MySQL download-install-setup:

<https://dev.mysql.com/downloads/file/?id=497106>

<https://www.mysqltutorial.org/install-mysql/#:~:text=Install%20MySQL%20via%20MySQL%20Installer&text=Install%20MySQL%20Step%203%20%E2%80%93%20Download,server%2C%20MySQL%20Workbench%2C%20etc.&text=Install%20MySQL%20Step%205%20%E2%80%93%20Choosing,are%20several%20setup%20types%20available.>

<https://www.youtube.com/watch?v=X_umYKqKaF0>

Project:

1. Create a SpringBoot project named studentboot with web, Spring Data JPA, SpringBoot Dev Tools and MySQL Server Driver packages. Extract that project.
2. Import the project in Eclipse.
3. Create the configuration class  
   Instead of XML, we perform annotation-based configuration. So, we create a class Config.java and specify the required configuration in it. However, there is one more configuration class OnlineCollaborateApplication.java. This class is provided by Spring Boot automatically.

package com.coll.OnlineCollaborate.config;

import java.util.Properties;

import javax.sql.DataSource;

import org.hibernate.SessionFactory;

import org.springframework.boot.autoconfigure.EnableAutoConfiguration;

import org.springframework.boot.autoconfigure.orm.jpa.HibernateJpaAutoConfiguration;

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.jdbc.datasource.DriverManagerDataSource;

import org.springframework.orm.hibernate5.HibernateTransactionManager;

import org.springframework.orm.hibernate5.LocalSessionFactoryBean;

import org.springframework.orm.hibernate5.LocalSessionFactoryBuilder;

import org.springframework.transaction.annotation.EnableTransactionManagement;

import org.springframework.web.servlet.ViewResolver;

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

@Configuration

@ComponentScans(value= {@ComponentScan("com.coll.OnlineCollaborate"),

@ComponentScan("model"),

@ComponentScan("controller"),

@ComponentScan("dao"),

@ComponentScan("daoImpl"),

@ComponentScan("config"),

@ComponentScan("serviceImpl"),

@ComponentScan("service")})

@EnableAutoConfiguration(exclude = {HibernateJpaAutoConfiguration.class})

@EnableTransactionManagement

public class HibernateConfig {

public static final String DATABASE\_URL="jdbc:mysql://localhost:3306/collaboration";

public static final String DATABASE\_DRIVER="com.mysql.cj.jdbc.Driver";

public static final String DATABASE\_DIALECT="org.hibernate.dialect.MySQLDialect";

public static final String DATABASE\_USERNAME="root";

public static final String DATABASE\_PASSWORD="nmj@1234567";

@Bean(name="dataSource")

public DataSource getDataSource() {

DriverManagerDataSource dataSource=new DriverManagerDataSource();

dataSource.setDriverClassName(DATABASE\_DRIVER);

dataSource.setUrl(DATABASE\_URL);

dataSource.setUsername(DATABASE\_USERNAME);

dataSource.setPassword(DATABASE\_PASSWORD);

return dataSource;

}

@Bean

public LocalSessionFactoryBean getSessionFactory() {

LocalSessionFactoryBean sessionFactory = new LocalSessionFactoryBean();

sessionFactory.setDataSource(getDataSource());

sessionFactory.setPackagesToScan("com.coll.OnlineCollaborate");

Properties hibernateProperties =new Properties();

hibernateProperties.put("hibernate.dialect",DATABASE\_DIALECT);

hibernateProperties.put("hibernate.show\_sql","true");

hibernateProperties.put("hibernate.hbm2ddl.auto","update");

sessionFactory.setHibernateProperties(hibernateProperties);

return sessionFactory;

}

@Bean

public HibernateTransactionManager getTransactionManger() {

HibernateTransactionManager txm=new HibernateTransactionManager();

txm.setSessionFactory(getSessionFactory().getObject());

return txm;

}

@Bean

public ViewResolver jspViewResolver() {

InternalResourceViewResolver viewResolver= new InternalResourceViewResolver();

viewResolver.setPrefix("/views/");

viewResolver.setSuffix(".jsp");

return viewResolver;

}

}

1. Create the entity class  
   Here, we are creating an Entity/POJO (Plain Old Java Object) class.

package com.coll.OnlineCollaborate.model;

import java.io.Serializable;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Transient;

import org.springframework.stereotype.Component;

@Component

@Entity

public class User extends DomainResponse implements Serializable {

private static final long serialVersionUID = 1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

private int userId;

private String firstName;

private String lastName;

private String username;

private String password;

private String email;

private String role;

private String status;

private boolean isOnline;

private boolean enabled;

public int getUserId() {

return userId;

}

public void setUserId(int userId) {

this.userId = userId;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

public boolean getIsOnline() {

return isOnline;

}

public void setIsOnline(boolean isOnline) {

this.isOnline = isOnline;

}

public boolean getEnabled() {

return enabled;

}

public void setEnabled(boolean enabled) {

this.enabled = enabled;

}

}

1. Create the DAO interface implementation class

package com.coll.OnlineCollaborate.dao;

import java.util.List;

import com.coll.OnlineCollaborate.model.User;

public interface IUserDao {

List<User> userListbyStatus(String status);

List<User> getAllUsers();

User getUserById(int userid);

User getUserByUsername(String username);

User validateUser(User user);

boolean addUser(User user);

boolean updateUser(User user);

boolean deleteUser(int userId);

boolean deactiveUser(int userId);

boolean activeUser(int userId);

List<User> getAllDeactiveUser();

boolean updateUserProfile(String file,Integer userId);

}

1. Create the DAO interface implementation class

package com.coll.OnlineCollaborate.daoImpl;

import java.util.List;

import org.hibernate.SessionFactory;

import org.hibernate.query.Query;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.coll.OnlineCollaborate.dao.IUserDao;

import com.coll.OnlineCollaborate.model.User;

@Repository("userDao")

@Transactional

public class UserDaoImpl implements IUserDao{

@Autowired

SessionFactory sessionFactory;

@Override

public List<User> userListbyStatus(String status) {

String q="from User where status='"+status+"'";

Query query=sessionFactory.getCurrentSession().createQuery(q);

return query.getResultList();

}

@Override

public List<User> getAllUsers() {

return sessionFactory.getCurrentSession().createQuery("from User",User.class).getResultList();

}

@Override

public User getUserById(int userId) {

// TODO Auto-generated method stub

return sessionFactory.getCurrentSession().get(User.class, Integer.valueOf(userId));

}

@Override

public User getUserByUsername(String username) {

String query="from User where username=:username";

return sessionFactory.getCurrentSession().createQuery(query,User.class).setParameter("username", username).getSingleResult();

}

@Override

public User validateUser(User user) {

String username=user.getUsername();

String password=user.getPassword();

String q="from User where username'"+username+"' and password='"+password+"' and enabled='true'";

Query query=sessionFactory.getCurrentSession().createQuery(q);

try {

user=(User)query.getSingleResult();

return user;

}

catch(Exception e) {

e.printStackTrace();

return null;

}

}

@Override

public boolean addUser(User user) {

try {

sessionFactory.getCurrentSession().save(user);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean updateUser(User user) {

try {

sessionFactory.getCurrentSession().update(user);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean deleteUser(int userId) {

try {

sessionFactory.getCurrentSession().delete(getUserById(userId));

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean deactiveUser(int userId) {

try {

User user=getUserById(userId);

user.setEnabled(false);

sessionFactory.getCurrentSession().update(user);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean updateUserProfile(String file, Integer userId) {

String q="update User set profile=:filename where userId=:id";

Query query=sessionFactory.getCurrentSession().createQuery(q);

query.setParameter("id", (Integer)userId);

query.setParameter("fileName", file);

try {

query.executeUpdate();

return true;

}

catch(Exception e) {

e.printStackTrace();

return false;

}

}

@Override

public boolean activeUser(int userId) {

try {

User user=getUserById(userId);

user.setEnabled(true);

sessionFactory.getCurrentSession().update(user);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public List<User> getAllDeactiveUser() {

return sessionFactory.getCurrentSession().createQuery("from User where enabled=false",User.class).getResultList();

}

}

1. Create the service layer interface

Here, we are creating a service layer interface that acts as a bridge between DAO and Entity classes.

package com.coll.OnlineCollaborate.service;

import java.util.List;

import com.coll.OnlineCollaborate.model.User;

public interface IUserService {

List<User> userListbyStatus(String status);

List<User> getAllUsers();

User getUserById(int userid);

User getUserByUsername(String username);

User validateUser(User user);

boolean addUser(User user);

boolean updateUser(User user);

boolean deleteUser(int userId);

boolean deactiveUser(int userId);

boolean activeUser(int userId);

List<User> getAllDeactiveUser();

boolean updateUserProfile(String file,Integer userId);

}

1. Create the service layer implementation class

package com.coll.OnlineCollaborate.serviceImpl;

import java.util.List;

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

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import com.coll.OnlineCollaborate.dao.IUserDao;

import com.coll.OnlineCollaborate.model.User;

import com.coll.OnlineCollaborate.service.IUserService;

@Service

@Transactional

public class UserServiceImpl implements IUserService{

@Autowired

IUserDao userDao;

@Override

public List<User> userListbyStatus(String status) {

// TODO Auto-generated method stub

return userDao.userListbyStatus(status);

}

@Override

public List<User> getAllUsers() {

// TODO Auto-generated method stub

return userDao.getAllUsers();

}

@Override

public User getUserById(int userid) {

// TODO Auto-generated method stub

return userDao.getUserById(userid);

}

@Override

public User getUserByUsername(String username) {

// TODO Auto-generated method stub

return userDao.getUserByUsername(username);

}

@Override

public User validateUser(User user) {

// TODO Auto-generated method stub

return userDao.validateUser(user);

}

@Override

public boolean addUser(User user) {

// TODO Auto-generated method stub

return userDao.addUser(user);

}

@Override

public boolean updateUser(User user) {

// TODO Auto-generated method stub

return userDao.updateUser(user);

}

@Override

public boolean deleteUser(int userId) {

// TODO Auto-generated method stub

return userDao.deleteUser(userId);

}

@Override

public boolean deactiveUser(int userId) {

// TODO Auto-generated method stub

return userDao.deactiveUser(userId);

}

@Override

public boolean updateUserProfile(String file, Integer userId) {

// TODO Auto-generated method stub

return userDao.updateUserProfile(file, userId);

}

@Override

public boolean activeUser(int userId) {

// TODO Auto-generated method stub

return userDao.activeUser(userId);

}

@Override

public List<User> getAllDeactiveUser() {

// TODO Auto-generated method stub

return userDao.getAllDeactiveUser();

}

}

1. Create the controller class

package com.coll.OnlineCollaborate.controller;

import java.util.List;

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

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.DeleteMapping;

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

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

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

import com.coll.OnlineCollaborate.model.User;

import com.coll.OnlineCollaborate.service.IUserService;

@RestController

@CrossOrigin(origins="http://localhost:4200")

@RequestMapping(value="/api")

public class UserController {

@Autowired

IUserService userService;

@PostMapping("save-user")

public boolean saveUser(@RequestBody User user) {

return userService.addUser(user);

}

@GetMapping("user-list")

public List<User> allUsers(){

return userService.getAllUsers();

}

@DeleteMapping("delete-user/{userId}")

public boolean deleteUser(@PathVariable("userId") int userId) {

return userService.deleteUser(userId);

}

@GetMapping("user/{userId}")

public User userById(@PathVariable("userId") int userId) {

return userService.getUserById(userId);

}

@PostMapping("update-user/{userId}")

public boolean updateUser(@RequestBody User user,@PathVariable("userId") int userId) {

user.setUserId(userId);

return userService.updateUser(user);

}

@GetMapping("deactive-list")

public List<User> AllDeactiveUser(){

return userService.getAllDeactiveUser();

}

@PostMapping("active-user/{userId}")

public boolean activeUser(@RequestBody User user,@PathVariable("userId") int userId) {

return userService.activeUser(userId);

}

}

1. Edit application.properties file  
   Here, we are editing the **application.properties** file present inside the **src/main/resources** folder. The following file contains the configuration properties.

# Database

db.driver=com.mysql.jdbc.Driver

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

db.username=root

db.password=nmj@1234567

# Hibernate

hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

hibernate.show\_sql=true

hibernate.hbm2ddl.auto=update

entitymanager.packagesToScan=com.niit.OnlineCollaborate.model

1. Open Visual Studio Code. Create new project inside Angular\_Workspace. Set the project name as StudentAngular

ng new OnlineCollaborate

1. Install Bootstrap CSS framework

Use the following command to install bootstrap in the project.

*D:\All\_Workspace\Angular\_Workspace\SrudentAngular>* ***npm install*** [***bootstrap@3.3.7***](mailto:bootstrap@3.3.7) ***--save***

1. Now, include the following code in the style.css file.

@import "~bootstrap/dist/css/bootstrap.css";

1. Install Angular Data Table

Use the following command to install bootstrap in the project.

*D:\All\_Workspace\Angular\_Workspace\SrudentAngular>* ***npm install angular-datatable --save***

1. It is required to include **DataTableModule** in imports array of **app.module.ts** file.
2. Generate Components  
   Open the project in visual studio and then use the following command to generate Angular components:

ng g c