****

***Online Collaborate***

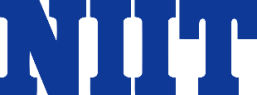
**Developed by:** Ganesh Pandi E

**Reg No:** R200002100293

**Batch Code: S**210167

**Name of the Cordinator:** Ms. Lopamudra Bera

**Date of Submission:** 15/03/2021



***CERTIFICATE***

This is to certify that “Ganesh Pandi E” student of Batch-S210167 has successfully completed their Project on “Online Collaborate” under the guidance of “Miss.Lopamudra Bera”.

Miss.Lopmudra Bera Ganesh Pandi E

***ACKNOWLEDGEMENT***

I would like to express my special thanks of gratitude to my teacher

“Miss.Lopamudra Bera” for their able guidance and support in completing my Project.

Date: Ganesh Pandi E

15/03/2021  R200002100293

***ABSTRACT***

This Project will mainly use for Online Collaboration, allowing registered User can to post blogs and their information on profile and easily Collaborate to all registered users like Social Service media.

Users will register themselves in the application and fill in their professional details and User can be able to posting blog by their own thoughts.

Admin can view all registered user details , active them or delete them.admin can view all posted blog by users, and approve the blog or delete it.

Admin and Users can see their profile and can update thire profiles.

|  |  |
| --- | --- |
| ***CONFIGURATION***   |  | | --- | | * Hardware   + - AMD Pro Processor 2.10 GHz x64 bit     - 4 Gigabytes of RAM * Operating System   + - Windows 10 Home * Software   + Visual Studio Code     - HTML     - CSS     - Typescript     - Bootstrap   + Eclipse IDE     - Java     - Hibernate     - Spring     - Spring Boot   + MySQL | |

***OBJECTIVE***

This Project will mainly use for professional networking, allowing registered users to post blogs and their information and its easily collaborate with all other register users.

Users will register themselves in the application and fill in their personal details. And Users can be able to see their profiles.

***STATEMENT OF REQUIREMENTS***

**Key Features of the application:**

* Its has Home page window that User/Admin can Login/Register.
* It has the registration window to register for user register by themselves.
* Application has the same login window for everyone and it detects by itself which designation you are in and give access accordingly.
* Admin can able to activate and deactivate an users profile and blog their posted.
* Admin can able to view all the User status and can update, delete, activate or deactivate them.
* User can add a blog posting on the their own blog and update or delete a blog while it is needed.

**User**

* View their own profile
* Update profile
* Activate or Deactivate a user
* View All user status
* Active or deactive user blogs

**Admin**

***Login / Registrations***

In the above model is that the overall functionality model of the application that we have already discussed in the previous chapter.

***HIGH LEVEL CASE DIAGRAM***

**Approved**

**Admin**

**User**

**Sent Request**

**Sent Response**

**BackEnd**

**MySQL**

**Database**

A High-level design document adds the necessary details to the current project description to represent a suitable model for coding.

***DATA STRUCTURE MODEL***

userId int not null auto\_increment,

firstName varchar

lastName varchar

username varchar

password varchar

email varchar

role varchar,

status varchar,

isOnline varchar

enabled varchar

****

blogCommentId integer not null

auto\_increment,

blogComment varchar

currentDate date,

onOfLikes integer not null,

titel varchar

userId integer not null,

userProfileId varchar

username varchar,

blogId integer ,

****

blogId integer not null

auto\_increment,

blogContent varchar

blogPosted dateTime,

blogTitel varchar

noFoComments integer not

null,

noOfLikes integer not null,

noOfViews integer not null,

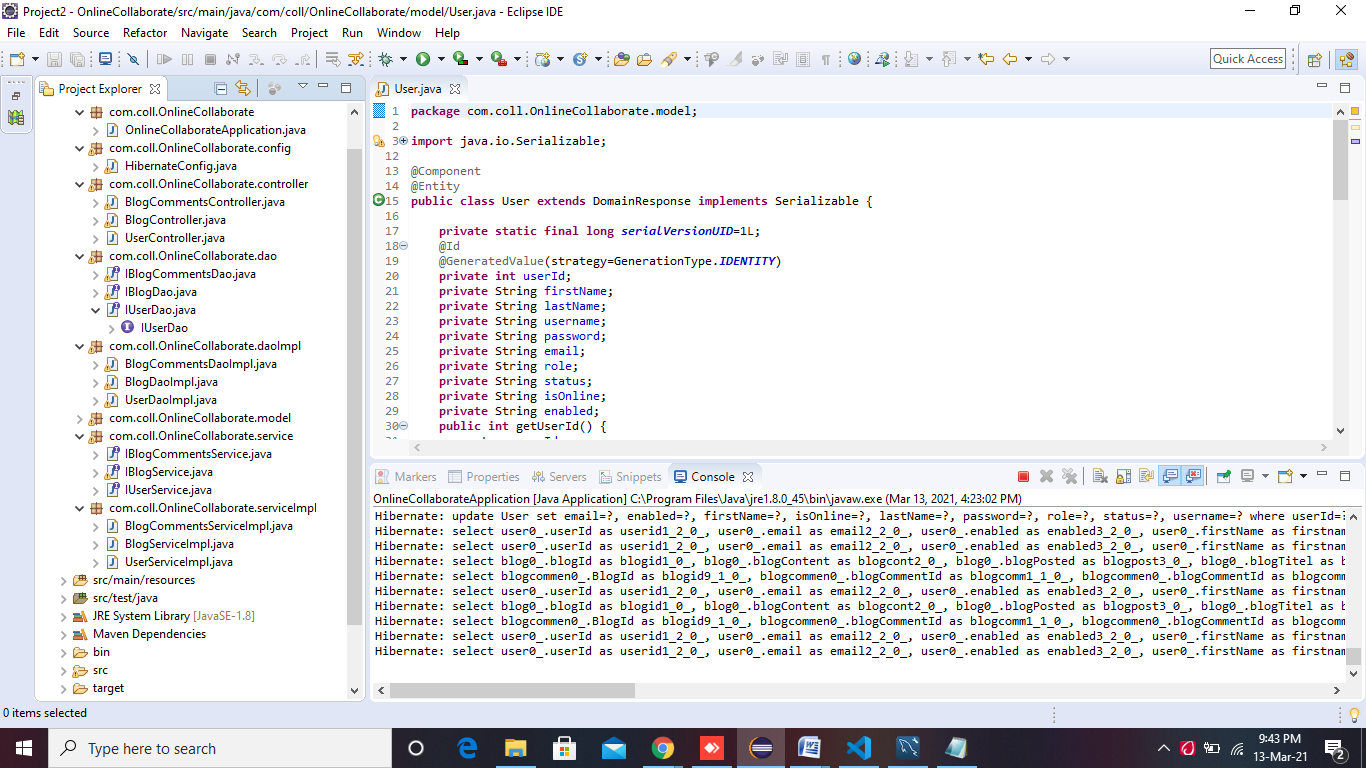
status varchar

userId integer not null,

username varchar

****

**BACKEND:-**

****

The structure of the project is present on the left most side of the window under Project Explorer. There you will get to see different kind of packages like config , controller, dao, daoimpl, entry, model,service,serviceImpl.which holds different classes that are categorized in terms of different packages. These are the classes which can handle the entire backend process of the project structure

**Procedure:-**

**🡪** Create a SpringBoot project named OnlineCollaborate with web, Spring Data JPA, SpringBoot Dev Tools and MySQL Server Driver packages. Extract that project

🡪 Import the project in Eclipse.

🡪Open **package** com.coll.OnlineCollaborate;

🡪Create **package** com.coll.OnlineCollaborate.config;

🡪Configuration File as HibernateConfig:-

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.http.converter.json.MappingJackson2HttpMessageConverter;

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;

import com.fasterxml.jackson.databind.DeserializationFeature;

import com.fasterxml.jackson.databind.ObjectMapper;

@Configuration

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

@ComponentScan("model"),

@ComponentScan("controller"),

@ComponentScan("dao"),

@ComponentScan("daoImpl"),

@ComponentScan("config"),

@ComponentScan("serviceImpl"),

@ComponentScan("service"),

@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="ganesh@7";

@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;

}

@Bean

public MappingJackson2HttpMessageConverter mappingJackson2HttpMessageConverter() {

MappingJackson2HttpMessageConverter jsonConverter = new MappingJackson2HttpMessageConverter();

ObjectMapper objectMapper = new ObjectMapper();

objectMapper.configure(DeserializationFeature.FAIL\_ON\_UNKNOWN\_PROPERTIES, false);

jsonConverter.setObjectMapper(objectMapper);

return jsonConverter;

}

}

🡪Create **package** com.coll.OnlineCollaborate.model

🡪 Create model classes:

**1.User 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 String isOnline;

private String 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 String getIsOnline() {

return isOnline;

}

public void setIsOnline(String isOnline) {

this.isOnline = isOnline;

}

public String getEnabled() {

return enabled;

}

public void setEnabled(String enabled) {

this.enabled = enabled;

}

}

**2.Blog:-**

package com.coll.OnlineCollaborate.model;

import java.io.Serializable;

import java.time.LocalDate;

import java.time.LocalDateTime;

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;

import com.fasterxml.jackson.databind.annotation.JsonDeserialize;

import com.fasterxml.jackson.datatype.jsr310.deser.LocalDateTimeDeserializer;

import com.fasterxml.jackson.datatype.jsr310.ser.LocalDateTimeSerializer;

@Component

@Entity

public class Blog extends DomainResponse implements Serializable{

private static final long serialVersionUID=1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

int blogId;

String blogTitel, blogContent;

@JsonDeserialize(using = LocalDateTimeDeserializer.class)

LocalDateTime blogPosted;

String status;

int noOfLikes,noFoComments, 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 getBlogTitel() {

return blogTitel;

}

public void setBlogTitel(String blogTitel) {

this.blogTitel = blogTitel;

}

public String getBlogContent() {

return blogContent;

}

public void setBlogContent(String blogContent) {

this.blogContent = blogContent;

}

public LocalDateTime getBlogPosted() {

return blogPosted;

}

public void setBlogPosted(LocalDateTime 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 getNoFoComments() {

return noFoComments;

}

public void setNoFoComments(int noFoComments) {

this.noFoComments = noFoComments;

}

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;

}

}

**3.BlogComments:-**

package com.coll.OnlineCollaborate.model;

import java.io.Serializable;

import java.time.LocalDate;

import java.util.Locale;

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 titel;

int onOfLikes;

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 getUserProfileId() {

return userProfileId;

}

public void setUserProfileId(String userProfileId) {

this.userProfileId = userProfileId;

}

public String getTitel() {

return titel;

}

public void setTitel(String titel) {

this.titel = titel;

}

public int getOnOfLikes() {

return onOfLikes;

}

public void setOnOfLikes(int onOfLikes) {

this.onOfLikes = onOfLikes;

}

public static long getSerialversionuid() {

return serialVersionUID;

}

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;

}

}

🡪Create **package** com.coll.OnlineCollaborate.dao;

🡪Create Interface :-

1**.IUserDao:-**

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 getUserByName(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);

boolean logoutUser(int userId);

List<User> getAllDeactiveUser();

boolean updateUserProfile(String file, Integer userId);

}

**2.IBlogDao:-**

package com.coll.OnlineCollaborate.dao;

import java.util.List;

import com.coll.OnlineCollaborate.model.Blog;

import com.coll.OnlineCollaborate.model.User;

public interface IBlogDao {

List<Blog> getAllBlogs();

List<Blog> getBlogsByStatus(String status);

List<Blog> getUsersBlogs(int id);

Blog getBlogById(int blogId);

boolean addBlog(Blog blog);

boolean updateBlog(Blog blog);

boolean activeBlog(int blogId);

List<Blog> getAllDeactiveBlog();

boolean deleteBlog(int blogId);

List<Blog> getBlogbyUserId(int userId);

}

**3.IBlogCommentsDao:-**

package com.coll.OnlineCollaborate.dao;

import java.util.List;

import com.coll.OnlineCollaborate.model.Blog;

import com.coll.OnlineCollaborate.model.BlogComments;

public interface IBlogCommentsDao {

List<BlogComments> getAllBlogComments();

List<BlogComments> getCommentsbyBlogId(int blogId);

BlogComments getBlogCommentsById(int blogCommentsId);

boolean addBlogComments(BlogComments blogComments);

boolean updateBlogComments(BlogComments blogComments);

boolean deleteBlogComments(int blogCommentsId);

}

🡪Create **package** com.coll.OnlineCollaborate.daoImpl;

🡪Create DaoImpl Class

**1.UserDaoImpl:-**

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 where enabled='true'",User.class).getResultList();

}

@Override

public User getUserById(int userId) {

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

}

@Override

public User getUserByName(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();

user.setIsOnline("true");

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");

user.setStatus("Inactive");

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");

user.setStatus("Active");

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();

}

@Override

public boolean logoutUser(int userId) {

try {

User user=getUserById(userId);

user.setIsOnline("false");

sessionFactory.getCurrentSession().update(user);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

}

**2.BlogDaoImpl:-**

package com.coll.OnlineCollaborate.daoImpl;

import java.time.LocalDate;

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;

import com.coll.OnlineCollaborate.model.User;

@Repository("blogDao")

@Transactional

public class BlogDaoImpl implements IBlogDao{

@Autowired

SessionFactory sessionFactory;

@Override

public List<Blog> getAllBlogs() {

return sessionFactory.getCurrentSession().createQuery("from Blog where status='Posted'",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> getUsersBlogs(int id) {

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

}

@Override

public Blog getBlogById(int blogId) {

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

}

@Override

public boolean addBlog(Blog blog) {

try {

System.out.println(blog.getBlogPosted());

sessionFactory.getCurrentSession().save(blog);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean updateBlog(Blog blog) {

try {

sessionFactory.getCurrentSession().update(blog);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean deleteBlog(int blogId) {

try {

System.out.println(blogId);

sessionFactory.getCurrentSession().delete(getBlogById(blogId));

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public boolean activeBlog(int blogId) {

try {

Blog blog=getBlogById(blogId);

blog.setStatus("Posted");

sessionFactory.getCurrentSession().update(blog);

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public List<Blog> getAllDeactiveBlog() {

return sessionFactory.getCurrentSession().createQuery("from Blog where status='Waiting For Apporved'",Blog.class).getResultList();

}

@Override

public List<Blog> getBlogbyUserId(int userId) {

return sessionFactory.getCurrentSession().createQuery("from Blog where userId='"+userId+"'",Blog.class).getResultList();

}

}

**3.BlogComentsDaoImpl:-**

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.Blog;

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 blogId) {

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

}

@Override

public boolean addBlogComments(BlogComments blogComments) {

try {

sessionFactory.getCurrentSession().save(blogComments);

return true;

}

catch(Exception ex) {

ex.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(int blogCommentsId) {

try {

sessionFactory.getCurrentSession().delete(getBlogCommentsById(blogCommentsId));

return true;

}

catch(Exception ex) {

ex.printStackTrace();

return false;

}

}

@Override

public List<BlogComments> getCommentsbyBlogId(int blogId) {

return (List<BlogComments>) sessionFactory.getCurrentSession().get(BlogComments.class,Integer.valueOf(blogId));

}

}

🡪Create **package** com.coll.OnlineCollaborate.service;

🡪Create **Service Inteface**

**1.IUserService:**

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 getUserByName(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);

boolean logoutUser(int userId);

List<User> getAllDeactiveUser();

boolean updateUserProfile(String file, Integer userId);

}

**2.IBlogService:-**

package com.coll.OnlineCollaborate.service;

import java.util.List;

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

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

import com.coll.OnlineCollaborate.model.Blog;

public interface IBlogService {

List<Blog> getAllBlogs();

List<Blog> getBlogsByStatus(String status);

List<Blog> getUsersBlogs(int id);

Blog getBlogById(int blogId);

boolean addBlog(Blog blog);

boolean updateBlog(Blog blog);

boolean activeBlog(int blogId);

List<Blog> getAllDeactiveBlog();

boolean deleteBlog(int blogId);

List<Blog> getBlogbyUserId(int userId);

}

**3.IBlogCommentsService:-**

package com.coll.OnlineCollaborate.service;

import java.util.List;

import com.coll.OnlineCollaborate.model.BlogComments;

public interface IBlogCommentsService {

List<BlogComments> getAllBlogComments();

List<BlogComments> getCommentsbyBlogId(int blogId);

BlogComments getBlogCommentsById(int blogCommentsId);

boolean addBlogComments(BlogComments blogComments);

boolean updateBlogComments(BlogComments blogComments);

boolean deleteBlogComments(int blogCommentsId);

}

🡪Create **package** com.coll.OnlineCollaborate.serviceImpl;

🡪Create **ServiceImpl Class:-**

**1. UserServiceImpl**:-

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 getUserByName(String username) {

// TODO Auto-generated method stub

return userDao.getUserByName(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();

}

@Override

public boolean logoutUser(int userId) {

// TODO Auto-generated method stub

return userDao.logoutUser(userId);

}

}

**2.BlogServiceImpl:-**

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 blogId) {

return blogDao.getUsersBlogs(blogId);

}

@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(int blogId) {

return blogDao.deleteBlog(blogId);

}

@Override

public boolean activeBlog(int blogId) {

// TODO Auto-generated method stub

return blogDao.activeBlog(blogId);

}

@Override

public List<Blog> getAllDeactiveBlog() {

// TODO Auto-generated method stub

return blogDao.getAllDeactiveBlog();

}

@Override

public List<Blog> getBlogbyUserId(int userId) {

// TODO Auto-generated method stub

return blogDao.getBlogbyUserId(userId);

}

}

**3.BlogCommentsServiceImpl:-**

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 BlogCommentsServiceImpl implements IBlogCommentsService {

@Autowired

IBlogCommentsDao blogCommentsDao;

@Override

public List<BlogComments> getAllBlogComments() {

return blogCommentsDao.getAllBlogComments();

}

@Override

public BlogComments getBlogCommentsById(int blogCommentsId) {

return blogCommentsDao.getBlogCommentsById(blogCommentsId);

}

@Override

public boolean addBlogComments(BlogComments blogComments) {

return blogCommentsDao.addBlogComments(blogComments);

}

@Override

public boolean updateBlogComments(BlogComments blogComments) {

return blogCommentsDao.updateBlogComments(blogComments);

}

@Override

public boolean deleteBlogComments(int blogCommentsId) {

return blogCommentsDao.deleteBlogComments(blogCommentsId);

}

@Override

public List<BlogComments> getCommentsbyBlogId(int blogId) {

return blogCommentsDao.getCommentsbyBlogId(blogId);

}

}

🡪Create **package** com.coll.OnlineCollaborate.controller;

🡪Create **Controllers:**

**1.UserController**

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);

}

@PostMapping("validate-user")

public User validateUser(@RequestBody User user) {

return userService.validateUser(user);

}

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

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

return userService.logoutUser(userId);

}

}

**2.BlogController:-**

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.model.User;

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/{blogId}")

public boolean deleteBlog(@PathVariable("blogId") int blogId) {

return blogService.deleteBlog(blogId);

}

@GetMapping("blog/{blogId}")

public Blog getblogById(@PathVariable("blogId") int blogId)

{

return blogService.getBlogById(blogId);

}

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

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

{

blog.setBlogId(blogId);

return blogService.updateBlog(blog);

}

@GetMapping("deactiveBlog-list")

public List<Blog> AllDeactiveBlog(){

return blogService.getAllDeactiveBlog();

}

@PostMapping("active-blog/{blogId}")

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

return blogService.activeBlog(blogId);

}

@GetMapping("myblogs/{userId}")

public List<Blog> getBlogbyUserId(@PathVariable("userId") int userId){

return blogService.getBlogbyUserId(userId);

}

}

**3.BlogCommentsController:-**

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.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 saveBlog(@RequestBody BlogComments blogComments)

{

return blogCommentsService.addBlogComments(blogComments);

}

@GetMapping("blogComments-list")

public List<BlogComments> allBlogComments()

{

return blogCommentsService.getAllBlogComments();

}

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

public boolean deleteBlogComments(@PathVariable("blogComments\_id") int blogComments\_id) {

return blogCommentsService.deleteBlogComments(blogComments\_id);

}

@GetMapping("blogComments/{blogComments\_id}")

public BlogComments blogCommentsById(@PathVariable("blogComments\_id") int blogComments\_id)

{

return blogCommentsService.getBlogCommentsById(blogComments\_id);

}

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

public boolean updateBlogComments(@PathVariable("blogComments") BlogComments blogComments)

{

return blogCommentsService.updateBlogComments(blogComments);

}

}

🡪 Open **OnlineCollaborateApplication**

package com.coll.OnlineCollaborate;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class OnlineCollaborateApplication {

public static void main(String[] args) {

SpringApplication.run(OnlineCollaborateApplication.class, args);

}

}

🡪 Right Click OnlineCollaborateApplication

🡪Click Run as

🡪 Click JavaApplication

**🡪Create Database**

create database collaboration;

🡪Active Databse using follwing commends:

use collaboration;

🡪Create Table for user

create tabl user(

userId int not null auto\_increment,

firstName varchar(30),

lastName varchar(30),

username varchar(20),

password varchar(30),

email varchar(30),

role varchar(30),

status varchar(10),

isOnline varchar(5),

enabled varchar(5),

primary key(userId)

);

select \* from user;

🡪Create Table for blog

create table Blog (

blogId integer not null auto\_increment,

blogContent varchar(255),

blogPosted dateTime,

blogTitel varchar(255),

noFoComments integer not null,

noOfLikes integer not null,

noOfViews integer not null,

status varchar(255),

userId integer not null,

username varchar(255),

primary key (blogId));

select \* from Blog;

🡪Create Table for Blogcomments

create table BlogComments (

blogCommentId integer not null auto\_increment,

blogComment varchar(255),

currentDate date,

onOfLikes integer not null,

titel varchar(255),

userId integer not null,

userProfileId varchar(255),

username varchar(255),

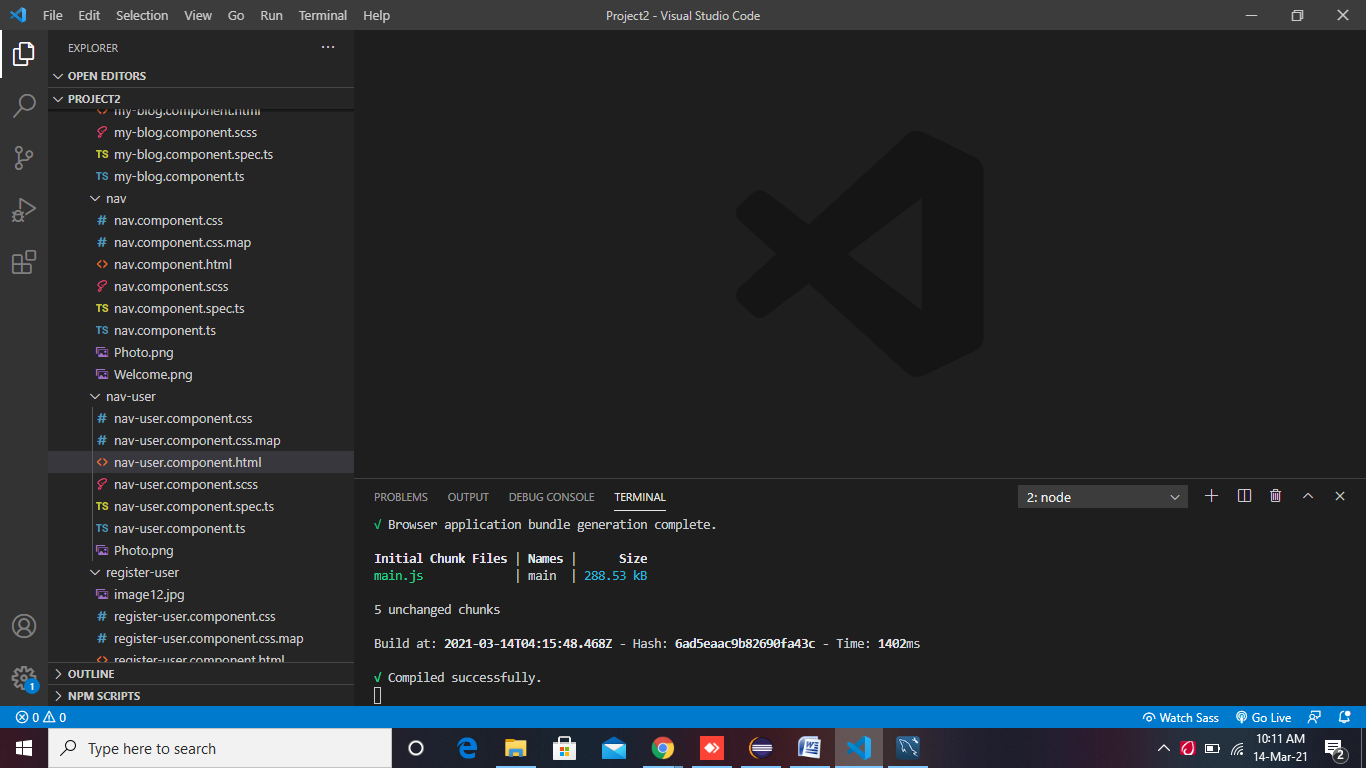
blogId integer ,

primary key (blogCommentId));

drop table blogcomments;

drop table blog;

Front End Procedure;-



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

Ng new OnlineCollaborateAngular

🡪 Install Bootstrap

Use the following command to install bootstrap in the project.

PS F:\AngularWorkSpace\OnlineCollaborateAngular> ***npm install*** [***bootstrap@3.3.7***](mailto:bootstrap@3.3.7) ***–save***

🡪 Install Angular Data Table

Use the following command to install bootstrap in the project.

PS F:\AngularWorkSpace\OnlineCollaborateAngular> ***npm install angular-datatable --save***

🡪Add Angular Material Using Follwing command

* ng add @angular/material (select custom)

🡪Create Navigation using angular material use the following commands

* ng generate @angular/material:navigation nav
* ng generate @angular/material:navigation nav-user

🡪 Generate Components  
Open the project in visual studio and then use the following command to generate Angular components:

* *ng g c home*
* *ng g c login-user*
* *ng g c register-use*
* *ng g c user-list*
* *ng g c user-profile*
* *ng g c blog-list*
* *ng g c bloglists*
* *ng g c my-blog*
* *ng g c active-user*
* *ng g c active-blog*

🡪 Let's also create a service class by using the following command: -

* ng g s user
* ng g s blog

🡪 Edit the **app.module.ts** file

* 1. **Import Routing** - Here, we are importing routing file (app-routing.module.ts) and include it in imports array.
  2. **Import ReactiveFormsModule** - Here, we are importing **ReactiveFormsModule** for reactive forms and specify it in imports array.
  3. **Import HttpModule** - Here, we are importing **HttpModule** for server requests and specifying it in imports array.
  4. **Register Service class** - Here, we are mentioning the service class in provider's array.

--//app.module.ts

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

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

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

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

import {DataTablesModule } from 'angular-datatables';

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

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

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

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

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

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

import { NavComponent } from './nav/nav.component';

import { LayoutModule } from '@angular/cdk/layout';

import { MatToolbarModule } from '@angular/material/toolbar';

import { MatButtonModule } from '@angular/material/button';

import { MatSidenavModule } from '@angular/material/sidenav';

import { MatIconModule } from '@angular/material/icon';

import { MatListModule } from '@angular/material/list';

import { UserProfileComponent } from './user-profile/user-profile.component';

import { BlogListComponent } from './blog-list/blog-list.component';

import { ActiveUserComponent } from './active-user/active-user.component';

import { HomeComponent } from './home/home.component';

import { NavUserComponent } from './nav-user/nav-user.component';

import { AboutUsComponent } from './about-us/about-us.component';

import { BloglistsComponent } from './bloglists/bloglists.component';

import { ActiveBlogComponent } from './active-blog/active-blog.component';

import { MyBlogComponent } from './my-blog/my-blog.component';

@NgModule({

  declarations: [

    AppComponent,

    RegisterUserComponent,

    UserListComponent,

    LoginUserComponent,

    NavComponent,

    UserProfileComponent,

    BlogListComponent,

    ActiveUserComponent,

    HomeComponent,

    NavUserComponent,

    AboutUsComponent,

    BloglistsComponent,

    ActiveBlogComponent,

    MyBlogComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

    ReactiveFormsModule,

    HttpClientModule,

    DataTablesModule, BrowserAnimationsModule, LayoutModule, MatToolbarModule, MatButtonModule, MatSidenavModule, MatIconModule, MatListModule

    ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

🡪 Edit the **app-routing.module.ts** file

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

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

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

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

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

import {ActiveUserComponent} from './active-user/active-user.component';

import {BlogListComponent } from './blog-list/blog-list.component';

import {UserProfileComponent} from './user-profile/user-profile.component';

import { HomeComponent } from './home/home.component';

import { NavComponent } from './nav/nav.component';

import { NavUserComponent } from './nav-user/nav-user.component';

import { BloglistsComponent } from './bloglists/bloglists.component';

import { ActiveBlogComponent } from './active-blog/active-blog.component';

import { AboutUsComponent } from './about-us/about-us.component';

import { MyBlogComponent } from './my-blog/my-blog.component';

const routes: Routes = [

  { path: '', redirectTo: 'home', pathMatch: 'full' },

  {path:'home', component: HomeComponent},

  {path:'about-us', component: AboutUsComponent},

  {path:'nav/:Id', component:NavComponent,

  children: [

    { path: 'user-profile/:Id' , component: UserProfileComponent},

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

    { path: 'active-user' , component: ActiveUserComponent},

  { path: 'blog-list/:Id' , component: BlogListComponent},

  { path: 'bloglists' , component: BloglistsComponent},

  { path: 'active-blog' , component: ActiveBlogComponent},

  { path: 'my-blog/:Id' , component: MyBlogComponent}

  ]

} ,

  {path:'nav-user/:Id', component:NavUserComponent,

   children:[

    { path: 'blog-list/:Id' , component: BlogListComponent},

    { path: 'user-profile/:Id' , component: UserProfileComponent},

    { path: 'my-blog/:Id' , component: MyBlogComponent}

   ]

},

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

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

  ];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

🡪 Edit the **app.component.html** file(remove existing code and replace with following code)

<router-outlet>

</router-outlet>

🡪 Create the **user.ts** class

🡪Let's create a class by using the following command: -

* *ng g class user*

🡪 Now, specify the required fields within the **User** class. (The purpose of this class is to map the specified fields with the fields of Spring entity class.)

export class User {

    userId:number;

    firstName:String;

    lastName:String;

    username:String;

    password:String;

    confirm\_password:String;

    email:String;

    role:String;

    status:String;

    isOnline:String;

    enabled:String;

}

🡪 Edit the **user.service.ts** file

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

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

import { Observable } from 'rxjs';

@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);

          }

        deactiveList():Observable<any> {

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

        }

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

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

            }

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

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

            }

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

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

                }

}

🡪 Create the **blog.ts** class

🡪Let's create a class by using the following command: -

* *ng g class blog*

🡪 Now, specify the required fields within the **Blog** class.

export class Blog{

    blogId:number;

    blogTitel:String;

    blogContent:String;

    blogPosted:any;

    status:String;

    noOfLikes:number;

    noFoComments:number;

    noOfViews:number;

    userId:number;

    username:String;

}

🡪 Edit the **blog.service.ts** file

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

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

import { Observable } from 'rxjs';

import { Blog } from './blog';

@Injectable({

  providedIn: 'root'

})

export class BlogService {

  blog: Blog=new Blog();

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

  constructor(private http:HttpClient){ }

  getBlogList(): Observable<any>{

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

  }

  createBlog(blog: object): Observable<object>{

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

  }

  deleteBlog(blogId:number): Observable<any>{

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

  }

  getBlog(blogId:number): Observable<Object>{

    return this.http.get(`${this.baseUrl}/blog/${blogId}`);

  }

  updateBlog(blogId:number, blog: object): Observable<Object>{

    return this.http.post(`${this.baseUrl}/update-blog/${blogId}`,blog);

  }

  deactiveBlogList():Observable<any> {

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

  }

  activeBlog(blogId: number): Observable<Object> {

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

      }

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

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

      }

}

🡪 Edit the **register-user.component.ts** file

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

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

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

import { User } from '../user';

@Component({

  selector: 'app-register-user',

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

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

})

export class RegisterUserComponent implements OnInit {

  constructor(private userservice: UserService) { }

  user: User = new User();

  submitted = false;

  ngOnInit(): void {

    this.submitted = false;

  }

  registrationform = new FormGroup({

    firstName: new FormControl('', [Validators.required, Validators.minLength(3)]),

    lastName: new FormControl('', [Validators.required, Validators.minLength(3)]),

    username: new FormControl('', [Validators.required, Validators.minLength(3)]),

    password: new FormControl('', [Validators.required, Validators.minLength(3)]),

    confirm\_password: new FormControl('', [Validators.required, Validators.minLength(3)]),

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

    role: new FormControl()

  });

  register(register: any) {

    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;

    console.log(this.user.firstName);

    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');

  }

  get Status() {

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

  }

  registrationForm() {

    this.submitted = false;

    this.registrationform.reset();

  }

  logInUser() {

    $(".pages").css("visibility","visible");

    $(".peru").css("visibility","hidden");

  }

}

🡪 Edit the **register-user.component.html** file

<div class="peru">

<div class="main">

    <div class="row">

        <h2 style="font-weight: bold; font-style: italic; text-align: center">Registration Form</h2>

        <div class="cols-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" class="form-control" formControlName="firstName" data-toggle="tooltip"

                            data-placement="right" titel="Enter First Name" placeholder="Enter Your First Name">

                        <div class="alert alert-info" \*ngIf="(FirstName.touched) && (FirstName.invalid)"

                            style="margin-top: 5px;">

                            <span \*ngIf="FirstName.errors.required">  <strong>First Name</strong> is Required</span>

                            <span \*ngIf="FirstName.errors.minlength">MinLength Error</span>

                        </div>

                    </div>

                    <div class="form-group">

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

                        <input type="text" class="form-control" formControlName="lastName" data-toggle="tooltip"

                            data-placement="right" titel="Enter Last Name" placeholder="Enter Your Last Name">

                        <div class="alert alert-info" \*ngIf="(LastName.touched) && (LastName.invalid)"

                            style="margin-top: 5px;">

                            <span \*ngIf="LastName.errors.required"><strong>Last Name</strong> is Required</span>

                            <span \*ngIf="LastName.errors.minlength">MinLength Error</span>

                        </div>

                    </div>

                    <div class="form-group">

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

                        <input type="text" class="form-control" formControlName="username" data-toggle="tooltip"

                            data-placement="right" titel="Enter Username" placeholder="Enter Your User Name">

                        <div class="alert alert-info" \*ngIf="(Username.touched) && (Username.invalid)"

                            style="margin-top: 5px;">

                            <span \*ngIf="Username.errors.required"><strong>Username</strong> is Required</span>

                            <span \*ngIf="Username.errors.minlength">MinLength Error</span>

                        </div>

                    </div>

                    <div class="form-group">

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

                        <input type="password" class="form-control" formControlName="password" data-toggle="tooltip"

                            data-placement="right" titel="Enter Password" placeholder="Enter Your Password">

                        <div class="alert alert-info" \*ngIf="(Password.touched) && (Password.invalid)"

                            style="margin-top: 5px;">

                            <span \*ngIf="Password.errors.required"><strong>Password</strong> is Required</span>

                            <span \*ngIf="Password.errors.minlength">MinLength Error</span>

                        </div>

                    </div>

                    <div class="form-group">

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

                        <input type="password" class="form-control" formControlName="confirm\_password"

                            data-toggle="tooltip" data-placement="right" titel="Enter Confirm Password"

                            placeholder="Enter Your Confirm Password">

                        <div class="alert alert-info" \*ngIf="(ConfirmPassword.touched) && (ConfirmPassword.invalid)"

                            style="margin-top: 5px;">

                            <span \*ngIf="ConfirmPassword.errors.required"><strong>Confirm Password</strong> is Required</span>

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

                                mathc</span>

                            <span \*ngIf="Password.errors.minlength">MinLength Error</span>

                        </div>

                    </div>

                    <div class="form-group">

                        <label for="email">E-Mail</label>

                        <input type="text" class="form-control" formControlName="email" data-toggle="tooltip"

                            data-placement="right" titel="Enter Email Id" placeholder="Enter Your E-Mail Address">

                        <div class="alert alert-info" \*ngIf="(Email.touched) && (Email.invalid)"

                            style="margin-top: 5px;">

                            <span \*ngIf="Email.errors.required"><strong>Email</strong> is Required</span>

                            <span \*ngIf="Email.errors.email"><strong>Invalid Email </strong> Format</span>

                        </div>

                    </div>

                    <div class="form-group">

                        <label for="role">Select Role:-</label>

                        <select class="form-control" formControlName="role" class="optionbox" data-toggle="tooltip" data-placement="right"

                            title="Select Role">

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

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

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

                        </select>

                    </div>

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

                    <a routerLink="login-user"  (click)="logInUser()">Alredy have an account..?</a>

                </form>

            </div>

        </div>

        <div [hidden]="!submitted">

            <h4 style="font-weight: bold; font-style: italic; text-align: center; color:white;">Register request sent Successfully..!!</h4>

        </div>

    </div>

</div>

</div>

🡪 Edit the **register-user.component.scss** file:

body {

    margin: 0;

    padding: 0;

    font-family: sans-serif;

    background: linear-gradient(to right, #b92b27, #1565c0)

}

.row {

    width: 500px;

    padding: 40px;

    position: absolute;

    border: 2px solid #3498db;

    border-radius: 25px;

    background:brown;

    text-align: center;

    transition: 0.25s;

    margin-top: 100px

}

.row input[type="text"],

.row input[type="password"],

.row select {

    border: 0;

    background: none;

    display: block;

    margin: 20px auto;

    text-align: center;

    border: 2px solid #3498db;

    padding: 10px 10px;

    width: 250px;

    outline: none;

    color: white;

    border-radius: 24px;

    transition: 0.25s

}

.row label{

    color:aliceblue;

}

.row h2,h4 {

    color: white;

    text-transform: uppercase;

    font-weight: 500;

}

.row label:hover{

   color: yellow;

}

.row h2:hover{

    color:yellow;

}

.row a{

    color:white;

}

.row a:hover{

    color:yellow;

}

.row input[type="text"]:focus,

.row input[type="password"]:focus,

.row select:focus {

    width: 300px;

    border-color: #2ecc71

}

.row option{

    background-color:black;

}

.row  button[type="submit"] {

    border: 0;

    background: none;

    display: block;

    margin: 20px auto;

    text-align: center;

    border: 2px solid #2ecc71;

    padding: 14px 40px;

    outline: none;

    color: white;

    border-radius: 24px;

    transition: 0.25s;

    cursor: pointer

}

.row

button[type="submit"]:hover {

    background: #2ecc71

}

::placeholder {

    color:white;

    opacity: 1; /\* Firefox \*/

  }

  :-ms-input-placeholder { /\* Internet Explorer 10-11 \*/

   color:white;

  }

  ::-ms-input-placeholder { /\* Microsoft Edge \*/

   color:white;

  }

  .pages{

    position:absolute;

    top:10%;

    left:30%;

    visibility: hidden;

}

.form-control{

    width:55%;

    border-radius:5px;

    border: 2px solid rgba(76, 46, 207, 0.856);

    font-size: 13px;

    font-family:monospace;

    }

🡪 Edit the **user-list.component.ts** file

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

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

import { User } from '../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 {

  constructor(private userservice: UserService) { }

  userArray: any[] = [];

  dtOptions: DataTables.Settings = {};

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

  users!: Observable <User[]>;

  user: User = new User();

  deleteMessage = false;

  userlist: any;

  isupdated = false;

  ngOnInit() {

    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(userId: number) {

    this.userservice.deleteUser(userId)

      .subscribe(

        (data) => {

          console.log(data);

          this.deleteMessage = true;

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

            this.users = data

          })

        },

        (error) => console.log(error));

  }

  updatUser(id: number) {

    this.userservice.getUser(id)

      .subscribe(

        (data) => {

          this.userlist = data;

          console.log(this.userlist);

        }),

        (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()

  });

  updateUser(updateusers: any) {

    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.status = this.Status.value;

    this.user.isOnline = this.IsOnline.value;

    this.user.enabled = this.Enabled.value;

    console.log(this.FirstName.value);

    this.userservice.updateUser(this.user.userId, this.user).subscribe(

      data => {

        this.isupdated = true;

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

          this.users = data

        })

      },

        error => {

          console.log(this.users);

          console.log(error)});

  }

  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');

  }

  get UserId() {

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

  }

  changeisUpdate() {

    this.isupdated = false;

  }

}

🡪 Edit the **user-list.component.html** file

<div class="panel">

  <div class="panel panel-default">

    <div class="panel-heading">

      <h1 style="font-weight: bold; font-style: italic; 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">×</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>UserID</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> &nbsp; &nbsp;

             <button (click)="updatUser(user.userId)" class='btn btn-info' data-toggle="modal"

                data-target="#myModal">Update</button>

            </td>

          </tr>

        </tbody><br>

      </table>

    </div>

  </div>

</div>

<div class="modal" id="myModal">

  <div class="modal-dialog">

    <div class="modal-content">

      <form [formGroup]="userupdateform" #updateusers (ngSubmit)="updateUser(updateusers)">

        <!-- Modal Header -->

        <div class="modal-header">

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

        </div>

        <!-- Modal body -->

        <div class="modal-body" \*ngIf="userlist">

          <div [hidden]="isupdated">

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

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

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

              <select class="form-control" formControlName="role" required>

                <option value="Admin" [selected]="'Admin' == userlist.role">Admin</option>

                <option value="User" [selected]="'User' == userlist.role">User</option>

              </select>

            </div>

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

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

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

            </div>

          </div>

          <div [hidden]="!isupdated">

            <h4>User Detail Updated!</h4>

          </div>

        </div>

        <!-- Modal footer -->

        <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>

🡪 Edit the **user-list.component.scss** file

.panel{

  position:relative;

  left:95px;

  background-color:lightskyblue;

  }

td {

  border-right: solid 0.5px  rgb(10, 10, 10);

  border-bottom:solid 0.5px black;

}

th{

  border-top:solid 0.5px black;

  border-bottom:solid 0.5px black; ;

}

.panel th:hover{

  color:white;

}

.panel tr:hover{

  font-weight: bold;

  color:blue;

}

.panel button:hover{

  font-weight: bolder;

  color:black;

}

.panel td:nth-child(1n+1):hover{

  color:black;

}

🡪 Edit the **login-user.component.ts** file

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

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

import { User } from '../user';

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

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

@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;

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

  ngOnInit(): void {

  }

  loginform=new FormGroup({

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

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

  })

  validateUser() {

    this.user=new User();

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

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

    this.userService.checkUser(this.user).subscribe (

      data => {

        console.log(data);

        if(data!=null) {

          this.currentUser=data;

         if(this.currentUser.role==="Admin"){

           this.router.navigateByUrl("/nav/" +`${this.currentUser.userId}`);

         }

         else{

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

         }

        }

        else {

          console.log("Object Empty");

        }

      },

      error => console.log(error)

    )

  }

  get Username() {

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

  }

  get Password() {

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

  }

  RegUser(){

    $(".pagess").css("visibility","visible");

    $(".card").css("visibility","hidden");

  }

}

🡪 Edit the **login-user.component.html** file

<div class="container">

  <div class="row">

      <div class="col-md-6">

          <div class="card">

  <form [formGroup]="loginform" action="#" #validateuser="ngForm" (ngSubmit)="validateUser()" class="form1"  class="box">

    <h1>Sign In</h1>

    <input type="text" class="form-control" name="username" required placeholder="Username" formControlName="username">

    <input type="password" class="form-control" name="password" required placeholder="Password" formControlName="password">

    <button type="submit" class="submit" align="center">Login</button>

     <a class="submit"  routerLink="register-user" (click)="RegUser()">Don't have an account..?</a>

  </form>

</div>

</div>

</div>

</div>

🡪 Edit the **login-user.component.scss** file

body {

    margin: 0;

    padding: 0;

    font-family: sans-serif;

   height: 400px;

    background: linear-gradient(to right, #b92b27, #1565c0)

}

.box {

    width: 500px;

    padding: 40px;

    position: absolute;

    border: 2px solid #3498db;

    border-radius: 25px;

    background:brown;

    text-align: center;

    transition: 0.25s;

    margin-top: 100px

}

.box input[type="text"],

.box input[type="password"] {

    border: 0;

    background: none;

    display: block;

    margin: 20px auto;

    text-align: center;

    border: 2px solid #3498db;

    padding: 10px 10px;

    width: 250px;

    outline: none;

    color:white;

    border-radius: 24px;

    transition: 0.25s

}

.box h1 {

    color: white;

    text-transform: uppercase;

    font-weight: 500

}

.box h1:hover{

    color:yellow;

}

.box input[type="text"]:focus,

.box input[type="password"]:focus {

    width: 300px;

    border-color: #2ecc71

}

.box a {

    color:aliceblue;

    text-indent: 30px;

  }

  .box a:hover{

      color:yellow;

  }

.box button[type="submit"] {

    border: 0;

    background: none;

    display: block;

    margin: 20px auto;

    text-align: center;

    border: 2px solid #2ecc71;

    padding: 14px 40px;

    outline: none;

    color: white;

    border-radius: 24px;

    transition: 0.25s;

    cursor: pointer

}

.box button[type="submit"]:hover {

    background: #2ecc71

}

::placeholder {

    color:white;

    opacity: 1; /\* Firefox \*/

  }

  :-ms-input-placeholder { /\* Internet Explorer 10-11 \*/

   color:white;

  }

  ::-ms-input-placeholder { /\* Microsoft Edge \*/

   color:white;

  }

  .pagess{

    position:absolute;

    top:-20%;

    left:35%;

    visibility: hidden;

}

🡪 Edit the **home.component.ts** file

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

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

@Component({

  selector: 'app-home',

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

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

})

export class HomeComponent implements OnInit {

  constructor(private router : Router) { }

  ngOnInit(): void {

  }

  logInUser() {

    $(".pages").css("visibility","visible");

  }

  RegUser(){

    $(".pagess").css("visibility","visible");

  }

}

🡪 Edit the **home.component.html** file

<div class="container">

    <div class="navbar">

        <div class="menu">

            <h3 class="logo"><span>Online  </span>Collborate</h3>

            <div class="hamburger-menu">

                <div class="bar"></div>

            </div>

        </div>

    </div>

    <div class="main-container">

        <div class="main">

            <header>

                <div class="overlay" id="one">

                    <h2 class="title">Hello there..!!</h2>

                    <p class="description">“click here to be part of  <span><strong>Online collaboration</strong> </span>”

                    </p>

                    <div class="bts">

                       <a mat-list-item  class="nav-link" (click)="logInUser()" class="btn">Login</a>/<a mat-list-item   class="nav-link" (click)="RegUser()" class="btn">Registration</a>

                      </div>

                </div>

            </header>

        </div>

        <div class="shadow one"></div>

        <div class="shadow two"></div>

    </div>

</div>

<div class="pages">

    <app-login-user></app-login-user>

</div>

<div class="pagess">

    <app-register-user></app-register-user>

</div>

<div class="middleTop row">

    <div class="footer-inner">

        <div class="col-md-2">

            <div class="middle-heading">Policy Info</div>

            <ul class="middle-content ">

                <li><a class="underLineOnHover" href="#" data-abc="true">Privacy Policy</a></li>

            </ul>

        </div>

        <div class="col-md-2">

            <div class="middle-heading">Company</div>

            <ul class="middle-content">

                <li><a class="underLineOnHover" href="#" data-abc="true">About Us</a></li>

            </ul>

        </div>

        <div class="col-md-2">

            <div class="middle-heading">Business</div>

            <ul class="middle-content">

                <li><a class="underLineOnHover" href="#" data-abc="true">Be an Affiliate</a></li>

            </ul>

        </div>

        <div class="col-md-2">

            <div class="middle-heading">Need Help ?</div>

            <ul class="middle-content">

                <li><a class="underLineOnHover" href="#" data-abc="true">FAQ</a></li>

                <li><a class="underLineOnHover" href="#" data-abc="true">Contact Us</a></li>

            </ul> <i class="originalalityfooterIcon footerIconsImg hide"></i>

        </div>

        <div class="col-md-4">

            <div class="middle-heading" style="margin-left: 0px;">About us</div> <span class="address" style="color:aliceblue;">1/239, sivakasi, VirudhuNagar, Tamilnadu-626124</span>

        </div>

    </div>

</div>

🡪 Edit the **home.component.scss** file

body{ font-family: 'Recursive', sans-serif; overflow: hidden; }

.container{

            height:900px;

            width: 100%; background-color: #131414;

            background-image: linear-gradient(135deg, #131414 0%, #000000 100%);

            transform-style: preserve-3d; overflow: hidden;

}

.navbar{ position: fixed;

         top: 0;

         left: 0;

         width: 100%;

         z-index: 10;

         height: 5rem;

}

.menu{ max-width: 72rem;

      width: 100%;

      height: 100%;

      margin: 0 auto;

      padding: 0 2rem;

      display: flex;

      justify-content: space-between;

      align-items: center;

      color: #fff;

}

.logo{

    font-size: 1.8rem; font-weight: 600; text-transform: uppercase;

    letter-spacing: 2px; line-height: 4rem; margin-top: 20px;

    color:white;

}

.logo span{ font-size: 1.8rem; margin-left: 5px; color: #e20f2f; }

.hamburger-menu{ height: 4rem; width: 3rem; cursor: pointer;

    display: flex; align-items: center; justify-content: flex-end; }

.bar{ width: 1.9rem; height: 1.5px; border-radius: 2px;

    background-color: #eee; transition: 0.5s; position: relative; }

.bar::before, .bar::after{ content: ""; position: absolute;

    width: inherit; height: inherit; background-color: #eee;

    transition: 0.5s; }

.bar::before{ transform: translateY(-9px); }

.bar::after{ transform: translateY(9px); }

.main-container{ overflow: hidden; }

.main{ position: relative; width: 100%; left: 0; z-index: 5;

    overflow: hidden; transform-origin: left;

    transform-style: preserve-3d; transition: 0.5s;

}

header{ min-height: 100vh; width: 100%;

    background: url("image25.jpg") no-repeat top center / cover;

    position: relative;

}

.overlay{ position: absolute; width: 100%; height: 100%; top: 0;

    left: 0; background-color: rgba(0, 0, 0, 0.712);

    display: flex; justify-content: center; align-items: center;

    flex-direction: column; color: #fff;

}

.inner{ max-width: 35rem; text-align: center; color: #fff;

    padding: 0 2rem;

}

.title{ font-size: 3rem; }

.description{ margin: 10px 0; text-align: center; width: 50%;

    font-size: 1.5rem;

}

#one h2:hover{

    color:chartreuse;

}

#one .bts{

    display:inline-block ;

    padding: 0.6rem 0.8rem;

    color:crimson;

    bottom:200px;

    right:50px;

    background-color: transparent;

    border:2px solid crimson;

    font-size: 1.3rem;

    border-radius: 25px;

    transition: 3s ease;

    transition-property: background-color, color;

}

#one .description:hover{

    color:yellow;

}

#one .bts:hover{

    color:white;

    background-color: crimson;

}

.btn{

     color: #fff; text-transform: uppercase;

}

#one .btn:hover{

    color:yellow;

}

#one p span{

    color:rgb(218, 58, 58);

    font-size: 150%;

}

#one p span:hover{

    color:black;

}

.container.active .bar{ transform: rotate(360deg);

    background-color: transparent;

}

.container.active .bar::before{

    transform: translateY(0) rotate(45deg);

}

.container.active .bar::after{

    transform: translateY(0) rotate(-45deg);

}

.container.active .main{

    animation: main-animation 0.5s ease; cursor: pointer;

    transform: perspective(1300px) rotateY(20deg) translateY(10px)

                translateZ(310px) scale(0.5);

}

@keyframes main-animation{

    from{ transform: translate(0); }

    to{

        transform: perspective(1300px) rotateY(20deg) translateY(10px)

                    translateZ(310px) scale(0.5);

    }

}

.links{ position: absolute; width: 30%; right: 0; top: 0;

    height: 100vh; z-index: 2; overflow: hidden;

    display: flex; justify-content: flex-start; align-items: center;

    margin-left: 10px;

}

ul{ list-style: none; }

ul li.active a{ color: #e20f2f; }

.links a{ text-decoration: none; color: #eee; padding: 0.7rem 0;

    display: inline-block; font-size: 1.8rem; font-weight: 300;

    text-transform: uppercase; letter-spacing: 1px; transition: 0.3s;

    opacity: 0; transform: translateY(10px);

    animation: hide 0.5s forwards ease;

}

.links a:hover{ color: #e20f2f; }

.container.active .links a{

    animation: appear 0.5s forwards ease var(--i);

}

.pages{

    position:absolute;

    top:10%;

    left:30%;

    visibility: hidden;

}

.pagess{

    position:absolute;

    top:-20%;

    left:35%;

    visibility: hidden;

}

.middleTop {

    margin: 0 auto;

    padding: 50px 0;

    max-width: 1600px;

    min-width: 1024px;

    background-color: #000000db

}

.middle-heading {

    font-family: "font-bold";

    font-size: 14px;

    color: #fff;

    margin-left: 38px;

    padding-bottom: 15px;

    text-transform: uppercase

}

.middle-content {

    font-size: 12px;

    color: #777 !important;

    list-style-type: none;

    max-width: 210px

}

.middleContent-footer .middleTop div {

    box-sizing: border-box

}

.middleBottom {

    background-color: #000;

    text-align: center;

    border-top: 1px solid #e2e2e2;

    border-bottom: 1px solid #e2e2e2;

    padding: 12px 0

}

.middleBottomFooter {

    text-transform: uppercase;

    font-size: 14px;

    color: #344c55;

    text-align: left;

    background-color: #000

}

.underLineOnHover {

    color: #fff

}

.bottomfooter {

    background-color: #000

}

.copyright {

    color: #fff

}

🡪 Edit the **nav.component.html** file

<mat-sidenav-container class="sidenav-container">

  <mat-sidenav #drawer class="sidenav" fixedInViewport

      [attr.role]="(isHandset$ | async) ? 'dialog' : 'navigation'"

      [mode]="(isHandset$ | async) ? 'over' : 'side'"

      [opened]="(isHandset$ | async) === false">

    <mat-toolbar><span class="material-icons">

      toc

      </span></mat-toolbar>

    <mat-nav-list>

        <a mat-list-item routerLink="user-profile/{{user.userId}}" class="nav-link" class="btn-primary active" (click)="Profile()" role="button">My Profile</a>

       <br/>

        <a mat-list-item routerLink="user-list" class="nav-link" class="btn-primary active" role="button">View User</a>

        <br/>

        <a mat-list-item routerLink="active-user" class="nav-link" class="btn-primary active" role="button">Active User</a>

        <br/>

       <a mat-list-item routerLink="bloglists" class="nav-link" class="btn-primary active" role="button">Blog Lists</a>

       <br/>

       <a mat-list-item routerLink="active-blog" class="nav-link" class="btn-primary active" role="button">Active Blog Lists</a>

      </mat-nav-list>

  </mat-sidenav>

  <mat-sidenav-content>

    <mat-toolbar>

      <span style="font-style: italic;">OnlineCollaborate</span>

      <span title="Logout" class="logout" class="material-icons" (click)="logout()">

        power\_settings\_new

        </span>

    </mat-toolbar>

  </mat-sidenav-content>

</mat-sidenav-container>

<div class="Profile">

 <p>Welcome</p>

  <h1 class="text-3xl">{{user.firstName+" "+user.lastName}}</h1>

  &nbsp; &nbsp; <span class="text-1xl">{{user.role}}</span>

</div>

<router-outlet></router-outlet>

🡪 Edit the **nav.component.ts** file

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

import { BreakpointObserver, Breakpoints } from '@angular/cdk/layout';

import { Observable } from 'rxjs';

import { map, shareReplay } from 'rxjs/operators';

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

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

import { param } from 'jquery';

@Component({

  selector: 'app-nav',

  templateUrl: './nav.component.html',

  styleUrls: ['./nav.component.scss']

})

export class NavComponent implements OnInit {

   Id:number;

   user:any;

  isHandset$: Observable<boolean> = this.breakpointObserver.observe(Breakpoints.Handset)

    .pipe(

      map(result => result.matches),

      shareReplay()

    );

  constructor(private breakpointObserver: BreakpointObserver, private route: ActivatedRoute, private userservice: UserService, private router:Router) {}

ngOnInit(){

   this.route.params.subscribe (

    (params:Params)=> {

      this.Id=+params["Id"];

      console.log(this.Id);

      this.userservice.getUser(this.Id).subscribe (

        data=>{

          this.user=data;

          console.log(this.user);

        }

   )

    }

   )

}

logout(){

   this.router.navigateByUrl("/home");

   this.userservice.logoutUser(this.Id).subscribe(

     data=>console.log(data)

   );

}

Profile(){

  $(".Profile").css("visibility","hidden");

}

}

🡪 Edit the **nav.component.scss** file

.sidenav-container {

  height: 10%;

  width: 1560px;

}

.sidenav {

  width: 200px;

}

.sidenav .mat-toolbar {

  background: inherit;

}

.mat-toolbar.mat-primary {

  position: sticky;

  top: 0;

  z-index: 1;

}

.logout{

  position: absolute;

  right:250px;

  color:blue;

  font-size: small;

  border-radius: 8px;

  border-color:black;

}

p{

  left:420px;

  top:60px;

  position: relative;

  color:green;

  font-family: Verdana, Geneva, Tahoma, sans-serif;

  font-size: large;

  font-style: italic;

  font-weight: bold;

  text-transform: uppercase;

}

mat-sidenav{

  background-color:darkgray;

}

mat-nav-list a{

   border-radius: 20px;

   background-color:darkgray;

   color:black;

   font-weight: bold;

}

mat-toolbar{

  background-color:mediumturquoise;

  font-family: Verdana;

}

mat-toolbar span:nth-child(2){

  position: relative;

  border: 2px solid red;

  right:-700px;

  color:red;

  border-radius: 5px;

}

mat-toolbar span:nth-child(2):hover{

    border-radius: 40px;

}

p{

  left:130px;

  top:10px;

  position: relative;

  color:blue;

  font-family: Verdana, Geneva, Tahoma, sans-serif;

  font-size: large;

  font-style: italic;

  font-weight: bold;

  text-transform: uppercase;

}

.Profile{ height: 200px; width: 30%;

  left:200px;

  background: url("Photo.png") no-repeat top center / cover;

  position: relative;

}

h1{

   top:20px;

  left:420px;

  color:darkorange;

  font-size: 30px;

  font-style: italic;

  position: relative;

}

.Profile span{

  top:20px;

  left:420px;

  font-size:20px;

  position: relative;

}

h1:hover{

  color:mediumseagreen;

}

🡪 Edit the **nav-user.component.ts** file

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

import { BreakpointObserver, Breakpoints } from '@angular/cdk/layout';

import { Observable } from 'rxjs';

import { map, shareReplay } from 'rxjs/operators';

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

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

import { param } from 'jquery';

@Component({

  selector: 'app-nav-user',

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

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

})

export class NavUserComponent implements OnInit {

  Id:number;

  user:any;

  isHandset$: Observable<boolean> = this.breakpointObserver.observe(Breakpoints.Handset)

    .pipe(

      map(result => result.matches),

      shareReplay()

    );

    constructor(private breakpointObserver: BreakpointObserver, private route: ActivatedRoute, private userservice: UserService, private router:Router) {}

    ngOnInit(){

       this.route.params.subscribe (

        (params:Params)=> {

          this.Id=+params["Id"];

          console.log(this.Id);

          this.userservice.getUser(this.Id).subscribe (

            data=>{

              this.user=data;

              console.log(this.user);

            }

       )

        }

       )

      }

      logout(){

        this.router.navigateByUrl("/home");

        this.userservice.logoutUser(this.Id).subscribe(

          data=>console.log(data)

        );

     }

     Profile(){

  $(".Profile").css("visibility","hidden");

}

    }

🡪 Edit the **nav-user.component.html** file

<mat-sidenav-container class="sidenav-container">

  <mat-sidenav #drawer class="sidenav" fixedInViewport

      [attr.role]="(isHandset$ | async) ? 'dialog' : 'navigation'"

      [mode]="(isHandset$ | async) ? 'over' : 'side'"

      [opened]="(isHandset$ | async) === false">

    <mat-toolbar><span class="material-icons">

      toc

      </span></mat-toolbar>

    <mat-nav-list>

        <a mat-list-item routerLink="user-profile/{{user.userId}}" class="nav-link" class="btn-primary active" (click)="Profile()" role="button" >Profile</a>

        <a mat-list-item routerLink="blog-list/{{user.userId}}" class="nav-link" class="btn-primary active" role="button">Blog</a>

         <a mat-list-item routerLink="my-blog/{{user.userId}}" class="nav-link" class="btn-primary active" role="button">My Blogs</a>

    </mat-nav-list>

  </mat-sidenav>

  <mat-sidenav-content>

    <mat-toolbar>

      <span>OnlineCollaborate</span>

      <span title="Logout" class="logout" class="material-icons" (click)="logout()">

        power\_settings\_new

        </span>

    </mat-toolbar>

  </mat-sidenav-content>

</mat-sidenav-container>

<div class="Profile">

  <p>Welcome</p>

   <h1 class="text-3xl">{{user.firstName+" "+user.lastName}}</h1>

   &nbsp; &nbsp; <span class="text-1xl">{{user.role}}</span>

 </div>

<router-outlet></router-outlet>

🡪 Edit the **nav-user.component.scss** file

.sidenav-container {

  height: 10%;

  width: 1560px;

}

.sidenav {

  width: 200px;

}

.sidenav .mat-toolbar {

  background: inherit;

}

.mat-toolbar.mat-primary {

  position: sticky;

  top: 0;

  z-index: 1;

}

mat-sidenav{

  background-color:darkgray;

}

mat-nav-list a{

   border-radius: 20px;

   background-color:darkgray;

   color:black;

   font-weight: bold;

}

mat-toolbar{

  background-color:mediumturquoise;

  font-family: Verdana;

}

mat-toolbar span:nth-child(2){

  position: relative;

  border: 2px solid red;

  right:-700px;

  color:red;

  border-radius: 5px;

}

mat-toolbar span:nth-child(2):hover{

    border-radius: 40px;

}

.welcome{

  position:relative;

  font-style: oblique;

  font-weight:large;

  left:250px;

  top:10px;

}

.welcome span{

  color:crimson;

  font-family: Verdana, Geneva, Tahoma, sans-serif;

  font-size: large;

  font-style: italic;

  font-weight: bold;

  text-transform: uppercase;

}

.logout{

  position: absolute;

  right:250px;

  color:blue;

  font-size: small;

  border-radius: 8px;

  border-color:black;

}

p{

  left:130px;

  top:10px;

  position: relative;

  color:blue;

  font-family: Verdana, Geneva, Tahoma, sans-serif;

  font-size: large;

  font-style: italic;

  font-weight: bold;

  text-transform: uppercase;

}

.Profile{ height: 200px; width: 30%;

  left:200px;

  background: url("Photo.png") no-repeat top center / cover;

  position: relative;

}

h1{

   top:20px;

  left:420px;

  color:darkorange;

  font-size: 30px;

  font-style: italic;

  position: relative;

}

.Profile span{

  top:20px;

  left:420px;

  font-size:20px;

  position: relative;

}

h1:hover{

  color:mediumseagreen;

}

🡪 Edit the **user-profile.component.ts** file

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

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

import { User } from '../user';

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

import { param } from 'jquery';

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

@Component({

  selector: 'app-user-profile',

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

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

})

export class UserProfileComponent implements OnInit {

  Id: number;

  user: any;

  users!: Observable <User[]>;

  userlist:any;

  isupdated =false;

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

  ngOnInit() {

    this.isupdated = false;

    this.route.params.subscribe(

      (params: Params) => {

        this.Id = +params["Id"];

        console.log(this.Id);

        this.userservice.getUser(this.Id).subscribe(

          data => {

            this.user = data;

            console.log(this.user);

          }

        )

      }

    )

  }

  updatUser(id: number) {

    this.userservice.getUser(id)

      .subscribe(

        (data) => {

          this.userlist = data;

          console.log(this.userlist);

        }),

        (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()

  });

  updateUser(updateusers: any) {

    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.status = this.Status.value;

    this.user.isOnline = this.IsOnline.value;

    this.user.enabled = this.Enabled.value;

    console.log(this.FirstName.value);

    this.userservice.updateUser(this.user.userId, this.user).subscribe(

      data => {

        this.isupdated = true;

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

          this.users = data

        })

      },

        error => {

          console.log(this.users);

          console.log(error)});

  }

  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');

  }

  get UserId() {

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

  }

  changeisUpdate() {

    this.isupdated = false;

  }

}

🡪 Edit the **user-profile.component.html** file

<div class="container d-flex justify-content-center mt-5">

    <div class="card">

        <div class="top-container"> <img src="https://img.icons8.com/bubbles/100/000000/user.png" class="img-fluid profile-image" width="70">

            <div class="ml-3">

                <h5 class="name">{{user.firstName+" "+user.lastName}}</h5>

                <p class="mail">{{user.email}}</p>

            </div>

        </div>

         <div class="recent-border mt-4"> Username:-<span class="recent-orders">{{user.username}}</span> &nbsp; &nbsp; Role:- <span class="wishlist">{{user.role}}</span>

            &nbsp; &nbsp; Status:- <span class="fashion-studio">{{user.status}}</span> </div>

    </div>

    <div class="button">

    <button  (click)="updatUser(user.userId)" class='btn btn-info' data-toggle="modal"

    data-target="#myModal">Update Profile</button>

  </div>

</div>

<div class="modal" id="myModal">

  <div class="modal-dialog">

    <div class="modal-content">

      <form [formGroup]="userupdateform" #updateusers (ngSubmit)="updateUser(updateusers)">

        <!-- Modal Header -->

        <div class="modal-header">

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

        </div>

        <!-- Modal body -->

        <div class="modal-body" \*ngIf="userlist">

          <div [hidden]="isupdated">

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

            <div class="form-group">

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

            </div>

            <div class="form-group">

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

            </div>

            <div class="form-group">

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

            </div>

            <div class="form-group">

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

            </div>

            <div class="form-group">

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

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

            </div>

            <div class="form-group">

              <select   class="form-control" formControlName="role" required>

                <option value="{{userlist.role}}" [selected]="'{{userlist.role}}' == userlist.role">{{userlist.role}}</option>

              </select>

            </div>

            <div class="form-group">

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

            </div>

            <div class="form-group">

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

            </div>

            <div class="form-group">

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

            </div>

          </div>

          <div [hidden]="!isupdated">

            <h4>User Detail Updated!</h4>

          </div>

        </div>

        <!-- Modal footer -->

        <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>

🡪 Edit the **user-profile.component.scss** file

body {

    background-color: #eee;

}

h5:hover{

    color:blueviolet;

}

p:hover{

   color:black;

   font-size: medium;

}

.card {

    top:-150px;

    position: relative;

    left:200px;

    background-color: #fff;

    width: 500px;

    border-radius: 33px;

    box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);

    padding: 2rem !important

}

.top-container {

    display: flex;

    align-items: center

}

.profile-image {

    border-radius: 10px;

    border: 2px solid #5957f9

}

.name {

    font-size: 15px;

    font-weight: bold;

    color: #272727;

    position: relative;

    top: 8px

}

.mail {

    font-size: 14px;

    color: grey;

    position: relative;

    top: 2px

}

.middle-container {

    background-color: #eee;

    border-radius: 12px

}

.middle-container:hover {

    border: 1px solid #5957f9

}

.dollar-div {

    background-color: #5957f9;

    padding: 12px;

    border-radius: 10px

}

.round-div {

    border-radius: 50%;

    width: 35px;

    height: 35px;

    background-color: #fff;

    display: flex;

    justify-content: center;

    align-items: center;

    text-align: center

}

.dollar {

    font-size: 16px !important;

    color: #5957f9 !important;

    font-weight: bold !important

}

.current-balance {

    font-size: 15px;

    color: #272727;

    font-weight: bold

}

.amount {

    color: black;

    font-size: 16px;

    font-weight: bold

}

.dollar-sign {

    font-size: 16px;

    color: #272727;

    font-weight: bold

}

.recent-border {

    border-left: 2px solid #5957f9;

    display: flex;

    align-items: center

}

.recent-border:hover {

    border-bottom: 1px solid #dee2e6 !important

}

.recent-orders {

    font-size: 16px;

    font-weight: 700;

    color: Black;

    margin-left: 2px

}

.wishlist {

    font-size: 16px;

    font-weight: 700;

    color: #272727

}

.wishlist-border:hover {

    border-bottom: 1px solid #dee2e6 !important

}

.fashion-studio {

    font-size: 16px;

    font-weight: 700;

    color: #272727

}

.fashion-studio-border:hover {

    border-bottom: 1px solid #dee2e6 !important

}

.recent-border span:nth-child(1):hover{

    color:blue;

}

.recent-border span:nth-child(2):hover{

    color:blue;

}

.recent-border span:nth-child(3):hover{

    color:blue;

}

.button button{

      position: relative;

      left:505px;

      top:-145px;

  }

🡪 Edit the **active-user.component.ts** file

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

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

import { User } from '../user';

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

@Component({

  selector: 'app-active-user',

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

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

})

export class ActiveUserComponent implements OnInit {

  dtOptions: DataTables.Settings={};

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

  constructor(private userservice: UserService) { }

  users!: Observable <User[]>;

  user: User =new User();

  deactiveList: any;

  isEnabled= false;

  ngOnInit(){

   this.isEnabled= false;

   this.dtOptions={

     pageLength: 6,

     stateSave: true,

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

     processing: true

   };

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

     this.users =data;

     this.dtTrigger.next();

   })

  }

  enabledUser(id: number){

    this.userservice.activeUser(id).

    subscribe(

      (data) =>{

        this.deactiveList =data;

        console.log(this.deactiveList);

        this.activeUser(id);

      },

      (error) => console.log(error)

    );

  }

  activeUserForm = new FormGroup({

    userId: new FormControl()

  });

  activeUser(id){

    this.user =new User();

    this.user.userId=id;

    this.userservice.activeUser(this.user.userId).subscribe(

    data => {

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

        this.users =data

        console.log(this.users)

      } )

    },

    error => console.log(error));

  }

  get UserId() {

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

  }

}

🡪 Edit the **active-user.component.html** file

<h1 style="font-weight: bold; font-style: italic; text-align: center; ">Users List</h1><br>

<div class="panel">

    <div class="panel panel-default">

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

            <div class="col-sm-4">

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

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

            <h2 style="text-align: center; font-weight: bold;">User Actived</h2>

          </div>

            </div>

        </div>

            <div class="panel-body">

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

                <thead class="thead-light">

                    <tr>

                        <th>UserID</th>

                        <th>Name</th>

                        <th>UserName</th>

                        <th>Enabled</th>

                    </tr>

                </thead>

                <tbody>

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

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

                        <td>{{user.firstName +' '+ user.lastName}}</td>

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

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

                        <td> <button (click)="enabledUser(user.userId)" class='btn btn-info' data-toggle="modal"

                            data-target="#myModal"><i class="fa fa-futboll-0">Active</i></button> &nbsp; &nbsp;</td>

                    </tr>

                </tbody>

                </table>

            </div>

    </div>

</div>

🡪 Edit the **active-user.component.scss** file

.panel{

    position:relative;

    left:95px;

    background-color:lightskyblue;

    }

td {

     border-right: solid 0.3px  rgb(10, 10, 10);

     border-bottom:solid 0.3px black;

      }

th{

     border-top:solid 0.5px black;

     border-bottom:solid 0.5px black; ;

      }

.panel-body th:hover{

     color:white;

}

.panel-body tr:hover{

     font-weight: bold;

     color:blue;

  }

  .panel button:hover{

    font-weight: bolder;

    color:black;

  }

  .panel-body td:nth-child(1n+1):hover{

    color:black;

  }

🡪 Edit the **blog-list.component.ts** file

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

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

import { Blog } from '../blog';

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

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

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

@Component({

  selector: 'app-blog-list',

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

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

})

export class BlogListComponent implements OnInit {

  Id: number;

  user: any;

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

  blog: Blog =new Blog();

  submitted = false;

  ngOnInit() {

    this.route.params.subscribe(

      (params: Params) => {

        this.Id = +params["Id"];

        console.log(this.Id);

        this.userservice.getUser(this.Id).subscribe(

          data => {

            this.user = data;

            console.log(this.user);

          }

        )

      }

    )

    this.submitted = false;

  }

  blogpostform= new FormGroup(

    {

      blogTitel: new FormControl(''),

      blogContent:new FormControl('')

    });

    blogpost(blogpost:any){

      this.blog =new Blog();

      this.blog.userId=this.user.userId;

      this.blog.username=this.user.username;

      this.blog.blogContent=this.BlogContent.value;

      this.blog.blogTitel=this.BlogTitel.value;

    if(this.user.role=="Admin"){

      this.blog.status="Posted";

    }

    else{

      this.blog.status="Waiting For Apporved";

    }

    this.submitted=true;

      console.log(this.blog.blogContent);

      this.save();

    }

    save(){

      this.blogservice.createBlog(this.blog)

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

      this.blog =new Blog();

    }

    get BlogContent(){

      return this.blogpostform.get('blogContent');

    }

    get BlogTitel(){

      return this.blogpostform.get('blogTitel');

    }

    blogPostForm(){

      this.submitted=false;

      this.blogpostform.reset();

    }

}

🡪 Edit the **blog-list.component.html** file

<div class="main">

<div class="container mt-5">

    <div class="d-flex justify-content-center row">

        <div class="col-md-8">

            <div class="d-flex flex-column comment-section">

                    <div class="d-flex flex-row user-info">

                          <div class="d-flex flex-column justify-content-start ml-2"><span class="d-block font-weight-bold name"><strong>{{user.firstName+" "+user.lastName}}</strong></span></div>

                    </div>

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

                    <form [formGroup]="blogpostform" (ngSubmit)="blogpost(blogpost)">

                    <div class="form-group">

                        <label for ="blogTitel">BlogTitel</label>

                        <input type="text" class="form-control" formControlName="blogTitel" data-toggle="tooltip"

                        data-placement="right" titel="Enter Blog Titel" placeholder="Enter Blog Titel">

                    </div>

                       <div class="form-group">

                        <textarea class="form-control" formControlName="blogContent" data-toggle="tooltip"

                        data-placement="right" titel="Enter  Blog Content" placeholder="Enter Blog Content"></textarea>

                    </div>

                        <button type="submit" class="btn btn-primary btn-sm shadow-none" >Post comment</button>

                    <button class="btn btn-outline-primary btn-sm ml-1 shadow-none" type="button">Cancel</button>

            </form>

        </div>

            </div>

            <div [hidden]="!submitted">

                <h4 style="font-weight: bold; font-style: italic; text-align: center; color:white;">Comment Blog Posted</h4>

            </div>

        </div>

    </div>

</div>

</div>

🡪 Edit the **blog-list.component.scss** file

body {

    background: #eee

}

.main{

    position:relative;

    left:100px;

    background-color:lightskyblue;

    }

.date {

    font-size: 11px

}

.comment-text {

    font-size: 12px

}

.fs-12 {

    font-size: 12px

}

.shadow-none {

    box-shadow: none

}

.name {

    color: #007bff

}

.cursor:hover {

    color: blue

}

.cursor {

    cursor: pointer

}

.textarea {

    resize: none

}

🡪 Edit the **blog-lists.component.ts** file

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

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

import {Blog} from '../blog';

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

@Component({

  selector: 'app-bloglists',

  templateUrl: './bloglists.component.html',

  styleUrls: ['./bloglists.component.scss']

})

export class BloglistsComponent implements OnInit {

  constructor(private blogservice: BlogService) { }

   blogArray:any[]=[];

   dtOptions: DataTables.Settings = {};

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

   blogs!: Observable <Blog[]>;

   blog: Blog = new Blog();

   deleteBlogMessage = false;

   bloglist: any;

   isupdated = false;

  ngOnInit() {

    this.isupdated = false;

    this.dtOptions = {

      pageLength: 6,

      stateSave: true,

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

      processing: true

    };

    this.blogservice.getBlogList().subscribe((data)=>

    {

      this.blogs = data;

      this.dtTrigger.next();

    })

  }

  deleteBlog(blogId: number){

    this.blogservice.deleteBlog(blogId)

    .subscribe(

      (data) =>{

        console.log(data);

        this.deleteBlogMessage = true;

        this.blogservice.getBlogList()

        .subscribe((data) =>

        {

          this.blog= data

        })

      },

      (error) => console.log(error));

  }

      updatBlog(id: number){

        this.blogservice.getBlog(id)

        .subscribe(

          (data) => {

            this.bloglist =data;

            console.log(this.bloglist);

          }

        ),

        (error) => console.log(error);

      }

      blogUpdateform = new FormGroup({

        blogId : new FormControl(),

        blogTitel: new FormControl(),

        blogContent: new FormControl(),

        status: new FormControl()

      });

      updateBlog(){

        this.blog = new Blog();

        this.blog.blogId= this.BlogId.value;

        this.blog.blogTitel= this.BlogTitel.value;

        this.blog.blogContent=this.BlogContent.value;

        this.blog.status=this.Status.value;

        console.log(this.BlogTitel.value);

        this.blogservice.updateBlog(this.blog.blogId, this.blog).subscribe(

          data => {

            this.isupdated =true;

            this.blogservice.getBlogList().subscribe(data => {

              this.blogs = data

            })

          },

          error => {

            console.log(this.blogs);

            console.log(error)});

      }

      get BlogId(){

        return this.blogUpdateform.get('blogId');

      }

      get BlogTitel(){

        return this.blogUpdateform.get('blogTitel');

      }

      get BlogContent(){

        return this.blogUpdateform.get('blogContent');

      }

      get Status(){

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

      }

      changeisUpdate(){

        this.isupdated = false;

      }

  }

🡪 Edit the **blog-lists.component.html** file

<div class="panel">

    <div class="panel panel-default">

        <h1 style="font-weight: bold; font-style: italic; text-align: center; ">Blog Lists</h1><br>

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

          <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">×</button>

              <strong>Blog has Been Deleted</strong>

            </div>

          </div>

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

        </div>

      </div>

      <div class="main">

  <div class="panel-body" \*ngFor="let blog of blogs"> {{blog.blogId}}.

        <div class="bg-white p-2">

             <div class="d-flex flex-row user-info"> <!--<img class="rounded-circle" src="https://i.imgur.com/RpzrMR2.jpg" width="40"> -->

                <div class="name"><span class="d-block font-weight-bold name">{{blog.username}}</span> &nbsp;-- <span>{{blog.blogPosted}}</span> &nbsp;<span class="date">{{blog.status}}</span></div>

            </div>

            <div class="mt-2">

                <p class="comment-text">{{blog.blogContent}}</p>

            </div>

        </div>

        <div class="bg-white">

            <div class="d-flex2 ">

                <i class="fa fa-thumbs-o-up"></i><span class="ml-1">Like</span> &nbsp; &nbsp;

                <i class="fa fa-commenting-o"></i><span class="ml-1">Comment</span>

            </div>

        </div>

        <br/>

        <div class="button">

        <button (click)="deleteBlog(blog.blogId)" class='btn btn-primary'><i class="fa fa-futboll-0">Delete</i></button> &nbsp; &nbsp;

             <button (click)="updatBlog(blog.blogId)" class='btn btn-info' data-toggle="modal"

                data-target="#myModal">Update</button>

              </div>

  </div>

</div>

    </div>

    <div class="modal" id="myModal">

      <div class="modal-dialog">

        <form [formGroup]="blogUpdateform"  (ngSubmit)="updateBlog()">

           <!-- Modal Header -->

        <div class="modal-header">

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

        </div>

        <div class="modal-body" \*ngIf="bloglist">

          <div [hidden]="isupdated">

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

            <div class="form-group">

              <label for="name">Blog Titel</label>

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

            </div>

            <div class="form-group">

              <label for="name">Blog Content</label>

              <textarea type="text" class="form-control" formControlName="blogContent" [(ngModel)]="bloglist.blogContent"></textarea>

            </div>

            <div class="form-group">

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

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

            </div>

          </div>

          <div [hidden]="!isupdated">

            <h4>Blog 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>

🡪 Edit the **blog-lists.component.scss** file

body {

    background: #eee;

}

.main{

    position:relative;

    left:200px;

    width: 500px;

    }

.bg-white span:nth-child(2){

   font-size: 12px;

}

.bg-white:nth-child(2){

    border-top: solid 1px black;

    background-color:lightskyblue;

}

.name span:nth-child(1){

color:blue;

}

.name span:nth-child(2){

    color:skyblue;

    }

    .name span:nth-child(3){

        color:green;

        font-size: 14px;

        }

.date {

    font-size: 11px;

}

.d-flex span:nth-child(1){

    font-size:medium;

    color:blue;

}

.comment-text {

    border:2px solid white;

    border-width: 20px;

    font-size: 15px;

}

.mt-2{

    border-top: solid 1px black;

    border-right: solid 1px black;;

     border-left:solid 1px black;

    position: relative;

    background-color:white;

}

.button button:nth-child(1):hover{

color:black;

}

.button button:nth-child(2):hover{

    color:darkblue;

    }

🡪 Edit the **my-blog.component.ts** file

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

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

import {Blog} from '../blog';

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

import { param } from 'jquery';

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

@Component({

  selector: 'app-my-blog',

  templateUrl: './my-blog.component.html',

  styleUrls: ['./my-blog.component.scss']

})

export class MyBlogComponent implements OnInit {

  Id:number;

  blog: any;

  constructor(private route: ActivatedRoute, private blogservice: BlogService) { }

  ngOnInit(): void {

     this.route.params.subscribe(

       (params: Params) =>{

         this.Id =+params["Id"];

         console.log(this.Id);

         this.blogservice.getUserBlog(this.Id).subscribe(

           data =>{

             this.blog = data;

             console.log(this.blog);

           }

         )

       }

     )

  }

}

🡪 Edit the **my-blog.component.html** file

<div class="panel">

    <div class="panel panel-default">

        <div class="panel-body" \*ngFor="let blogs of blog" > {{blogs.blogId}}.

            <div class="bg-white p-2">

                 <div class="d-flex flex-row user-info"> <!--<img class="rounded-circle" src="https://i.imgur.com/RpzrMR2.jpg" width="40"> -->

                    <div class="name"><span class="d-block font-weight-bold name">{{blogs.username}}</span> &nbsp;--

                         <div \*ngFor="let date of blogs.blogPosted"> <span>{{date.time}}</span> </div>&nbsp;<span class="date">{{blogs.status}}</span></div>

                </div>

                <div class="mt-2">

                    <p class="comment-text">{{blogs.blogContent}}</p>

                </div>

            </div>

            <div class="border"></div>

      </div>

    </div>

</div>

🡪 Edit the **my-blog.component.scss** file

body {

    background: #eee;

}

.panel{

    position:relative;

    left:100px;

    width: 500px;

    }

.bg-white span:nth-child(2){

   font-size: 12px;

}

.bg-white:nth-child(2){

    border-top: solid 1px black;

    background-color:lightskyblue;

}

.name span:nth-child(1){

color:blue;

}

.name span:nth-child(2){

    color:skyblue;

    }

    .name span:nth-child(3){

        color:green;

        font-size: 14px;

        }

.date {

    font-size: 11px;

}

.d-flex span:nth-child(1){

    font-size:medium;

    color:blue;

}

.comment-text {

    border:2px solid white;

    border-width: 20px;

    font-size: 15px;

}

.mt-2{

    border-top: solid 1px black;

    border-left:solid 1px black;

    border-right: solid 1px black;;

    position: relative;

    background-color:white;

}

.button button:hover{

    color:darkred;

}

.border{

    border-top:1px solid black;

}

🡪 Edit the **active-blog.component.ts** file

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

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

import {Blog} from '../blog';

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

import { User } from '../user';

@Component({

  selector: 'app-active-blog',

  templateUrl: './active-blog.component.html',

  styleUrls: ['./active-blog.component.scss']

})

export class ActiveBlogComponent implements OnInit {

  dtOptions: DataTables.Settings={};

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

  constructor(private blogservice: BlogService) { }

  blogs!: Observable <Blog[]>;

  blog: Blog =new Blog();

  deactiveBlogList:any;

  isPosted= false;

  ngOnInit(): void {

    this.isPosted=false;

    this.dtOptions={

      pageLength: 6,

      stateSave: true,

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

      processing: true

    };

    this.blogservice.deactiveBlogList().subscribe(data =>{

      this.blogs =data;

      this.dtTrigger.next();

    })

  }

  postBlog(id: number){

    this.blogservice.activeBlog(id).subscribe(

      (data) =>{

        this.deactiveBlogList =data;

        console.log(this.deactiveBlogList);

         this.postingBlog(id);

      },

      (error) => console.log(error)

    );

  }

  postBlogForm =new FormGroup({

    blogId: new FormControl()

  });

  postingBlog(id){

    this.blog = new Blog();

    this.blog.blogId=id;

    this.blogservice.activeBlog(this.blog.blogId).subscribe(

      data => {

        this.blogservice.deactiveBlogList().subscribe(

          data =>{

            this.blogs =data

            console.log(this.blogs)

          }

        )

      },

      error =>console.log(error)

    );

  }

  get BlogId(){

    return this.postBlogForm.get('blogId');

  }

}

🡪 Edit the **active-blog.component.html** file

<h1 style="font-weight: bold; font-style: italic; text-align: center; ">Waiting Blog List</h1><br>

<div class="panel">

    <div class="panel panel-default">

        <div class="blog" [hidden]="!isPosted">

            <div class="cols-sm-4">

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

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

                    <h2 style="text-align: center; font-weight: bold;">Blog Posted</h2>

                  </div>

            </div>

        </div>

        <div class="panel-body" \*ngFor="let blog of blogs"> {{blog.blogId}}.

            <div class="bg-white p-2">

                 <div class="d-flex flex-row user-info"> <!--<img class="rounded-circle" src="https://i.imgur.com/RpzrMR2.jpg" width="40"> -->

                    <div class="name"><span class="d-block font-weight-bold name">{{blog.username}}</span> &nbsp;-- <span>{{blog.blogPosted}}</span> &nbsp;<span class="date">{{blog.status}}</span></div>

                </div>

                <div class="mt-2">

                    <p class="comment-text">{{blog.blogContent}}</p>

                </div>

            </div>

            <div class="border"></div>

            <br/>

            <div class="button">

            <button (click)="postBlog(blog.blogId)" class='btn btn-info' data-toggle="modal"

                            data-target="#myModal"><i class="fa fa-futboll-0">Active Blog</i></button>

                        </div>

      </div>

    </div>

</div>

🡪 Edit the **active-blog.component.scss** file

body {

    background: #eee;

}

.panel{

    position:relative;

    left:100px;

    width: 500px;

    }

.bg-white span:nth-child(2){

   font-size: 12px;

}

.bg-white:nth-child(2){

    border-top: solid 1px black;

    background-color:lightskyblue;

}

.name span:nth-child(1){

color:blue;

}

.name span:nth-child(2){

    color:skyblue;

    }

    .name span:nth-child(3){

        color:green;

        font-size: 14px;

        }

.date {

    font-size: 11px;

}

.d-flex span:nth-child(1){

    font-size:medium;

    color:blue;

}

.comment-text {

    border:2px solid white;

    border-width: 20px;

    font-size: 15px;

}

.mt-2{

    border-top: solid 1px black;

    border-left:solid 1px black;

    border-right: solid 1px black;;

    position: relative;

    background-color:white;

}

.button button:hover{

    color:darkred;

}

.border{

    border-top:1px solid black;

}

🡪Install following:

npm install jquery --save

npm install datatables.net --save

npm install datatables.net-dt --save

npm install angular-datatables@6.0.0 --save

npm install @types/jquery --save-dev

npm install @types/datatables.net --save-dev

    🡪 Edit angular.json file at line 30. styles and scripts:

"styles": [

"src/styles.scss",

"node\_modules/datatables.net-dt/css/jquery.dataTables.css",

"node\_modules/bootstrap/dist/css/bootstrap.css"

],

"scripts": [

"node\_modules/jquery/dist/jquery.js",

"node\_modules/datatables.net/js/jquery.dataTables.js",

"node\_modules/bootstrap/dist/js/bootstrap.js"

]

🡪Save All.

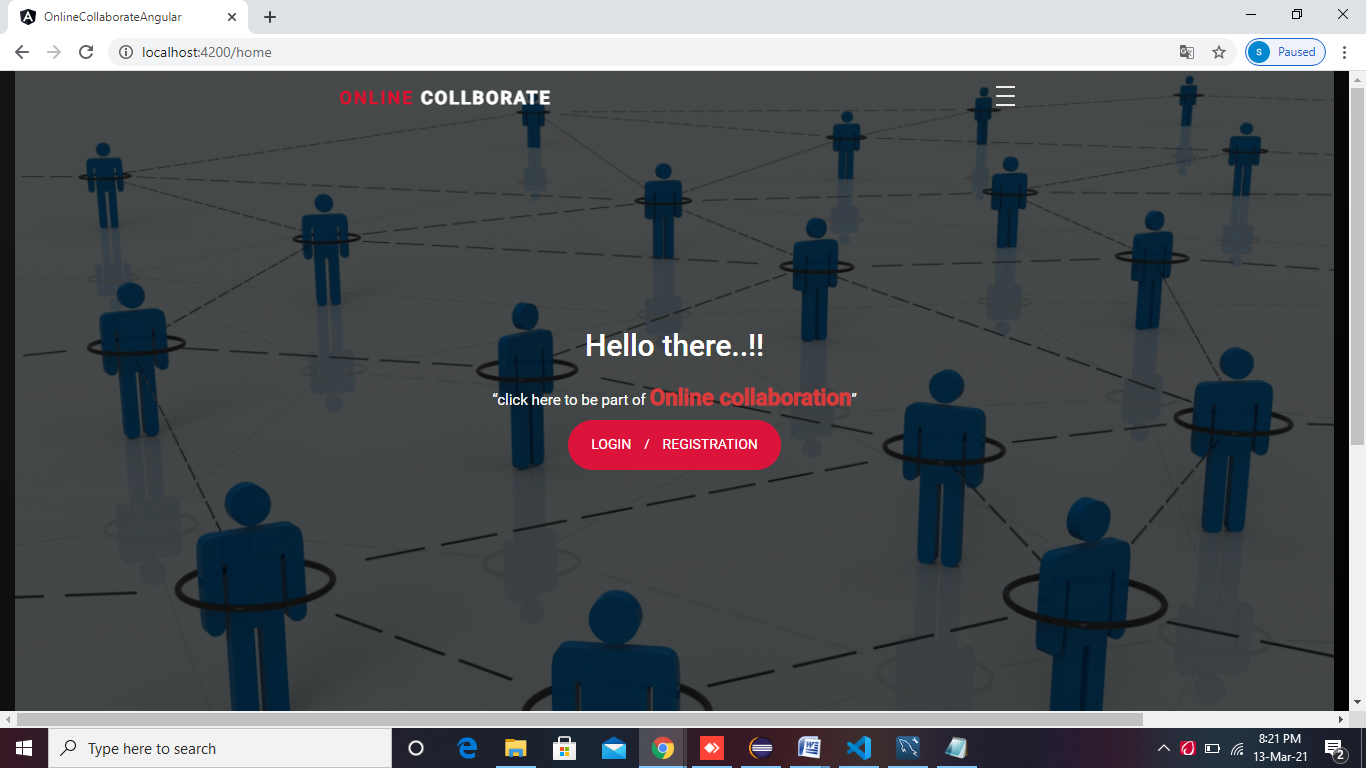
🡪Run the SpringBoot Application

🡪Run the Angular project

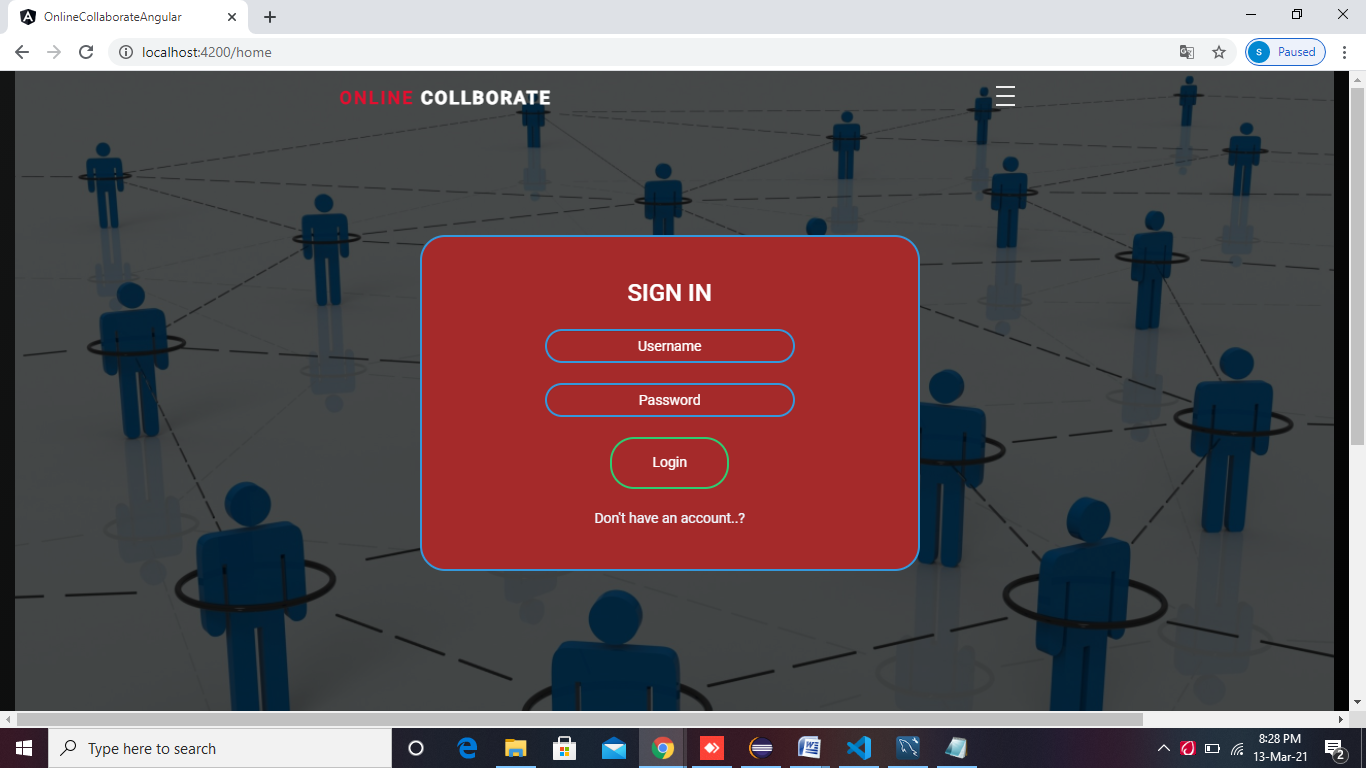
***USER INTERFACE DESIGN***

The common user interface for the entire system that includes an Admin

and User.

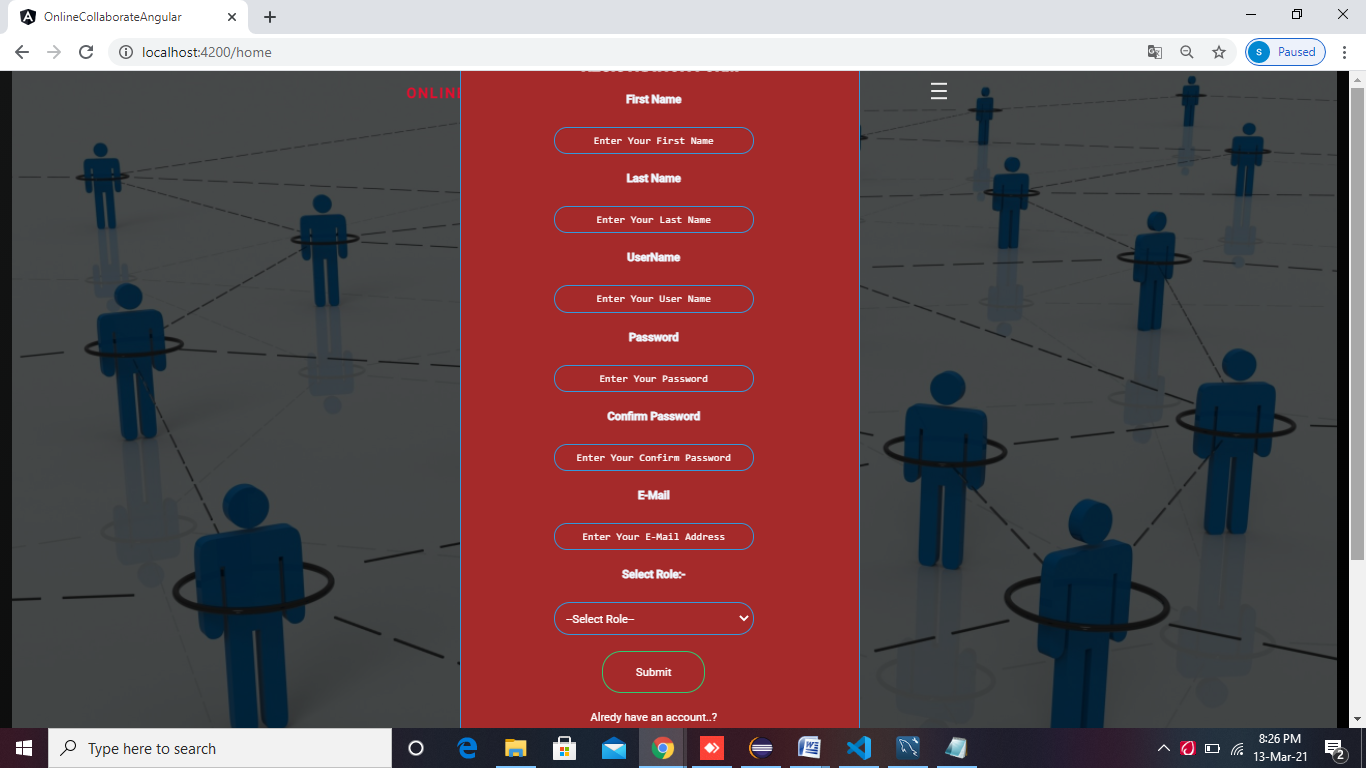


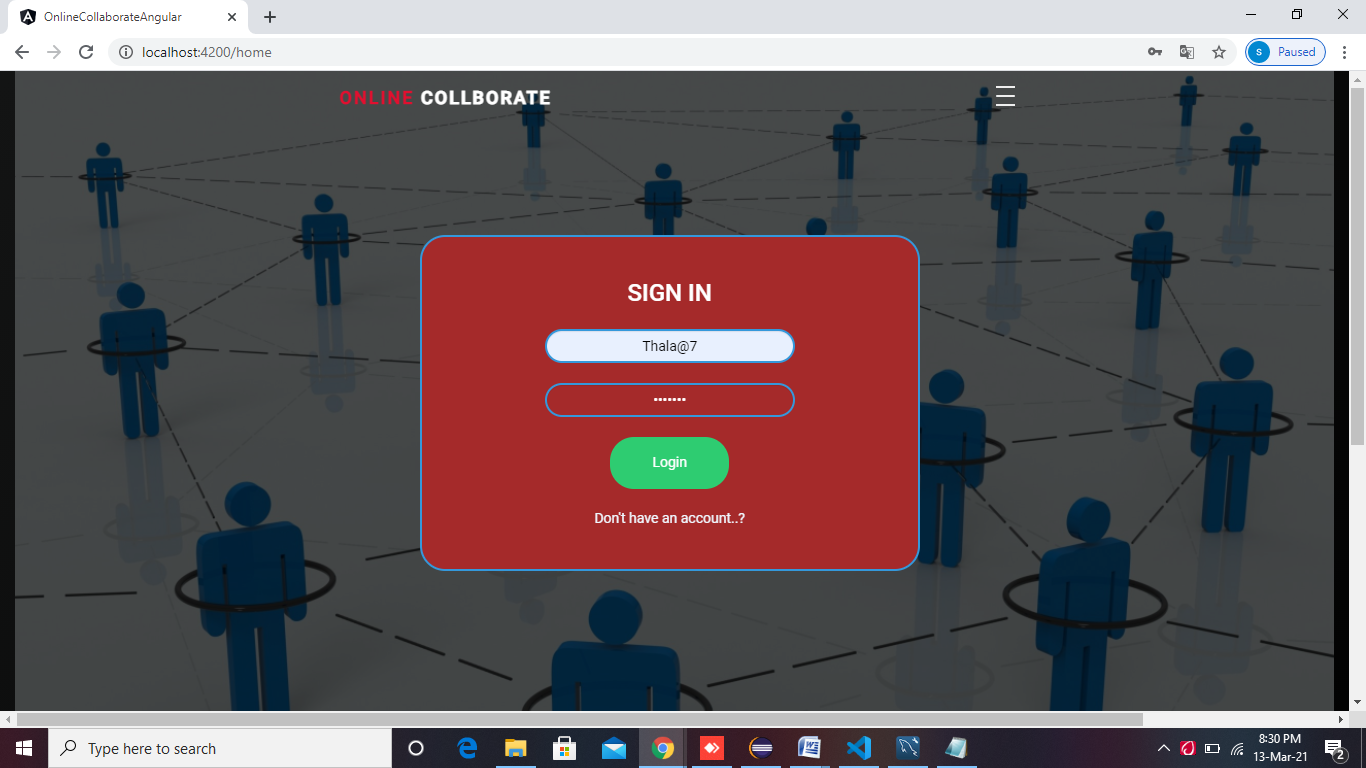
The Admin/User has to be Click “Login/Regstration” button for login or registered.



The Admin/User has to be registered in Online Collaboration Admin/User before login. This frame will connect to the database and checks the Admin/User has authorization and then also set which designation they belong to Admin Window, User Window and then will give access to the Admin/User . If Admin/User doesn’t have authorization for login then Admin/User has to click “REGISTRATION” button for Registration.

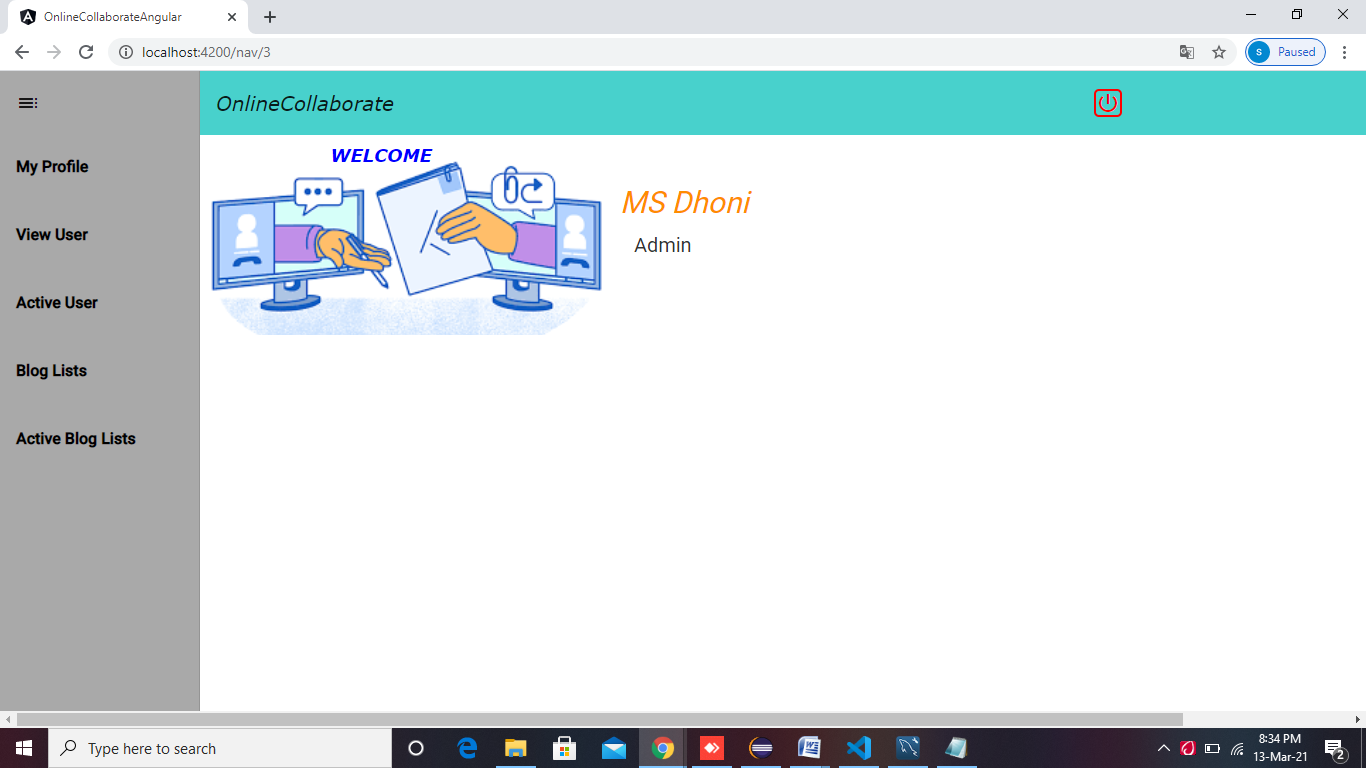
This is the Registration form that the Admin/User has to fill their details to get registered before the Online Collaboration

******

****

This is how an Admin/User can login to their portal. This is a Admin login credentials. This navigates us to Admin portal. While registering with the portal, one need to choose that the user is which an Admin/User.It will walk us through which portal we are having the access to

Admin Window Screen:-

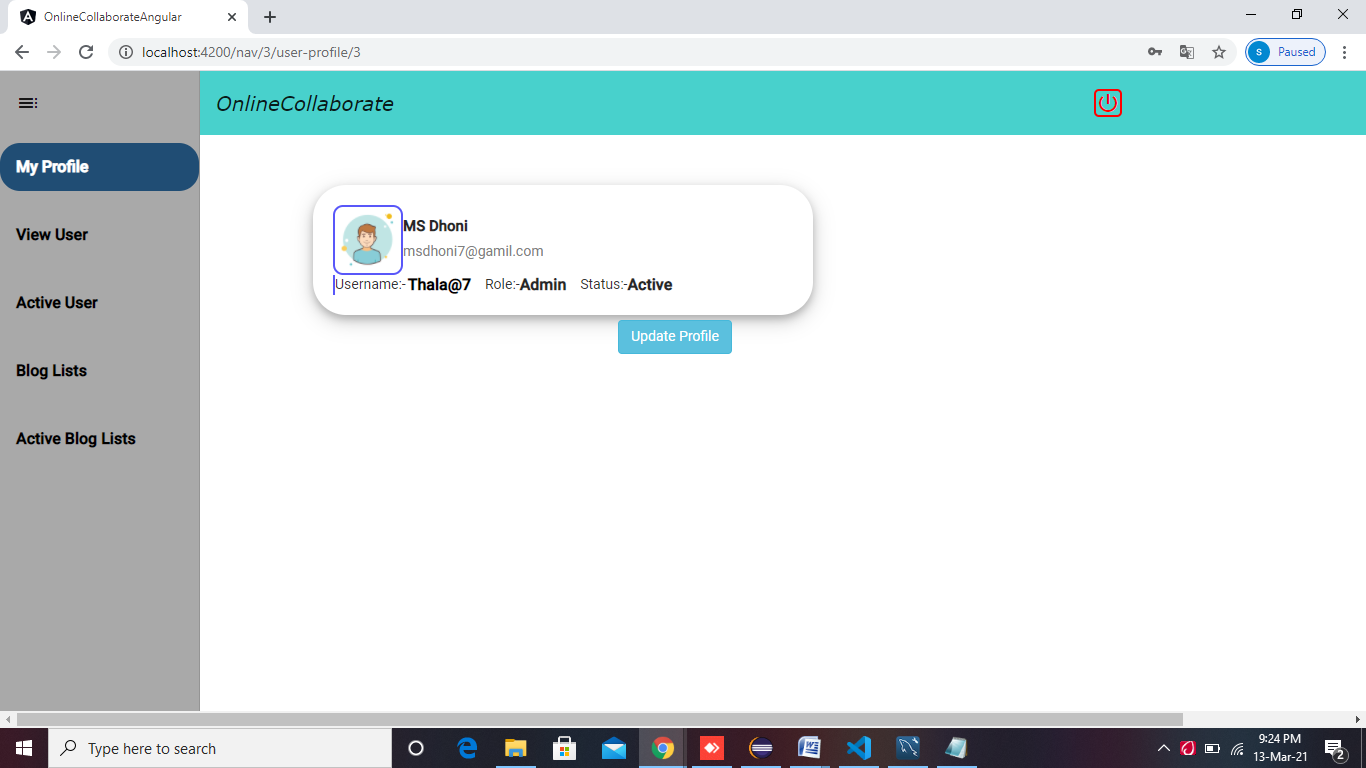
****

This is how the Admin Window looks like. Here, can see that what Admin can have the access to do.

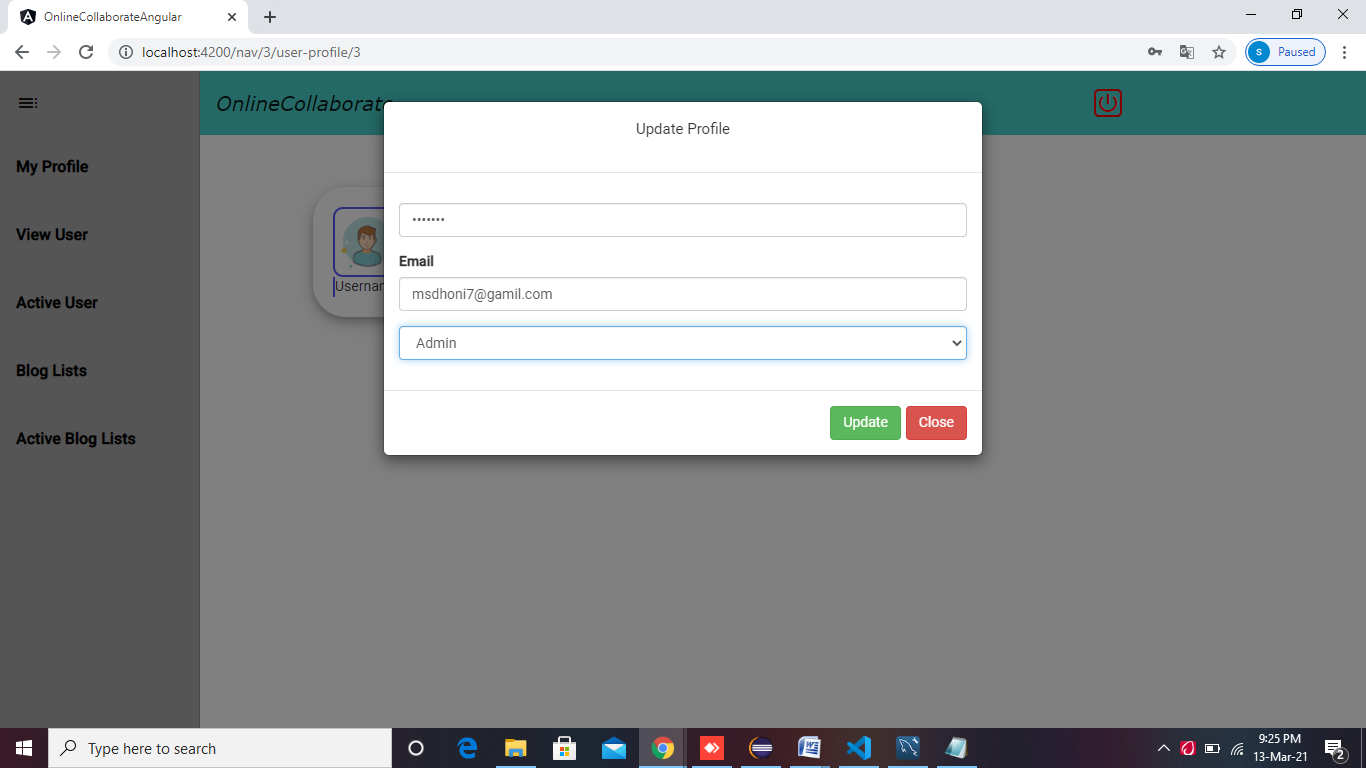
**Admin Can:**

* View Thier own profile
* Update Own Profile
* View All Activate User List
* Activating Users
* View All Deactive UserList
* View All Active Blog List
* View All Deactive Blog List
* Activating Blogs

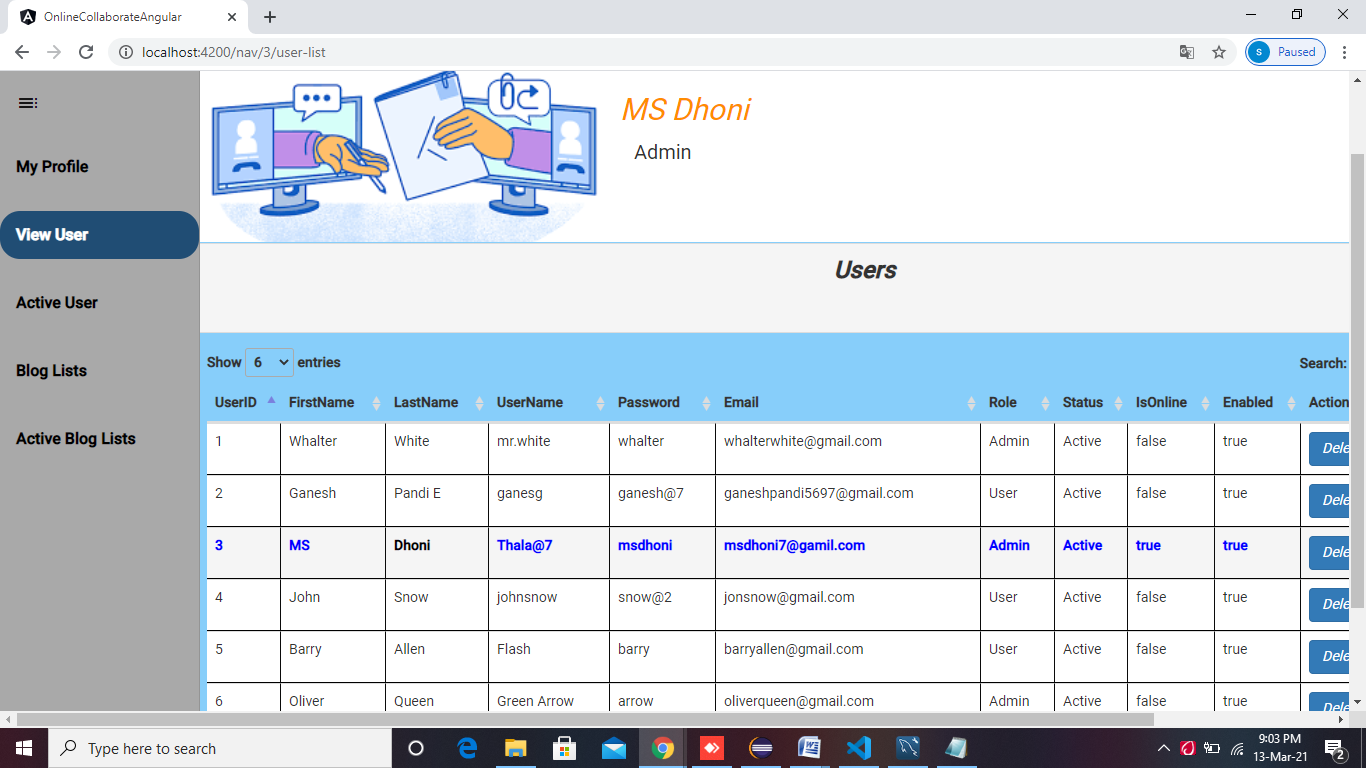
Admin Can see their own profile

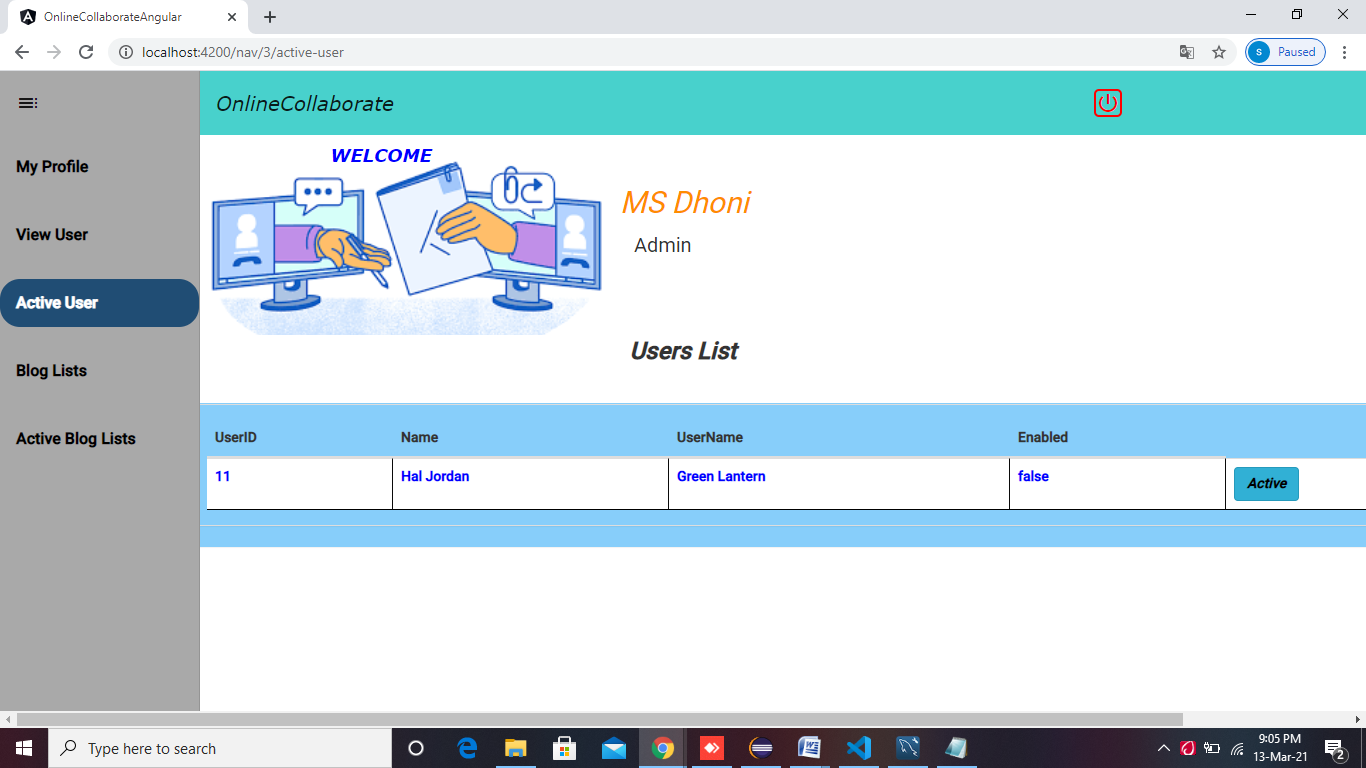


Admin Can Update their own profile details like email and password

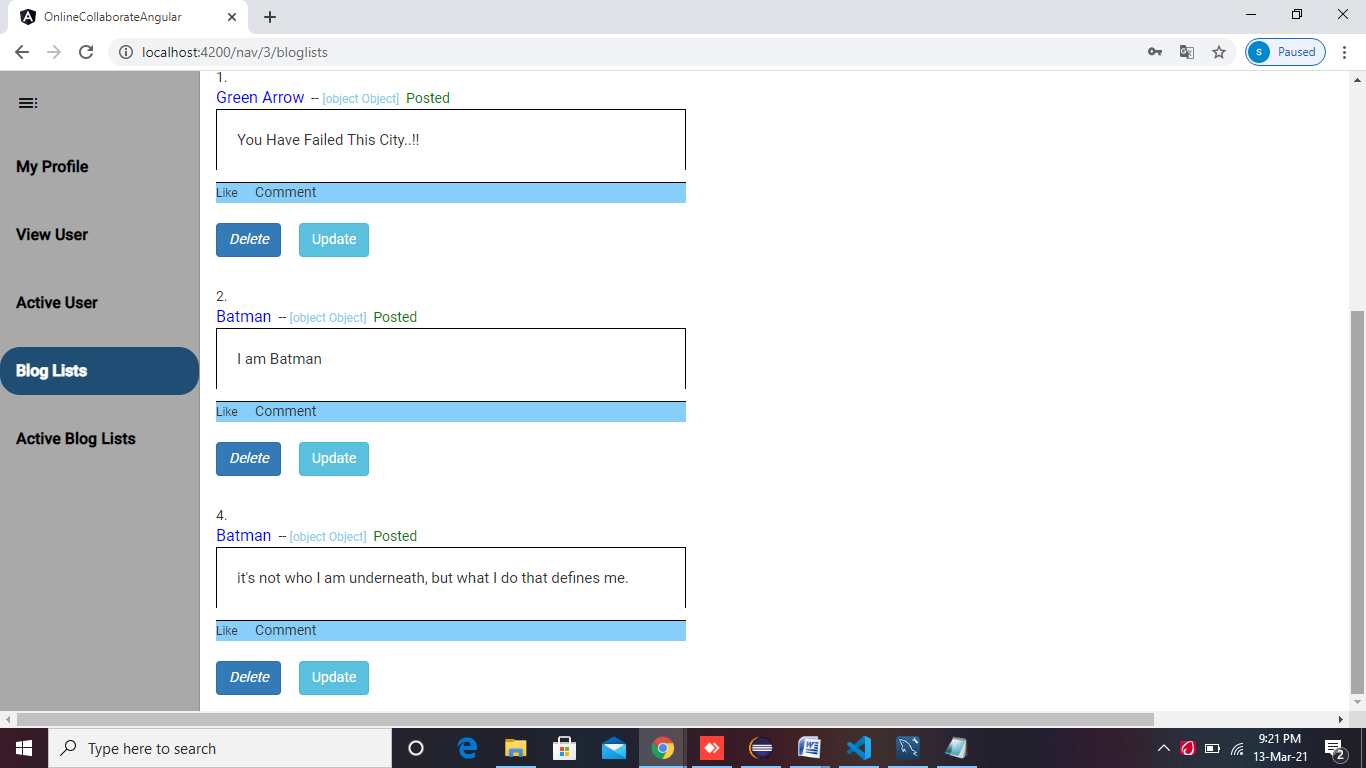


Admin Can See All Active User Lists by Enabled

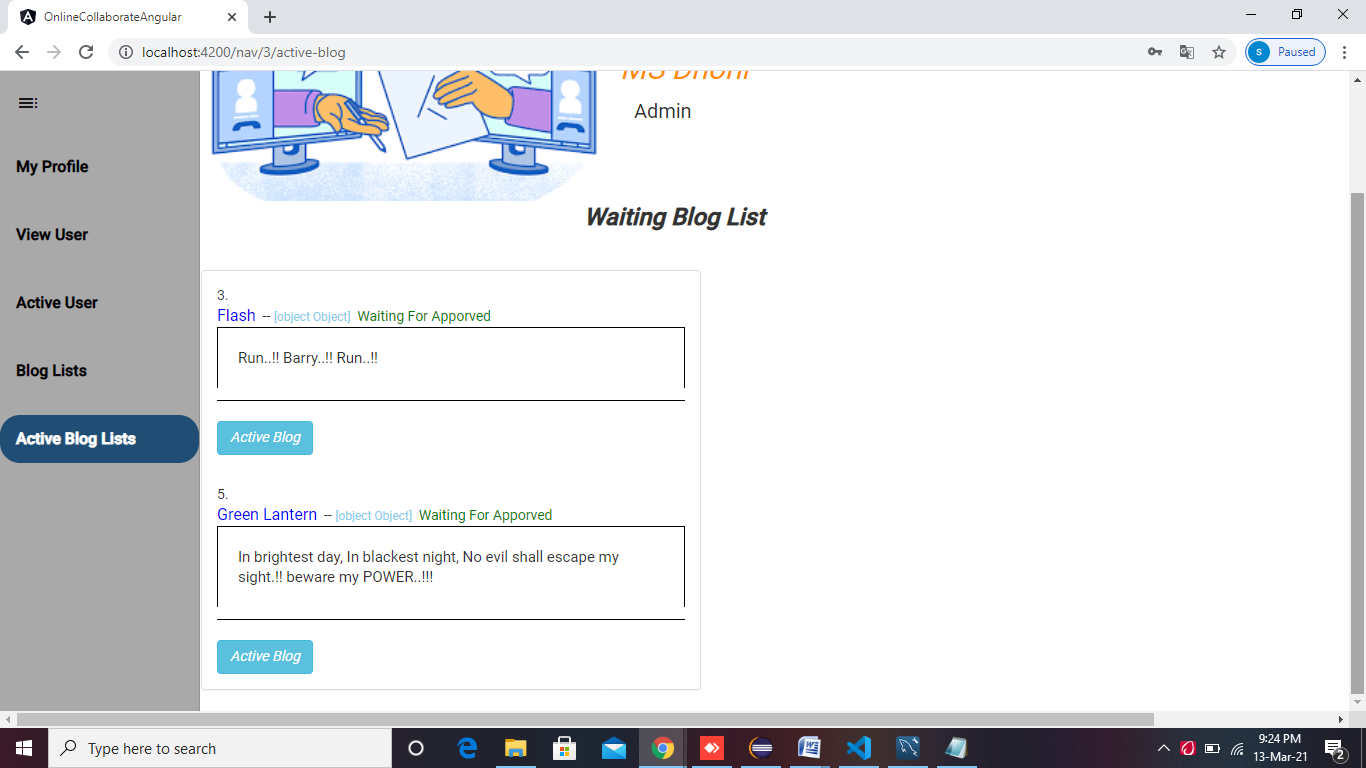


Admin Can See All Deactive User Lists and Make them active by click Active Button.

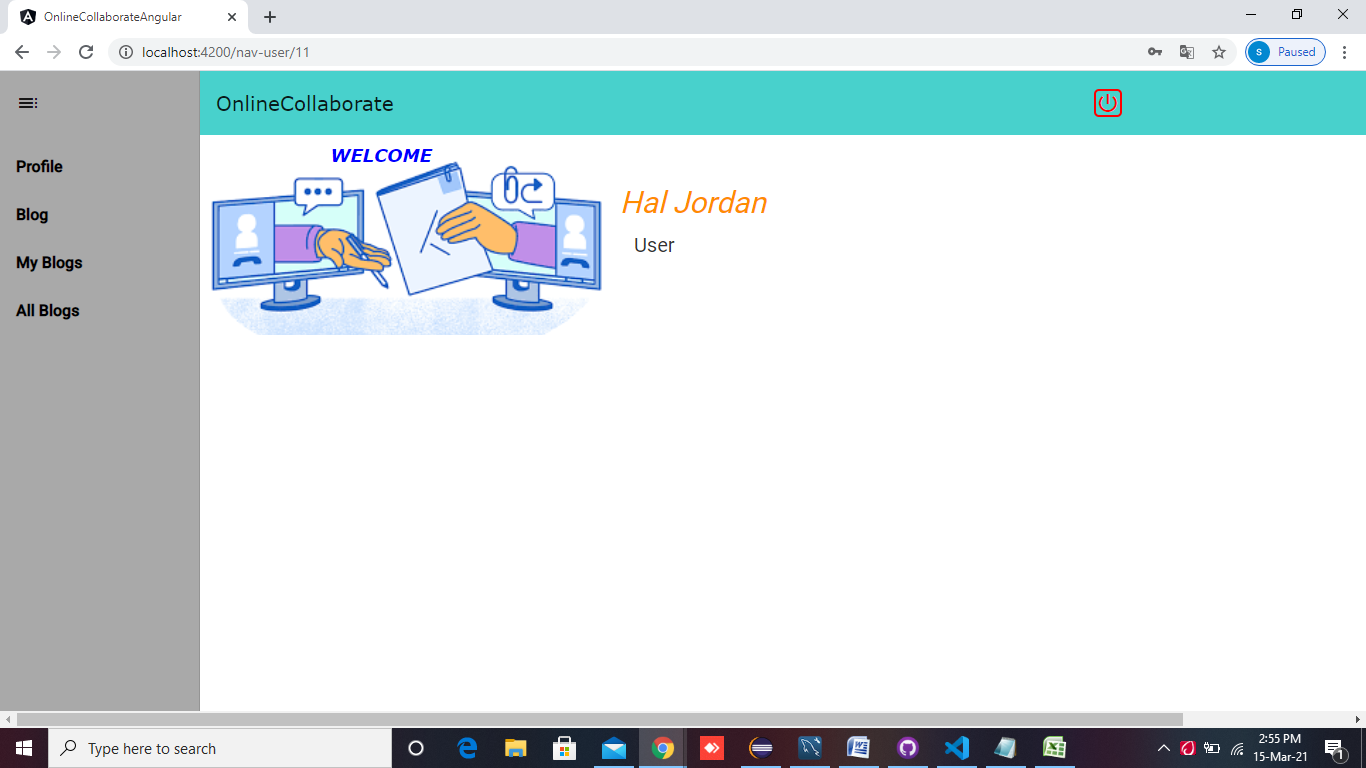
Admin can see all the posted/active blog list and delete if Admin wants



Admin Also can see deactive blog List/wating for approval blogs List



User Window screen:

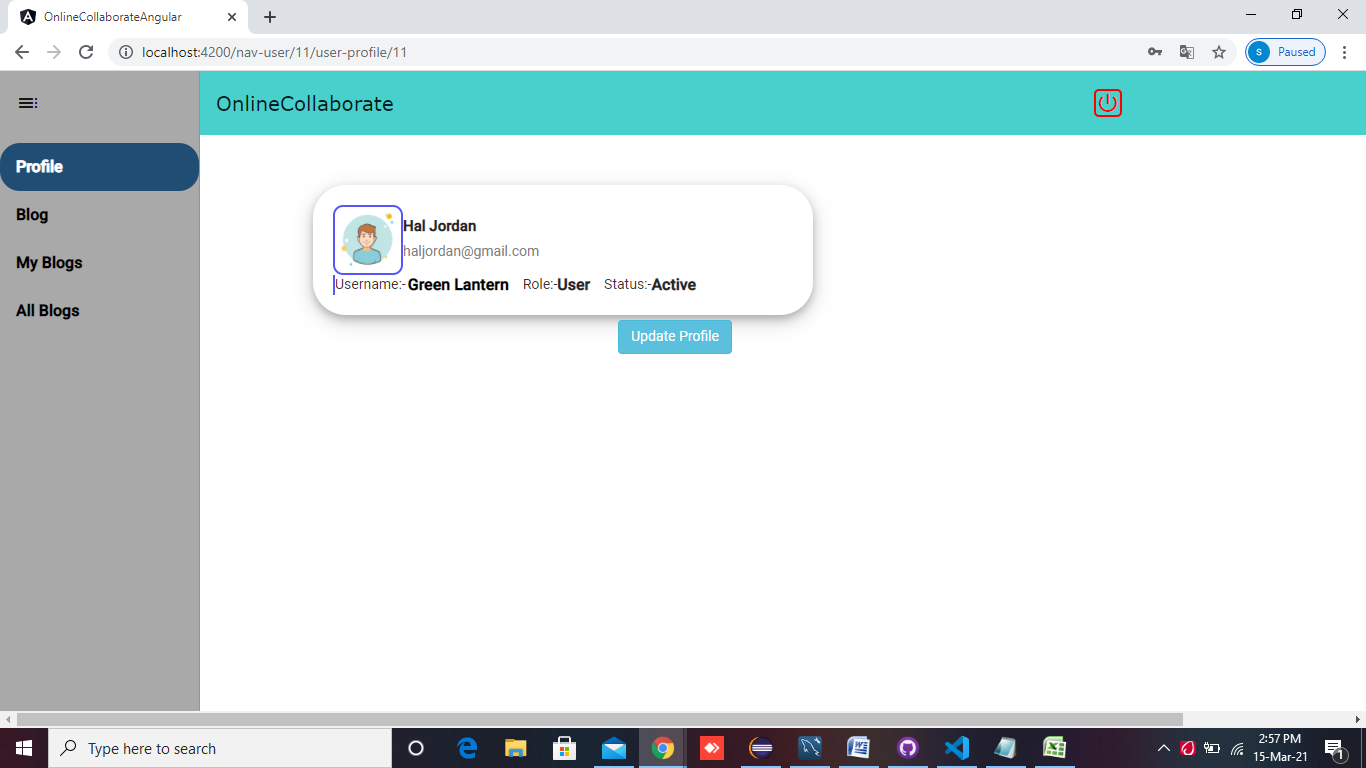


This is how the User Window looks like. Here, can see that what User can have the access to do.

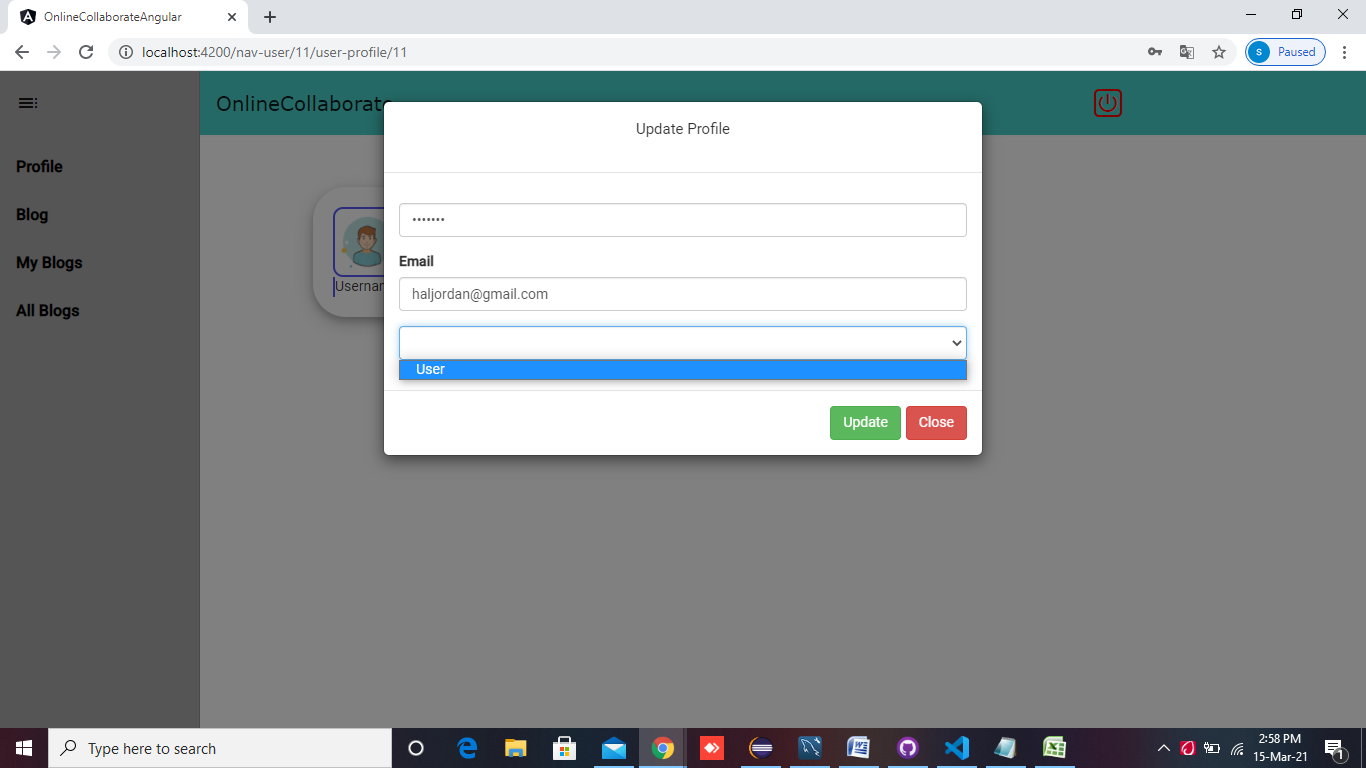
**User Can:**

* View Their own profile
* Update their own profile
* Posting Blog
* See Their all Posted Blogs

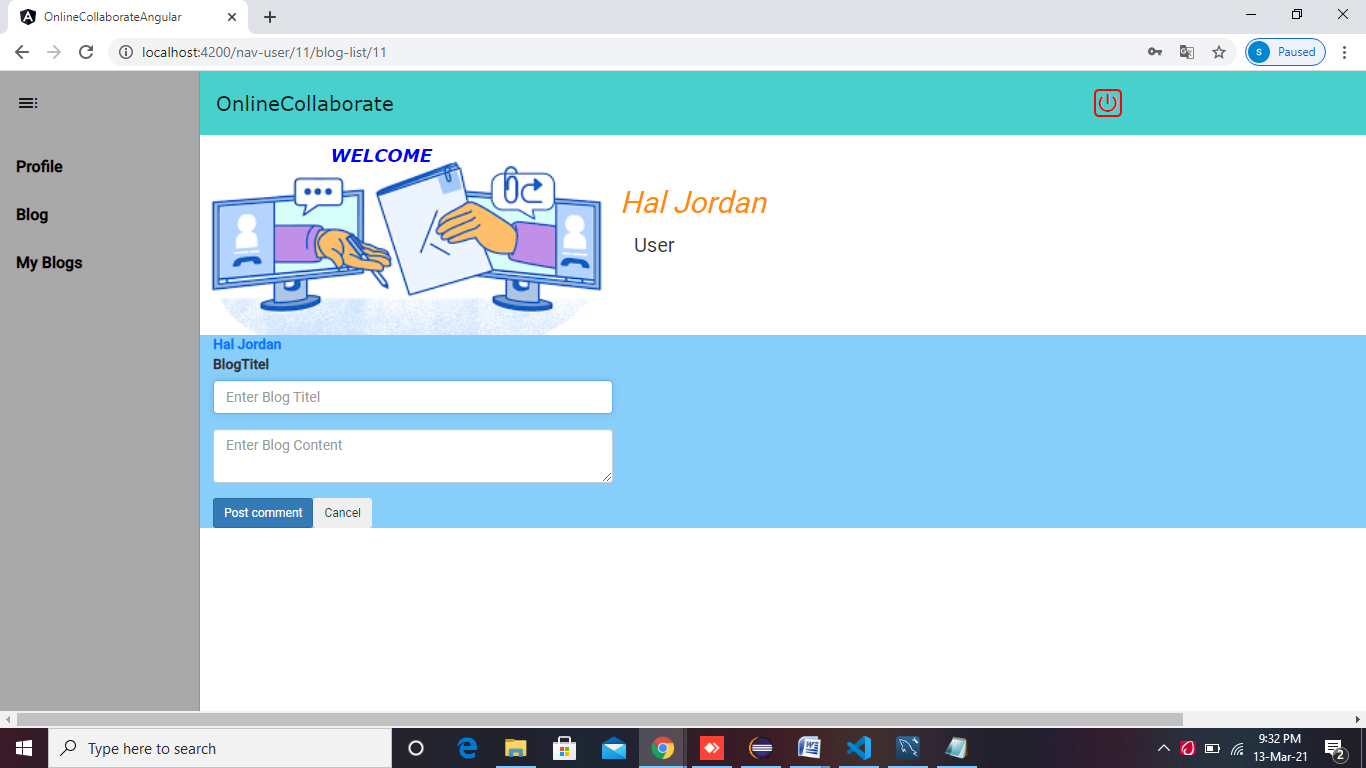
View Their Own Profile:



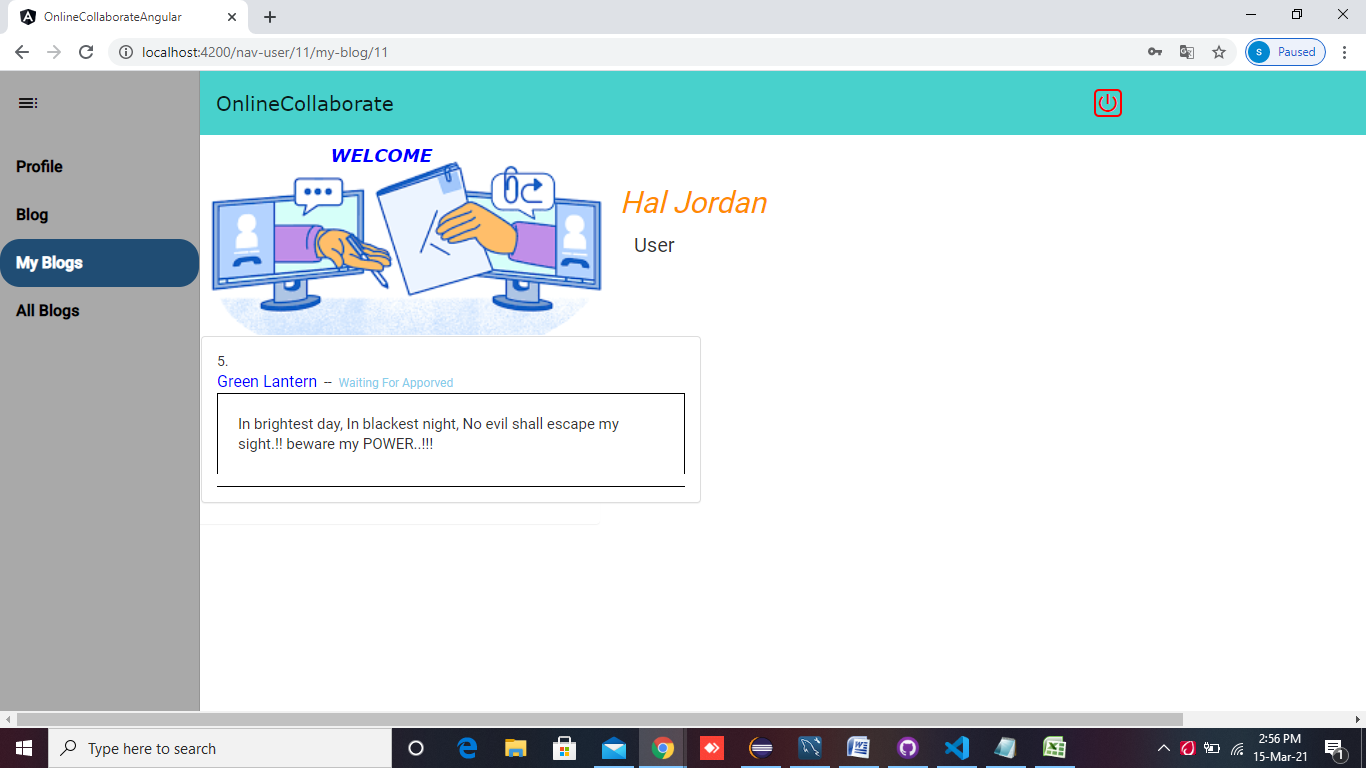
Update Their Own Proifle like Email and password



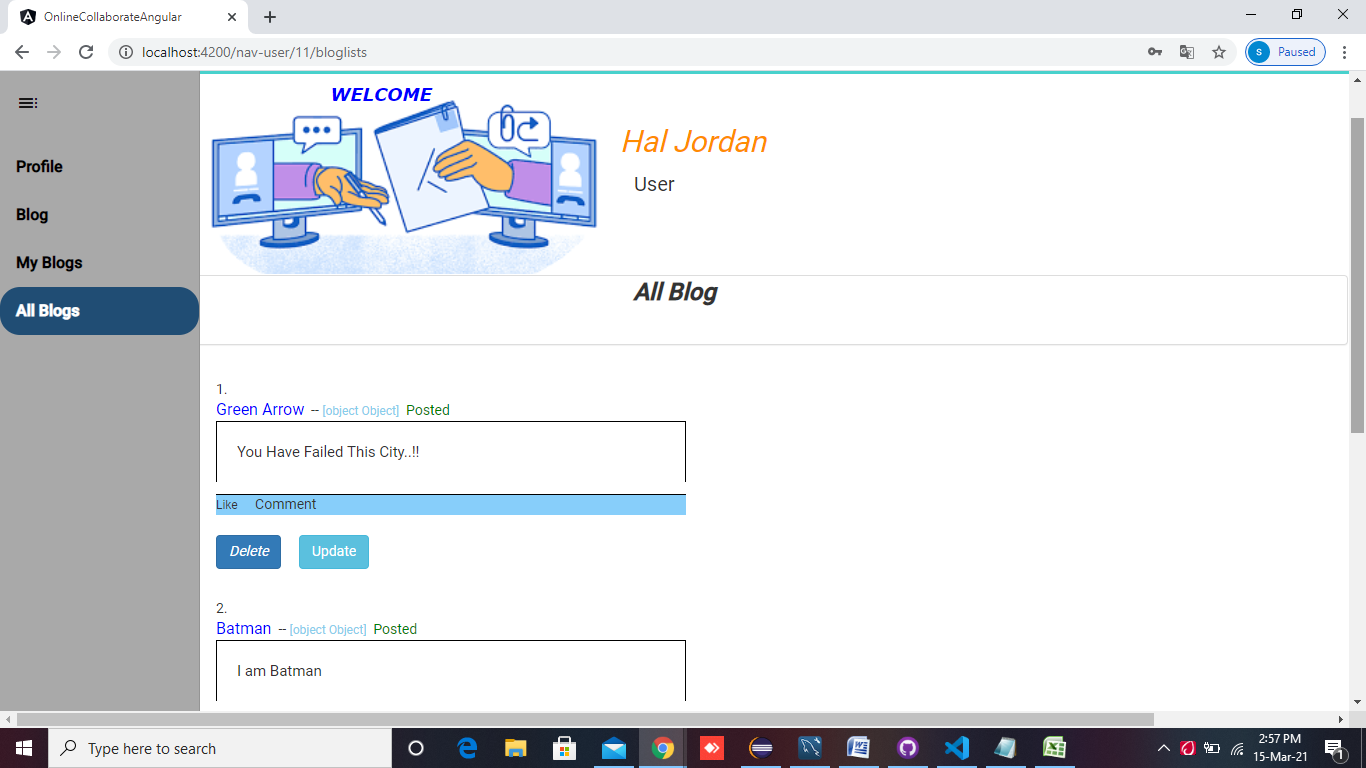
User Can Posting blog:



User Can see their all Posted/Waiting For Apporval Blogs



User Can see all posted blogs by other users



**REFERENCSE**

* <https://www.freecodecamp.org/news/the-best-angular-examples/>
* <https://bbbootstrap.com/snippets/>
* <https://medium.com/ngx/3-ways-to-implement-conditional-validation-of-reactive-forms-c59ed6fc3325>
* <https://bbbootstrap.com/snippets/neomorphism-animation-90020343>
* <https://bbbootstrap.com/snippets/bootstrap-profile-card-template-hover-11673763>
* <https://www.javatpoint.com/angular-spring-login-and-logout-example>
* <https://stackblitz.com/edit/selection-list-activelinks-pweu8z?file=app/selection-list/selection-list.component.html>
* <https://stackoverflow.com/questions/19917922/how-do-i-put-multiple-links-on-the-same-line-html5>
* <https://www.quora.com/How-can-I-resolve-the-org-springframework-web-HttpRequestMethodNotSupportedException-Request-met-hod-GET-not-supported-in-springboot>
* <https://bbbootstrap.com/snippets/animated-login-form-95290954>
* <https://codeburst.io/blogging-app-using-angular-9-and-firebase-98e36168d850>
* <https://dzone.com/articles/tutorial-connect-your-angular-app-to-mysql>

**BIBLIOGRAPHY**

Implementing Data Persistence Using the Hibernate Framework Student Guide-NIIT,

Building Web Applications Using the Spring Framework Student Guide-NIIT,

Building RESTful Web Services with Spring Student Guide-NIIT,

Beginning Angular with Typescript Student Guide-NIIT,

Angular by Example Student Guide-NIIT,

Html5, CSS3 and JavaScript-NIIT,

SQL Programming-Student guide-NIIT,

<https://github.com/mehulk05/Angular-Blogging-App>

<https://github.com/mehulk05/Personal-Portfolio-Angular-Updated>

<https://material.io/resources/icons/?icon=visibility&style=baseline>

**FUTURE ASPECTS**

* The User Can Follow their fellow users.
* The Admin Can send job details, and the user can apply for them,
* User Can able to interact with admin and other users by text message, posting comments of blogs and give a like for their blogs.
* Admin/User can set and change their profile pictures.
* Admin Can post Events and users can participates on the events