

#Connection Properties#

db.driver=com.mysql.jdbc.Driver

db.url=jdbc:mysql://localhost:3306/warehouse

db.un=root

db.pwd=root

#ORM Properties #

orm.dialect=org.hibernate.dialect.MySQL55Dialect

orm.showsql=true

orm.fmtsql=true

orm.ddlauto=create

# MVC PROPERTIES#

mvc.prefix=/WEB-INF/views/

mvc.suffix=.jsp

AppConfig.java

**package** com.app.config;

**import** java.util.Properties;

**import** org.apache.commons.dbcp2.BasicDataSource;

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

**import** org.springframework.context.annotation.Bean;

**import** org.springframework.context.annotation.ComponentScan;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.context.annotation.PropertySource;

**import** org.springframework.core.env.Environment;

**import** org.springframework.orm.hibernate5.HibernateTemplate;

**import** org.springframework.orm.hibernate5.HibernateTransactionManager;

**import** org.springframework.orm.hibernate5.LocalSessionFactoryBean;

**import** org.springframework.transaction.annotation.EnableTransactionManagement;

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

**import** org.springframework.web.servlet.view.InternalResourceViewResolver;

@Configuration

@PropertySource("classpath:app.properties")

@ComponentScan("com.app")

@EnableWebMvc //Active MVC Concept in Spring

@EnableTransactionManagement //Tx enabled

**public** **class** AppConfig {

@Autowired

**private** Environment env;

//1. DataSource object

@Bean

**public** BasicDataSource dsObj() {

BasicDataSource d=**new** BasicDataSource();

d.setDriverClassName(env.getProperty("db.driver"));

d.setUrl(env.getProperty("db.url"));

d.setUsername(env.getProperty("db.un"));

d.setPassword(env.getProperty("db.pwd"));

**return** d;

}

//2. SessionFactory object

@Bean

**public** LocalSessionFactoryBean sf() {

LocalSessionFactoryBean s=**new** LocalSessionFactoryBean();

s.setDataSource(dsObj());//ref tag.

s.setHibernateProperties(propObj());

s.setAnnotatedClasses(**null**);

**return** s;

}

//SAC

@Bean

**public** Properties propObj() {

Properties p=**new** Properties();

p.put("hibernate.dialect",env.getProperty("orm.dialect"));

p.put("hibernate.show\_sql",env.getProperty("orm.showsql"));

p.put("hibernate.format\_sql",env.getProperty("orm.fmtsql"));

p.put("hibernate.hbm2ddl.auto",env.getProperty("orm.ddlauto"));

**return** p;

}

//3. HibernateTemplate

@Bean

**public** HibernateTemplate ht() {

HibernateTemplate ht=**new** HibernateTemplate();

ht.setSessionFactory(sf().getObject());

**return** ht;

}

//4. Hibernate Tx Manager

@Bean

**public** HibernateTransactionManager htxm() {

HibernateTransactionManager h=**new** HibernateTransactionManager();

h.setSessionFactory(sf().getObject());

**return** h;

}

//5. View Resolver

@Bean

**public** InternalResourceViewResolver ivr() {

InternalResourceViewResolver v=**new** InternalResourceViewResolver();

v.setPrefix(env.getProperty("mvc.prefix"));

v.setSuffix(env.getProperty("mvc.suffix"));

**return** v;

}

}

AppInit.java

**package** com.app.init;

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

**import** com.app.config.AppConfig;

**public** **class** AppInit **extends** AbstractAnnotationConfigDispatcherServletInitializer{

@Override

**protected** Class<?>[] getRootConfigClasses() {

**return** **new** Class[] {AppConfig.**class**};

}

@Override

**protected** Class<?>[] getServletConfigClasses() {

**return** **null**;

}

@Override

**protected** String[] getServletMappings() {

**return** **new** String[] {"/"};

}

}