Online Collaborate

Developed By

Name: K Naveenraj Reg.No: S200040300164

NIIT

Online Collaborate

Batch Code - S210167

Start Date - 11/11/20

End Date - 15/03/21

Name Of the Coordinator - Lopamudra Bera

Name Of the Developer - K Naveenraj

Date of Submission - 15.03.2020

NIIT

CERTIFICATE

This is to certify that this report, titled **Online Collaborate** embodies the original work done by **Mr.K Naveenraj**, in partial fullfillment of his course requirement at NIIT.

Coordinator : LOPAMUDRA BERA

Acknowledgement

I would like to thank my mentor Lopamudra Bera and also the institute NIIT who helped me achieve this project on Online Collaborate. It helped me to do a lot on the research on the project and the environment.

I am really thankful for the staffs for helping me around in learning a lot of information during my learning process.

Abstract

This project manages the entire process of Blogging website to help Bloggers and Users to use the site with ease.

Its also provides single window system to Blogger who can showcase their ideas and skills in their respective fields or showcase their passion.

Configuration

Hardware

Dell Insprion 15 (i5 processor),
16Gb RAM,
2TB Hard Disk.

Operating System

Windows 10 X64

Software

- Eclipse IDE
  - ✓ Java
  - ✓ Spring Boot
  - ✓ Hibernate
  - ✓ MySQL

- Visual Studio
  - ✓ Angular
  - ✓ Html
  - ✓ SCSS
  - ✓ Typescript
  - ✓ jQuery

TABLE OF CONTENTS

Chapter 1- Introduction

1.0 Aim
1.1 Objectives

Chapter 2- Project Requirement Specification

2.0 Literature Research
2.1 Statement of Requirements

Chapter 3- Project Analysis

3.0 Project Plan
3.1 System Architecture
3.2 Business Process Model
3.3 Software Requirement Specification
3.4 High Level Use Case Diagrams

Chapter 4- Project Design

4.0 Low Level Use Case Diagrams
4.1 User Interface Design
4.2 Systems Input and Output Design
4.3 Database Structure
4.4 Sample Code

Challenges
Observations
References
Appendix

AIM

The main objective of the Online Blogging System is to manage the details of Blogs,Topic,Idea,Content,Entries. It manages all the information about Blogs, Views, Entries, Blogs. The project is totally built at administrative end and thus only the administrator is guaranteed the access.

OBJECTIVES

Following are the objectives to be achieved through Online Collaboration Application

- User registration
- Profile Validation
- Activation/Approving
- Posting Blogs
- View Blogs
- Job Postings
- Recruitment

This objectives are to be automated which were previously done manually.

STATEMENT OF REQUIREMENTS

PROJECT OBJECTIVES

Title	Online Collaborate
Subtitle	Blog Management System
Author(s)	K Naveenraj
Author's E-mail	naveenrajkrishnan@gmail.com
Author's Phone	9942543332
Description	Blogging Website
Version	1.01

About Your Company

Online Collaborate is a public collaboration website which provides a platform to connect with people to share their works.

Need for Collaboration

- It helps us problem-solve
- Collaboration brings people and organizations closer
- Collaboration helps people learn from each other
- It opens up new channels for communication
- Collaboration boosts morale across your organization
- It leads to higher retention rates
- Collaboration makes us more efficient workers

PROJECT LIFE CYCLE MODEL

The Initiation Phase: The initiation phase aims to define and authorize the project.

Vision: Registration ,Authorization,Blogging

The Planning Phase: The purpose of this phase is to lay down a detailed strategy of how the project has to be performed and how to make it a success.

Strategic Planning- overall approach to the project

Implementation Planning-ways to apply the decisions

The Execution Phase: In this phase, the decisions and activities defined during the planning phase are implemented.

The Termination Phase: This is the last phase of any project, and it marks the official closure of the project.

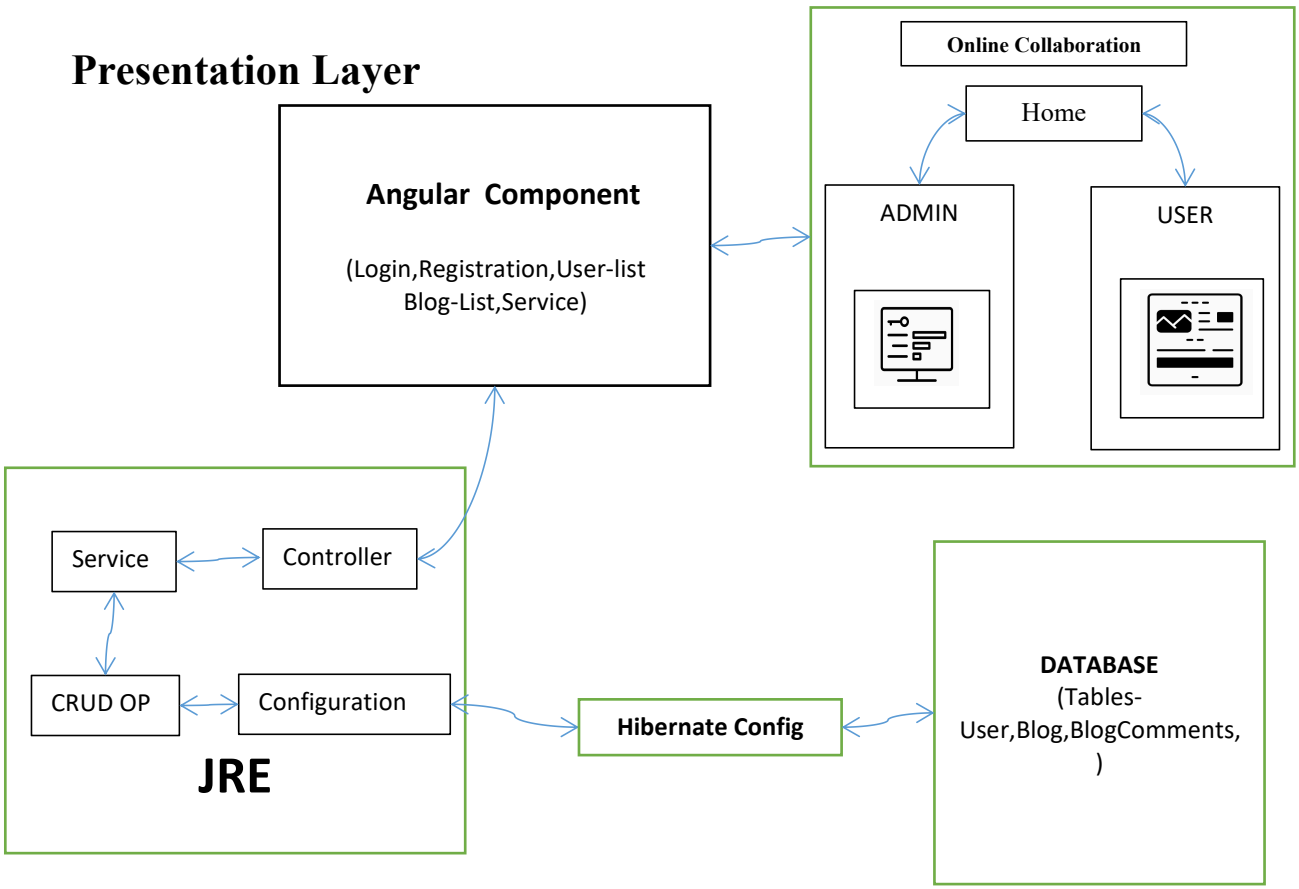
SYSTEM ARCHITECTURE

3-Tier Architecture

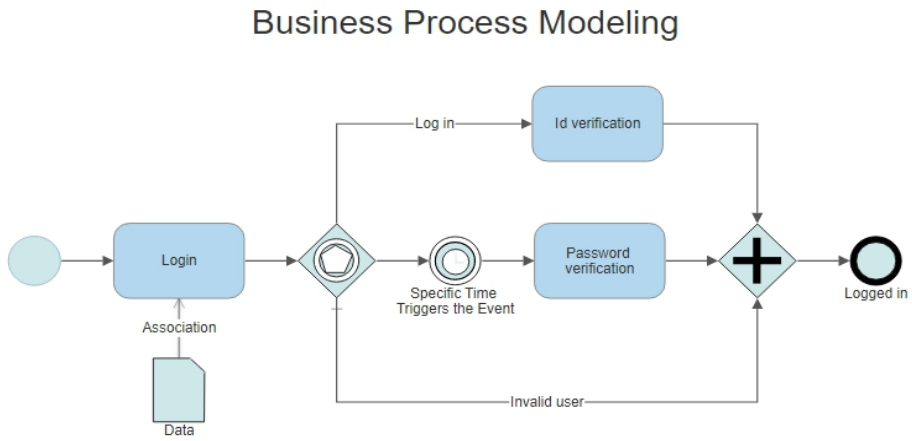
1. A **Presentation Layer** that sends content to browsers in the form of HTML/JS/CSS. This might leverage frameworks like React, Angular, Ember, Aurora, etc.

2. An **Application Layer** that uses an application server and processes the business logic for the application. This might be written in C#, Java, C++, Python, Ruby, etc.

3. A **Data Layer** which is a database management system that provides access to application data. This could be MSSQL, MySQL, Oracle, or PostgreSQL, Mongo, etc.



BUSINESS PROCESS MODEL



Business process modeling (or) process modeling, is the analytical representation or put simply an illustration of an organization's business processes. Modeling processes is a critical component for effective business process management.

Benefits of business process modeling:

- Gives everyone a clear understanding of how the process works
- Provides consistency and controls the process
- Identifies and eliminates redundancies and inefficiencies
- Sets a clear starting and ending to the process

SOFTWARE REQUIREMNETS SPECIFICATION

Table of Contents

1. Introduction
   - 1.1 Purpose
   - 1.2 Scope
   - 1.3 Definitions,acronyms and abbreviations
   - 1.4 References
   - 1.5 Overview

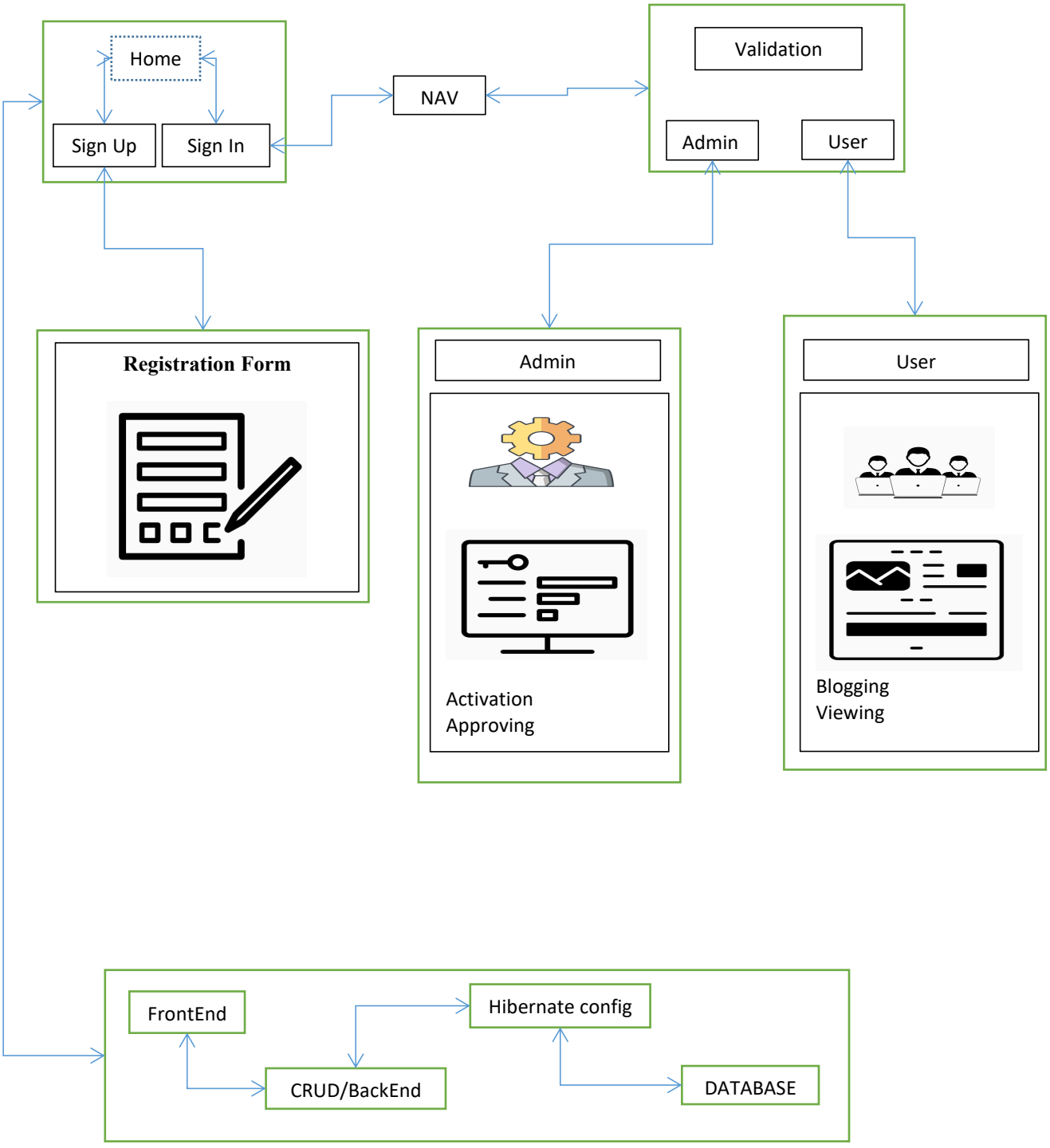
2. General Description
   - 2.1 Product perspective
   - 2.2 Product functions
   - 2.3 User characteristics
   - 2.4 Constraints
   - 2.5 Assumption and dependencies

3. Specific Requirements
   - 3.1 Functional requirements
   - 3.2 Non-functional requirements
   - 3.3 External interface requirements
   - 3.4 Performance requirements
   - 3.5 Design constraints
   - 3.6 Attributes
   - 3.7 Other requirements

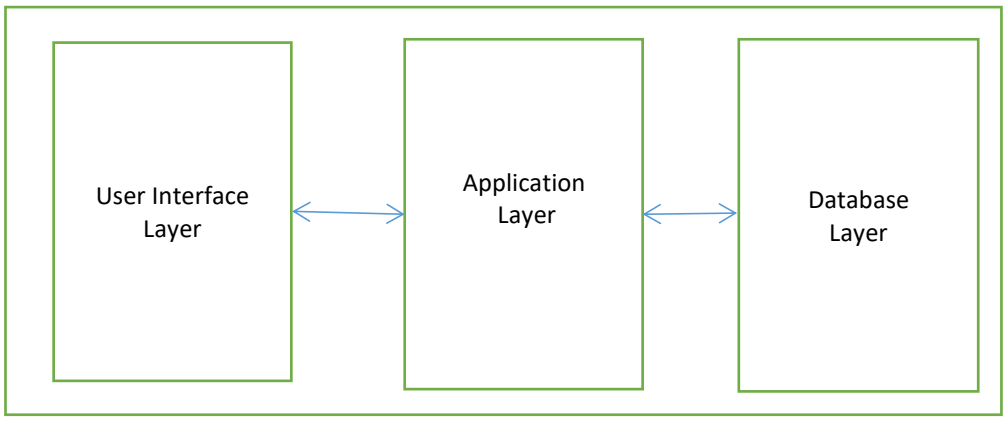
4. Appendices

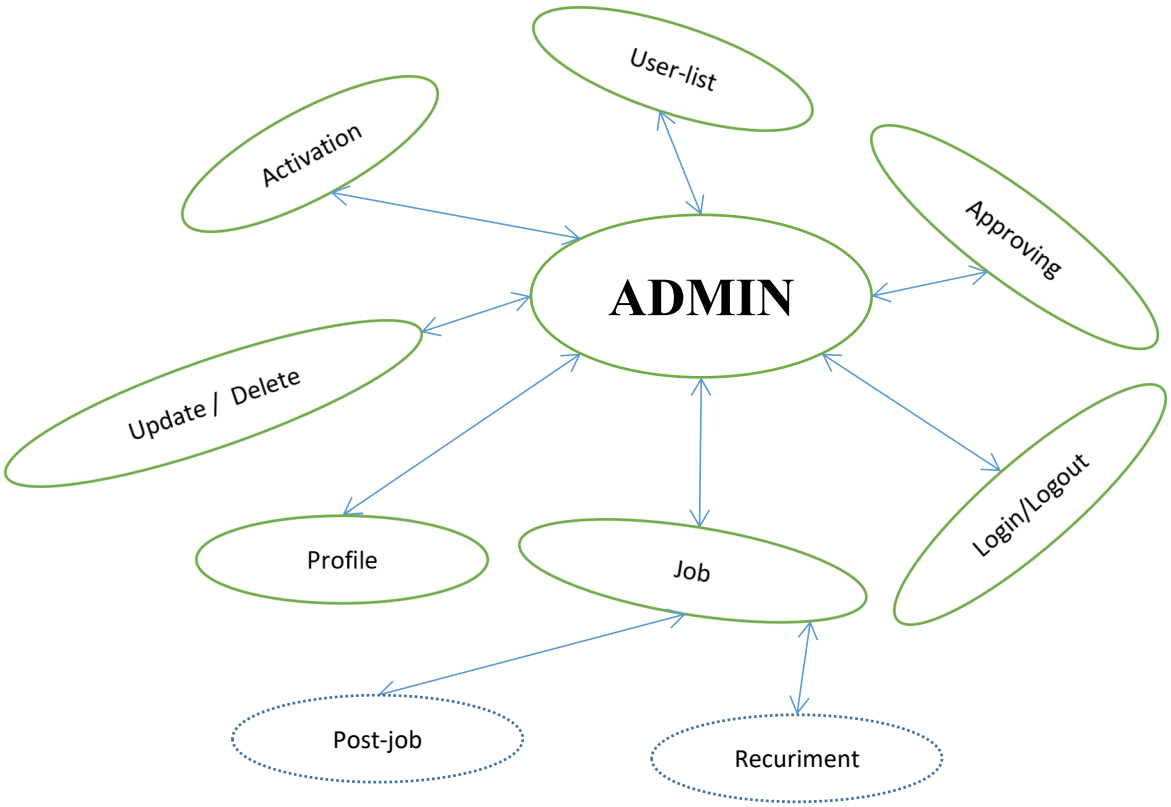
5. Index

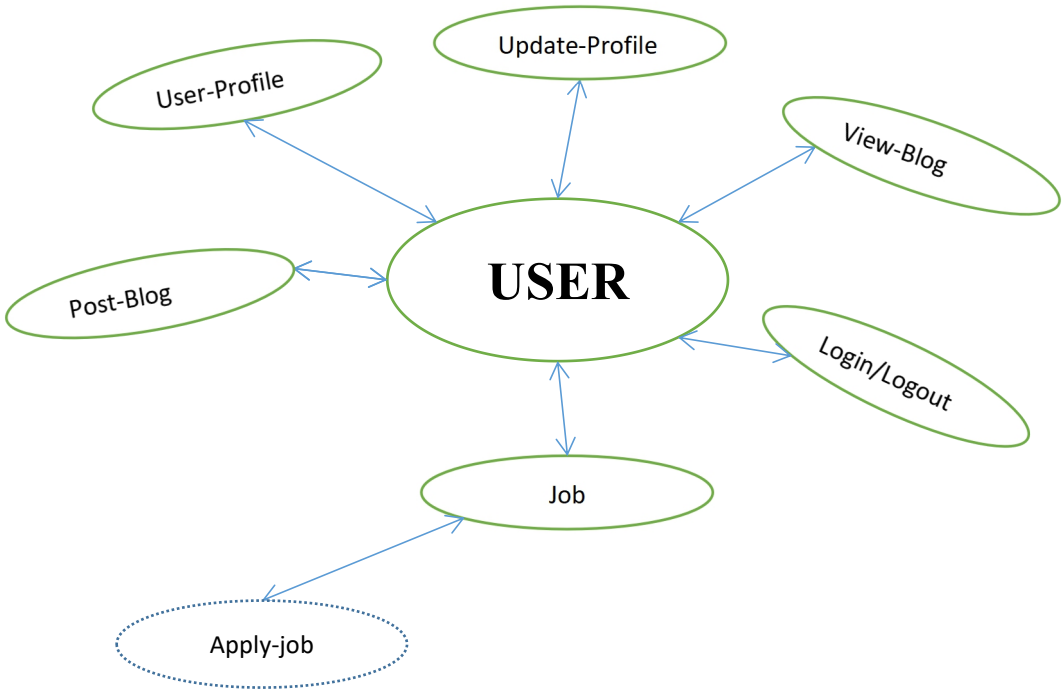
HIGH LEVEL USER CASE DIAGRAMS



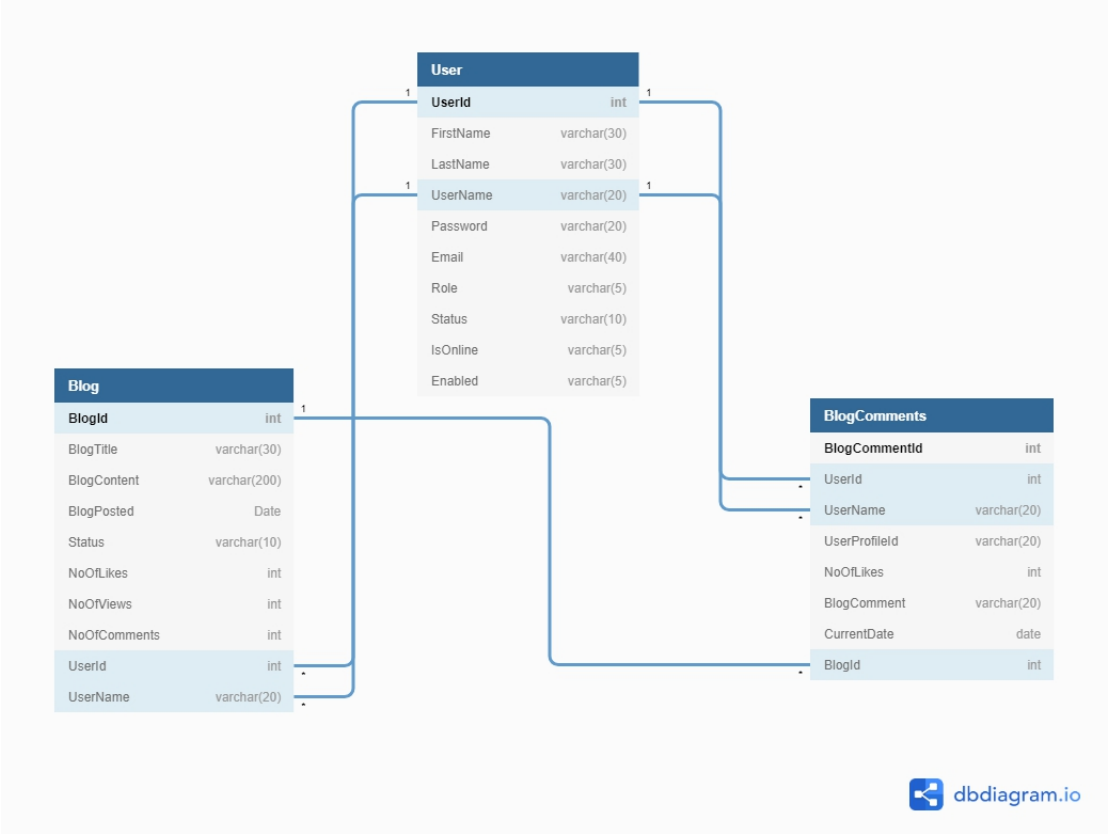
LOW LEVEL USER CASE DIAGRAMS






Update-Profile
User-Profile
View-Blog
USER
Post-Blog
Login/Logout
Job
Apply-job

DATABASE STRUCTURE



MySQL download-install-setup:
https://dev.mysql.com/downloads/file/?id=497106
https://www.mysqltutorial.org/install-mysql/#:~:text=Install%20MySQL%20via%20MySQL%20Installer&text=Install%20MySQL%20Step%203%20%E2%80%93%20Download,server%2C%20MySQL%20Workbench%2C%20etc.&text=Install%20MySQL%20Step%205%20%E2%80%93%20Choosing,are%20several%20setup%20types%20available.
https://www.youtube.com/watch?v=X_umYKqKaF0

create database collaboration;

use collaboration;

```
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 varchar(5),
Enabled varchar(5),
primary key(UserId));

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 table BlogComments(
BlogCommentId int not null auto_increment,
UserId int,
UserName varchar(20),
UserProfileId varchar(20),
NoOfLikes int,
BlogComment varchar(20),
CurrentDate date,
BlogId int,
primary key(BlogCommentId),
    );
```

Project:

1. Create a SpringBoot project named OnlineCollaboration 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 and specify the required configuration in it. However, there is one more configuration class OnlineCollaborateApplication.java. This class is provided by Spring Boot automatically.

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>2.4.2</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.coll</groupId>
	<artifactId>OnlineCollaborate</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>OnlineCollaborate</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-mail</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-validation</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </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>
```

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.HibernateJpaAutoConfi
guration;
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.MappingJackson2HttpMessage
Converter;
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.EnableTransactionManagem
ent;
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"),
                       })
@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="krishnanCse@76";

        @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_DIALEC
T);
                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;
   }
}
```

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);
	}

}
```

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

4(a) User.model

```
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;
	}


}
```

5.Create the DAO interface implementation class

5(a) User.dao

```
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();
        List<User> getAllActiveUser();
        boolean updateUserProfile(String file, Integer userId);

    }
```

6.Create the DAO interface implementation class

6(a) User.daoImpl

```
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 wheere 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 getUserByName(String username) {
            String query="from User where username=:username";
            return
sessionFactory.getCurrentSession().createQuery(query,User.class).setPar
ameter("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;
            }
        }

        @Override
        public List<User> getAllActiveUser() {
             return
sessionFactory.getCurrentSession().createQuery("from User where
enabled='true'",User.class).getResultList();

        }
}
```

7.Create the service layer interface

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

7(a) User.service

```
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();
	List<User> getAllActiveUser();
	boolean updateUserProfile(String file, Integer userId);

}
```

8.Create the service layer implementation class

8(a) User.serviceImpl

```
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);
	}

	@Override
	public List<User> getAllActiveUser() {
		// TODO Auto-generated method stub
		return userDao.getAllActiveUser();
	}
}
```

9.Create the controller class

9(a) User.controller

```
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);
	}

	@GetMapping("active-list")
	public List<User> AllActiveUser(){
		return userService.getAllActiveUser();
	}

	@PostMapping("deactive-user/{userId}")
	public boolean deactiveUser(@RequestBody User user,
@PathVariable("userId") int userId) {
```

```
		return userService.deactiveUser(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);
	}
}
```

4(b)Blog.model

```
package com.coll.OnlineCollaborate.model;

import java.io.Serializable;
import java.time.LocalDate;
import java.util.List;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToMany;

import org.springframework.stereotype.Component;

import com.fasterxml.jackson.annotation.JsonIgnore;
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;
	}
}
```

5(b) Blog.dao

```
package com.coll.OnlineCollaborate.dao;

import java.util.List;

import com.coll.OnlineCollaborate.model.Blog;

public interface IBlogDao {

	List<Blog> getAllBlogs();
	List<Blog> getBlogsByStatus(String status);
	List<Blog> getUsersBlogs(int id);
	Blog getBlogById(int blogId);
	boolean addBlog(Blog blog);
	boolean updateBlog(Blog blog);
	boolean deleteBlog(int blogId);
}
```

6(b) Blog.daoImpl

```
package com.coll.OnlineCollaborate.daoImpl;

import java.util.List;

import org.hibernate.SessionFactory;
import org.hibernate.query.Query;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;

import com.coll.OnlineCollaborate.dao.IBlogDao;
import com.coll.OnlineCollaborate.model.Blog;


@Repository("blogDao")
@Transactional
public class BlogDaoImpl implements IBlogDao {

	@Autowired
	SessionFactory sessionFactory;
```

```
	@Override
	public List<Blog> getAllBlogs() {

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

	@Override
	public List<Blog> getBlogsByStatus(String status) {
		String q="from Blog where status='"+status+"'";
		Query
query=sessionFactory.getCurrentSession().createQuery(q);
		return query.getResultList();
	}

	@Override
	public List<Blog> getUsersBlogs(int blogId) {
		// TODO Auto-generated method stub
		return null;
	}

	@Override
	public Blog getBlogById(int blogId) {
		return
sessionFactory.getCurrentSession().get(Blog.class,Integer.valueOf(blogId)
);
	}

	@Override
	public boolean addBlog(Blog blog) {
		try
		{
			sessionFactory.getCurrentSession().save(blog);
			return true;
		}
		catch(Exception ex)
		{
			ex.printStackTrace();
			return false;
		}
	}
```

```
@Override
public boolean updateBlog(Blog blog) {
	try
	{
		sessionFactory.getCurrentSession().update(blog);
		return true;
	}
	catch(Exception ex)
	{
		ex.printStackTrace();
		return false;
	}
}

@Override
public boolean deleteBlog(Blog blogId) {
	try
	{
		sessionFactory.getCurrentSession().delete(blogId);
		return true;
	}
	catch(Exception ex)
	{
		ex.printStackTrace();
		return false;
	}
}

@Override
public List<Blog> mainList() {
	// TODO Auto-generated method stub
	return null;
}}
```

7(b) Blog.service

```
package com.coll.OnlineCollaborate.service;

import java.util.List;

import com.coll.OnlineCollaborate.model.Blog;

public interface IBlogService {

	List<Blog> getAllBlogs();
	List<Blog> getBlogsByStatus(String status);
	List<Blog> getUsersBlogs(int blogId);
	List<Blog> mainList();
	Blog getBlogById(int blogId);
	boolean addBlog(Blog blog);
	boolean updateBlog(Blog blog);
	boolean deleteBlog(Blog blogId);
}
```

8(b) Blog.serviceImpl

```
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 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 blogId) {
```

```
		return blogDao.deleteBlog(blogId);
	}
}
```

9(b) Blog.controller

```
package com.coll.OnlineCollaborate.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.coll.OnlineCollaborate.model.Blog;
import com.coll.OnlineCollaborate.service.IBlogService;


@RestController
@CrossOrigin(origins="http://localhost:4200")
@RequestMapping(value="/api")
public class BlogController {

	@Autowired
	IBlogService blogService;

	@PostMapping("save-blog")
	public boolean saveBlog(@RequestBody Blog blog)
	{
		return blogService.addBlog(blog);
	}

	@GetMapping("blog-list")
	public List<Blog> allBlog()
	{
```

```
		return blogService.getAllBlogs();
	}

	@DeleteMapping("delete-blog/{blog_id}")
	public boolean deleteBlog(@PathVariable("blog_id") Blog blog_id)
	{
		return blogService.deleteBlog(blog_id);
	}

	@GetMapping("blog/{blog_id}")
	public Blog blogById(@PathVariable("blog_id") int blog_id)
	{
		return blogService.getBlogById(blog_id);
	}

	@PostMapping("update-blog/{blog}")
	public boolean updateBlog(@PathVariable("blog") Blog blog)
	{
		return blogService.updateBlog(blog);
	}
}
```

4(c) Blogcomments.model

```
package com.coll.OnlineCollaborate.model;

import java.io.Serializable;
import java.time.LocalDate;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;

import org.springframework.stereotype.Component;

import com.fasterxml.jackson.annotation.JsonBackReference;

@Component
```

```
@Entity
public class BlogComments implements Serializable {

	private static final long serialVersionUID = 1L;
	@Id
	@GeneratedValue(strategy=GenerationType.IDENTITY)
	int blogCommentId;
	int userId;
	String username;
	String userProfileId;
	String title;
	int noOfLikes;
	String blogComment;
	LocalDate currentDate;
	@ManyToOne
	@JoinColumn(name="BlogId")
	@JsonBackReference
	Blog blog;

	public int getBlogCommentId() {
		return blogCommentId;
	}
	public void setBlogCommentId(int blogCommentId) {
		this.blogCommentId = blogCommentId;
	}
	public int getUserId() {
		return userId;
	}
	public void setUserId(int userId) {
		this.userId = userId;
	}
	public String getUsername() {
		return username;
	}
	public void setUsername(String username) {
		this.username = username;
	}
	public String 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;
	}
}
```

5(c) Blogcomments.dao

```
package com.coll.OnlineCollaborate.dao;

import java.util.List;

import com.coll.OnlineCollaborate.model.BlogComments;

public interface IBlogCommentsDao {

	List<BlogComments> getAllBlogComments();
	BlogComments getBlogCommentsById(int blogCommentId);
	boolean addBlogComments(BlogComments blogComments);
	boolean updateBlogComments(BlogComments blogComments);
	boolean deleteBlogComments(BlogComments blogCommentId);
}
```

6(c) Blogcomments.daoImpl

```
package com.coll.OnlineCollaborate.daoImpl;

import java.util.List;

import org.hibernate.SessionFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;

import com.coll.OnlineCollaborate.dao.IBlogCommentsDao;
import com.coll.OnlineCollaborate.model.BlogComments;

@Repository("blogCommentsDao")
@Transactional
public class BlogCommentsDaoImpl implements IBlogCommentsDao{

	@Autowired
	SessionFactory sessionFactory;

	@Override
	public List<BlogComments> getAllBlogComments() {
```

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

	@Override
	public BlogComments getBlogCommentsById(int blogCommentId)
{
		return
sessionFactory.getCurrentSession().get(BlogComments.class,Integer.valu
eOf(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
blogCommentId) {
		try
		{

	sessionFactory.getCurrentSession().delete(blogCommentId);
			return true;
		}
		catch(Exception ex)
		{
			ex.printStackTrace();
			return false;
		}
	}
}
```

7(c) Blogcomments.service

```
package com.coll.OnlineCollaborate.service;

import java.util.List;

import com.coll.OnlineCollaborate.model.BlogComments;

public interface IBlogCommentsService {

	List<BlogComments> getAllBlogComments();
	BlogComments getBlogCommentsById(int blogCommentId);
	boolean addBlogComments(BlogComments blogComments);
	boolean updateBlogComments(BlogComments blogComments);
	boolean deleteBlogComments(BlogComments blogCommentId);
}
```

8(c) Blogcomments.serviceImpl

```
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 blogCommentId)
{

		return
blogCommentsDao.getBlogCommentsById(blogCommentId);
	}

	@Override
	public boolean addBlogComments(BlogComments blogComments)
{
```

```
		return
blogCommentsDao.addBlogComments(blogComments);
	}

	@Override
	public boolean updateBlogComments(BlogComments
blogComments) {

		return
blogCommentsDao.updateBlogComments(blogComments);
	}

	@Override
	public boolean deleteBlogComments(BlogComments
blogCommentId) {

		return
blogCommentsDao.deleteBlogComments(blogCommentId);
	}
}
```

9(c) Blogcomments.controller

```
package com.coll.OnlineCollaborate.controller;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import com.coll.OnlineCollaborate.model.BlogComments;
import com.coll.OnlineCollaborate.service.IBlogCommentsService;

@RestController
@CrossOrigin(origins="http://localhost:4200")
@RequestMapping(value="/api")
```

```
public class BlogCommentsController {

	@Autowired
	IBlogCommentsService blogCommentsService;

	@PostMapping("save-blogComments")
	public boolean saveBlogComments(@RequestBody
BlogComments blogComments)
	{
		return
blogCommentsService.addBlogComments(blogComments);
	}

	@GetMapping("blogComments-list")
	public List<BlogComments> allBlogComments()
	{
		return blogCommentsService.getAllBlogComments();
	}

	@DeleteMapping("delete-blogComments/{blogComment_id}")
	public boolean
deleteBlogComments(@PathVariable("blogComment_id")
BlogComments blogComment_id)
	{
		return
blogCommentsService.deleteBlogComments(blogComment_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 updateBlog(@PathVariable("blogComments")
BlogComments blogComments)
	{
		return
blogCommentsService.updateBlogComments(blogComments);
```

```
        }
}
```

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

 # Database

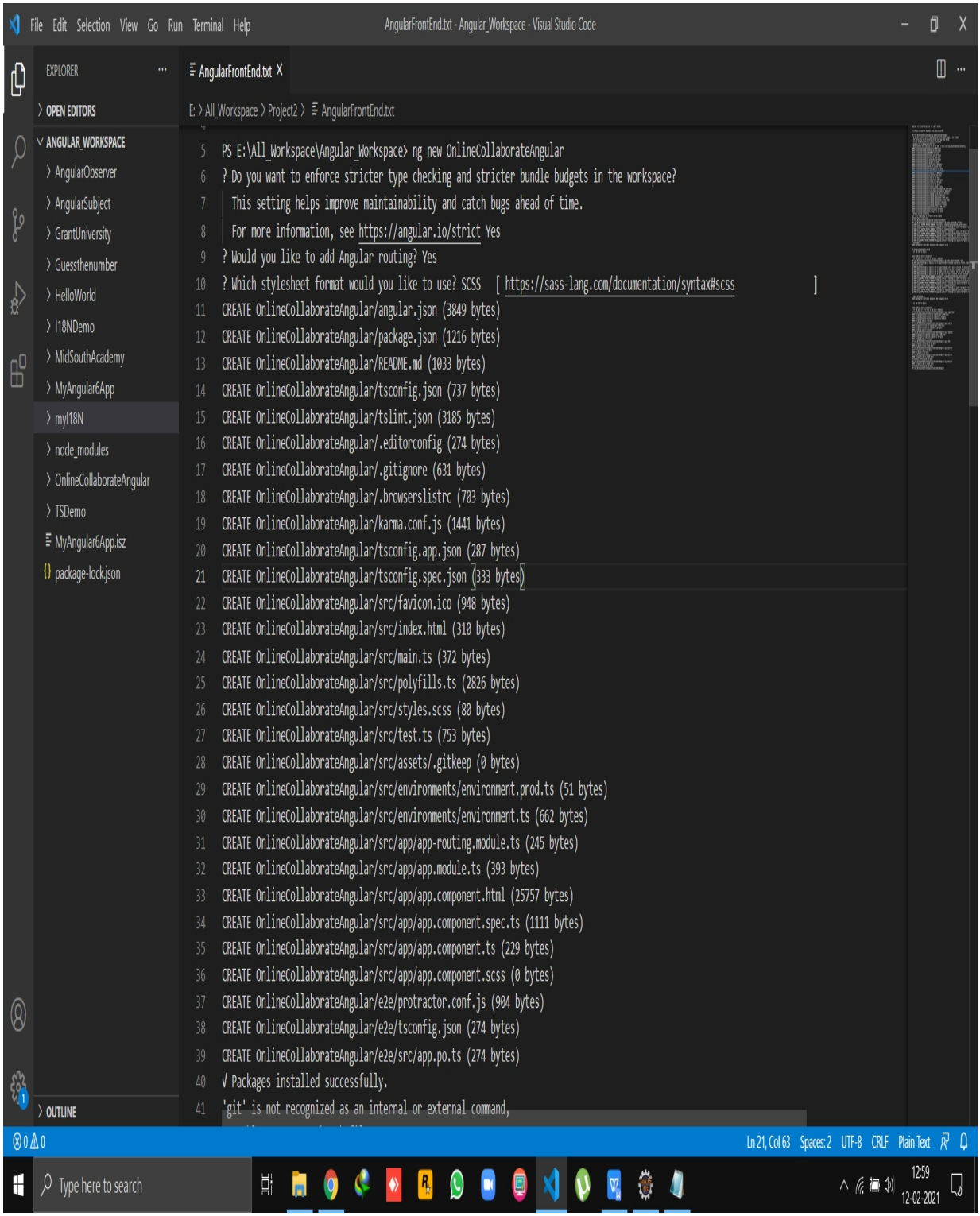
```
public static final String
DATABASE_URL="jdbc:mysql://localhost:3306/collaboration";
        public static final String
DATABASE_DRIVER="com.mysql.cj.jdbc.Driver";
        public static final String
DATABASE_DIALECT="org.hibernate.dialect.MySQLDialect";
        public static final String DATABASE_USERNAME="root";
        public static final String
DATABASE_PASSWORD="krishnanCse@76";
```

 # Hibernate

```
sessionFactory.setPackagesToScan("com.coll.OnlineCollaborate");
                Properties hibernateProperties=new Properties();

        hibernateProperties.put("hibernate.dialect",DATABASE_DIALECT);
        hibernateProperties.put("hibernate.show_sql","true");
        hibernateProperties.put("hibernate.hbm2dll.auto","update");
        sessionFactory.setHibernateProperties(hibernateProperties);
```

11. Open Visual Studio Code. Create new project inside Angular_Workspace. Set the project name as OnlineCollaborateAngular

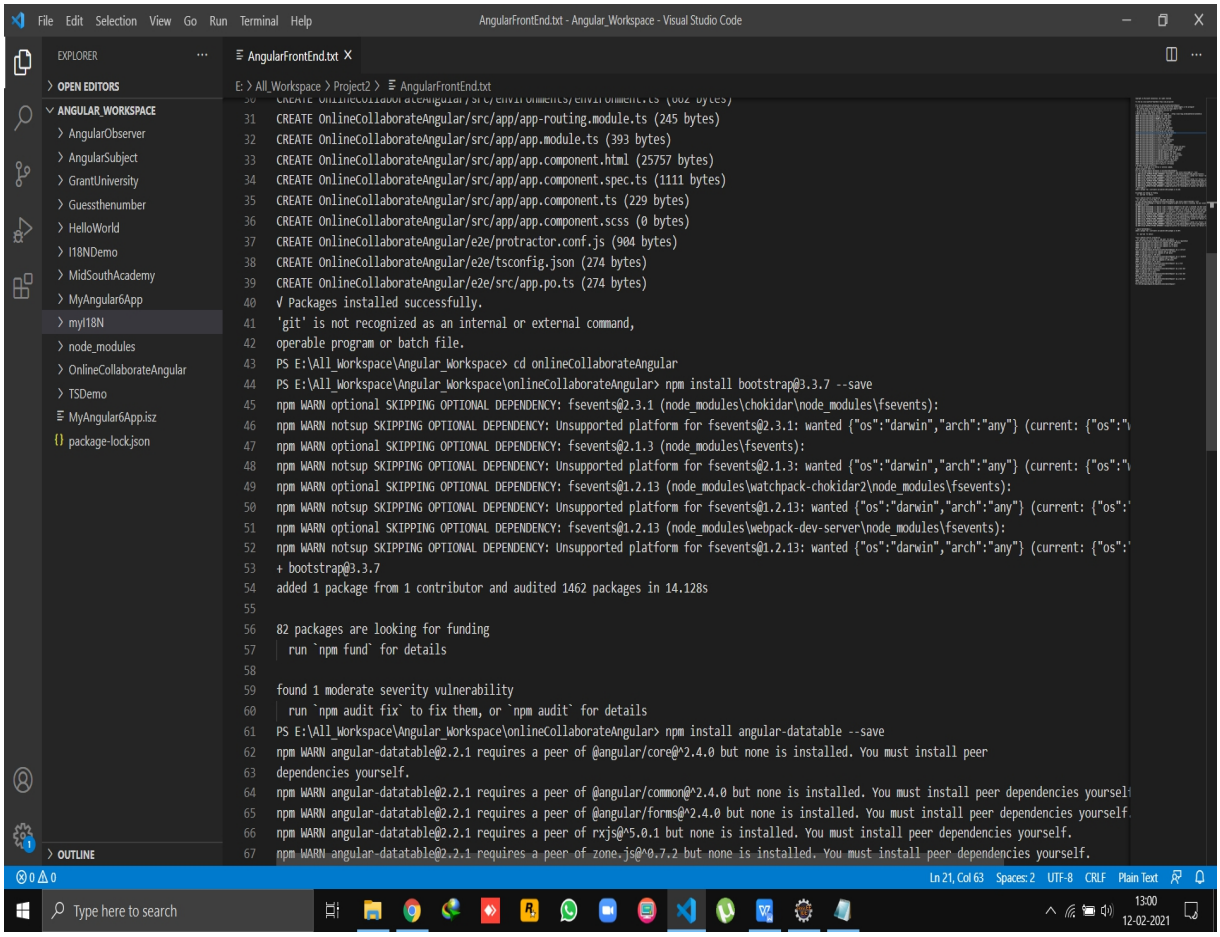
ng new OnlineCollaborateAngular



12.Install Bootstrap CSS framework

Use the following command to install bootstrap in the project.

E:\All_Workspace\Angular_Workspace\OnlineCollaborateAngular > ***npm install bootstrap@3.3.7 --save***



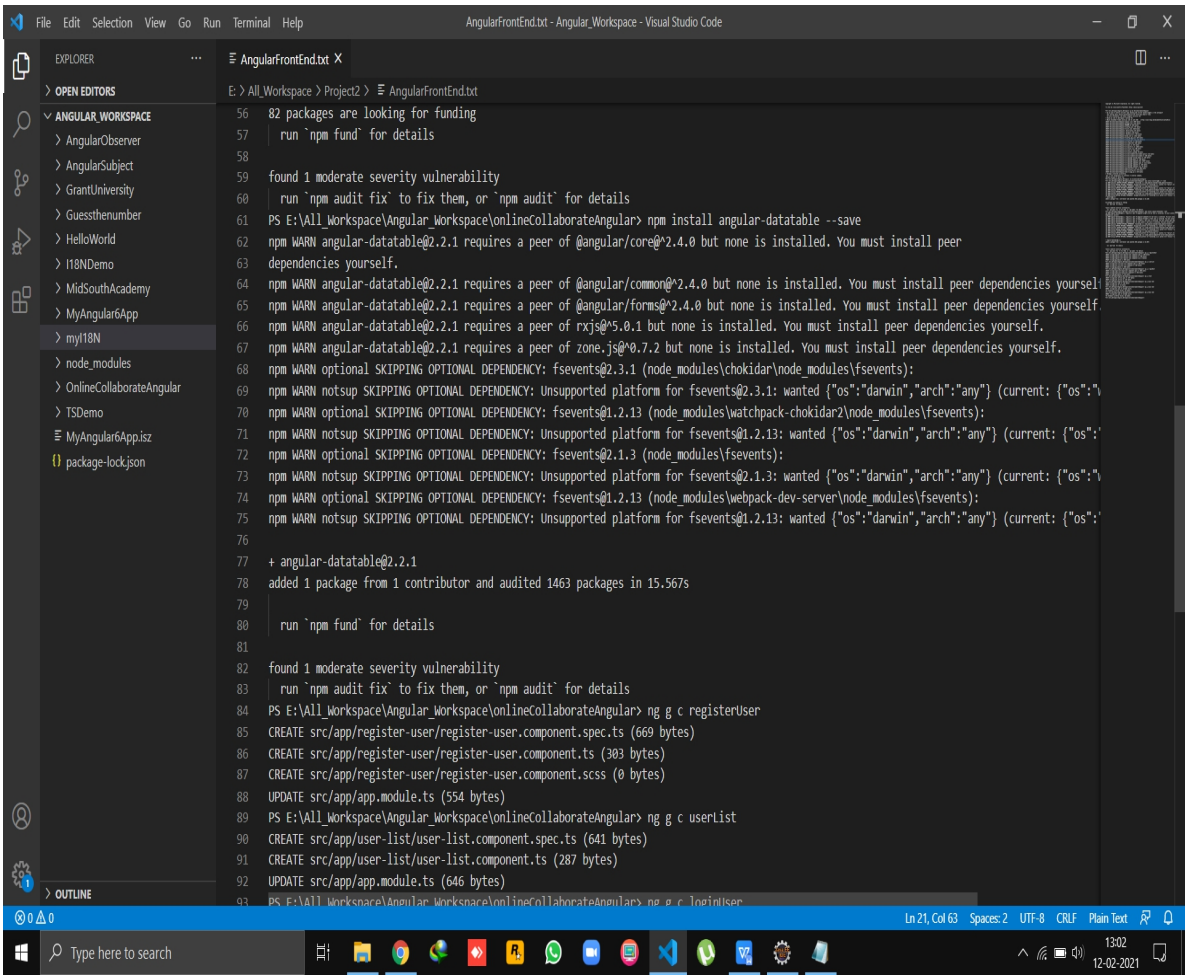
13.Now, include the following code in the style.css file.

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

14.Install Angular Data Table

Use the following command to install bootstrap in the project.

E:\All_Workspace\Angular_Workspace\OnlineCollaborateAngular > ***npm install angular-datatable --save***



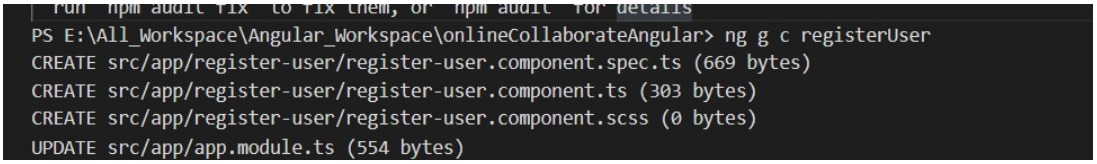
15. It is required to include **DataTablesModule** in imports array of **app.module.ts** file.

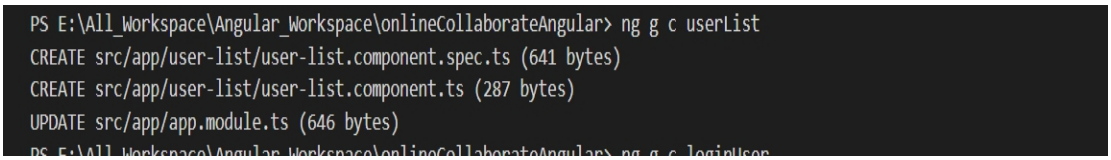
16. Generate Components
Open the project in visual studio and then use the following command to generate Angular components:

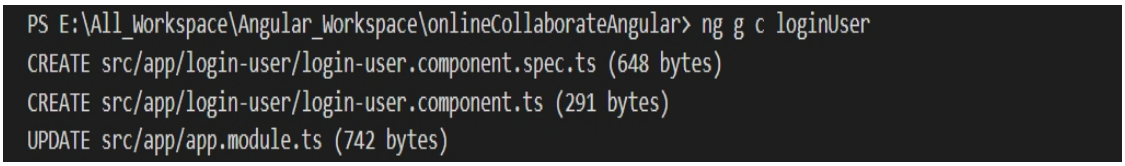
ng g c registerUser

ng g c userList

ng g c loginUser

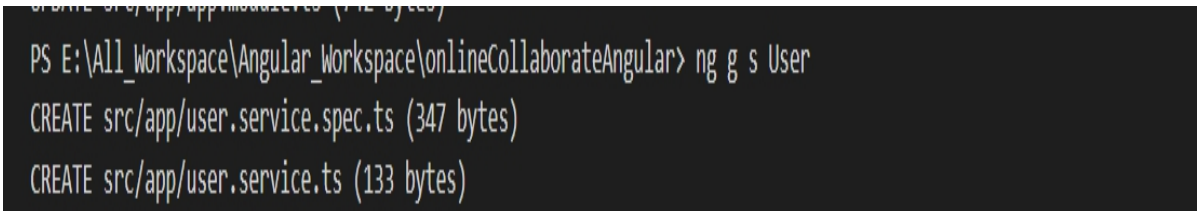






17. Let's also create a service class by using the following command: -

ng g s User



18. Edit the **app.module.ts** file
    a. **Import Routing** - Here, we are importing routing file (app-routing.module.ts) and include it in imports array.
    b. **Import ReactiveFormsModule** - Here, we are importing **ReactiveFormsModule** for reactive forms and specify it in imports array.
    c. **Import HttpModule** - Here, we are importing **HttpModule** for server requests and specifying it in imports array.
    d. **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 { NavigationComponent } from
'./navigation/navigation.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 { NavigationUserComponent } from './navigation-user/navigation-user.component';
import { AboutUsComponent } from './about-us/about-us.component';
import { EditorModule } from '@tinymce/tinymce-angular';
import { DeactivateUserComponent } from './deactivate-user/deactivate-user.component';
import { AddPostComponent } from './add-post/add-post.component';
import {MatProgressBarModule} from "@angular/material/progress-bar";
import { UpdateUserprofileComponent } from './update-userprofile/update-userprofile.component';

@NgModule({
  declarations: [
    AppComponent,
    RegisterUserComponent,
    UserListComponent,
    LoginUserComponent,
    NavigationComponent,
    UserProfileComponent,
    BlogListComponent,
    ActiveUserComponent,
    HomeComponent,
    NavigationUserComponent,
    AboutUsComponent,
    DeactivateUserComponent,
    AddPostComponent,
    UpdateUserprofileComponent,  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    ReactiveFormsModule,
    HttpClientModule,
    EditorModule,
    MatProgressBarModule,
    DataTablesModule, BrowserAnimationsModule, LayoutModule,
MatToolbarModule, MatButtonModule, MatSidenavModule,
MatIconModule, MatListModule],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

19 . 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 { NavigationComponent } from
'./navigation/navigation.component';
import { NavigationUserComponent } from './navigation-user/navigation-
user.component';
import { AboutUsComponent } from './about-us/about-us.component';
import { DeactivateUserComponent } from './deactivate-user/deactivate-
user.component';
import { AddPostComponent } from './add-post/add-post.component';

const routes: Routes = [
  { path: '', redirectTo: 'home', pathMatch: 'full' },
  {path:'home', component: HomeComponent},
  {path: 'about-us',component:AboutUsComponent},
  {path:'nav/:Id', component:NavigationComponent,
  children: [
    { path: 'user-list', component: UserListComponent },
    { path: 'active-user' , component: ActiveUserComponent},
    { path: 'deactivate-user' , component: DeactivateUserComponent},
  { path: 'blog-list' , component: BlogListComponent},
  { path: 'user-profile/:Id' , component: UserProfileComponent}
  ]
} ,
  {path:'nav-user/:Id', component:NavigationUserComponent,
   children:[
    { path: 'add-post' , component: AddPostComponent},
    { path: 'blog-list' , component: BlogListComponent},
    { path: 'user-profile/:Id' , component: UserProfileComponent},
   ]
```

```
},
  { path: 'register-user', component: RegisterUserComponent },
  { path: 'login-user' , component: LoginUserComponent} ,];
@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

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

```
<router-outlet>

</router-outlet>
```

app.component.specs.ts

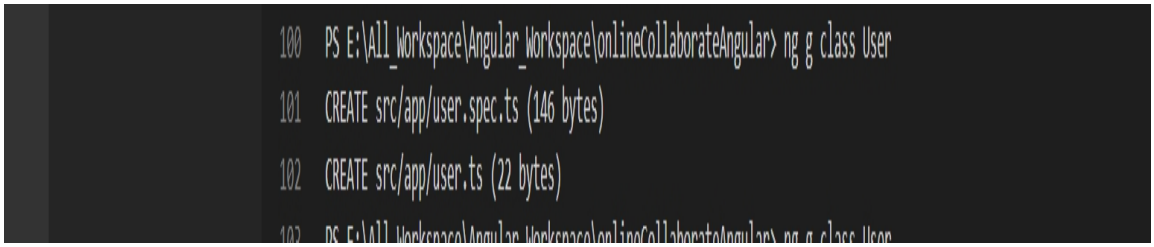
```
import { TestBed } from '@angular/core/testing';
import { RouterTestingModule } from '@angular/router/testing';
import { AppComponent } from './app.component';

describe('AppComponent', () => {
  beforeEach(async () => {
    await TestBed.configureTestingModule({
      imports: [
        RouterTestingModule
      ],
      declarations: [
        AppComponent
      ],
    }).compileComponents();
  });

  it('should create the app', () => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.componentInstance;
    expect(app).toBeTruthy();
  });

  it(`should have as title 'OnlineCollaborateAngular'`, () => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.componentInstance;
```

```
    expect(app.title).toEqual('OnlineCollaborateAngular');
  });

  it('should render title', () => {
    const fixture = TestBed.createComponent(AppComponent);
    fixture.detectChanges();
    const compiled = fixture.nativeElement;
    expect(compiled.querySelector('.content
span').textContent).toContain('OnlineCollaborateAngular app is
running!');
  });
});
```

21 .Create the **User.ts** class

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

ng g class User



22 . 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 {
    user_id!: number;
        first_name!: String;
        last_name!: String;
        username!: String;
        password!: String;
        confirm_password!: String;
        email!: String;
        role!: String;
        status!: String;
        isOnline!: boolean;
```

```
    enabled!: boolean;

}
```

23 . 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'});
              }
updateUserProfile(userId: number, value: any): Observable<Object> {
                return this.http.post(`${this.baseUrl}/update-
userprofile/${userId}`, value);
                  }
}
```

24 .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 ] ),
lastName:new FormControl(" , [Validators.required ] ),
username:new FormControl(" , [Validators.required  ] ),
password:new FormControl(" , [Validators.required ] ),
confirm_password:new FormControl(" , [Validators.required ] ),
email:new FormControl(",[Validators.required,Validators.email]),
role:new FormControl()
 });

 register()
 {
  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');
  }


  registrationForm(){
    this.submitted=false;
    this.registrationform.reset();
  }

  logInUser() {
    $(".pages").css("visibility","visible");
    $(".peru").css("visibility","hidden");
  }
}
```

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

```
<h3 style="text-align: center">Register User</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()">
          <div class="form-group">
              <label for="firstName">First Name</label>
              <input type="text" class="form-control"
formControlName="firstName" data-toggle="tooltip"
```

```
            data-placement="right" title="Enter First Name"
placeholder="Enter Your FirstName">
          <div class="alert alert-danger" *ngIf = "(FirstName.touched)
&& (FirstName.invalid)"
            style="margin-top: 5px;">
              <span *ngIf="FirstName.errors.required">First Name 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" title="Enter Last Name"
placeholder="Enter Your LastName">
          <div class="alert alert-danger" *ngIf = "(LastName.touched) &&
(LastName.invalid)"
            style="margin-top: 5px;">
              <span *ngIf="LastName.errors.required">LastName 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" title="Enter Username"
placeholder="Enter Your Username">
          <div class="alert alert-danger" *ngIf = "(Username.touched) &&
(Username.invalid)"
            style="margin-top: 5px;">
              <span *ngIf="Username.errors.required">Username 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" title="Enter Password"
placeholder="Enter Your Password">
            <div class="alert alert-danger" *ngIf = "(Password.touched) &&
(Password.invalid)"
                style="margin-top: 5px;">
                <span *ngIf="Password.errors.required">Password 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" title="Enter Confirm Password"
placeholder="Enter Your Confirm Password">
            <div class="alert alert-danger" *ngIf =
"(ConfirmPassword.touched) && (ConfirmPassword.invalid)"
                style="margin-top: 5px;">
                <span *ngIf="ConfirmPassword.errors.required">Confirm
Password is Required</span>
                <span *ngIf="ConfirmPassword.errors.pattern">Password and
Confirm Password does not match.</span>
                <span *ngIf = "ConfirmPassword.errors.minlength">
MinLength Error </span>
            </div>
        </div>

        <div class="form-group">
            <label for="email">Email</label>
            <input type="text" class="form-control"
formControlName="email"
                data-toggle="tooltip" data-placement="right" title="Enter
Email Id" placeholder="Enter Your Email Id">
```

```
            <div class="alert alert-danger" *ngIf = "(Email.touched) &&
(Email.invalid)"
            style="margin-top: 5px;">
              <span *ngIf="Email.errors.required">Email is
Required</span>
                <span *ngIf = "Email.errors.email">
                   Invalid Email Format
                </span>
            </div>
        </div>

        <div class="form-group">
           <label for="role">Role</label>
            <select class="form-control" formControlName="role" data-
toggle="tooltip"
                 data-placement="right" title="Select 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 [hidden]="!submitted">
      <h4 style="font-weight: bold; font-style: italic; text-align: center;
color:green;">Congratulations..!!! You have Registered
successFully..!!</h4>
</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: 300px;
    left: 150%;
    padding: 20px;
    position:relative;
    text-align:center;
    transition: 0.25s;
    margin-top: 125px
}

.row input[type="text"],
.row input[type="password"],
.row select {
    border: 0;
    background: none;
    display: block;
    margin: 10px auto;
    text-align: center;
    border: 3px solid #3498db;
    padding: 5px 10px;
    width: 200px;
    outline: none;
    color: white;
    border-radius: 25px;
    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:palegoldenrod;
}
.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:rebeccapurple
}

::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:10%;
     visibility: hidden;

}
.form-control{
    width:100%;
    border-radius:5px;
    border: 2px solid rgba(76, 46, 207, 0.856);
    font-size: 13px;
    font-family:monospace;

    }
```

26 .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:any;
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:any) => 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;
  }


}
```

27 . 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>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.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)="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;

}
```

28 **. 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
```

29 .**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"
          ]
```

30 .**Save All.**

31 .**Run the SpringBoot Application**

32 .**Run the Angular project**

Login -User Component

Login-user.component.ts

```
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");
  }

}
```

Login-user.component.html

```
<div class="login-box">
  <h2>Login</h2>
  <form [formGroup]="loginform"  (ngSubmit)="validateUser()">
    <!--Username-->
    <div class="user-box">
      <input type="text" class="form-control"
formControlName="username" data-toggle="tooltip"
             data-placement="right" title="Enter Username"
placeholder="Username">
      <div class="alert alert-danger" *ngIf = "(Username.touched) &&
(Username.invalid)"
            style="margin-top: 5px;">
              <span *ngIf="Username.errors.required">Username is
Required</span>
            </div>
    </div>

    <!--Password-->
    <div class="user-box">
      <input type="password" class="form-control"
formControlName="password" data-toggle="tooltip"
      data-placement="right" title="Enter Password"
placeholder="Password">
   <div class="alert alert-danger" *ngIf = "(Password.touched) &&
(Password.invalid)"
     style="margin-top: 5px;">
       <span *ngIf="Password.errors.required">Password is
Required</span>
     </div>
    </div>

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

  </form>
</div>
```

Login-user.component.scss

```
html {
  height: 100%;
}
body {
  margin:0;
  padding:0;
  font-family: sans-serif;
  background: linear-gradient(#141e30, #243b55);
}

.login-box {
  position:fixed;
  top: 50%;
  right: 50%;
  width: 400px;
  padding: 40px;
  transform: translate(-50%, -50%);
  background: rgba(6, 1, 31, 0.938);
  box-sizing: border-box;
  box-shadow: 0 15px 25px rgba(0,0,0,.6);
  border-radius: 20px;
}

.login-box h2 {
  margin: 0 0 30px;
  padding: 0;
  color: #fff;
  text-align: center;
}

.login-box .user-box {
  position: relative;
}

.login-box .user-box input {
  width: 100%;
  padding: 10px 0;
  font-size: 16px;
  color: #fff;
  margin-bottom: 30px;
  border: none;
  border-bottom: 1px solid #fff;
```

```
  outline: none;
  background: transparent;
}
.login-box .user-box label {
  position: absolute;
  top:0;
  left: 0;
  padding: 10px 0;
  font-size: 16px;
  color: #fff;
  pointer-events: none;
  transition: .5s;
}

.login-box .user-box input:focus ~ label,
.login-box .user-box input:valid ~ label {
  top: -20px;
  left: 0;
  color: #03e9f4;
  font-size: 12px;
}

.login-box form a {
  position: relative;
  display: inline-block;
  padding: 10px 20px;
  color: #03e9f4;
  font-size: 16px;
  text-decoration: none;
  text-transform: uppercase;
  overflow: hidden;
  transition: .5s;
  margin-top: 40px;
  letter-spacing: 4px
}

.login-box a:hover {
  background: #03e9f4;
  color: #fff;
  border-radius: 5px;
  box-shadow: 0 0 5px #03e9f4,
              0 0 25px #03e9f4,
              0 0 50px #03e9f4,
```

```
            0 0 100px #03e9f4;
}

.login-box a span {
  position: absolute;
  display: block;
}

.login-box a span:nth-child(1) {
  top: 0;
  left: -100%;
  width: 100%;
  height: 2px;
  background: linear-gradient(90deg, transparent, #03e9f4);
  animation: btn-anim1 1s linear infinite;
}

@keyframes btn-anim1 {
  0% {
    left: -100%;
  }
  50%,100% {
    left: 100%;
  }
}

.login-box a span:nth-child(2) {
  top: -100%;
  right: 0;
  width: 2px;
  height: 100%;
  background: linear-gradient(180deg, transparent, #03e9f4);
  animation: btn-anim2 1s linear infinite;
  animation-delay: .25s
}

@keyframes btn-anim2 {
  0% {
    top: -100%;
  }
  50%,100% {
    top: 100%;
  }
```

```
}

.login-box a span:nth-child(3) {
  bottom: 0;
  right: -100%;
  width: 100%;
  height: 2px;
  background: linear-gradient(270deg, transparent, #03e9f4);
  animation: btn-anim3 1s linear infinite;
  animation-delay: .5s
}

@keyframes btn-anim3 {
  0% {
    right: -100%;
  }
  50%,100% {
    right: 100%;
  }
}

.login-box a span:nth-child(4) {
  bottom: -100%;
  left: 0;
  width: 2px;
  height: 100%;
  background: linear-gradient(360deg, transparent, #03e9f4);
  animation: btn-anim4 1s linear infinite;
  animation-delay: .75s
}

@keyframes btn-anim4 {
  0% {
    bottom: -100%;
  }
  50%,100% {
    bottom: 100%;
  }
}
```

Active User Component

Active-user.component.ts

```
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:any;
  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:number){
    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');
  }

}
```

Active-user.component.html

```
<h1 style="font-weight: bold; font-style: italic; text-align: center; ">Users Deactive 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 Activated</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 }}</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">Activate</button>    </td>
               </tr>
            </tbody>
            </table>
         </div>
   </div>
</div>
```

Active-user.component.scss

```
.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;

  }
```

Deactivate User Component

Deactivate-user.component.ts

```
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-deactivate-user',
  templateUrl: './deactivate-user.component.html',
  styleUrls: ['./deactivate-user.component.scss']
})
export class DeactivateUserComponent implements OnInit {

  dtOptions: DataTables.Settings={};
  dtTrigger: Subject<any>= new Subject();

  constructor(private userservice: UserService) { }

  users:any;
  user: User =new User();
  activeList: any;
  isEnabled= true;


  ngOnInit(){
   this.isEnabled= true;
   this.dtOptions={
     pageLength: 6,
     stateSave: true,
     lengthMenu: [[6, 16, 20, -1], [6, 16, 20, "All"]],
     processing: true
   };
   this.userservice.activeList().subscribe((data: any) =>{
     this.users =data;
     this.dtTrigger.next();
   })
  }

  enabledUser(id: number){
```

```
  this.userservice.deactiveUser(id).
  subscribe(
    (data: any) =>{
      this.activeList =data;
      console.log(this.activeList);
      this.deactiveUser(id);

    },
    (error: any) => console.log(error)
  );
}

deactiveUserForm = new FormGroup({
  userId: new FormControl()
});

deactiveUser(id:number){
  this.user =new User();
  this.user.userId=id;
  this.userservice.deactiveUser(this.user.userId).subscribe(
    (    data: any) => {
    this.userservice.activeList().subscribe( (data: any) =>{
      this.users =data
      console.log(this.users)
    } )
  },
    (    error: any) => console.log(error));
}

get UserId() {
  return this.deactiveUserForm.get('userId');
}

}
```

Deactivate-user.component.html

```
<h1 style="font-weight: bold; font-style: italic; text-align: center; ">Users
Deactive List</h1><br>
<div class="panel">
   <div class="panel panel-default">



         <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 }}</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">Activate</button>  
 </td>
               </tr>

            </tbody>
            </table>
         </div>
   </div>
</div>
```

Deactivate-user.component.scss

```
.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;

  }
```

Deactivate-user.component.specs.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { DeactivateUserComponent } from './deactivate-user.component';

describe('DeactivateUserComponent', () => {
  let component: DeactivateUserComponent;
  let fixture: ComponentFixture<DeactivateUserComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
```

```
    declarations: [ DeactivateUserComponent ]
   })
   .compileComponents();
 });

 beforeEach(() => {
   fixture = TestBed.createComponent(DeactivateUserComponent);
   component = fixture.componentInstance;
   fixture.detectChanges();
 });

 it('should create', () => {
   expect(component).toBeTruthy();
 });
});
```

User-profile component

User-profile.component.ts

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

```
 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);
        }
   )
    }
   )
}
}
```

User-profile.component.html

```
<div>
   <div class="main">
     <div class="page-content page-container" id="page-content">
        <div class="padding">
           <div class="row container d-flex justify-content-center">
              <div class="col-xl-6 col-md-12">
                 <div class="card user-card-full">
                    <div class="row m-l-0 m-r-0">
                       <div class="col-sm-4 bg-c-lite-green user-profile">
                          <div class="card-block text-center text-white">
                             <div class="m-b-25"> <img
src="https://img.icons8.com/bubbles/100/000000/user.png" class="img-
radius" alt="User-Profile-Image"> </div>
                             <h6 class="f-w-600">{{user.username}}</h6>
                             <p>{{user.role}}</p> <i class=" mdi mdi-square-
edit-outline feather icon-edit m-t-10 f-16"></i>
                          </div>
                       </div>
                       <div class="col-sm-8">
                          <div class="card-block">
                             <h6 class="m-b-20 p-b-5 b-b-default f-w-
600">Information</h6>
```

```
                                <div class="row">
                                    <div class="col-sm-6">
                                        <p class="m-b-10 f-w-600">Name</p>
                                        <h6 class="text-muted f-w-
400">{{user.firstName +" "+ user.lastName}}</h6>
                                    </div>
                                    <div class="col-sm-6">
                                        <p class="m-b-10 f-w-600">Email</p>
                                        <h6 class="text-muted f-w-
400">{{user.email}}</h6>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>   </div   </div>  </div>
```

User-profile.component.scss

```
.main{
   position:relative;
  left:95px;
}
body {
   background-color: #f9f9fa;
   position: static;
   left:100px;
}

.padding {
   padding: 3rem !important
}

.user-card-full {
   overflow: hidden
}
 h6:hover{
   color:blue;
   font-weight: bold;
}
p:hover{
   color:blue;
```

```
    font-weight: bold;
}

.card {
    border-radius: 5px;
    -webkit-box-shadow: 0 1px 20px 0 rgba(69, 90, 100, 0.08);
    box-shadow: 0 1px 20px 0 rgba(69, 90, 100, 0.08);
    border: none;
    margin-bottom: 30px
}

.m-r-0 {
    margin-right: 0px
}

.m-l-0 {
    margin-left: 0px
}

.user-card-full .user-profile {
    border-radius: 5px 0 0 5px
}

.bg-c-lite-green {
    background: -webkit-gradient(linear, left top, right top, from(#f29263),
to(#ee5a6f));
    background: linear-gradient(to right, #ee5a6f, #f29263)
}

.user-profile {
    padding: 20px 0
}

.card-block {
    padding: 1.25rem
}

.m-b-25 {
    margin-bottom: 25px
}

.img-radius {
    border-radius: 5px
```

```
}

h6 {
    font-size: 14px
}

.card .card-block p {
    line-height: 25px
}

@media only screen and (min-width: 1400px) {
    p {
        font-size: 14px
    }
}

.card-block {
    padding: 1.25rem
}

.b-b-default {
    border-bottom: 1px solid #e0e0e0
}

.m-b-20 {
    margin-bottom: 20px
}

.p-b-5 {
    padding-bottom: 5px !important
}

.card .card-block p {
    line-height: 25px
}

.m-b-10 {
    margin-bottom: 10px
}

.text-muted {
    color: #919aa3 !important
}
```

```
.b-b-default {
    border-bottom: 1px solid #e0e0e0
}

.f-w-600 {
    font-weight: 600
}

.m-b-20 {
    margin-bottom: 20px
}

.m-t-40 {
    margin-top: 20px
}

.p-b-5 {
    padding-bottom: 5px !important
}

.m-b-10 {
    margin-bottom: 10px
}

.m-t-40 {
    margin-top: 20px
}

.user-card-full .social-link li {
    display: inline-block
}

.user-card-full .social-link li a {
    font-size: 20px;
    margin: 0 10px 0 0;
    -webkit-transition: all 0.3s ease-in-out;
    transition: all 0.3s ease-in-out
}
```

About-us component

About-us.component.ts

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

@Component({
  selector: 'app-about-us',
  templateUrl: './about-us.component.html',
  styleUrls: ['./about-us.component.scss']
})
export class AboutUsComponent implements OnInit {

  constructor() { }

  ngOnInit(): void {
  }

}
```

About-us.component.html

```
<h1 class="logo"><span>Relaxato  </span>City</h1>
<html>
<head>
<style>
p {
  color: navy;
  text-indent: 30px;
  text-transform: uppercase;
}
</style>
</head>
<body>
    <p style="text-align:center">
        Welcome to Relaxato !!!
        <br>
        Relaxato City is an Photography Blogging Website.
        <br>
        Here you can share Photos,vlogs
        <br>
        You can also share your travel experiences with others
    </p>
```

```
</body>
</html>
```

About-us.component.scss

```
.logo{
   font-size: 2.5rem; font-weight: 600; text-transform: uppercase;
   letter-spacing: 10px; line-height: 4rem; margin-top: 20px;
   color:crimson;
}

.logo span{ font-size: 2.5rem;margin-left: 40pc; margin-right: 50px;
color:chocolate; }
```

Add-post Component

Add-post.component.ts

```
import { Component, OnInit } from '@angular/core';
import {BlogService} from '../blog.service';
import { Blog } from '../blog';
import { UserService } from '../user.service';
import { Validators, FormControl, FormGroup, FormBuilder } from
'@angular/forms';
import { DatePipe, getLocaleDateFormat } from '@angular/common';
import { Observable } from 'rxjs';
import { map, shareReplay } from 'rxjs/operators';
import { ActivatedRoute, Params, Router } from '@angular/router';
import { BreakpointObserver, Breakpoints } from '@angular/cdk/layout';


@Component({
  selector: 'app-add-post',
  templateUrl: './add-post.component.html',
  styleUrls: ['./add-post.component.scss']
})
export class AddPostComponent implements OnInit {


  constructor(private blogservice: BlogService,private route:
ActivatedRoute, private userservice: UserService) { }
  blog: Blog =new Blog();
```

```
submitted = false;
user:any;

ngOnInit(): void {

   this.submitted = false;
}

addblogform=new FormGroup({
  blogTitle:new FormControl(" , [Validators.required ] ),
  blogContent:new FormControl(" , [Validators.required ] ),
    });


    addblog()
    {
      this.blog=new Blog();
      this.blog.blogTitle=this.BlogTitle!.value;
      this.blog.blogContent=this.BlogContent!.value;
      this.blog.blogPosted=DatePipe;
      this.blog.status="false";
      this.blog.noOfLikes=0;
      this.blog.noOfViews=0;
      this.blog.noofComments=0;

      this.submitted = true;

    this.save();
    }

    save() {
      this.blogservice.createBlog(this.blog)
        .subscribe(data => console.log(data), error => console.log(error));
      this.blog = new Blog();
    }

    get BlogTitle(){
      return this.addblogform.get('blogTitle');
    }

    get BlogContent(){
      return this.addblogform.get('blogContent');
    }
```

```
    addblogForm(){
      this.submitted=false;
      this.addblogform.reset();
    }
}
```

Add-post.component.html

```
<div class="add-post-content">
  <div class="container">
    <div>
      <h3 class="new-post-title">Create New Post</h3>
    </div>
    <hr>
    <form [formGroup]="addblogform" (ngSubmit)="addblog()">
      <div class="form-group">
        <label class="post-title">Title</label>
        <input type="text" [formControlName]="'blogTitle'" class="form-
control" placeholder="Title">
      </div>
      <div class="form-group">
        <label class="post-content">Body</label>
        <editor [formControlName]="'blogContent'" [init]="{plugins:
'link'}"></editor>
      </div>
      <div class="form-group">
        <button type="submit" class="btn btn-primary">Post</button>
      </div>
    </form >
  </div>
</div>
```

Add-post.component.scss

```
.add-post-content{
    padding-top: 20px;
  }
  .new-post-title{
    color: royalblue;
  }
  .post-content, .post-title{
    color: royalblue;
  }
```

Add-post.component.specs.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { AddPostComponent } from './add-post.component';

describe('AddPostComponent', () => {
  let component: AddPostComponent;
  let fixture: ComponentFixture<AddPostComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ AddPostComponent ]
    })
    .compileComponents();
  });

  beforeEach(() => {
    fixture = TestBed.createComponent(AddPostComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

Blog Component

Blog.ts

```
export class Blog{
   blogId!:number;
        blogTitel!: String;
   blogContent!: String;
   blogPosted:any;
   status!: String;
        noOfLikes!: number;
   noFoComments!: number;
   noOfViews!: number;
        userId!: number;
        username!: String;
}
```

Blog.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class BlogService {

  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);
  }

  deletBlog(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, value:any): Observable<Object>{
    return this.http.post(`${this.baseUrl}/update-blog/${blogId}`,value);
  }
}
```

Blog.service.specs.ts

```
import { TestBed } from '@angular/core/testing';

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

describe('BlogService', () => {
  let service: BlogService;

  beforeEach(() => {
    TestBed.configureTestingModule({});
    service = TestBed.inject(BlogService);
  });

  it('should be created', () => {
    expect(service).toBeTruthy();
  });
});
```

Blog-list.component.ts

```
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-blog-list',
  templateUrl: './blog-list.component.html',
  styleUrls: ['./blog-list.component.scss']
```

```
})
export class BlogListComponent implements OnInit {

  constructor(private blogservice: BlogService) { }
  blog: Blog =new Blog();
  submitted = false;
  ngOnInit(): void {
    this.submitted = false;
  }
}
```

Blog-list.component.html

```
<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="bg-white p-2">
                  <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></strong></span></div>
                  </div>
                  <div class="mt-2">
                     <p class="comment-text"></p>
                  </div>
               </div>
               <div class="bg-light p-2">
                  <div class="d-flex flex-row align-items-start"><textarea
class="form-control ml-1 shadow-none textarea"></textarea></div>
                  <div class="mt-2 text-right"><button class="btn btn-
primary btn-sm shadow-none" type="button">Post
comment</button><button class="btn btn-outline-primary btn-sm ml-1
shadow-none" type="button">Cancel</button></div>
               </div>
            </div>
         </div>
      </div>
   </div>
   </div>
```

Blog-list.component.scss

```
body {
   background: #eee
}
.main{
   position:relative;
   left:95px;
   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
}
```

Blog-list.component.specs.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

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

describe('BlogListComponent', () => {
  let component: BlogListComponent;
  let fixture: ComponentFixture<BlogListComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ BlogListComponent ]
    })
    .compileComponents();
  });

  beforeEach(() => {
    fixture = TestBed.createComponent(BlogListComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

Home Component

Home.component.ts

```
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");
  }
}
```

Home.component.html

```
<div class="container">
    <div class="navbar">
        <div class="menu">
            <h1 class="logo"><span>Relaxato  </span>City</h1>
            <div class="bts">
                <a href="/about-us" class="about-us">About us</a>
            </div>
        </div>
    </div>
    <div class="main-container">
        <div class="main">
            <header>
                <div class="overlay" id="one">
                    <h2 class="title">Welcome to Relaxato City</h2>
                    <div class="bts">
                      <a mat-list-item  class="nav-link" (click)="logInUser()"
class="btn">Sign In</a>/<a mat-list-item   class="nav-link"
(click)="RegUser()" class="btn">Sign Up</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>
```

Home.component.scss

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

.container{ max-height: 100vh;
          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: 2.5rem; font-weight: 600; text-transform: uppercase;
```

```
    letter-spacing: 10px; line-height: 4rem; margin-top: 20px;
    color:darkgoldenrod;
}

.logo span{ font-size: 2.5rem;margin-left: 10pc; margin-right: 50px;
color:darkgoldenrod; }

.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;
   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;
}

.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;
}
```

Home.component.specs.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

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

describe('HomeComponent', () => {
  let component: HomeComponent;
  let fixture: ComponentFixture<HomeComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ HomeComponent ]
    })
    .compileComponents();
  });

  beforeEach(() => {
    fixture = TestBed.createComponent(HomeComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  it('should create', () => {
    expect(component).toBeTruthy();
  });
});
```

Navigation Components

Navigation.component.ts

```
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-navigation',
  templateUrl: './navigation.component.html',
  styleUrls: ['./navigation.component.scss']
})
export class NavigationComponent 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: any)=>console.log(data)
  );
}

}
```

Navigation.component.html

```
<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>Menu</mat-toolbar>
    <mat-nav-list>

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

  </mat-sidenav>
  <mat-sidenav-content>
    <mat-toolbar color="primary">

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

```
        <button class="logout" (click)="logout()">LogOut</button>
      </mat-toolbar>
    </mat-sidenav-content>

  </mat-sidenav-container>
  <p class="welcome">Welcome  <span>{{user.firstName+"
"+user.lastName}}</span></p>
  <router-outlet></router-outlet>
```

Navigation.component.scss

```
.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;
}
.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;
}
```

Navigation.component.specs.ts

```
import { LayoutModule } from '@angular/cdk/layout';
import { waitForAsync, ComponentFixture, TestBed } from
'@angular/core/testing';
import { NoopAnimationsModule } from '@angular/platform-
browser/animations';
import { MatButtonModule } from '@angular/material/button';
import { MatIconModule } from '@angular/material/icon';
import { MatListModule } from '@angular/material/list';
import { MatSidenavModule } from '@angular/material/sidenav';
import { MatToolbarModule } from '@angular/material/toolbar';

import { NavigationComponent } from './navigation.component';

describe('NavigationComponent', () => {
  let component: NavigationComponent;
  let fixture: ComponentFixture<NavigationComponent>;

  beforeEach(waitForAsync(() => {
    TestBed.configureTestingModule({
      declarations: [NavigationComponent],
      imports: [
        NoopAnimationsModule,
        LayoutModule,
        MatButtonModule,
        MatIconModule,
        MatListModule,
        MatSidenavModule,
        MatToolbarModule,
      ]
    }).compileComponents();
  }));
```

```
    beforeEach(() => {
      fixture = TestBed.createComponent(NavigationComponent);
      component = fixture.componentInstance;
      fixture.detectChanges();
    });

    it('should compile', () => {
      expect(component).toBeTruthy();
    });
  });
```

Navigation-user.component.ts

```
  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: './navigation-user.component.html',
    styleUrls: ['./navigation-user.component.scss']
  })
  export class NavigationUserComponent 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: any)=>console.log(data)
    );
  }
 }
```

Navigation-user.component.html

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

   </mat-nav-list>
  </mat-sidenav>
  <mat-sidenav-content>
   <mat-toolbar color="primary">
```

```
          <span>RelaxatoCity</span>
          <button class="logout" (click)="logout()">LogOut</button>
        </mat-toolbar>
      </mat-sidenav-content>
    </mat-sidenav-container>
    <p class="welcome">Welcome  <span>{{user.firstName+"
"+user.lastName}}</span></p>
    <router-outlet></router-outlet>
```

Navigation-user.component.scss

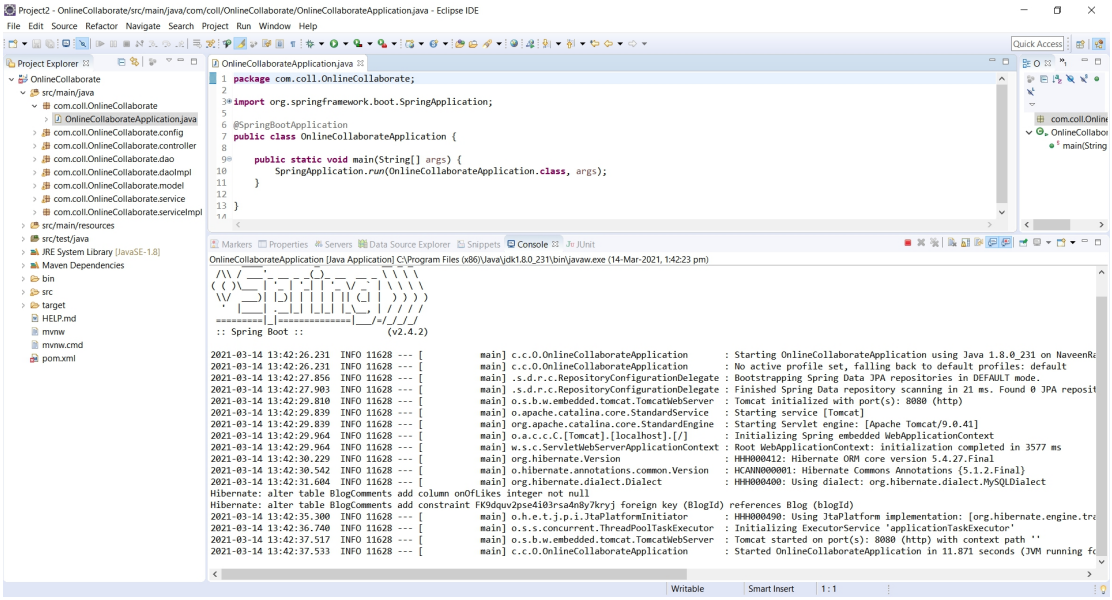
```
.sidenav-container {
  height: 10%;
  width: 1560px;
}

.sidenav {
  width: 200px;
}

.sidenav .mat-toolbar {
  background: inherit;
}

.mat-toolbar.mat-primary {
  position: sticky;
  top: 0;
  z-index: 1;
}
.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;
 }
```

Navigation-user.component.specs.ts

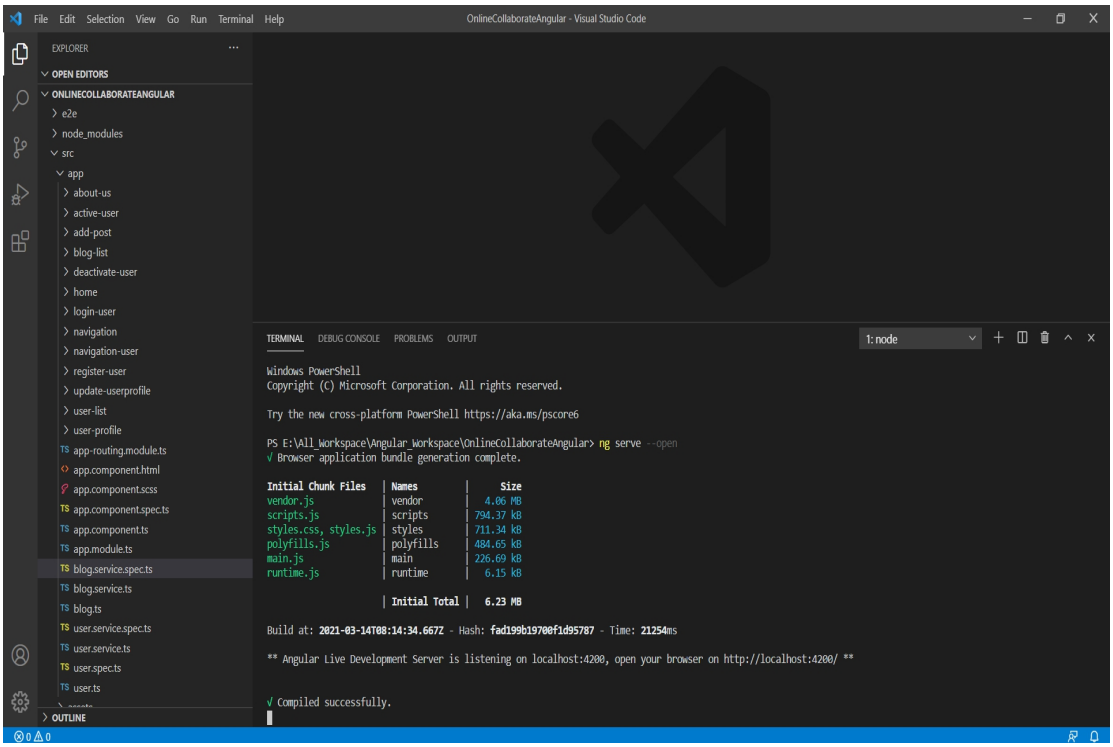
```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { NavigationUserComponent } from './navigation-
user.component';

describe('NavigationUserComponent', () => {
  let component: NavigationUserComponent;
  let fixture: ComponentFixture<NavigationUserComponent>;

  beforeEach(async () => {
   await TestBed.configureTestingModule({
    declarations: [ NavigationUserComponent ]
   })
   .compileComponents();
  });

  beforeEach(() => {
   fixture = TestBed.createComponent(NavigationUserComponent);
   component = fixture.componentInstance;
   fixture.detectChanges();
  });

  it('should create', () => {
   expect(component).toBeTruthy();
  });
 });
```

Run OnlineCollaboratApplication.java file

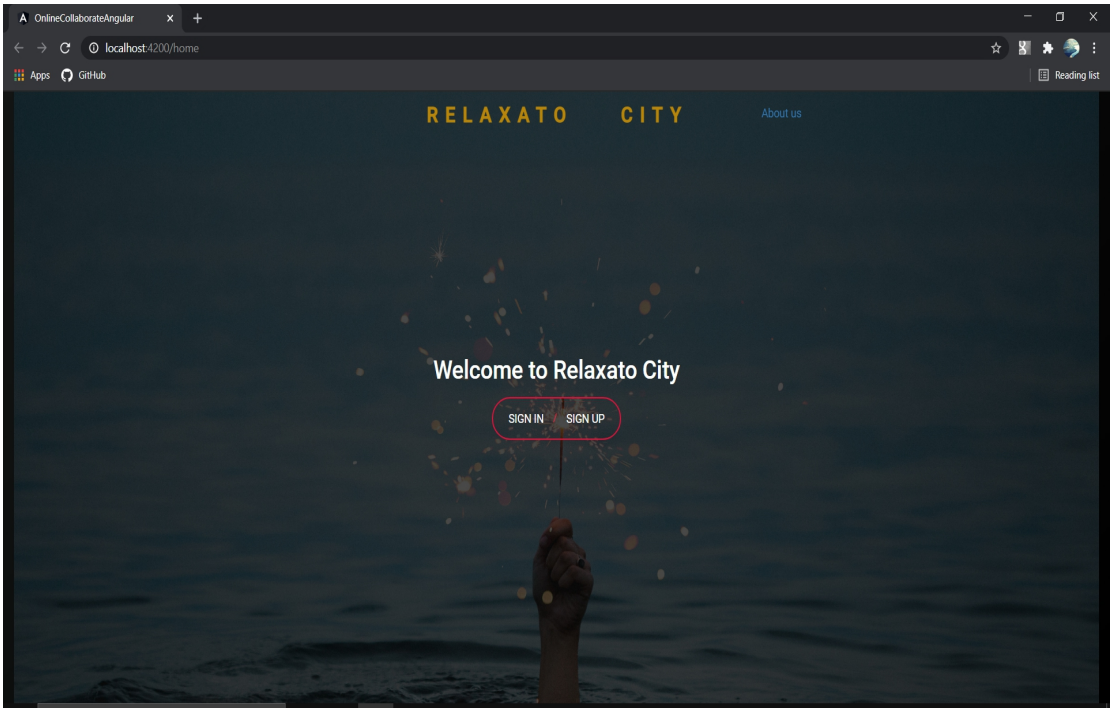


Starting the Angular

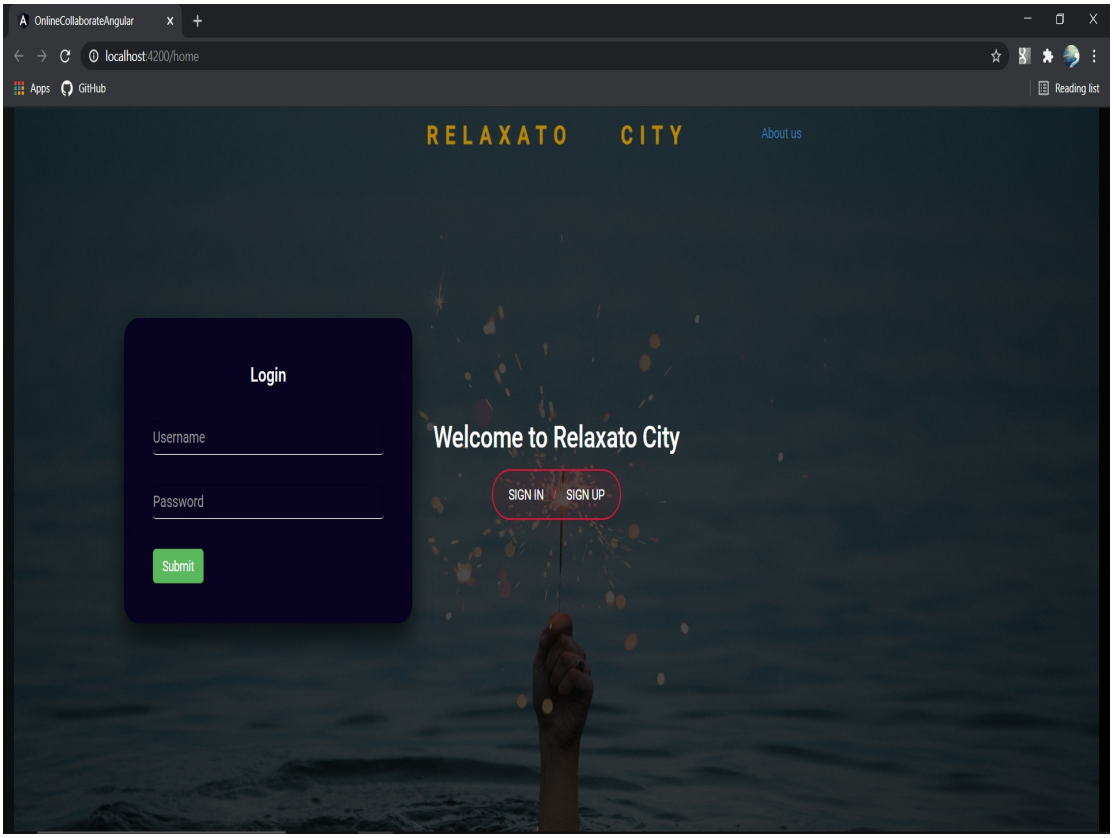
Run the cmd - ng serve --open



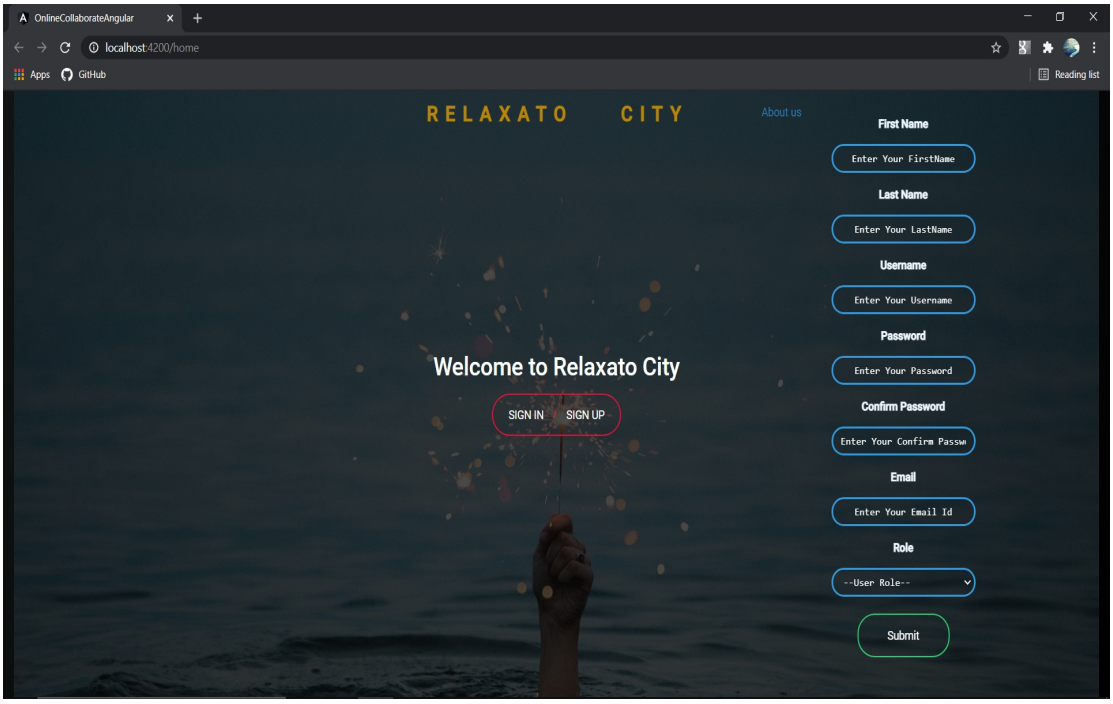
Home Page

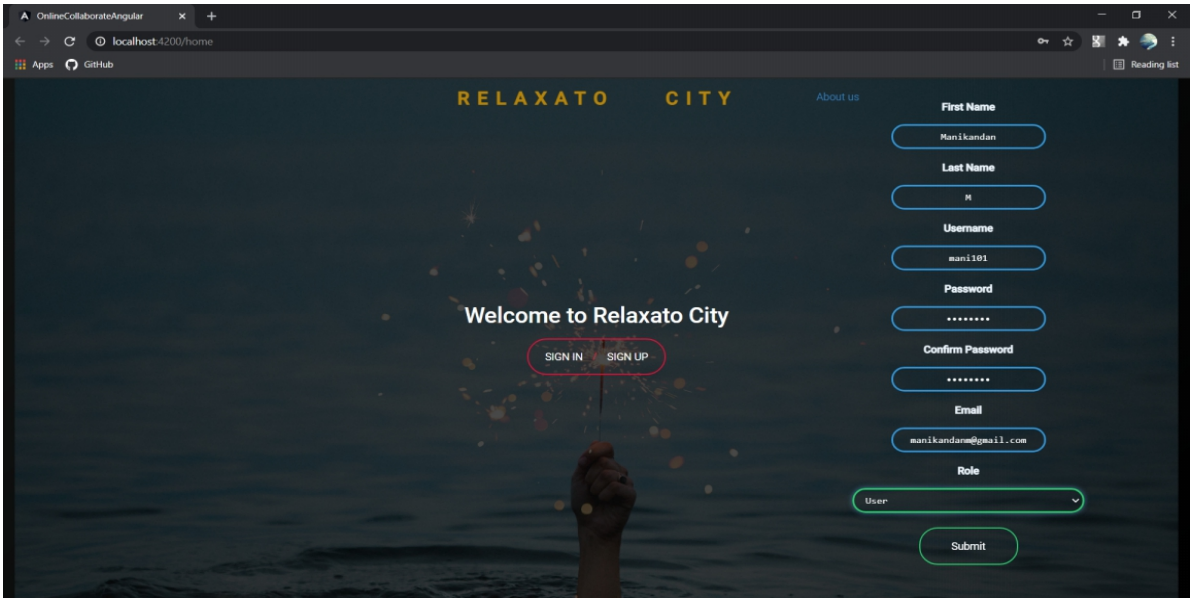


Login Page

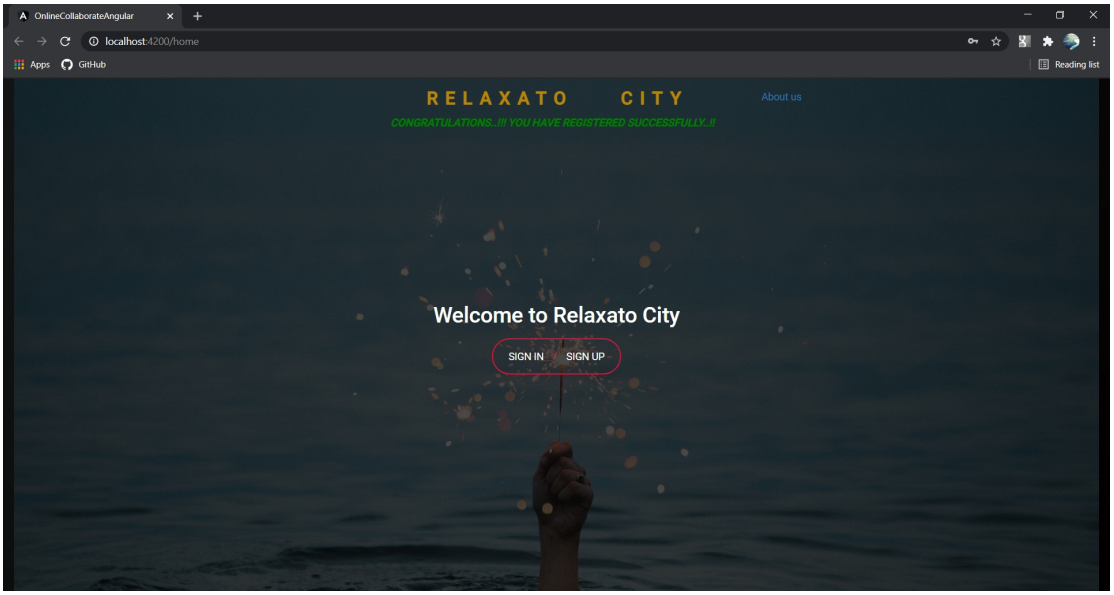


Registration Page

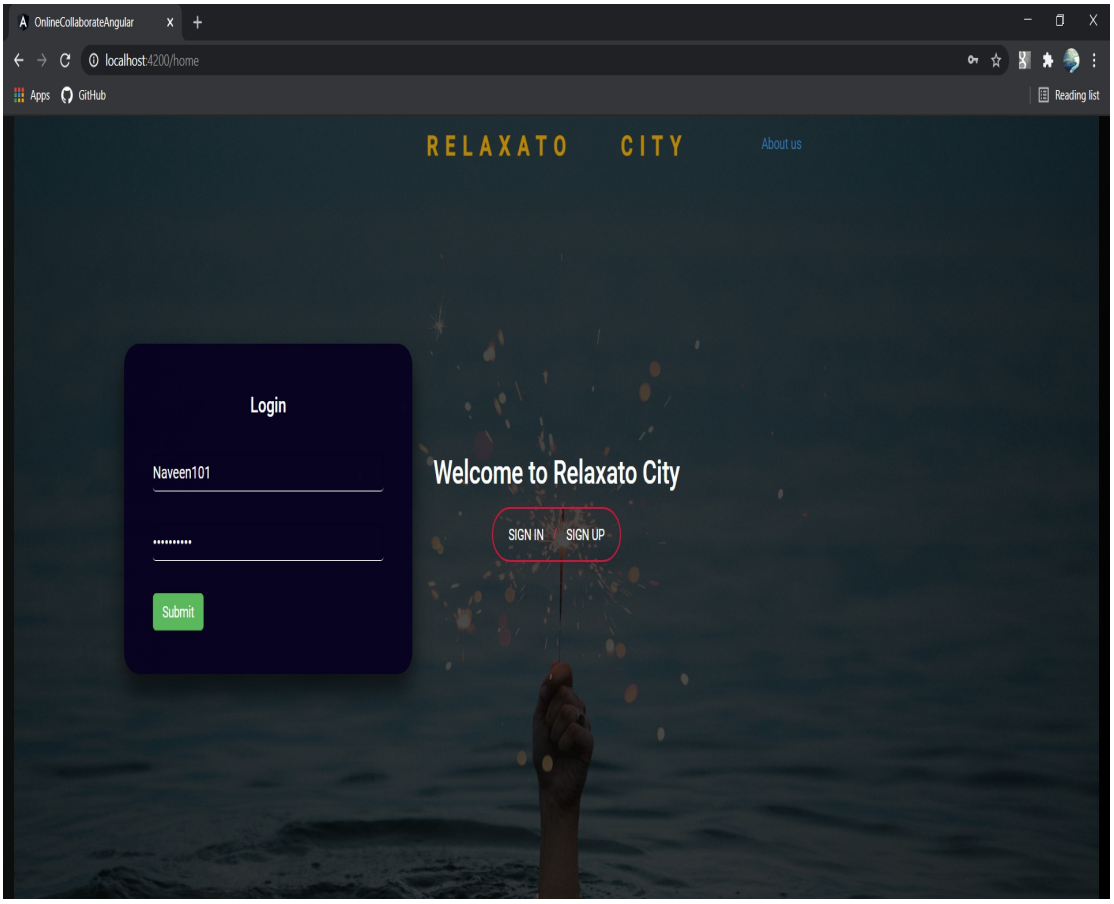




Registration Success

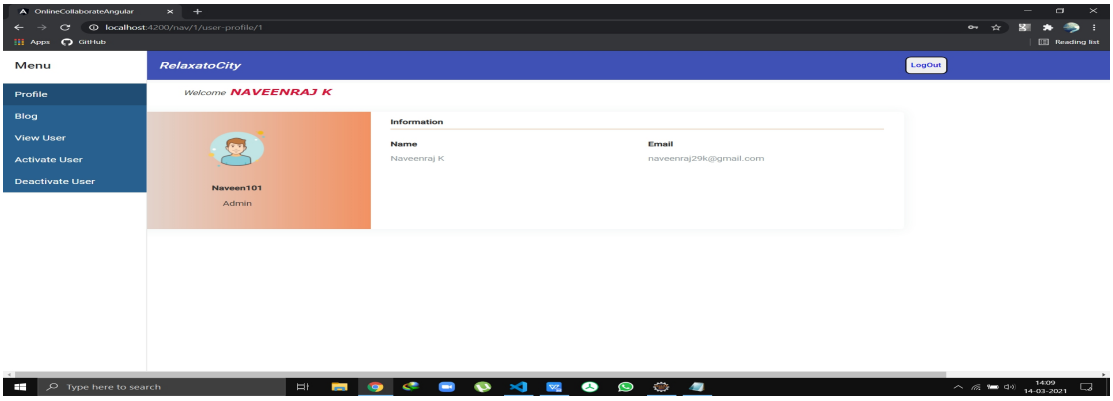


Admin Login

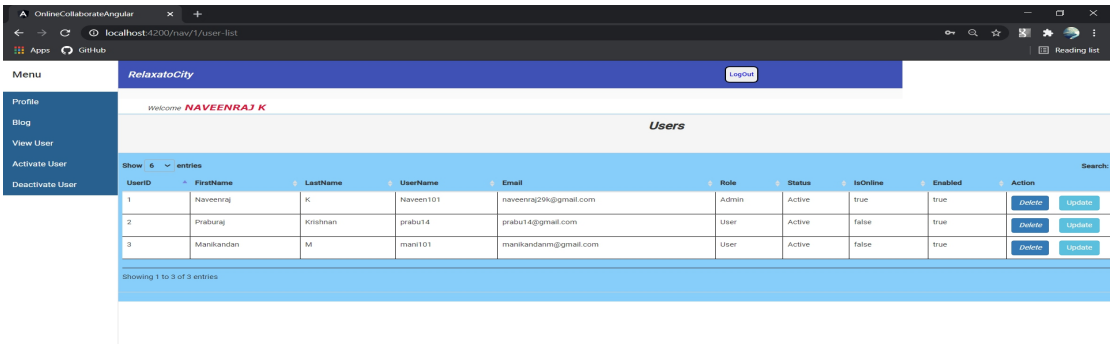


Admin Home

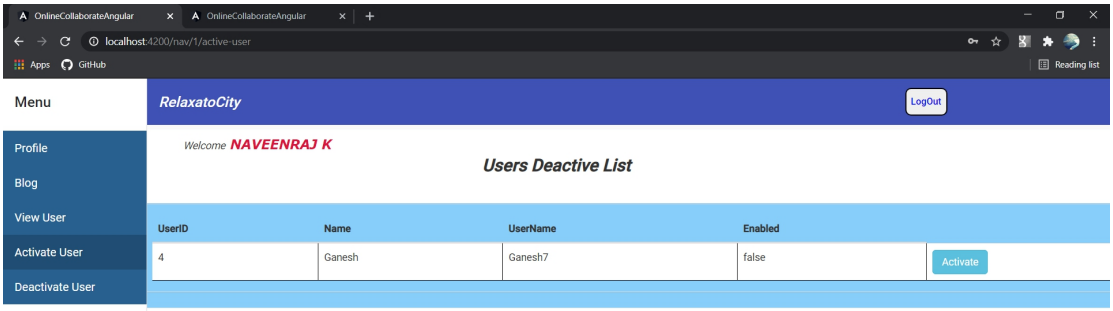
Profile



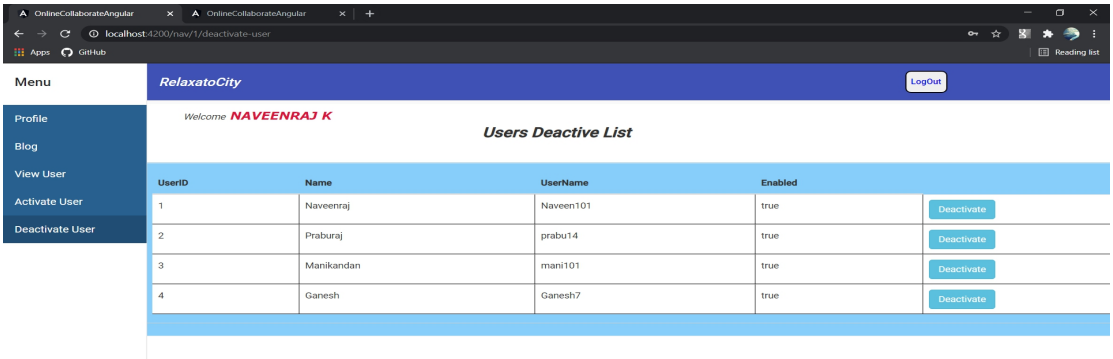
View User List



Users Deactive List

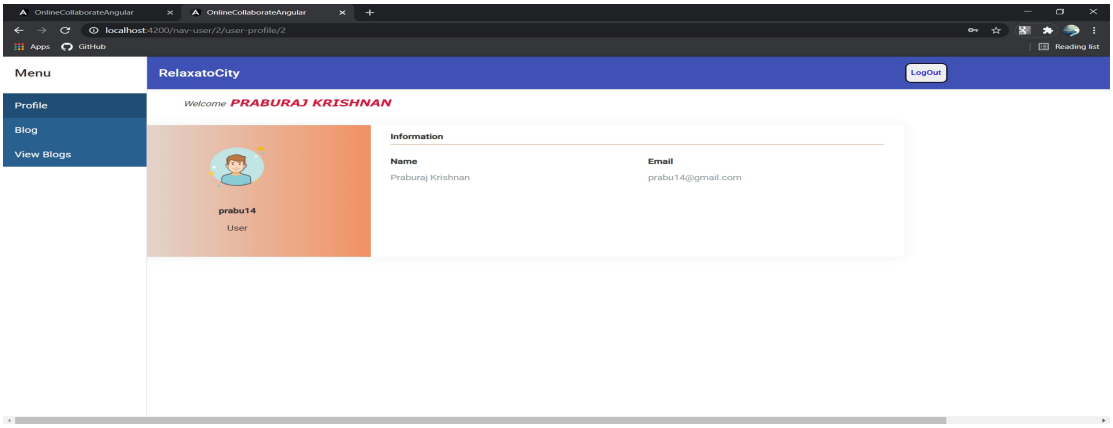


Users Active List



User Home

User Profile



Blog