|  |
| --- |
| **ONLINE COLLABORATION** |

|  |
| --- |
| Developed By  Nagaram Bhavani Reg.No:R210060100056    **NIIT** |

|  |
| --- |
| **ON LINE COLLABORATION** |

|  |
| --- |
| **Batch code:S210192**  **Start Date:23-02-2021**  **End Date:**  **Name of Coordinator: Lopamudra Bera**  **Name of Developer : Nagaram Bhavani** |

|  |
| --- |
| **ABSTRACT** |

|  |
| --- |
| Online collaboration is used to store user data and to create blogs.To Provide Attractive userInterface Application.to create blog,.In this We used Java , Spring ,hibernate,angular and MySQL. |

|  |
| --- |
| **NIIT** |

|  |
| --- |
| **CERTIFICATE**  This is to certify that this report,titled as **OnlinCollaboration** embodies the original work done by **Nagaram Bhavani** , in partial fulfilment of his course requirement at NIIT.  Coordinator:  **Lopamudra bera** |

|  |
| --- |
| **SYSTEM REQUIREMENTS** |

|  |
| --- |
| **SOFTWARE REQUIREMENTS:**  Operating System : Windows 10  User Interface : Angular,React  Programming Language : Java  IDE/Workbench : My Eclipse 8.6  Database : Mysql  **HARWARE REQUIREMENTS:**  Processor : Intel core i3 or above  Hard Disk : 500GB or more  RAM : 8GB or more |

|  |
| --- |
| **DESIGN** |

|  |
| --- |
| **USE CASE DIAGRAM**  Use case diagrams model the functionality of a system using actors and use cases. Use cases are services or functions provided by the system to its users.  CLIENT |

|  |
| --- |
| **MYSQL DATABASE:** |

* Open MySQL Workbench
* Create a schema named “collaborate”
* Create initially following tables:
* Active the Schema
  + **use collaborate;**
* Create table named “User”

create table User(

UserId int not null auto\_increment,

FirstName varchar(30),

LastName varchar(30),

UserName varchar(20),

Password varchar(20),

email varchar(40),

Role varchar(5),

Status varchar(10),

IsOnline boolean,

Enabled boolean,

primary key(UserId)

)

* Insert one record with Admin role into User table:
* View the record:

**select \* from User**

* Create table named “Blog”

create table Blog(

BlogId int not null auto\_increment,

BlogTitle varchar(30),

BlogContent varchar(200),

BlogPosted Date,

status varchar(10),

NoOfLikes int,

NoOfViews int,

NoOfComments int,

UserId int,

Username varchar(20),

primary key(BlogId)

)

* Create BlogComments table

create table BlogComments(

BlogCommentId int not null auto\_increment,

UserId int,

Username varchar(20),

UserProfileId varchar(20),

Title varchar(30),

NonOfLikes int,

BlogComment varchar(50),

CurrentDate Date,

BlogId int,primary key(BlogCommentId))

|  |
| --- |
| **PROJECT : BACKEND** |

1. Create a SpringBoot project named “OnlineCollaborate”(You can suggest any fesible project name according to project specification) 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 HibernateConfig.java inside com.coll.OnlineCollaborate.config package 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.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("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="niit@123";

@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 getTransactionManager() {

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 Entity(Model) classes. Here, we are creating an Entity/POJO (Plain Old Java Object) class inside com.coll.OnlineCollaborate.model package.

a . Create a class inside the above said package named “User”.

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;

}

public static long getSerialversionuid() {

return serialVersionUID;

}

* 1. Create another class inside the above said package named “Domain Response”.

**package** com.coll.OnlineCollaborate.model;

**public** **class** DomainResponse {

**int** responseCode;

String responseMessage;

**public** DomainResponse() {

**super**();

// **TODO** Auto-generated constructor stub

}

**public** DomainResponse(**int** responseCode, String responseMessage) {

**super**();

**this**.responseCode = responseCode;

**this**.responseMessage = responseMessage;

}

**public** **int** getResponseCode() {

**return** responseCode;

}

**public** **void** setResponseCode(**int** responseCode) {

**this**.responseCode = responseCode;

}

**public** String getResponseMessage() {

**return** responseMessage;

}

**public** **void** setResponseMessage(String responseMessage) {

**this**.responseMessage = responseMessage;

}

}

* 1. Create another class inside the above said package named “Blog”.

package com.coll.OnlineCollaborate.model;

import java.io.Serializable;

import java.time.LocalDate;

import java.util.List;

import javax.persistence.CascadeType;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.OneToMany;

import org.springframework.stereotype.Component;

import com.fasterxml.jackson.annotation.JsonManagedReference;

@Component

@Entity

public class Blog extends DomainResponse implements Serializable{

private static final long serialVersionUID = 1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

int blogId;

String blogTitle, blogContent;

LocalDate blogPosted;

String status;

int noOfLikes, noOfComments, noOfViews;

int userId;

String username;

@OneToMany(mappedBy="blog", fetch=FetchType.EAGER, cascade=CascadeType.ALL)

@JsonManagedReference

List<BlogComments> blogComments;

**public** **int** getBlogId() {

**return** blogId;

}

**public** **void** setBlogId(**int** blogId) {

**this**.blogId = blogId;

}

**public** String getBlogTitle() {

**return** blogTitle;

}

**public** **void** setBlogTitle(String blogTitle) {

**this**.blogTitle = blogTitle;

}

**public** String getBlogContent() {

**return** blogContent;

}

**public** **void** setBlogContent(String blogContent) {

**this**.blogContent = blogContent;

}

**public** LocalDate getBlogPosted() {

**return** blogPosted;

}

**public** **void** setBlogPosted(LocalDate blogPosted) {

**this**.blogPosted = blogPosted;

}

**public** String getStatus() {

**return** status;

}

**public** **void** setStatus(String status) {

**this**.status = status;

}

**public** **int** getNoOfLikes() {

**return** noOfLikes;

}

**public** **void** setNoOfLikes(**int** noOfLikes) {

**this**.noOfLikes = noOfLikes;

}

**public** **int** getNoOfComments() {

**return** noOfComments;

}

**public** **void** setNoOfComments(**int** noOfComments) {

**this**.noOfComments = noOfComments;

}

**public** **int** getNoOfViews() {

**return** noOfViews;

}

**public** **void** setNoOfViews(**int** noOfViews) {

**this**.noOfViews = noOfViews;

}

**public** **int** getUserId() {

**return** userId;

}

**public** **void** setUserId(**int** userId) {

**this**.userId = userId;

}

**public** String getUsername() {

**return** username;

}

**public** **void** setUsername(String username) {

**this**.username = username;

}

**public** List<BlogComments> getBlogComments() {

**return** blogComments;

}

**public** **void** setBlogComments(List<BlogComments> blogComments) {

**this**.blogComments = blogComments;

}

**public** **static** **long** getSerialversionuid() {

**return** ***serialVersionUID***;

}

}

* 1. Create another class inside the above said package named “BlogComments”.

package com.coll.OnlineCollaborate.model;

import java.io.Serializable;

import java.time.LocalDate;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

import org.springframework.stereotype.Component;

import com.fasterxml.jackson.annotation.JsonBackReference;

@Component

@Entity

public class BlogComments implements Serializable{

private static final long serialVersionUID = 1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

int blogCommentId;

int userId;

String username;

String userProfileId;

String title;

int noOfLikes;

String blogComment;

LocalDate currentDate;

@ManyToOne

@JoinColumn(name="BlogId")

@JsonBackReference

Blog blog;

**public** **int** getBlogCommentId() {

**return** blogCommentId;

}

**public** **void** setBlogCommentId(**int** blogCommentId) {

**this**.blogCommentId = blogCommentId;

}

**public** **int** getUserId() {

**return** userId;

}

**public** **void** setUserId(**int** userId) {

**this**.userId = userId;

}

**public** String getUsername() {

**return** username;

}

**public** **void** setUsername(String username) {

**this**.username = username;

}

**public** String getUserProfiledId() {

**return** userProfiledId;

}

**public** **void** setUserProfiledId(String userProfiledId) {

**this**.userProfiledId = userProfiledId;

}

**public** String getTitle() {

**return** title;

}

**public** **void** setTitle(String title) {

**this**.title = title;

}

**public** **int** getNoOfLikes() {

**return** noOfLikes;

}

**public** **void** setNoOfLikes(**int** noOfLikes) {

**this**.noOfLikes = noOfLikes;

}

**public** String getBlogComment() {

**return** blogComment;

}

**public** **void** setBlogComment(String blogComment) {

**this**.blogComment = blogComment;

}

**public** LocalDate getCurrentDate() {

**return** currentDate;

}

**public** **void** setCurrentDate(LocalDate currentDate) {

**this**.currentDate = currentDate;

}

**public** Blog getBlog() {

**return** blog;

}

**public** **void** setBlog(Blog blog) {

**this**.blog = blog;

}

**public** **static** **long** getSerialversionuid() {

**return** ***serialVersionUID***;

}

1. Create the DAO interfaces inside com.coll.OnlineCollaborate.dao package:
   1. Create an Interface named IUserDao.java

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 updateUserProfile(String file, Integer userId);

}

* 1. Create an Interface named IBlogDao.java

package com.coll.OnlineCollaborate.dao;

import java.util.List;

import com.coll.OnlineCollaborate.model.Blog;

public interface IBlogDao {

List<Blog> getAllBlogs();

List<Blog> getBlogsByStatus(String status);

List<Blog> getUsersBlogs(int id);

List<Blog> mainList();

Blog getBlogById(int blogId);

boolean addBlog(Blog blog);

boolean updateBlog(Blog blog);

boolean deleteBlog(Blog blog);

}

* 1. Create an Interface named IBlogCommentsDao.java

package com.coll.OnlineCollaborate.dao;

import java.util.List;

import com.coll.OnlineCollaborate.model.BlogComments;

public interface IBlogCommentsDao {

List<BlogComments> getAllBlogComments();

BlogComments getBlogCommentsById(int blogComemntId);

boolean addBlogComments(BlogComments blogComments);

boolean updateBlogComments(BlogComments blogComments);

boolean deleteBlogComments(BlogComments blogComments);

}

1. Create the DAO interface implementation classes inside com.coll.OnlineCollaborate.daoImpl package:
   1. Create a class named UserDaoImpl.java inside the above said package:

package com.coll.OnlineCollaborate.daoImpl;

import java.util.List;

import org.hibernate.query.Query;

import org.hibernate.SessionFactory;

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) {

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+"'";

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;

}

}

}

* 1. Create a class named BlogDaoImpl.java inside the above said package:

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.IBlogDao;

import com.coll.OnlineCollaborate.model.Blog;

@Repository("blogDao")

@Transactional

public class BlogDaoImpl implements IBlogDao{

@Autowired

SessionFactory sessionFactory;

@Override

public List<Blog> getAllBlogs() {

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

}

@Override

public List<Blog> getBlogsByStatus(String status) {

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

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

return query.getResultList();

}

@Override

public List<Blog> getUserBlogs(int id) {

return null;

}

@Override

public Blog getBlogById(int blogId) {

return sessionFactory.getCurrentSession().get(Blog.class,Integer.valueOf(blogId));

}

@Override

public boolean addBlog(Blog blog) {

try {

sessionFactory.getCurrentSession().save(blog);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean updateBlog(Blog blog) {

try {

sessionFactory.getCurrentSession().saveOrUpdate(blog);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean deleteBlog(Blog blog) {

try {

sessionFactory.getCurrentSession().delete(blog);

return true;

}

catch(Exception e) {

e.printStackTrace();

return false;

}

}

@Override

public List<Blog> mainList() {

// TODO Auto-generated method stub

return null;

}

}

* 1. Create a class named BlogCommentsDaoImpl.java inside the above said package:

package com.coll.OnlineCollaborate.daoImpl;

import java.util.List;

import org.hibernate.SessionFactory;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.coll.OnlineCollaborate.dao.IBlogCommentsDao;

import com.coll.OnlineCollaborate.model.BlogComments;

@Repository("blogCommentsDao")

@Transactional

public class BlogCommentsDaoImpl implements IBlogCommentsDao {

@Autowired

SessionFactory sessionFactory;

@Override

public List<BlogComments> getAllBlogComments() {

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

}

@Override

public BlogComments getBlogCommentsById(int blogCommentId) {

return sessionFactory.getCurrentSession().get(BlogComments.class, Integer.valueOf(blogCommentId));

}

@Override

public boolean addBlogComments(BlogComments blogComments) {

try {

sessionFactory.getCurrentSession().save(blogComments);

return true;

}

catch(Exception e) {

e.printStackTrace();

return false;

}

}

@Override

public boolean updateBlogComments(BlogComments blogComments) {

try {

sessionFactory.getCurrentSession().update(blogComments);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean deleteBlogComments(BlogComments blogComments) {

try {

sessionFactory.getCurrentSession().delete(blogComments);

return true;

}

catch(Exception e) {

e.printStackTrace();

return false;

}

}

1. Create the service interfaces inside com.coll.OnlineCollaborate.service package:
   1. Create an interface named IUserService.java inside the above said package:

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 updateUserProfile(String file, Integer userId);

}

* 1. Create an interface named IBlogService.java inside the above said package:

package com.coll.OnlineCollaborate.service;

import java.util.List;

import com.coll.OnlineCollaborate.model.Blog;

public interface IBlogService {

List<Blog> getAllBlogs();

List<Blog> getBlogsByStatus(String status);

List<Blog> getUsersBlogs(int id);

List<Blog> mainList();

Blog getBlogById(int blogId);

boolean addBlog(Blog blog);

boolean updateBlog(Blog blog);

boolean deleteBlog(Blog blog);

}

* 1. Create an interface named IBlogCommentsService.java inside the above said package:

package com.coll.OnlineCollaborate.service;

import java.util.List;

import com.coll.OnlineCollaborate.model.BlogComments;

public interface IBlogCommentsService {

List<BlogComments> getAllBlogComments();

BlogComments getBlogCommentsById(int blogComemntId);

boolean addBlogComments(BlogComments blogComments);

boolean updateBlogComments(BlogComments blogComments);

boolean deleteBlogComments(BlogComments blogComments);

}

1. Create the service implementation classes inside com.coll.OnlineCollaborate.serviceImpl package:
   1. Create a class named UserServiceImpl.java inside the above said package:

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) {

return userDao.userListbyStatus(status);

}

@Override

public List<User> getAllUsers() {

return userDao.getAllUsers();

}

@Override

public User getUserById(int userId) {

return userDao.getUserById(userId);

}

@Override

public User getUserByUsername(String username) {

return userDao.getUserByUsername(username);

}

@Override

public User validateUser(User user) {

return userDao.validateUser(user);

}

@Override

public boolean addUser(User user) {

return userDao.addUser(user);

}

@Override

public boolean updateUser(User user) {

return userDao.updateUser(user);

}

@Override

public boolean deleteUser(int userId) {

return userDao.deleteUser(userId);

}

@Override

public boolean deactiveUser(int userId) {

return userDao.deactiveUser(userId);

}

@Override

public boolean updateUserProfile(String file, Integer userId) {

return userDao.updateUserProfile(file, userId);

}

}

* 1. Create a class named BlogServiceImpl.java inside the above said package:

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.IBlogDao;

import com.coll.OnlineCollaborate.model.Blog;

import com.coll.OnlineCollaborate.service.IBlogService;

@Service

@Transactional

public class BlogServiceImpl implements IBlogService {

@Autowired

IBlogDao blogDao;

@Override

public List<Blog> getAllBlogs() {

return blogDao.getAllBlogs();

}

@Override

public List<Blog> getBlogsByStatus(String status) {

return blogDao.getBlogsByStatus(status);

}

@Override

public List<Blog> getUsersBlogs(int id) {

return blogDao.getUserBlogs(id);

}

@Override

public List<Blog> mainList() {

return blogDao.mainList();

}

@Override

public Blog getBlogById(int blogId) {

return blogDao.getBlogById(blogId);

}

@Override

public boolean addBlog(Blog blog) {

return blogDao.addBlog(blog);

}

@Override

public boolean updateBlog(Blog blog) {

return blogDao.updateBlog(blog);

}

@Override

public boolean deleteBlog(Blog blog) {

return blogDao.deleteBlog(blog);

}

}

* 1. Create a class named BlogCommentsServiceImpl.java inside the above said package:

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.IBlogCommentsDao;

import com.coll.OnlineCollaborate.model.BlogComments;

import com.coll.OnlineCollaborate.service.IBlogCommentsService;

@Service

@Transactional

public class BlogCommentsSeriveImpl implements IBlogCommentsService {

@Autowired

IBlogCommentsDao blogCommentsDao;

@Override

public List<BlogComments> getAllBlogComments() {

return blogCommentsDao.getAllBlogComments();

}

@Override

public BlogComments getBlogCommentsById(int blogCommentId) {

return blogCommentsDao.getBlogCommentsById(blogCommentId);

}

@Override

public boolean addBlogComments(BlogComments blogComments) {

return blogCommentsDao.addBlogComments(blogComments);

}

@Override

public boolean updateBlogComments(BlogComments blogComments) {

return blogCommentsDao.updateBlogComments(blogComments);

}

@Override

public boolean deleteBlogComments(BlogComments blogComments) {

return blogCommentsDao.deleteBlogComments(blogComments);

}

}

1. Create the controller classes inside com.coll.OnlineCollaborate.controller package:
   1. Create a class named UserController.java inside the above package:

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.RequestMethod;

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);

}

@RequestMapping(value="login/{username,password}", method=RequestMethod.POST)

public User validateUser(@RequestBody User user,@PathVariable("username") String username, @PathVariable("password") String password) {

user.setUsername(username);

user.setPassword(password);

return userService.validateUser(user);

}

}

* 1. Create a class named BlogController.java inside the above package:

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.Blog;

import com.coll.OnlineCollaborate.service.IBlogService;

@RestController

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

@RequestMapping(value="/api")

public class BlogController {

@Autowired

IBlogService blogService;

@PostMapping("save-blog")

public boolean saveBlog(@RequestBody Blog blog) {

return blogService.addBlog(blog);

}

@GetMapping("blog-list")

public List<Blog> allBlog(){

return blogService.getAllBlogs();

}

@DeleteMapping("delete-blog/{blog}")

public boolean deleteBlog(@PathVariable("blog") Blog blog) {

return blogService.deleteBlog(blog);

}

@GetMapping("blog/{blogId}")

public Blog getBlogById(@PathVariable("blogId") int blogId) {

return blogService.getBlogById(blogId);

}

@PostMapping("update-blog/{blog}")

public boolean updateBlog(@RequestBody Blog blog,@PathVariable("blogId") int blogId) {

blog.setBlogId(blogId);

return blogService.updateBlog(blog);

}

}

* 1. Create a class named BlogCommentsrController.java inside the above package:

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.BlogComments;

import com.coll.OnlineCollaborate.service.IBlogCommentsService;

@RestController

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

@RequestMapping(value="/api")

public class BlogCommentsController {

@Autowired

IBlogCommentsService blogCommentsService;

@PostMapping("save-blogComments")

public boolean saveBlogComments(@RequestBody BlogComments blogComments) {

return blogCommentsService.addBlogComments(blogComments);

}

@GetMapping("blogComments-list")

public List<BlogComments> allBlogComments(){

return blogCommentsService.getAllBlogComments();

}

@DeleteMapping("delete-blogComments/{blogComments}")

public boolean deleteBlogComments(@PathVariable("blogComments") BlogComments blogComments) {

return blogCommentsService.deleteBlogComments(blogComments);

}

@GetMapping("blogComments/{blogCommentsId}")

public BlogComments BlogCommentsById (@PathVariable("blogCommentsId")int blogCommentsId) {

return blogCommentsService.getBlogCommentsById(blogCommentsId);

}

@PostMapping("update-blogComments/{blogComments}")

public boolean updateBlogComments(@RequestBody BlogComments blogComments,@PathVariable("blogCommentsId")int blogCommentsId) {

blogComments.setBlogCommentId(blogCommentsId);

return blogCommentsService.updateBlogComments(blogComments);

}

}

1. Save All
2. Right click on OnlineCollaborateApplication.java class and Run as Java Application.

|  |
| --- |
| **Output** |

**Output:**

****

|  |
| --- |
| **frontend** |

**App.module.ts**

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import {DataTablesModule} from 'angular-datatables';

import {FormsModule, ReactiveFormsModule} from '@angular/forms';

import {HttpClientModule} from '@angular/common/http';

import { RegisterUserComponent } from './components/register-user/register-user.component';

import { UserListComponent } from './components/user-list/user-list.component';

import { LoginUserComponent } from './components/login-user/login-user.component';

import { ActivateUserComponent } from './components/activate-user/activate-user.component';

import { AddBlogComponent } from './components/add-blog/add-blog.component';

import { ApproveBlogComponent } from './components/approve-blog/approve-blog.component';

import { ViewBlogComponent } from './components/view-blog/view-blog.component';

import {UserService} from './services/user.service';

import {BlogService} from './services/blog.service';

import { AdminHomeComponent } from './components/admin-home/admin-home.component';

import { UserHomeComponent } from './components/user-home/user-home.component';

@NgModule({

  declarations: [

    AppComponent,

    RegisterUserComponent,

    UserListComponent,

    LoginUserComponent,

    ActivateUserComponent,

    AddBlogComponent,

    ApproveBlogComponent,

    ViewBlogComponent,

    AdminHomeComponent,

    UserHomeComponent,

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

    DataTablesModule,

    HttpClientModule,

    ReactiveFormsModule

  ],

  providers: [UserService, BlogService],

  bootstrap: [AppComponent]

})

export class AppModule {

}

**App.routing.module.ts**

import { NgModule } from '@angular/core';

import { Routes, RouterModule } from '@angular/router';

import { RegisterUserComponent } from './components/register-user/register-user.component';

import { UserListComponent } from './components/user-list/user-list.component';

import { LoginUserComponent } from './components/login-user/login-user.component';

import { ActivateUserComponent } from './components/activate-user/activate-user.component';

import { AddBlogComponent } from './components/add-blog/add-blog.component';

import { ApproveBlogComponent } from './components/approve-blog/approve-blog.component';

import { ViewBlogComponent } from './components/view-blog/view-blog.component';

import { AdminHomeComponent } from './components/admin-home/admin-home.component';

import { UserHomeComponent } from './components/user-home/user-home.component';

const routes: Routes = [

  {path:'',redirectTo:'login-user', pathMatch:'full'},

  {path:'register-user', component:RegisterUserComponent},

  {path:'login-user', component:LoginUserComponent},

  {path:'user-list', component:UserListComponent},

  {path:'activate-user', component:ActivateUserComponent},

  {path:'add-blog', component:AddBlogComponent},

  {path:'approve-blog', component:ApproveBlogComponent},

  {path:'view-blog', component:ViewBlogComponent},

  {path:'admin-home/:id', component:AdminHomeComponent},

  {path:'user-home/:id', component:UserHomeComponent}

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

**app.component.html**

<div class="container-fluid">

  <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

    <ul class="navbar-nav">

      <li class="nav-item">

        <a routerLink="user-list" class="nav-link btn btn-primary" role="button">View Users</a>

      </li>

      <li class="nav-item">

        <a routerLink="register-user" class="nav-link btn btn-primary" role="button">Registration</a>

      </li>

      <li class="nav-item">

        <a routerLink="login-user" class="nav-link btn btn-primary active" role="button">Login</a>

      </li>

    </ul>

  </nav>

  <router-outlet></router-outlet>

</div>

**User.service.ts**

import { Injectable } from '@angular/core';

import {HttpClient} from '@angular/common/http';

import {Observable} from 'rxjs';

import { LoginUserComponent } from '../components/login-user/login-user.component';

import { LocalizedString } from '@angular/compiler';

@Injectable({

  providedIn: 'root'

})

export class UserService {

  private baseUrl='http://localhost:8080/api/';

  constructor(private http:HttpClient) { }

  getUserList():Observable<any>{

    return this.http.get(`${this.baseUrl}`+'user-list');

  }

  createUser(user: object):Observable<Object>{

    return this.http.post(`${this.baseUrl}`+'save-user',user);

  }

  deleteUser(userId: number):Observable<any>{

    return this.http.delete(`${this.baseUrl}/delete-user/${userId}`,{responseType:'text'});

  }

  getUser(userId: number):Observable<Object>{

    return this.http.get(`${this.baseUrl}/user/${userId}`);

  }

  updateUser(userId: number, value: any):Observable<Object>{

    return this.http.post(`${this.baseUrl}/update-user/${userId}`,value);

  }

  activateUser(userId: number):Observable<any>{

    return this.http.post(`${this.baseUrl}/active-user/${userId}`,{responseType:'text'});

  }

  validateUser(user:object):Observable<any>{

    return this.http.post(`${this.baseUrl}`+'validate-user',user);

  }

  logout(userId: number):Observable<any>{

    return this.http.post(`${this.baseUrl}/logout/${userId}`,{responseType:'text'});

  }

}

**User.list.component.html**

<div class="panel panel-default">

    <div class="panel-heading">

        <h1 style="text-align: center;">Users</h1><br/>

        <div class="row" [hidden]="!deleteMessage">

            <div class="col-sm-4"></div>

            <div class="col-sm-4">

                <div class="alert alert-info alert-dismissible">

                    <button type="button" class="close" data-dismiss="alert">X</button>

                    <strong>User data deleted</strong>

                </div>

            </div>

            <div class="col-sm-4"></div>

        </div>

    </div>

    <div class="panel-body">

        <table class="table table-hover table-sm" datatable [dtOptions]="dtOptions" [dtTrigger]="dtTrigger">

            <thead class="thead-light">

                <tr>

                    <th>User Id</th>

                    <th>FirstName</th>

                    <th>LastName</th>

                    <th>Username</th>

                    <th>Password</th>

                    <th>Email</th>

                    <th>Role</th>

                    <th>Status</th>

                    <th>IsOnline</th>

                    <th>Enabled</th>

                    <th>Action</th>

                </tr>

            </thead>

            <tbody>

                <tr \*ngFor="let user of users">

                    <td>{{user.userId}}</td>

                    <td>{{user.firstName}}</td>

                    <td>{{user.lastName}}</td>

                    <td>{{user.username}}</td>

                    <td>{{user.password}}</td>

                    <td>{{user.email}}</td>

                    <td>{{user.role}}</td>

                    <td>{{user.status}}</td>

                    <td>{{user.isOnline}}</td>

                    <td>{{user.enabled}}</td>

                    <td>

                        <button (click)="deleteUser(user.userId)" class="btn btn-primary">

                            <i class="fa fa-futboll-0">Delete</i>

                        </button>

                        <button (click)="updateUser(user.userId)" class="btn btn-info" data-toggle="modal" data-target="#myModal">

                            Update

                        </button>

                    </td>

                </tr>

            </tbody>

        </table>

    </div>

</div>

<!--myModal-->

<div class="modal" id="myModal">

    <div class="modal-dialog">

        <div class="modal-content">

            <form [formGroup]="userupdateform" (ngSubmit)="update(user)">

                <!--Modal Header-->

                <div class="modal-header">

                    <h4 class="modal-title" style="text-align: center;">Update User</h4>

                </div>

                <!--Modal Body-->

                <div class="modal-body" \*ngFor="let user of userlist">

                    <div [hidden]="isupdated">

                        <input type="hidden" class="form-control" formControlName="userId" [(ngModel)]="user.userId">

                        <div class="form-group">

                            <label for="firstName">FirstName</label>

                            <input type="text" class="form-control" formControlName="firstName" [(ngModel)]="user.firstName" disabled="true">

                        </div>

                        <div class="form-group">

                            <label for="lastName">LastName</label>

                            <input type="text" class="form-control" formControlName="lastName" [(ngModel)]="user.lastName" disabled="true">

                        </div>

                        <div class="form-group">

                            <label for="username">Username</label>

                            <input type="text" class="form-control" formControlName="username" [(ngModel)]="user.username" disabled="true">

                        </div>

                        <div class="form-group">

                            <label for="password">FirstName</label>

                            <input type="password" class="form-control" formControlName="password" [(ngModel)]="user.password">

                        </div>

                        <div class="form-group">

                            <label for="email">Email</label>

                            <input type="text" class="form-control" formControlName="email" [(ngModel)]="user.email">

                        </div>

                        <div class="form-group">

                            <label for="role">Role</label>

                            <input type="text" class="form-control" formControlName="role" [(ngModel)]="user.role" disabled="true">

                        </div>

                        <div class="form-group">

                            <label for="firstName">FirstName</label>

                            <input type="text" class="form-control" formControlName="firstName" [(ngModel)]="user.firstName" disabled="true">

                        </div>

                        <div class="form-group">

                            <label for="status">Status</label>

                            <input type="text" class="form-control" formControlName="status" [(ngModel)]="user.status" disabled="true">

                        </div>

                        <div class="form-group">

                            <label for="isOnline">IsOnline</label>

                            <input type="text" class="form-control" formControlName="isOnline" [(ngModel)]="user.isOnline" disabled="true">

                        </div>

                        <div class="form-group">

                            <label for="enabled">Enabled</label>

                            <input type="text" class="form-control" formControlName="enabled" [(ngModel)]="user.enabled" disabled="true">

                        </div>

                    </div>

                    <div [hidden]="!isupdated">

                        <h4>User Detail Updated!</h4>

                    </div>

                </div>

                <div class="modal-footer">

                    <button type="submit" class="btn btn-success" [hidden]="isupdated">Update</button>

                    <button type="button" class="btn btn-danger" data-dismiss="modal" (click)="changeisUpdate()">Close</button>

                </div>

            </form>

        </div>

    </div>

</div>

**User-list.component.ts**

import { Component, OnInit } from '@angular/core';

import {UserService} from '../../services/user.service';

import {User} from '../../model/user';

import {Observable, Subject} from 'rxjs';

import {Validators, FormControl, FormGroup, FormBuilder} from '@angular/forms';

import {DataTablesModule} from 'angular-datatables'

@Component({

  selector: 'app-user-list',

  templateUrl: './user-list.component.html',

  styleUrls: ['./user-list.component.scss']

})

export class UserListComponent implements OnInit {

  usersArray:any=[];

  dtOptions: DataTables.Settings={};

  dtTrigger: Subject<any>=new Subject();

  users: Observable<User[]>;

  user: User=new User();

  deleteMessage=false;

  userlist:any;

  isupdated=false;

  constructor(private userservice: UserService) { }

  ngOnInit(): void {

    this.isupdated=false;

    this.dtOptions={

      pageLength:6,

      stateSave:true,

      lengthMenu:[[6,16,20,-1],[6,16,20,"All"]],

      processing:true

    };

    this.userservice.getUserList().subscribe(data=>{

      this.users=data;

      this.dtTrigger.next();

    })

  }

  deleteUser(id:number){

    this.userservice.deleteUser(id)

    .subscribe(data=>{

      console.log(data);

      this.deleteMessage=true;

      this.userservice.getUserList().subscribe(data=>{

        this.users=data;

      })

    }, error=>console.log(error));

  }

  updateUser(id:number){

    this.userservice.getUser(id)

    .subscribe(data=>{

      this.userlist=data;

    }, error=>console.log(error));

  }

  userupdateform=new FormGroup({

    userId:new FormControl(),

    firstName:new FormControl(),

    lastName:new FormControl(),

    username:new FormControl(),

    password:new FormControl(),

    email:new FormControl(),

    role:new FormControl(),

    status:new FormControl(),

    isOnline:new FormControl(),

    enabled:new FormControl()

  });

  update(user){

    this.user=new User();

    this.user.userId=this.UserId.value,

    this.user.firstName=this.FirstName.value;

    this.user.lastName=this.LastName.value;

    this.user.username=this.Username.value;

    this.user.password=this.Password.value;

    this.user.email=this.Email.value;

    this.user.role=this.Role.value;

    this.user.enabled=true;

    this.user.status="Active";

    this.userservice.updateUser(this.user.userId,this.user).subscribe(data=>{

      this.isupdated=true;

      this.userservice.getUserList().subscribe(data=>{

        this.users=data;

      })

    },error=>console.log(error));

  }

  get UserId(){

    return this.userupdateform.get('userId');

  }

  get FirstName(){

    return this.userupdateform.get('firstName');

  }

  get LastName(){

    return this.userupdateform.get('lastName');

  }

  get Username(){

    return this.userupdateform.get('username');

  }

  get Password(){

    return this.userupdateform.get('password');

  }

  get Email(){

    return this.userupdateform.get('email');

  }

  get Role(){

    return this.userupdateform.get('role');

  }

  get Status(){

    return this.userupdateform.get('status');

  }

  get IsOnline(){

    return this.userupdateform.get('isOnline');

  }

  get Enabled(){

    return this.userupdateform.get('enabled');

  }

  changeisUpdate(){

    this.isupdated=false;

  }

}

**Register-usre.compoent.html**

<h3>Register Here</h3>

<div class="row">

    <div class="col-sm-4"></div>

    <div class="col-sm-4">

        <div [hidden]="submitted" style="width: 400px;">

            <form [formGroup]="registrationform" (ngSubmit)="register(register)">

                <div class="form-group">

                    <label for="firstName">First Name</label>

                    <input type="text" formControlName="firstName" class="form-control" data-toggle="tooltip" data-placement="right" title="Enter First Name">

                    <div class="alert alert-danger" \*ngIf="(FirstName.touched) && (FirstName.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="FirstName.error.required">First Name is required</span>

                    </div>

                </div>

                <div class="form-group">

                    <label for="lastName">Last Name</label>

                    <input type="text" formControlName="lastName" class="form-control" data-toggle="tooltip" data-placement="right" title="Enter Last Name">

                    <div class="alert alert-danger" \*ngIf="(LastName.touched) && (LastName.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="LastName.error.required">Last Name is required</span>

                    </div>

                </div>

                <div class="form-group">

                    <label for="username">Username</label>

                    <input type="text" formControlName="username" class="form-control" data-toggle="tooltip" data-placement="right" title="Enter Username">

                    <div class="alert alert-danger" \*ngIf="(Username.touched) && (Username.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="Username.error.required">Username is required</span>

                    </div>

                </div>

                <div class="form-group">

                    <label for="password">Password</label>

                    <input type="password" formControlName="password" class="form-control" data-toggle="tooltip" data-placement="right" title="Enter Password">

                    <div class="alert alert-danger" \*ngIf="(Password.touched) && (Password.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="Password.error.required">Password is required</span>

                    </div>

                </div>

                <div class="form-group">

                    <label for="confirm\_password">Confirm Password</label>

                    <input type="password" formControlName="confirm\_password" class="form-control" data-toggle="tooltip" data-placement="right" title="Confirm Password" pattern="{{Password.value}}">

                    <div class="alert alert-danger" \*ngIf="(ConfirmPassword.touched) && (ConfirmPassword.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="ConfirmPassword.error.required">Confirm Password is required</span>

                        <span \*ngIf="ConfirmPassword.error.pattern">Password and Confirm Password does not match</span>

                    </div>

                </div>

                <div class="form-group">

                    <label for="email">Email</label>

                    <input type="text" formControlName="email" class="form-control" data-toggle="tooltip" data-placement="right" title="Enter Email Id">

                    <div class="alert alert-danger" \*ngIf="(Email.touched) && (Email.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="Email.error.required">Email is required</span>

                        <span \*ngIf="Email.error.email">Invalid Email format</span>

                    </div>

                </div>

                <div class="form-group">

                    <label for="role">Role</label>

                    <select formControlName="role" class="form-control" data-toggle="tooltip" data-placement="right" title="Select user Role">

                        <option value="null">-- User Role --</option>

                        <option value="Admin">Admin</option>

                        <option value="User">User</option>

                    </select>

                </div>

                <button type="submit" class="btn btn-success">Submit</button>

            </form>

        </div>

    </div>

    <div>

        <div class="col-sm-4"></div>

    </div>

    <div class="col-sm-4">

        <div [hidden]="!submitted">

            <h4>Congratulations! You have registered successfully!</h4>

        </div>

    </div>

</div>

**Register-user.component.ts**

import { Component, OnInit } from '@angular/core';

import {UserService} from '../../services/user.service';

import {FormControl, FormGroup, Validators} from '@angular/forms';

import {User} from '../../model/user';

@Component({

  selector: 'app-register-user',

  templateUrl: './register-user.component.html',

  styleUrls: ['./register-user.component.scss']

})

export class RegisterUserComponent implements OnInit {

  user:User=new User();

  submitted=false;

  constructor(private userservice:UserService) { }

  ngOnInit(): void {

    this.submitted=false;

  }

  registrationform=new FormGroup({

  firstName:new FormControl('',[Validators.required]),

    lastName:new FormControl('',[Validators.required]),

    username:new FormControl('',[Validators.required]),

    password:new FormControl('',[Validators.required]),

    confirm\_password:new FormControl('',[Validators.required]),

    email:new FormControl('',[Validators.required, Validators.email]),

    role:new FormControl(),

  });

  register(register){

  this.user=new User();

    this.user.firstName=this.FirstName.value;

    this.user.lastName=this.LastName.value;

    this.user.username=this.Username.value;

    if(this.Password.value===this.ConfirmPassword.value)

      this.user.password=this.Password.value;

    this.user.email=this.Email.value;

    this.user.role=this.Role.value;

    if(this.user.role==="Admin"){

      this.user.enabled=true;

      this.user.status="Active";

    }

    else{

      this.user.enabled=false;

      this.user.status="Inactive";

    }

    this.user.isOnline=false;

    this.submitted=true;

    this.save();

  }

  save(){

    this.userservice.createUser(this.user)

    .subscribe(data=>console.log(data), error=>console.log(error));

    this.user=new User();

  }

  get FirstName(){

    return this.registrationform.get('firstName');

  }

  get LastName(){

    return this.registrationform.get('lastName');

  }

  get Username(){

    return this.registrationform.get('username');

  }

  get Password(){

    return this.registrationform.get('password');

  }

  get ConfirmPassword(){

    return this.registrationform.get('confirm\_password');

  }

  get Email(){

    return this.registrationform.get('email');

  }

  get Role(){

    return this.registrationform.get('role');

  }

  registrationForm(){

    this.submitted=false;

    this.registrationform.reset();

  }

}

**Login-user.component.html**

<h3>Login Here</h3>

<div class="row">

    <div class="col-sm-4"></div>

    <div class="col-sm-4">

        <div [hidden]="submitted" style="width: 400px;">

            <form [formGroup]="loginform" (ngSubmit)="login(login)">

                <div class="form-group">

                    <label for="username">Username</label>

                    <input type="text" formControlName="username" class="form-control" data-toggle="tooltip" data-placement="right" title="Enter Username">

                    <div class="alert alert-danger" \*ngIf="(Username.touched) && (Username.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="Username.error.required">Username is required</span>

                    </div>

                </div>

                <div class="form-group">

                    <label for="password">Password</label>

                    <input type="password" formControlName="password" class="form-control" data-toggle="tooltip" data-placement="right" title="Enter Password">

                    <div class="alert alert-danger" \*ngIf="(Password.touched) && (Password.invalid)" style="margin-top: 5px;">

                        <span \*ngIf="Password.error.required">Password is required</span>

                    </div>

              </div>

                <button type="submit" class="btn btn-success">Submit</button>

            </form>

        </div>

    <div>

</div>

**Login-user.component.ts**

 import { Component, OnInit } from '@angular/core';

import {FormControl, FormGroup, NgForm, Validators} from '@angular/forms';

import {User} from '../../model/user';

import { UserService } from '../../services/user.service';

import {Router} from '@angular/router';

import { data } from 'jquery';

@Component({

  selector: 'app-login-user',

  templateUrl: './login-user.component.html',

  styleUrls: ['./login-user.component.scss']

})

export class LoginUserComponent implements OnInit {

  user:User=new User();

  currentUser:any;

  submitted=false;

  constructor(private userservice:UserService,private router:Router) { }

  ngOnInit(): void {

   this.submitted=false;

  }

  login(login){

  this.user=new User();

    this.user.username=this.Username.value;

      this.user.password=this.Password.value;

    this.userservice.validateUser(this.user)

    .subscribe(data=>{

      console.log(data);

    if(data!=null){

      this.currentUser=data;

      if(this.currentUser.role==="Admin"&& this.currentUser.isOnline===true){

        this.router.navigateByUrl('/admin-home/'+`${this.currentUser.userId}`);

        }

        else{

          this.router.navigateByUrl('/user-home/'+`${this.currentUser.userId}`);

        }

      console.log(this.currentUser.userId);

    }else{

      console.log("ObjectEmpty");

    }

  },error=>console.log(error));

}

  loginform=new FormGroup({

    username:new FormControl('',[Validators.required]),

    password:new FormControl('',[Validators.required]),

  });

  get Username(){

    return this.loginform.get('username');

  }

  get Password(){

    return this.loginform.get('password');

  }

  loginForm(){

    this.submitted=false;

    this.loginform.reset();

  }

}

**Admin-home.component.ts**

import { Component, OnInit } from '@angular/core';

import { FormControl, FormGroup, Validators } from '@angular/forms';

import { User } from 'src/app/model/user';

import {ActivatedRoute,Params} from '@angular/router';

import { UserService } from 'src/app/services/user.service';

@Component({

  selector: 'app-admin-home',

  templateUrl: './admin-home.component.html',

  styleUrls: ['./admin-home.component.scss']

})

export class AdminHomeComponent implements OnInit {

  user:User=new User();

id:number;

  constructor(private userservice:UserService,private route:ActivatedRoute) { }

  ngOnInit(): void {

    this.route.params

    .subscribe(

      (params:Params)=>{

        this.id = +params['id'];

       }

    );

  }

}

**User-home.component.ts**

import { Component, OnInit } from '@angular/core';

import { FormControl, FormGroup, Validators } from '@angular/forms';

import { User } from 'src/app/model/user';

import { UserService } from 'src/app/services/user.service';

import {ActivatedRoute,Params} from '@angular/router';

@Component({

  selector: 'app-user-home',

  templateUrl: './user-home.component.html',

  styleUrls: ['./user-home.component.scss']

})

export class UserHomeComponent implements OnInit {

  userhome:User=new User();

 id: number;

  constructor(private userservice:UserService,private route:ActivatedRoute) { }

  ngOnInit(): void {

    this.route.params

    .subscribe(

      (params:Params)=>{

        this.id = +params['id'];

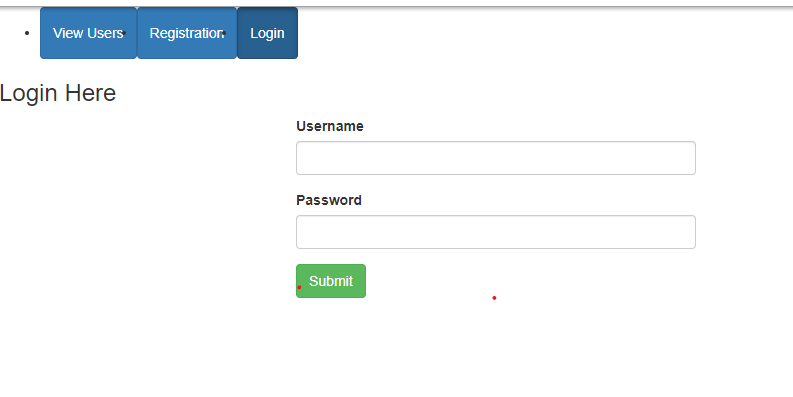
       }

    );

      }

}

**Output:**

****

