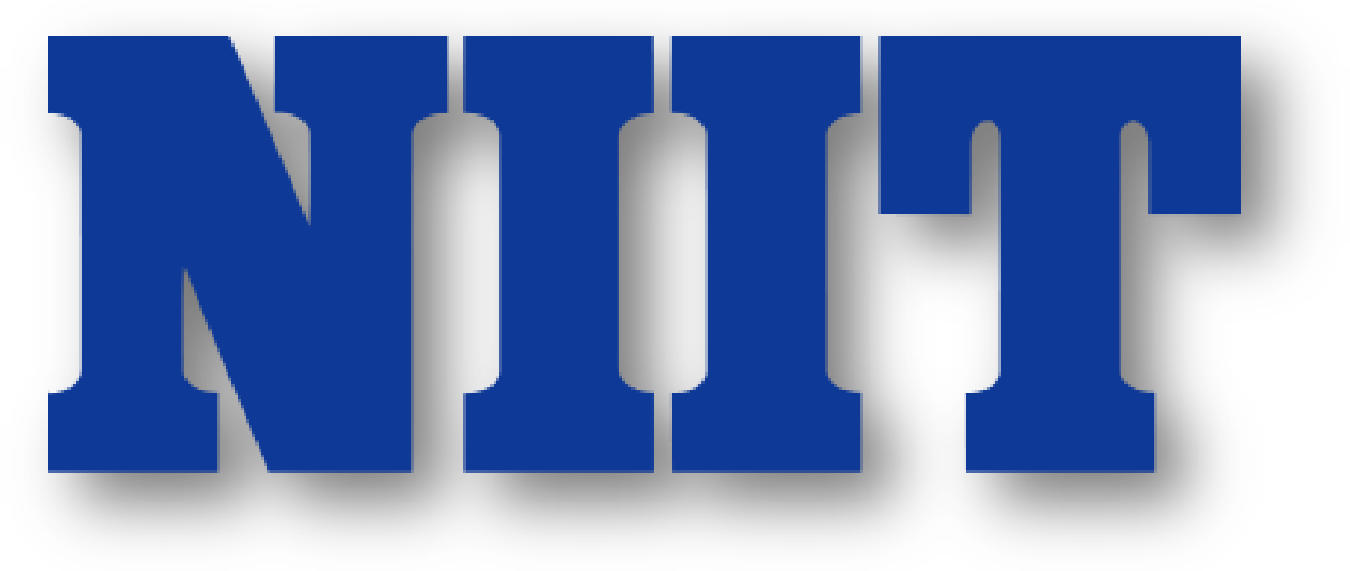
PROJECT ON

**Online Collaboration Application**

DEVELOPED By – Ritik Kumar



**Online Collaboration Application**

Batch Code : S210192

Start Date : 23 February 2021

End Date : 14th January 2021

Name of the

Coordinator : Mrs. Lopamudra Bera

Name of the

Developer : Ritik Kumar

Date of

Submission : 15th January 2021

**Online Collaboration Application**

CERTIFICATE

This is to certify that this report, titled Online Collaboration Application embodies the original work done by Ritik Kumar, in partial fulfillment of their course requirement at NIIT.

CO-ORDINATOR:

Acknowledgement

*We have benefitted a lot from the feedback and suggestions given to us by Mrs. Lopamudra Bera and other faculty members.*

Configuration

Hardware

Processor : Intel i3 or higher.

RAM : 2GB(minimum)

Speed : 1.5GHz

Secondary

Storage : 10GB

Software :

Java Version : JDK 1.8

Database

Management: MySQL

IDE : Eclipse, Visual Studio Code

OS : Windows 10.

Implementation and Diagrams

[3]Java has been one of the most popular programming language for many years. Java is Object Oriented. However it is not considered as pure object oriented as it provides support for primitive data types (like int, char, etc).The Java codes are first compiled into byte code (machine independent code). Then the byte code is run on **J**ava **V**irtual **M**achine (JVM) regardless of the underlying architecture. Java syntax is similar to C/C++. But Java does not provide low level programming functionalities like pointers. Also, Java codes are always written in the form of classes and objects.

Java is used in all kind of applications like Mobile Applications (Android isJava based), desktop applications, web applications, client server applications, enterprise applications and many more.

There are current two sets of Java APIs for graphics programming: AWT (Abstract Windowing Toolkit), Swing AWT API was introduced in JDK 1.0. Most of the AWT components have become obsolete and should be replaced by newer Swing components.Swing API, a much more comprehensive set of

graphics libraries that enhances the AWT, was introduced as part of Java Foundation Classes (JFC) after the release of JDK 1.1. JFC consists of Swing, Java2D,

Some Code Snippets

Database:

1. Open MySQL Workbench.
2. Create a schema named collaborate
3. Create initially following tables:

* Active the Schema

use collaborate;

* Create User table

create table User

(

UserId int not null auto\_increment,

FirstName varchar(30),

LastName varchar(30),

Username varchar(20),

Password varchar(20),

Email varchar(40),

Role varchar(5),

Status varchar(10),

IsOnline boolean,

Enabled boolean,

primary key(UserId)

)

* Create Blog table

create table Blog

(

BlogId int not null auto\_increment,

BlogTitle varchar(30),

BlogContent varchar(200),

BlogPosted Date,

status varchar(10),

NoOfLikes int,

NoOfViews int,

NoOfComments int,

UserId int,

Username varchar(20),

primary key(BlogId)

)

* Create BlogComments table

create table BlogComments

(

BlogCommentId int not null auto\_increment,

UserId int,

Username varchar(20),

UserProfileId varchar(20),

Title varchar(30),

NoOfLikes int,

BlogComment varchar(50),

CurrentDate Date,

primary key(BlogCommentId)

)

Project: Backend

1. Create a SpringBoot project named “OnlineCollaboration”(You can suggest any fesible project name according to project specification) with web, Spring Data JPA, SpringBoot Dev Tools and MySQL Server Driver packages. Extract that project.
2. Import the project in Eclipse.
3. Create the configuration class  
   Instead of XML, we perform annotation-based configuration. So, we create a class HibernateConfig.java inside com.coll.OnlineCollaborate.config package and specify the required configuration in it. However, there is one more configuration class OnlineCollaborateApplication.java. This class is provided by Spring Boot automatically.

**(1)Pom.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.4.3</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.coll</groupId>

<artifactId>OnlineCollaboration</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>OnlineCollaboration</name>

<description>Demo project for Spring Boot</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

<optional>true</optional>

</dependency>

<dependency>

<groupId>com.microsoft.sqlserver</groupId>

<artifactId>mssql-jdbc</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

**(2)HibernateConfig.java**

package com.coll.OnlineCollaboration.config;

import java.util.Properties;

import javax.sql.DataSource;

import org.springframework.boot.autoconfigure.EnableAutoConfiguration;

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

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.ComponentScan;

import org.springframework.context.annotation.ComponentScans;

import org.springframework.context.annotation.Configuration;

import org.springframework.jdbc.datasource.DriverManagerDataSource;

import org.springframework.orm.hibernate5.HibernateTransactionManager;

import org.springframework.orm.hibernate5.LocalSessionFactoryBean;

import org.springframework.transaction.annotation.EnableTransactionManagement;

import org.springframework.web.servlet.ViewResolver;

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

@Configuration

@ComponentScans(value=

{

@ComponentScan("com.coll.OnlineCollaboration"),

@ComponentScan("model"),

@ComponentScan("controller"),

@ComponentScan("dao"),

@ComponentScan("service")

})

@EnableAutoConfiguration(exclude=

{

HibernateJpaAutoConfiguration.class

})

@EnableTransactionManagement

public class HibernateConfig {

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

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="123456";

@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.OnlineCollaboration");

Properties hibernateProperties=new Properties();

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

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

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

sessionFactory.setHibernateProperties(hibernateProperties);

return sessionFactory;

}

@Bean

public HibernateTransactionManager getTransactionManager()

{

HibernateTransactionManager txm=new HibernateTransactionManager();

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

return txm;

}

@Bean

public ViewResolver jspViewResolver()

{

InternalResourceViewResolver viewResolver=new InternalResourceViewResolver();

viewResolver.setPrefix("/views/");

viewResolver.setSuffix(".jsp");

return viewResolver;

}

}

**(3)OnlineCollaborationApplication.java**

package com.coll.OnlineCollaboration;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class OnlineCollaborationApplication {

public static void main(String[] args) {

SpringApplication.run(OnlineCollaborationApplication.class, args);

}

}

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

### **4(a) DomainResponse.java**

**package** com.coll.OnlineCollaboration.model;

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

**int** responseCode;

String responseMessage;

**public** DomainResponse() {

**super**();

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

}

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

**super**();

**this**.responseCode = responseCode;

**this**.responseMessage = responseMessage;

}

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

**return** responseCode;

}

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

**this**.responseCode = responseCode;

}

**public** String getResponseMessage() {

**return** responseMessage;

}

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

**this**.responseMessage = responseMessage;

}

}

### **4(b) User.java**

package com.coll.OnlineCollaboration.model;

import java.io.Serializable;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import org.springframework.stereotype.Component;

@Component

@Entity

public class User extends DomainResponse implements Serializable{

private static final long serialVersionUID=1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

private int userId;

private String firstName;

private String lastName;

private String username;

private String password;

private String email;

private String role;

private String status;

private boolean isOnline;

private boolean enabled;

public int getUserId() {

return userId;

}

public void setUserId(int userId) {

this.userId = userId;

}

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

public boolean getIsOnline() {

return isOnline;

}

public void setIsOnline(boolean isOnline) {

this.isOnline = isOnline;

}

public boolean getEnabled() {

return enabled;

}

public void setEnabled(boolean enabled) {

this.enabled = enabled;

}

}

### **4(c) Blog.java**

package com.coll.OnlineCollaboration.model;

import java.io.Serializable;

import java.time.LocalDate;

import java.util.List;

import javax.persistence.CascadeType;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.OneToMany;

import org.springframework.stereotype.Component;

import com.fasterxml.jackson.annotation.JsonManagedReference;

@Component

@Entity

public class Blog extends DomainResponse implements Serializable{

private static final long serialVersionUID=1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

int blogId;

String blogTitle, blogContent;

LocalDate blogPosted;

String status;

int noOfLikes, noOfComments, noOfViews;

int userId;

String username;

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

@JsonManagedReference

List<BlogComments> blogComments;

public int getBlogId() {

return blogId;

}

public void setBlogId(int blogId) {

this.blogId = blogId;

}

public String getBlogTitle() {

return blogTitle;

}

public void setBlogTitle(String blogTitle) {

this.blogTitle = blogTitle;

}

public String getBlogContent() {

return blogContent;

}

public void setBlogContent(String blogContent) {

this.blogContent = blogContent;

}

public LocalDate getBlogPosted() {

return blogPosted;

}

public void setBlogPosted(LocalDate blogPosted) {

this.blogPosted = blogPosted;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

public int getNoOfLikes() {

return noOfLikes;

}

public void setNoOfLikes(int noOfLikes) {

this.noOfLikes = noOfLikes;

}

public int getNoOfComments() {

return noOfComments;

}

public void setNoOfComments(int noOfComments) {

this.noOfComments = noOfComments;

}

public int getNoOfViews() {

return noOfViews;

}

public void setNoOfViews(int noOfViews) {

this.noOfViews = noOfViews;

}

public int getUserId() {

return userId;

}

public void setUserId(int userId) {

this.userId = userId;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public List<BlogComments> getBlogComments() {

return blogComments;

}

public void setBlogComments(List<BlogComments> blogComments) {

this.blogComments = blogComments;

}

public static long getSerialversionuid() {

return serialVersionUID;

}

}

### **4(d) BlogComments.java**

package com.coll.OnlineCollaboration.model;

import java.io.Serializable;

import java.time.LocalDate;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

import org.springframework.stereotype.Component;

import com.fasterxml.jackson.annotation.JsonBackReference;

@Component

@Entity

public class BlogComments implements Serializable{

private static final long serialVersionUID=1L;

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

int blogCommentId;

int userId;

String username;

String userProfileId;

String title;

int noOfLikes;

String blogComment;

LocalDate currentDate;

@ManyToOne

@JoinColumn(name="BlogId")

@JsonBackReference

Blog blog;

public int getBlogCommentId() {

return blogCommentId;

}

public void setBlogCommentId(int blogCommentId) {

this.blogCommentId = blogCommentId;

}

public int getUserId() {

return userId;

}

public void setUserId(int userId) {

this.userId = userId;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getUserProfileId() {

return userProfileId;

}

public void setUserProfileId(String userProfileId) {

this.userProfileId = userProfileId;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public int getNoOfLikes() {

return noOfLikes;

}

public void setNoOfLikes(int noOfLikes) {

this.noOfLikes = noOfLikes;

}

public String getBlogComment() {

return blogComment;

}

public void setBlogComment(String blogComment) {

this.blogComment = blogComment;

}

public LocalDate getCurrentDate() {

return currentDate;

}

public void setCurrentDate(LocalDate currentDate) {

this.currentDate = currentDate;

}

public Blog getBlog() {

return blog;

}

public void setBlog(Blog blog) {

this.blog = blog;

}

public static long getSerialversionuid() {

return serialVersionUID;

}

}

1. Create the DAO interfaces inside com.coll.OnlineCollaboration.dao package:

**5(a) IUserDao.java**

package com.coll.OnlineCollaboration.dao;

import java.util.List;

import com.coll.OnlineCollaboration.model.User;

public interface IUserDao {

List<User> userListbyStatus(String status);

List<User> getAllUsers();

User getUserById(int userId);

User getUserByUsername(String username);

User validateUser(User user);

boolean addUser(User user);

boolean updateUser(User user);

boolean deleteUser(int userId);

boolean deactivateUser(int userId);

List<User> getAllInactivateUsers();

boolean activeUser(int userId);

boolean updateUserProfile(String file, Integer userId);

}

**5(b) IBlogDao.java**

package com.coll.OnlineCollaboration.dao;

import java.util.List;

import com.coll.OnlineCollaboration.model.Blog;

public interface IBlogDao {

List<Blog> getAllBlogs();

List<Blog> getBlogsByStatus(String status);

List<Blog> getUsersBlogs(int userId);

List<Blog> mainList();

Blog getBlogById(int blogId);

boolean addBlog(Blog blog);

boolean updateBlog(Blog blog);

boolean deleteBlog(Blog blog);

}

**5(c) IBlogCommentsDao.java**

package com.coll.OnlineCollaboration.dao;

import java.util.List;

import com.coll.OnlineCollaboration.model.BlogComments;

public interface IBlogCommentsDao {

List<BlogComments> getAllBlogComments();

BlogComments getBlogCommentsById(int blogCommentId);

boolean addBlogComments(BlogComments blogComments);

boolean updateBlogComments(BlogComments blogComments);

boolean deleteBlogComments(BlogComments blogComments);

}

1. Create the DAO interface implementation classes inside com.coll.OnlineCollaboration.daoImpl package:

**6(a) UserDaoImpl.java**

package com.coll.OnlineCollaboration.daoImpl;

import java.util.List;

import org.hibernate.query.Query;

import org.hibernate.SessionFactory;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.coll.OnlineCollaboration.dao.IUserDao;

import com.coll.OnlineCollaboration.model.User;

@Repository("userDao")

@Transactional

public class UserDaoImpl implements IUserDao{

@Autowired

SessionFactory sessionFactory;

@Override

public List<User> userListbyStatus(String status) {

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

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

return query.getResultList();

}

@Override

public List<User> getAllUsers() {

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

}

@Override

public User getUserById(int userId) {

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

}

@Override

public User getUserByUsername(String username) {

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

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

}

@Override

public User validateUser(User user) {

String username="user.getUsername";

String password="user.getPassword";

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

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

try

{

user=(User)query.getSingleResult();

return user;

}

catch(Exception e)

{

e.printStackTrace();

return null;

}

}

@Override

public boolean addUser(User user) {

try

{

sessionFactory.getCurrentSession().save(user);

return true;

}

catch(Exception ex)

{

ex.printStackTrace();

return false;

}

}

@Override

public boolean updateUser(User user) {

try

{

sessionFactory.getCurrentSession().update(user);

return true;

}

catch(Exception ex)

{

ex.printStackTrace();

return false;

}

}

@Override

public boolean deleteUser(int userId) {

try

{

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

return true;

}

catch(Exception ex)

{

ex.printStackTrace();

return false;

}

}

@Override

public boolean deactivateUser(int userId) {

try

{

User user=getUserById(userId);

user.setEnabled(false);

sessionFactory.getCurrentSession().update(user);

return true;

}

catch(Exception ex)

{

ex.printStackTrace();

return false;

}

}

@Override

public List<User> getAllInactivateUsers() {

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

}

@Override

public boolean activeUser(int userId) {

try

{

User user=getUserById(userId);

user.setEnabled(true);

sessionFactory.getCurrentSession().update(user);

return true;

}

catch(Exception ex)

{

ex.printStackTrace();

return false;

}

}

@Override

public 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;

}

}

}

**6(b) BlogDaoImpl.java**

package com.coll.OnlineCollaboration.daoImpl;

import java.util.List;

import org.hibernate.query.Query;

import org.hibernate.SessionFactory;

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

import org.springframework.stereotype.Repository;

import org.springframework.transaction.annotation.Transactional;

import com.coll.OnlineCollaboration.dao.IBlogDao;

import com.coll.OnlineCollaboration.model.Blog;

@Repository("blogDao")

@Transactional

public class BlogDaoImpl implements IBlogDao{

@Autowired

SessionFactory sessionFactory;

@Override

public List<Blog> getAllBlogs() {

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

}

@Override

public List<Blog> getBlogsByStatus(String status) {

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

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

return query.getResultList();

}

@Override

public List<Blog> getUsersBlogs(int userId) {

String q="from Blog where UserId='"+userId+"'";

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

return query.getResultList();

}

@Override

public List<Blog> mainList() {

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

{

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(Blog blog) {

try

{

sessionFactory.getCurrentSession().delete(blog);

return true;

}

catch(Exception ex)

{

ex.printStackTrace();

return false;

}

}

}

**6(c) BlogCommentsDaoImpl.java**

package com.coll.OnlineCollaboration.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.OnlineCollaboration.dao.IBlogCommentsDao;

import com.coll.OnlineCollaboration.model.BlogComments;

@Repository("blogCommentsDao")

@Transactional

public class BlogCommentsDaoImpl implements IBlogCommentsDao{

@Autowired

SessionFactory sessionFactory;

@Override

public List<BlogComments> getAllBlogComments() {

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

}

@Override

public BlogComments getBlogCommentsById(int blogCommentId) {

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

}

@Override

public boolean addBlogComments(BlogComments blogComments) {

try

{

sessionFactory.getCurrentSession().save(blogComments);

return true;

}

catch(Exception 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(BlogComments blogComments) {

try

{

sessionFactory.getCurrentSession().delete(blogComments);

return true;

}

catch(Exception ex)

{

ex.printStackTrace();

return false;

}

}

}

1. Create the service interfaces inside com.coll.OnlineCollaboration.service package:

**7(a) IUserService.java**

package com.coll.OnlineCollaboration.service;

import java.util.List;

import com.coll.OnlineCollaboration.model.User;

public interface IUserService {

List<User> userListbyStatus(String status);

List<User> getAllUsers();

User getUserById(int userId);

User getUserByUsername(String username);

User validateUser(User user);

boolean addUser(User user);

boolean updateUser(User user);

boolean deleteUser(int userId);

boolean deactiveUser(int userId);

List<User> getAllInactivateUsers();

boolean activeUser(int userId);

boolean updateUserProfile(String file, Integer userId);

}

**7(b) IBlogService.java**

package com.coll.OnlineCollaboration.service;

import java.util.List;

import com.coll.OnlineCollaboration.model.Blog;

public interface IBlogService {

List<Blog> getAllBlogs();

List<Blog> getBlogsByStatus(String status);

List<Blog> getUsersBlogs(int id);

List<Blog> mainList();

Blog getBlogById(int blogId);

boolean addBlog(Blog blog);

boolean updateBlog(Blog blog);

boolean deleteBlog(Blog blog);

}

**7(c) IBlogCommentsService.java**

package com.coll.OnlineCollaboration.service;

import java.util.List;

import com.coll.OnlineCollaboration.model.BlogComments;

public interface IBlogCommentsService {

List<BlogComments> getAllBlogComments();

BlogComments getBlogCommentsById(int blogCommentId);

boolean addBlogComments(BlogComments blogComments);

boolean updateBlogComments(BlogComments blogComments);

boolean deleteBlogComments(BlogComments blogComments);

}

1. Create the service implementation classes inside com.coll.OnlineCollaboration.serviceImpl package:

**8(a) UserServiceImpl.java**

package com.coll.OnlineCollaboration.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.OnlineCollaboration.dao.IUserDao;

import com.coll.OnlineCollaboration.model.User;

import com.coll.OnlineCollaboration.service.IUserService;

@Service

@Transactional

public class UserServiceImpl implements IUserService{

@Autowired

IUserDao userDao;

@Override

public List<User> userListbyStatus(String status) {

return userDao.userListbyStatus(status);

}

@Override

public List<User> getAllUsers() {

return userDao.getAllUsers();

}

@Override

public User getUserById(int userId) {

return userDao.getUserById(userId);

}

@Override

public User getUserByUsername(String username) {

return userDao.getUserByUsername(username);

}

@Override

public User validateUser(User user) {

return userDao.validateUser(user);

}

@Override

public boolean addUser(User user) {

return userDao.addUser(user);

}

@Override

public boolean updateUser(User user) {

return userDao.updateUser(user);

}

@Override

public boolean deleteUser(int userId) {

return userDao.deleteUser(userId);

}

@Override

public boolean deactiveUser(int userId) {

return userDao.deactivateUser(userId);

}

@Override

public boolean updateUserProfile(String file, Integer userId) {

return userDao.updateUserProfile(file, userId);

}

@Override

public List<User> getAllInactivateUsers() {

return userDao.getAllInactivateUsers();

}

@Override

public boolean activeUser(int userId) {

return userDao.activeUser(userId);

}

}

**8(b) BlogServiceImpl.java**

package com.coll.OnlineCollaboration.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.OnlineCollaboration.dao.IBlogDao;

import com.coll.OnlineCollaboration.model.Blog;

import com.coll.OnlineCollaboration.service.IBlogService;

@Service

@Transactional

public class BlogServiceImpl implements IBlogService{

@Autowired

IBlogDao blogDao;

@Override

public List<Blog> getAllBlogs() {

return blogDao.getAllBlogs();

}

@Override

public List<Blog> getBlogsByStatus(String status) {

return blogDao.getBlogsByStatus(status);

}

@Override

public List<Blog> getUsersBlogs(int id) {

return blogDao.getUsersBlogs(id);

}

@Override

public List<Blog> mainList() {

return blogDao.mainList();

}

@Override

public Blog getBlogById(int blogId) {

return blogDao.getBlogById(blogId);

}

@Override

public boolean addBlog(Blog blog) {

return blogDao.addBlog(blog);

}

@Override

public boolean updateBlog(Blog blog) {

return blogDao.updateBlog(blog);

}

@Override

public boolean deleteBlog(Blog blog) {

return blogDao.deleteBlog(blog);

}

}

**8(c) BlogCommentsServiceImpl.java**

package com.coll.OnlineCollaboration.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.OnlineCollaboration.dao.IBlogCommentsDao;

import com.coll.OnlineCollaboration.model.BlogComments;

import com.coll.OnlineCollaboration.service.IBlogCommentsService;

@Service

@Transactional

public class BlogCommentsServiceImpl implements IBlogCommentsService{

@Autowired

IBlogCommentsService blogCommentsService;

@Override

public List<BlogComments> getAllBlogComments() {

return blogCommentsService.getAllBlogComments();

}

@Override

public BlogComments getBlogCommentsById(int blogCommentId) {

return blogCommentsService.getBlogCommentsById(blogCommentId);

}

@Override

public boolean addBlogComments(BlogComments blogComments) {

return blogCommentsService.addBlogComments(blogComments);

}

@Override

public boolean updateBlogComments(BlogComments blogComments) {

return blogCommentsService.updateBlogComments(blogComments);

}

@Override

public boolean deleteBlogComments(BlogComments blogComments) {

return blogCommentsService.deleteBlogComments(blogComments);

}

}

1. Create the controller classes inside com.coll.OnlineCollaboration.controller package:

**9(a) UserController.java**

package com.coll.OnlineCollaboration.controller;

import java.util.List;

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

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

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

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

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

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

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

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

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

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

import com.coll.OnlineCollaboration.model.User;

import com.coll.OnlineCollaboration.service.IUserService;

@RestController

@CrossOrigin(origins="http://localhost:8080")

@RequestMapping(value="/api")

public class UserController {

@Autowired

IUserService userService;

@PostMapping("save-user")

public boolean saveUser(@RequestBody User user)

{

return userService.addUser(user);

}

@GetMapping("user-list")

public List<User> allUsers()

{

return userService.getAllUsers();

}

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

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

{

return userService.deleteUser(userId);

}

@GetMapping("user/{userId}")

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

{

return userService.getUserById(userId);

}

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

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

{

user.setUserId(userId);

return userService.updateUser(user);

}

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

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

{

user.setUsername(username);

user.setPassword(password);

return userService.validateUser(user);

}

}

**9(b) BlogController.java**

package com.coll.OnlineCollaboration.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.OnlineCollaboration.model.Blog;

import com.coll.OnlineCollaboration.service.IBlogService;

@RestController

@CrossOrigin(origins="http://localhost:8080")

@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") Blog blogId)

{

return blogService.deleteBlog(blogId);

}

@GetMapping("blog/{blogId}")

public Blog blogById(@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);

}

}

**9(c) BlogCommentsController.java**

package com.coll.OnlineCollaboration.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.OnlineCollaboration.model.BlogComments;

import com.coll.OnlineCollaboration.service.IBlogCommentsService;

@RestController

@CrossOrigin(origins="http://localhost:8080")

@RequestMapping(value="/api")

public class BlogCommentsController {

@Autowired

IBlogCommentsService blogCommentsService;

@PostMapping("save-blogComments")

public boolean saveBlogComments(@RequestBody BlogComments blogComments)

{

return blogCommentsService.addBlogComments(blogComments);

}

@GetMapping("blogComments-list")

public List<BlogComments> allBlogComments()

{

return blogCommentsService.getAllBlogComments();

}

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

public boolean deleteBlogComments(@PathVariable("blogCommentId") BlogComments blogCommentId)

{

return blogCommentsService.deleteBlogComments(blogCommentId);

}

@GetMapping("blogComments/{blogCommentId}")

public BlogComments blogCommentById(@PathVariable("blogCommentId") int blogCommentId)

{

return blogCommentsService.getBlogCommentsById(blogCommentId);

}

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

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

{

blogComments.setBlogCommentId(blogCommentId);

return blogCommentsService.updateBlogComments(blogComments);

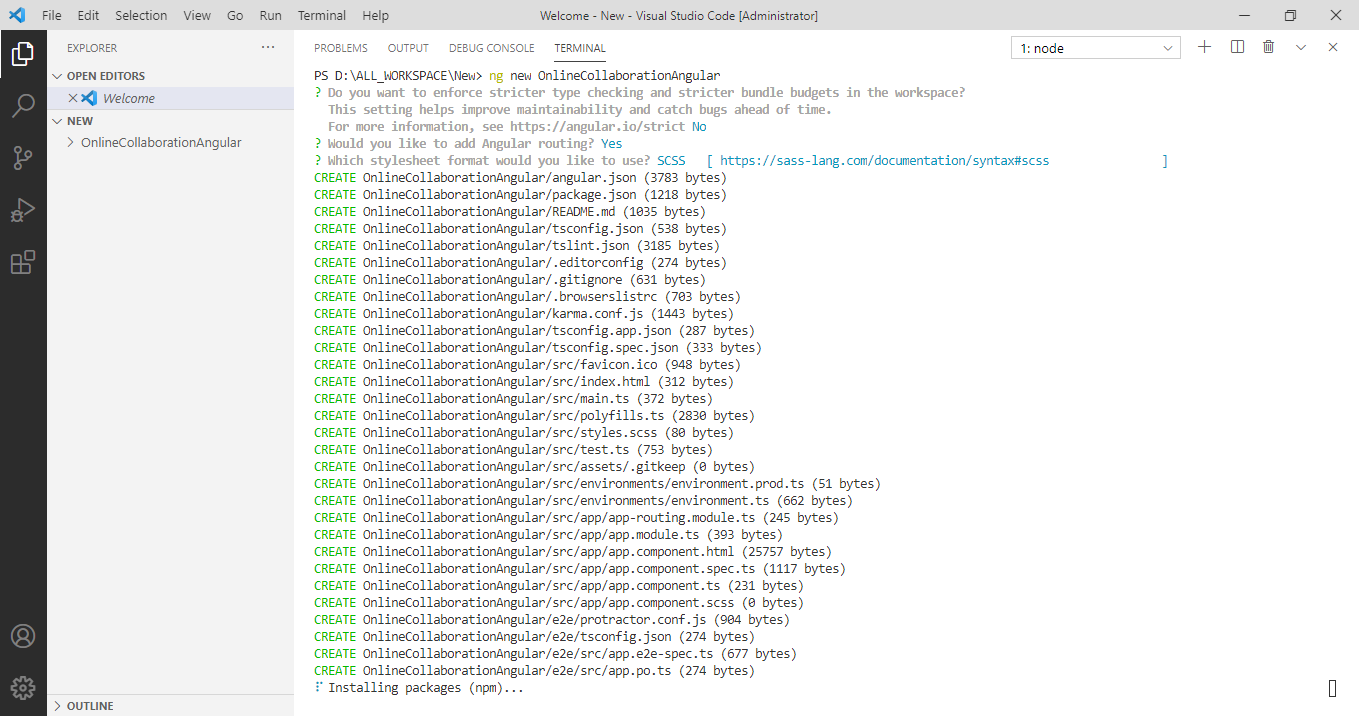
}

}

Project: Frontend

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

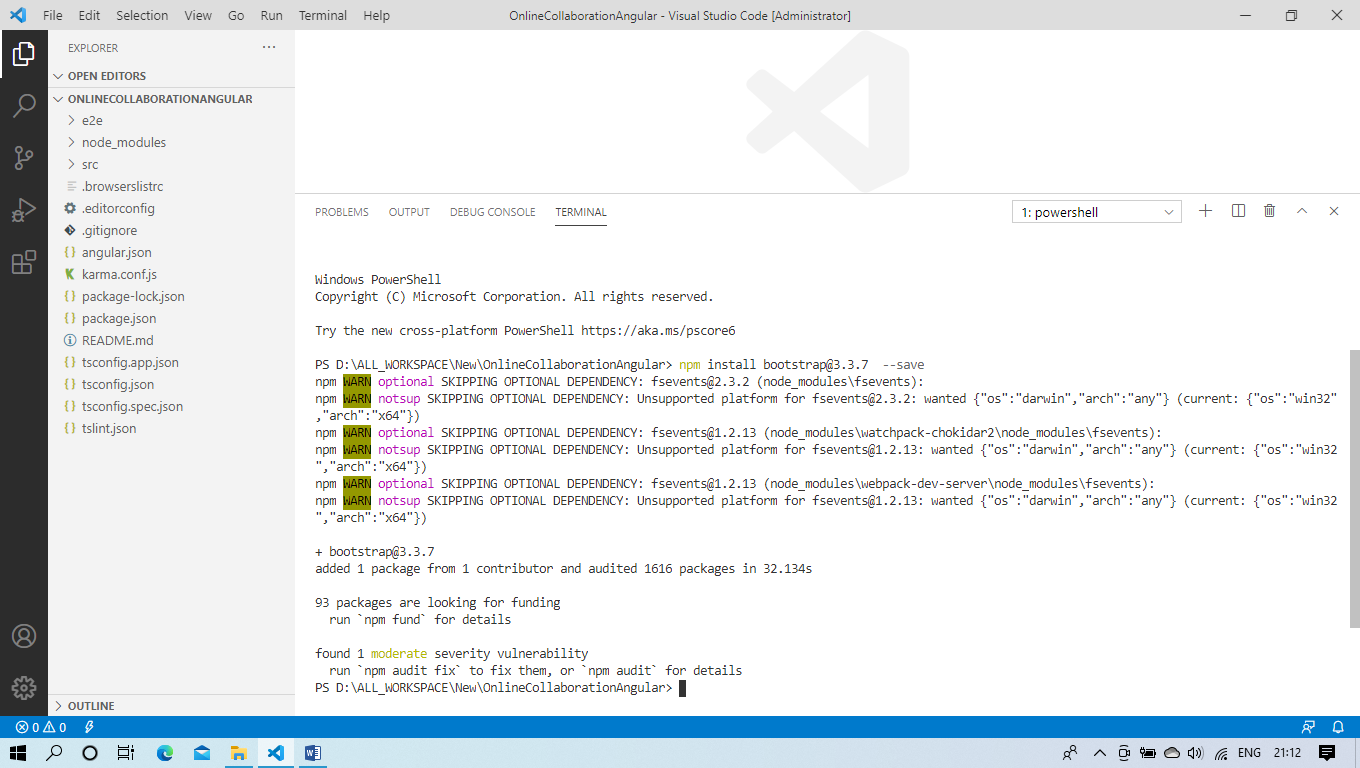
=> ng new OnlineCollaborationAngular



1. Install Bootstrap CSS framework

Use the following command to install bootstrap in the project.

***=> npm install*** [***bootstrap@3.3.7***](mailto:bootstrap@3.3.7) ***--save***



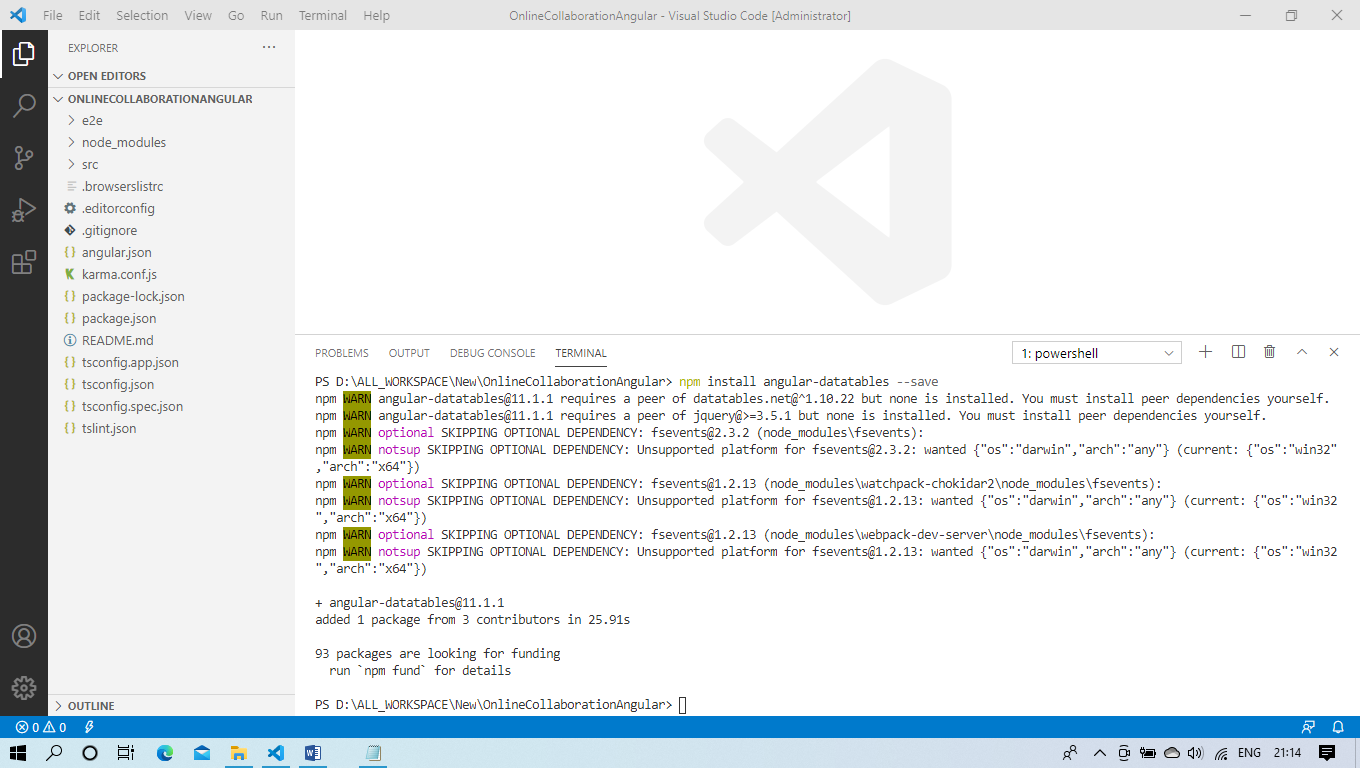
1. Include the following code in the styles.scss file:

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

1. Install Angular Data Table

Use the following command to install bootstrap in the project.

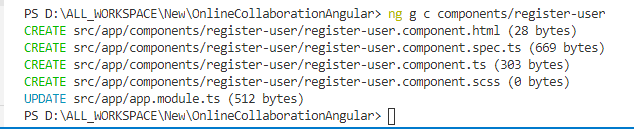
* ***npm install angular-datatables --save***



(14) Generate following Components:

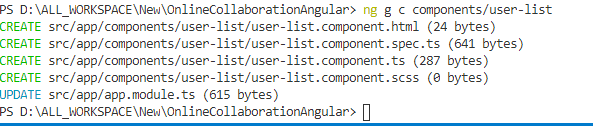
a. register-user:

ng g c components/register-user



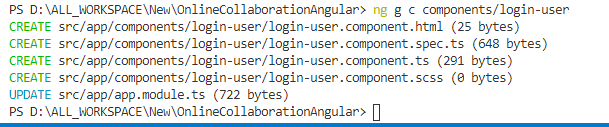
b. user-list:

ng g c components/user-list



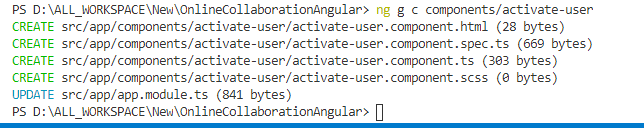
c. login-user:

ng g c components/login-user



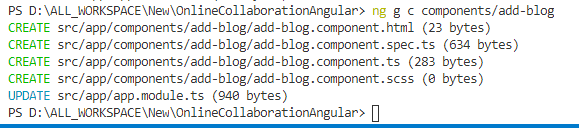
d. activate-user:

ng g c components/activate-user



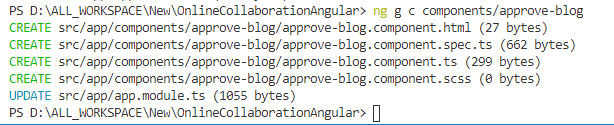
e. add-blog:

ng g c components/add-blog



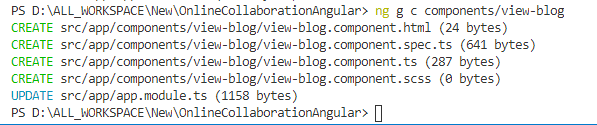
f. approve-blog:

ng g c components/approve-blog



g. view-blog:

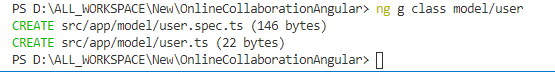
ng g c components/view-blog



(15). Create models classes:

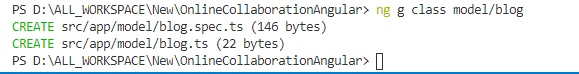
a. user:

ng g class model/user



b. blog:

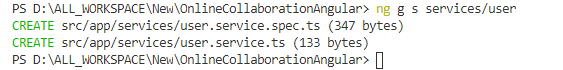
ng g class model/blog



(16). Create services:

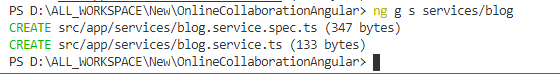
a. user

ng g s services/user



b. blog:

ng g s services/blog



(17) Open app.module.ts file and include required modules:

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

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

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

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

import { DataTablesModule } from 'angular-datatables';

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

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

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

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

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

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

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

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

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

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

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

@NgModule({

  declarations: [

    AppComponent,

    RegisterUserComponent,

    UserListComponent,

    LoginUserComponent,

    ActivateUserComponent,

    AddBlogComponent,

    ApproveBlogComponent,

    ViewBlogComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule,

    DataTablesModule,

    HttpClientModule,

    ReactiveFormsModule

  ],

  providers: [UserService, BlogService],

  bootstrap: [AppComponent]

})

export class AppModule { }

(18) Open app-routing.module.ts file and edit as following:

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

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

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

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

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

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

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

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

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

const routes: Routes = [

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

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

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

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

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

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

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

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

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

(19) Open app.component.html(landing page) file and edit as following:

<div class="container-fluid">

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

    <ul class="navbar-nav">

      <li class="nav-item">

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

      </li>

      <li class="nav-item">

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

      </li>

      <li class="nav-item">

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

      </li>

    </ul>

  </nav>

  <router-outlet></router-outlet>

</div>

(20) Open user.ts file(model class) and define the class:

export class User {

    userId:number;

    firstName:String;

    lastName:String;

    username:String;

    password:String;

    confirm\_password:String;

    email:String;

    role:String;

    status:String;

    isOnline:boolean;

    enabled:boolean;

}

(21) Open blog.ts file(model class) and define the class:

export class Blog {

    blogId:String;

    blogTitle:String;

    blogContent:String;

    blogPosted:Date;

    status:String;

    noOfLikes: number;

    noOfViews:number;

    noOfComments:number;

    userId:number;

    username:String;

}

(22) Open user.service.ts file and define the service class:

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

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

import { Observable, 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);

  }

  activateUser(userId: number):Observable<any>

  {

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

  }

}

(23) Opened register-user.component.ts file. and edited as following:

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

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

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

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

@Component({

  selector: 'app-register-user',

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

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

})

export class RegisterUserComponent implements OnInit {

  user:User=new User();

  submitted=false;

  constructor(private userservice:UserService) { }

  ngOnInit(): void {

    this.submitted=false;

  }

  registrationform=new FormGroup({

  });

  register(register)

  {

  }

  save()

  {

    this.userservice.createUser(this.user)

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

    this.user=new User();

  }

  get FirstName()

  {

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

  }

  get LastName()

  {

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

  }

  get Username()

  {

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

  }

  get Password()

  {

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

  }

  get ConfirmPassword()

  {

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

  }

  get Email()

  {

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

  }

  get Role()

  {

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

  }

  registrationForm()

  {

    this.submitted=false;

    this.registrationform.reset();

  }

}

(24) Open register-user.component.html file and modify as folloiwng:

<h3>Register Here</h3>

<div class="row">

    <div class="col-sm-4"></div>

    <div class="col-sm-4">

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

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

                <div class="form-group">

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

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

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

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

                    </div>

                </div>

                <div class="form-group">

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

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

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

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

                    </div>

                </div>

                <div class="form-group">

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

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

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

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

                    </div>

                </div>

                <div class="form-group">

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

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

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

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

                    </div>

                </div>

                <div class="form-group">

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

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

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

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

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

                    </div>

                </div>

                <div class="form-group">

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

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

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

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

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

                    </div>

                </div>

                <div class="form-group">

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

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

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

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

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

                    </select>

                </div>

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

            </form>

        </div>

    </div>

    <div>

        <div class="col-sm-4"></div>

    </div>

    <div class="col-sm-4">

        <div [hidden]="!submitted">

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

        </div>

    </div>

</div>

(25) Open user-list.comment.ts and modify as following:

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

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

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

import { Observable, Subject } from 'rxjs';

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

import { DataTablesModule } from 'angular-datatables';

@Component({

  selector: 'app-user-list',

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

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

})

export class UserListComponent implements OnInit {

  usersArray:any=[];

  dtOptions: DataTables.Settings={};

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

  users: Observable<User[]>;

  user: User=new User();

  deleteMessage=false;

  userlist:any;

  isupdated=false;

  constructor(private userservice: UserService) { }

  ngOnInit(): void {

    this.isupdated=false;

    this.dtOptions=

    {

      pageLength:6,

      stateSave:true,

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

      processing:true

    };

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

      this.users=data;

      this.dtTrigger.next();

    })

  }

  deleteUser(id:number)

  {

    this.userservice.deleteUser(id)

    .subscribe(data=>{

      console.log(data);

      this.deleteMessage=true;

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

        this.users=data;

      })

    }, error=>console.log(error));

  }

  updateUser(id:number)

  {

    this.userservice.getUser(id)

    .subscribe(data=>{

      this.userlist=data;

    }, error=>console.log(error));

  }

  userupdateform=new FormGroup({

    userId:new FormControl(),

    firstName:new FormControl(),

    lastName:new FormControl(),

    userName:new FormControl(),

    password:new FormControl(),

    email:new FormControl(),

    role:new FormControl(),

    status:new FormControl(),

    isOnline:new FormControl(),

    enabled:new FormControl()

  })

  update(user)

  {

    this.user=new User();

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

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

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

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

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

    this.user.email=this.Email.value;

    this.user.role=this.Role.value;

    this.user.enabled=true;

    this.user.status="Active";

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

      this.isupdated=true;

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

        this.users=data;

      })

    },error=>console.log(error));

  }

  get UserId(){

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

  }

  get FirstName(){

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

  }

  get LastName(){

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

  }

  get Username(){

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

  }

  get Password(){

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

  }

  get Email(){

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

  }

  get Role(){

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

  }

  get Status(){

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

  }

  get IsOnline(){

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

  }

  get Enabled(){

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

  }

  changeisUpdate(){

    this.isupdated=false;

  }

}

(26) Open user-list.component.html file and modify as following:

<div class="panel panel-default">

    <div class="panel-heading">

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

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

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

            <div class="col-sm-4">

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

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

                    <strong>User data deleted</strong>

                </div>

            </div>

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

        </div>

    </div>

    <div class="panel-body">

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

            <thead class="thead-light">

                <tr>

                    <th>User Id</th>

                    <th>FirstName</th>

                    <th>LastName</th>

                    <th>Username</th>

                    <th>Password</th>

                    <th>Email</th>

                    <th>Role</th>

                    <th>Status</th>

                    <th>IsOnline</th>

                    <th>Enabled</th>

                    <th>Action</th>

                </tr>

            </thead>

            <tbody>

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

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

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

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

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

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

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

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

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

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

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

                    <td>

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

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

                        </button>

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

                            Update

                        </button>

                    </td>

                </tr>

            </tbody>

        </table>

    </div>

</div>

<!-- myModal -->

<div class="modal" id="myModal">

    <div class="modal-dialog">

        <div class="modal-content">

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

                <!-- Modal Header -->

                <div class="modal-header">

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

                </div>

                <!-- Modal Body -->

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

                    <div [hidden]="isupdated">

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

                        <div class="form-group">

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

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

                        </div>

                        <div class="form-group">

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

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

                        </div>

                        <div class="form-group">

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

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

                        </div>

                        <div class="form-group">

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

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

                        </div>

                        <div class="form-group">

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

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

                        </div>

                        <div class="form-group">

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

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

                        </div>

                        <div class="form-group">

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

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

                        </div>

                        <div class="form-group">

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

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

                        </div>

                    </div>

                    <div [hidden]="!isupdated">

                        <h4>User Detail Updated!</h4>

                    </div>

                </div>

                <div class="modal-footer">

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

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

                </div>

            </form>

        </div>

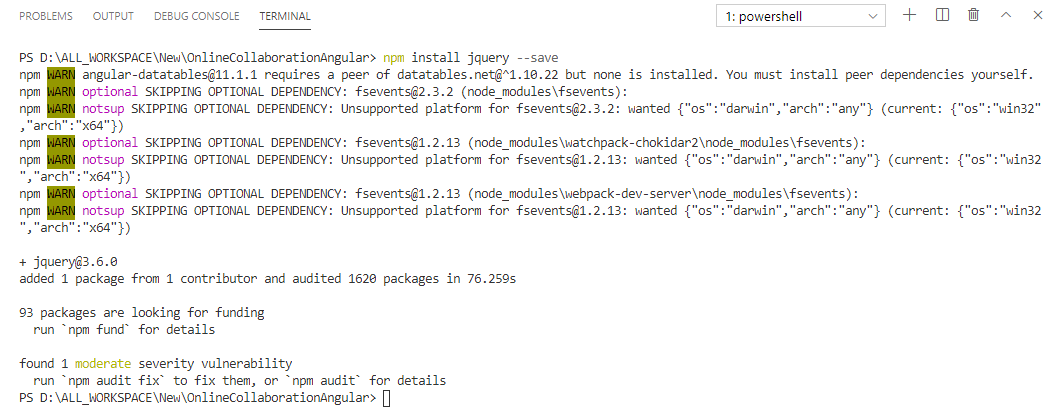
    </div>

</div>

(27) Install following modules :

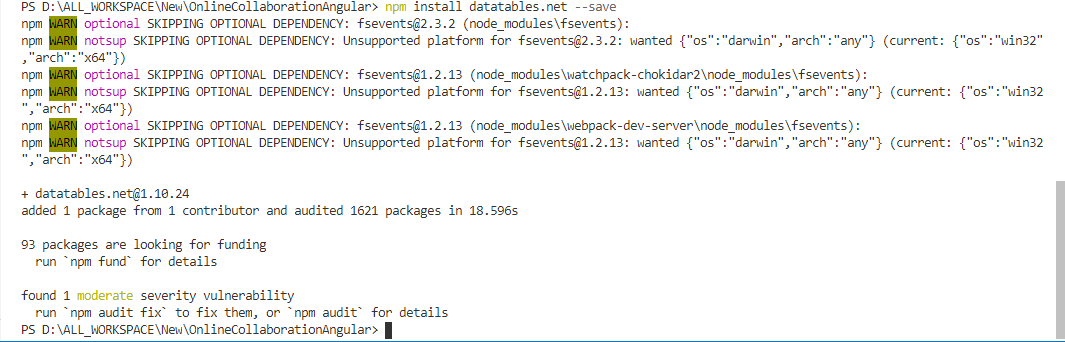
1. Use the following command to install jquery in the project.

* *npm install jquery –save*



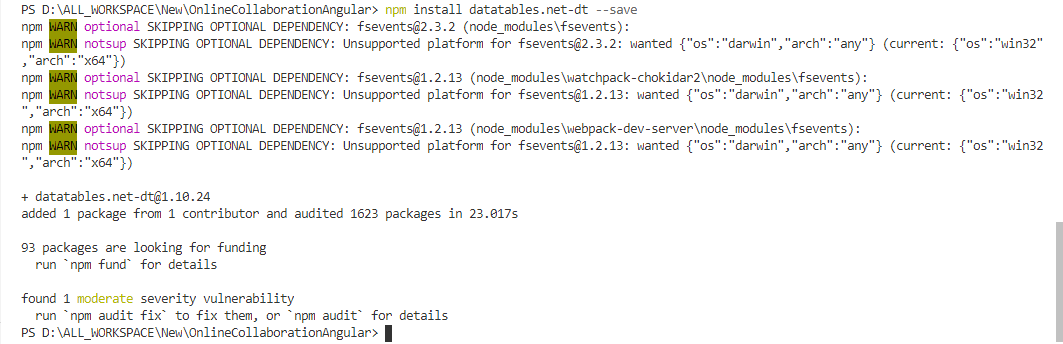
1. Use the following command to install datatables.net in the project.

* *npm install datatables.net --save*



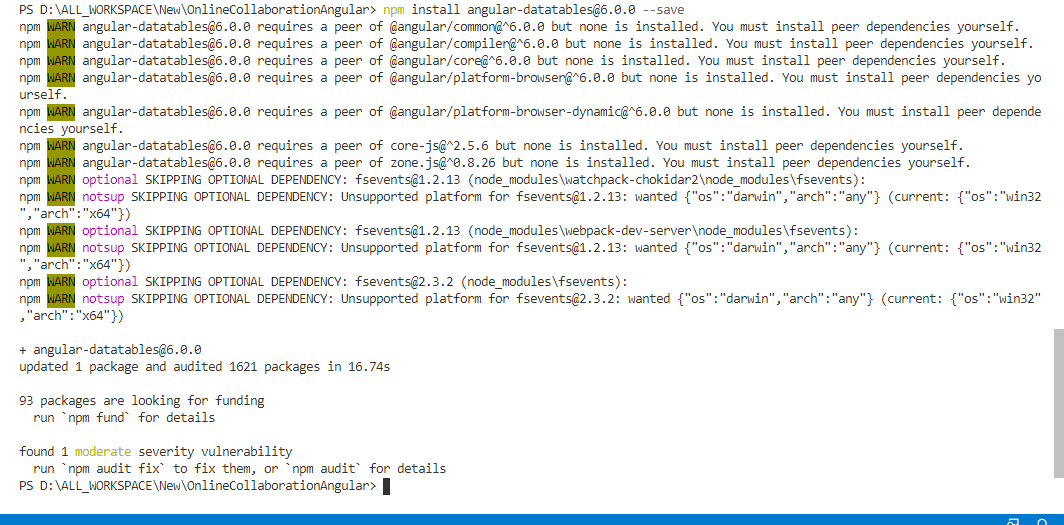
1. Use the following command to install datatables.net-dt in the project.

* *npm install datatables.net-dt --save*



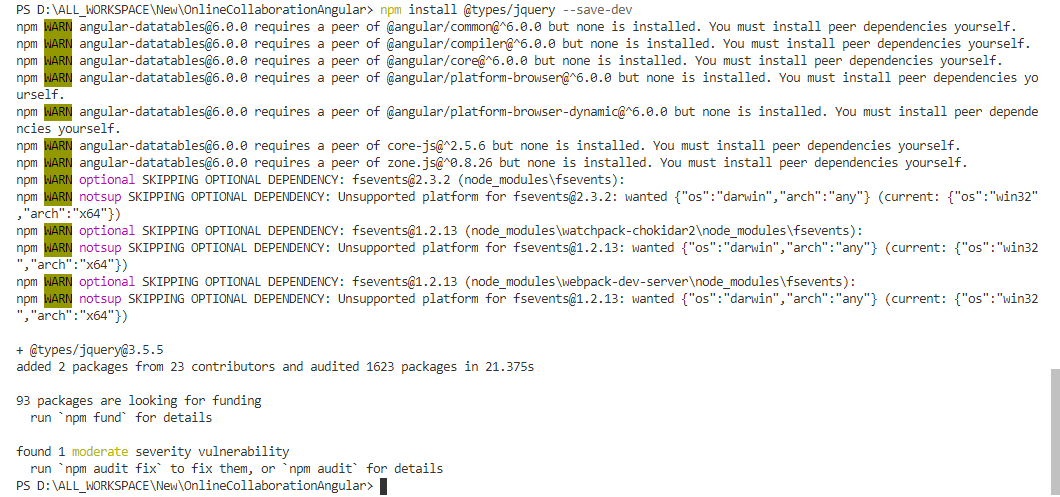
1. Use the following command to install angular-datatables in the project.

* *npm install angular-datatables@6.0.0 --save*



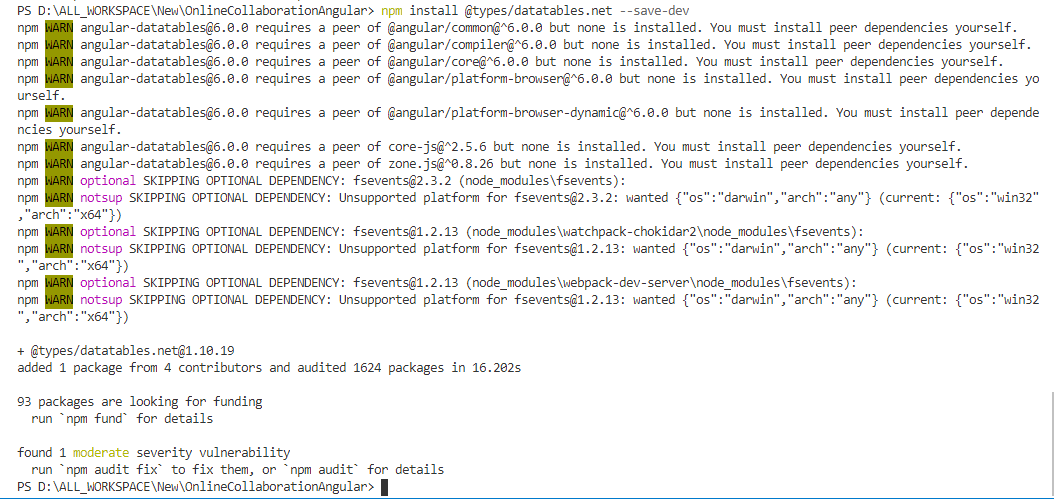
1. Use the following command to install @types/jquery in the project.

* *npm install @types/jquery --save-dev*



1. Use the following command to install @types/datatables in the project.

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



(28) Open angular.json file and at line no. 30: styles and script arrays, modify as following:

"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"

            ]

(29) Save all….

(30) Run….

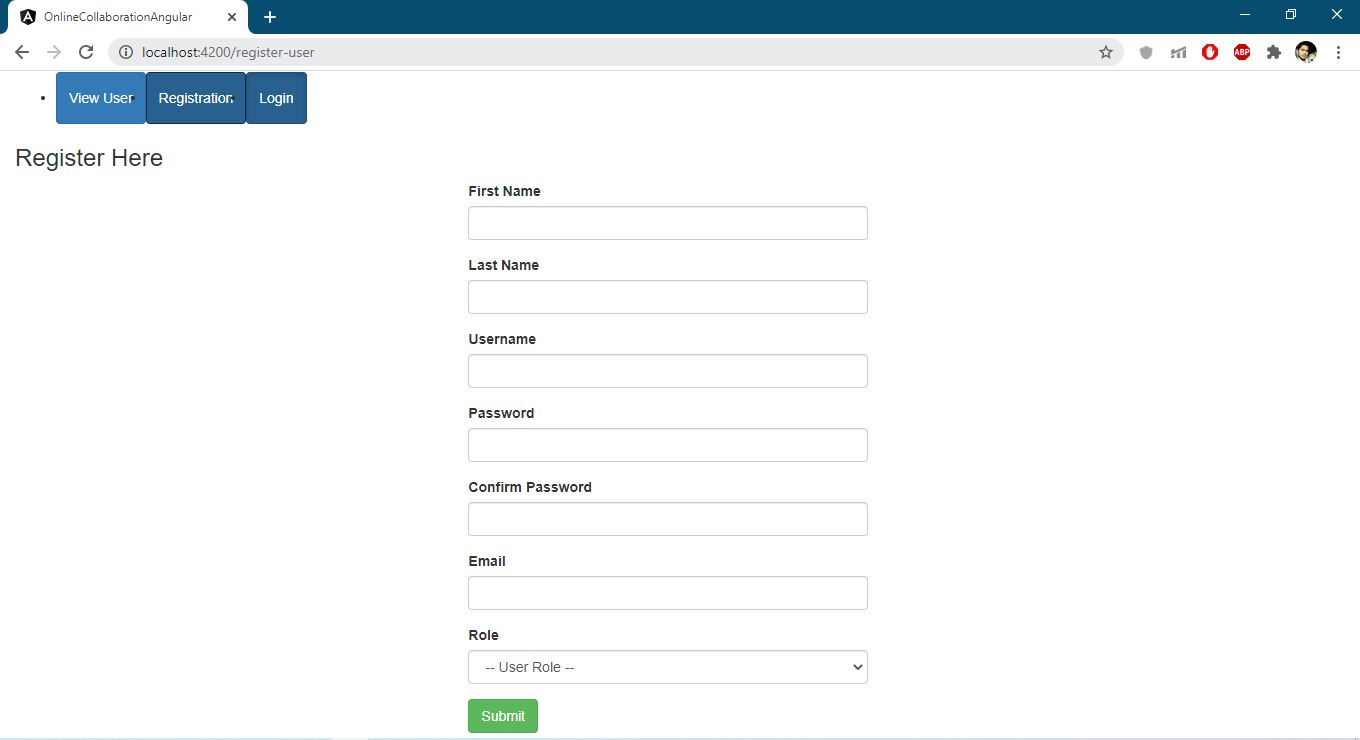


Fig :- Registration Page

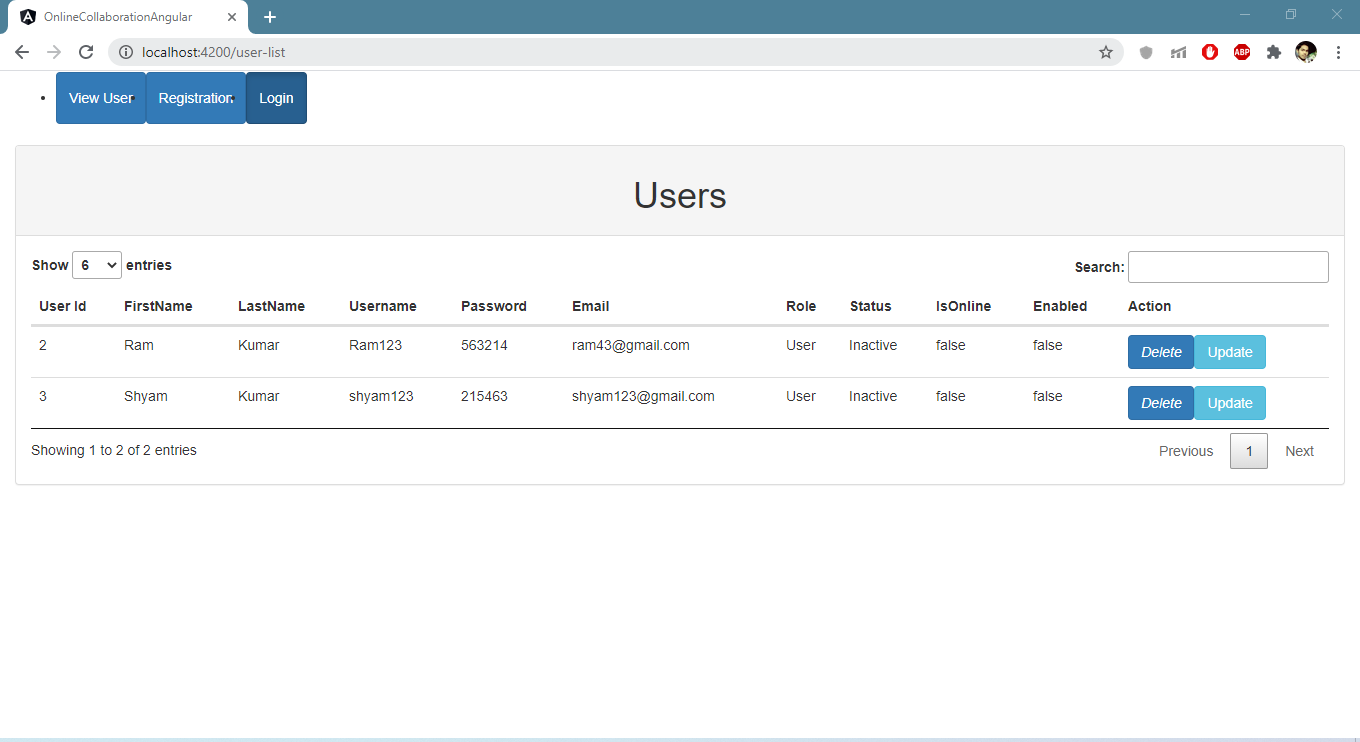


Fig :- View User