**Task :** Blog module

a)Add dto,dao and daoimpl

1.Add a model class , Blog.java.

@Entity

@Table(name = "Blogs")

**public** **class** Blog **extends** DomainResponse **implements** Serializable{

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

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

}

/\*\*

\*

\*/

**private** **static** **final** **long** ***serialVersionUID*** = -5902335015810630363L;

@Id

@GeneratedValue(strategy = GenerationType.***SEQUENCE***, generator = "generator")

@SequenceGenerator(name = "generator", sequenceName = "blogs\_seq", allocationSize = 1)

**private** **int** blogId;

**private** String blogHeader;

**private** String blogCategory;

**private** String blogUser;

**private** String blogContent;

**private** String isActive="false";

**private** **int** blogLikes=0;

**private** String blogDate=**new** Date().toString();

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

**return** blogId;

}

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

**this**.blogId = blogId;

}

**public** String getBlogHeader() {

**return** blogHeader;

}

**public** **void** setBlogHeader(String blogHeader) {

**this**.blogHeader = blogHeader;

}

**public** String getBlogCategory() {

**return** blogCategory;

}

**public** **void** setBlogCategory(String blogCategory) {

**this**.blogCategory = blogCategory;

}

**public** String getBlogUser() {

**return** blogUser;

}

**public** **void** setBlogUser(String blogUser) {

**this**.blogUser = blogUser;

}

**public** String getBlogContent() {

**return** blogContent;

}

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

**this**.blogContent = blogContent;

}

**public** String getIsActive() {

**return** isActive;

}

**public** **void** setIsActive(String isActive) {

**this**.isActive = isActive;

}

**public** **int** getBlogLikes() {

**return** blogLikes;

}

**public** **void** setBlogLikes(**int** blogLikes) {

**this**.blogLikes = blogLikes;

}

**public** String getBlogDate() {

**return** blogDate;

}

**public** **void** setBlogDate(String blogDate) {

**this**.blogDate = blogDate;

}

**public** Blog(**int** blogId, String blogHeader, String blogCategory, String blogUser, String blogContent) {

**super**();

**this**.blogId = blogId;

**this**.blogHeader = blogHeader;

**this**.blogCategory = blogCategory;

**this**.blogUser = blogUser;

**this**.blogContent = blogContent;

**this**.isActive = "false";

**this**.blogLikes = 0;

**this**.blogDate = **new** Date().toString();

}

**public** Blog() {

**super**();

}

}

2.Add a dao interface,IBlogDAO.java.

**public** **interface** IBlogDAO {

**public** Blog getBlog(**int** id);

**public** List<Blog> getAllBlogs();

**public** List<Blog> getDBlogs();

**public** List<Blog> getABlogs();

**public** List<Blog> getBlogsByUserName(String userName);

**public** Blog getBlogById(**int** blogId);

**public** **boolean** activateBlog(**int** id);

**public** **boolean** addBlog(Blog b);

**public** **boolean** updateBlog(Blog b);

**public** **boolean** deleteBlog(**int** id);

**public** **boolean** likeBlog(**int** id);

}

3.Add a daoimpl class , BlogDAOImpl.java.

@Repository("blogDAO")

@Transactional

@EnableTransactionManagement

**public** **class** BlogDAOImpl **implements** IBlogDAO

{

@Autowired

**private** SessionFactory sessionFactory;

@Override

**public** Blog getBlog(**int** id) {

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

}

@Override

**public** List<Blog> getAllBlogs() {

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

}

@Override

**public** List<Blog> getDBlogs() {

**return** sessionFactory.getCurrentSession().createQuery("FROM Blog b where b.isActive='false'", Blog.**class**).getResultList();

}

@Override

**public** List<Blog> getABlogs() {

**return** sessionFactory.getCurrentSession().createQuery("FROM Blog b where b.isActive='true'", Blog.**class**).getResultList();

}

@Override

**public** List<Blog> getBlogsByUserName(String userName) {

**return** sessionFactory.getCurrentSession().createQuery("FROM Blog b where b.blogUser=:userName", Blog.**class**).getResultList();

}

@Override

**public** Blog getBlogById(**int** blogId) {

**return** (Blog)sessionFactory.getCurrentSession().createQuery("FROM Blog b where b.blogId=:blogId", Blog.**class**).setParameter("blogId",blogId).getResultList().get(0);

}

@Override

**public** **boolean** activateBlog(**int** id) {

**try** {

Blog blog=getBlog(id);

blog.setIsActive("true");

sessionFactory.getCurrentSession().update(blog);

**return** **true**;

} **catch** (Exception ex) {

ex.printStackTrace();

**return** **false**;

}

}

@Override

**public** **boolean** addBlog(Blog b) {

**try** {

sessionFactory.getCurrentSession().persist(b);

**return** **true**;

} **catch** (Exception ex) {

ex.printStackTrace();

**return** **false**;

}

}

@Override

**public** **boolean** updateBlog(Blog b) {

**try** {

sessionFactory.getCurrentSession().update(b);

**return** **true**;

} **catch** (Exception ex) {

ex.printStackTrace();

**return** **false**;

}

}

@Override

**public** **boolean** deleteBlog(**int** id) {

**try** {

sessionFactory.getCurrentSession().delete(getBlog(id));

**return** **true**;

} **catch** (Exception ex) {

ex.printStackTrace();

**return** **false**;

}

}

@Override

**public** **boolean** likeBlog(**int** id) {

**try** {

Blog blog=getBlog(id);

blog.setBlogLikes(blog.getBlogLikes()+1);;

sessionFactory.getCurrentSession().update(blog);

**return** **true**;

} **catch** (Exception ex) {

ex.printStackTrace();

**return** **false**;

}

}}

b)Add the controller.

1.Add a java class,BlogController.java.

@CrossOrigin(origins = "http://localhost:8887")

@RestController

@RequestMapping("/blog")

**public** **class** BlogController

{

@Autowired

IBlogDAO blogDAO;

@Autowired

**private** HttpServletRequest request;

@PostMapping("/add")

**public** ResponseEntity<Blog> addBlog(@RequestBody Blog blog) {

blog.setBlogDate(**new** Date().toString());

blog.setBlogLikes(0);

blog.setIsActive("false");

**if** (blogDAO.addBlog(blog)) {

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

blog.setResponseMessage("Your post has been successfully registered .");

**return** **new** ResponseEntity<Blog>(blog, HttpStatus.***OK***);

} **else** {

**return** **new** ResponseEntity<Blog>(blog, HttpStatus.***INTERNAL\_SERVER\_ERROR***);

}

}

@GetMapping("/viewD")

**public** ResponseEntity<Blog> getDBlogs() {

**return** **new** ResponseEntity(blogDAO.getDBlogs(), HttpStatus.***OK***);

}

@PostMapping("/delete")

**public** ResponseEntity<String> deleteUser(@RequestBody **int** id) {

**if** (blogDAO.deleteBlog(id)) {

**return** **new** ResponseEntity<String>("Blog successfully deleted .", HttpStatus.***OK***);

} **else** {

**return** **new** ResponseEntity<String>("Deletion failure.", HttpStatus.***INTERNAL\_SERVER\_ERROR***);

}

}

@PostMapping("/activate")

**public** ResponseEntity<String> activateBlog(@RequestBody **int** id) {

**if** (blogDAO.activateBlog(id)) {

**return** **new** ResponseEntity<String>("Blog successfully activated .", HttpStatus.***OK***);

} **else** {

**return** **new** ResponseEntity<String>("Blog activation failure.", HttpStatus.***INTERNAL\_SERVER\_ERROR***);

}

}

//@GetMapping("/singleBlog")

@RequestMapping(value = "/singleBlog/{blogId}")

**public** ResponseEntity<Blog> getSingleBlog(@PathVariable("blogId") **int** blogId){

//@RequestParam("blogHeader") String blogHeader) {

**return** **new** ResponseEntity(blogDAO.getBlogById(blogId), HttpStatus.***OK***);

}

@PostMapping("/like")

**public** ResponseEntity<String> likeBlog(@RequestBody **int** id) {

**if** (blogDAO.likeBlog(id)) {

**return** **new** ResponseEntity<String>("Blog successfully liked .", HttpStatus.***OK***);

} **else** {

**return** **new** ResponseEntity<String>("Blog liking failure.", HttpStatus.***INTERNAL\_SERVER\_ERROR***);

}}

c)Front end application.

1.Add a folder inside the components folder , blog.

2.Add an html file , blog.html.

<div class="row">

<div class="col-md-2"></div>

<div class="col-md-8">

<h1>Post a Blog</h1>

<form name="blogForm" ng-submit="submitForm(blogForm.$valid)" novalidate>

<div class="form-group">

<label for="blogHeader">Blog Header</label>

<input type="text" name="blogHeader" ng-model="blogCtrl.blog.blogHeader" />

</div>

<div class="form-group">

<label for="blogCategory">Blog Category</label>

<input type="text" name="blogCategory" ng-model="blogCtrl.blog.blogCategory" />

</div>

<div class="form-group">

<label for="blogUser">Blog User</label>

<input type="text" name="blogUser" ng-model="blogCtrl.blog.blogUser" />

</div>

<div class="form-group">

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

<!--<input type="text" name="blogContent" ng-model="blogCtrl.blog.blogContent" />-->

<textarea name="blogContent" cols="40" rows="5" ng-model="blogCtrl.blog.blogContent"></textarea>

</div>

<div class="form-group">

<input type="button" ng-click="blogCtrl.blogAdd()" class="btn btn-default" value="Post Blog" />

</div>

</form>

</div>

<div class="col-md-2"></div>

</div>

</div>

2.Add a js file , BlogService.js.

var BlogModule = angular.module('BlogModule', []);

BlogModule.service('BlogService', ['$http', '$q', 'REST\_URI', function ($http, $q, REST\_URI) {

this.blogAdd = function (blog) {

var deffered = $q.defer();

console.log(blog);

$http.post(REST\_URI + '/blog/add', blog).then(

function (response) {

deffered.resolve(response.data.responseMessage);

},

function (error) {

deffered.reject(error);

}

);

return deffered.promise;

}

this.viewD = function () {

var deffered = $q.defer();

$http.get(REST\_URI + '/blog/viewD').then(

function (response) {

deffered.resolve(response.data);

},

function (error) {

deffered.reject(error);

}

);

return deffered.promise;

}

this.deleteBlog = function (id) {

var deffered = $q.defer();

console.log(id);

$http.post(REST\_URI + '/blog/delete', id).then(

function (response) {

deffered.resolve(response.data.text());

},

function (error) {

deffered.reject(error);

}

);

this.init();

return deffered.promise;

}

this.activateBlog = function (id) {

var deffered = $q.defer();

console.log(id);

$http.post(REST\_URI + '/blog/activate', id).then(

function (response) {

deffered.resolve(response.data.text());

},

function (error) {

deffered.reject(error);

}

);

this.init();

return deffered.promise;

}

this.viewA = function () {

var deffered = $q.defer();

$http.get(REST\_URI + '/blog/viewA').then(

function (response) {

deffered.resolve(response.data);

},

function (error) {

deffered.reject(error);

}

);

return deffered.promise;

}

this.singleBlogV = function (id) {

var deffered = $q.defer();

$http.get(REST\_URI + '/blog/singleBlog/' + id).then(

function (response) {

deffered.resolve(response.data);

},

function (error) {

deffered.reject(error);

}

);

return deffered.promise;

}

this.likeBlog = function (id) {

var deffered = $q.defer();

console.log(id);

$http.post(REST\_URI + '/blog/like', id).then(

function (response) {

deffered.resolve(response.data.text());

},

function (error) {

deffered.reject(error);

}

);

this.init();

return deffered.promise;

}

}]);

3.Add a js file , BlogController.js.

BlogModule.controller('BlogController', ['BlogService', '$location', '$rootScope', function (BlogService, $location, $rootScope) {

var me = this;

me.blog = {};

me.message = '';

me.blogsD = {};

me.blogsA = {};

me.singleBlogC = {};

$rootScope.demo;

me.blogAdd = function () {

console.log(me.blog);

BlogService.blogAdd(me.blog).then(

function (response) {

me.message = response;

console.log(me.message);

},

function (error) {

console.log(me.blog);

}

);

}

BlogService.viewD().then(

function (response) {

me.blogsD = response;

console.log(response);

},

function (error) {

console.log(error);

}

);

me.deleteBlog = function (id) {

BlogService.deleteBlog(id).then(

function (response) {

console.log(response);

window.location.reload();

},

function (error) {

console.log(error);

}

);

}

me.activateBlog = function (id) {

BlogService.activateBlog(id).then(

function (response) {

console.log(response);

window.location.reload();

},

function (error) {

console.log(error);

}

);

}

BlogService.viewA().then(

function (response) {

me.blogsA = response;

console.log(response);

},

function (error) {

console.log(error);

}

);

me.singleBlog = function (id) {

console.log(id);

BlogService.singleBlogV(id).then(

function (response) {

me.singleBlogC = response;

//console.log(response);

// alert(me.singleBlogC);

console.log(me.singleBlogC);

$rootScope.demo = response;

//sessionStorage.setItem('blog', $rootScope.demo);

$location.path("/singleBlogView");

// $rootScope.demo=sessionStorage.getItem('blog');

},

function (error) {

console.log(error);

}

);

}

me.likeBlog = function (id) {

BlogService.likeBlog(id).then(

function (response) {

console.log(response);

me.singleBlog(id);

$location.path("/singleBlogView");

},

function (error) {

console.log(error);

}

);

}

}]);

4.Add the BlogModule inside the app.module.js.

5.In the app.routes.js ,add the routes.

"/blogAdd":

{

templateUrl: 'app/components/blog/blog.html',

controller: 'BlogController',

controllerAs: 'blogCtrl',

requireLogin: false,

roles: ['GUEST']

},

"/blogs":

{

templateUrl: 'app/components/blog/blogs.html',

controller: 'BlogController',

controllerAs: 'blogCtrl',

requireLogin: false,

roles: ['GUEST']

},

"/viewBlogs":

{

templateUrl: 'app/components/blog/viewBlogs.html',

controller: 'BlogController',

controllerAs: 'blogCtrl',

requireLogin: false,

roles: ['GUEST']

},

"/singleBlogView":

{

templateUrl: 'app/components/blog/singleBlog.html',

controller: 'BlogController',

controllerAs: 'blogCtrl',

requireLogin: false,

roles: ['GUEST']

},

6.Add a html page , blogs.html.

<div class="row">

<div class="col-md-12">

<center>

<h1 style="text-decoration:underline">Blog's Details</h1>

</center>

<br/>

<center>

<h3>Non Approved Blogs</h3>

</center>

<table border="1" class="table table-bordered table-hover table-striped">

<tr>

<th>Blog Id</th>

<th>Blog Writer</th>

<th>Blog Title</th>

<th>Blog Category</th>

<th>Blog Date</th>

<th>Blog Content</th>

<th>Approve</th>

<th>Disapprove</th>

</tr>

<tr ng-repeat="x in blogCtrl.blogsD">

<td>{{x.blogId}}</td>

<td>{{x.blogUser}}</td>

<td>{{x.blogHeader}}</td>

<td>{{x.blogCategory}}</td>

<td>{{x.blogDate}}</td>

<td>{{x.blogContent}}</td>

<td><button ng-click="blogCtrl.activateBlog(x.blogId)" class="btn btn-default" onclick="this.parentElement.style.display='none';"><span class="glyphicon glyphicon-ok"></span></button></td>

<td><button ng-click="blogCtrl.deleteBlog(x.blogId)" class="btn btn-default" onclick="this.parentElement.style.display='none';"><span class="glyphicon glyphicon-remove"></span></button></td>

</tr>

</table>

</div>

</div>

7.Add html file,viewBlogs.html.

<div class="row">

<div class="col-md-12">

<center>

<h1 style="text-decoration:underline">BLOGS</h1>

</center>

<br/>

<table border="1" class="table table-bordered table-hover table-striped">

<tr>

<th>Blog Id</th>

<th>Blog Writer</th>

<th>Blog Title</th>

<th>Blog Category</th>

<th>Blog Date</th>

<th>Blog Likes</th>

<!--<th>Blog Content</th>-->

</tr>

<tr ng-repeat="x in blogCtrl.blogsA">

<td>{{x.blogId}}</td>

<td>{{x.blogUser}}</td>

<td><button ng-click="blogCtrl.singleBlog(x.blogId)" style=" background:none!important;border:none;padding:0!important;font: inherit;color:blue;border-bottom:1px solid blue;cursor: pointer;">{{x.blogHeader}}</button></td>

<td>{{x.blogCategory}}</td>

<td>{{x.blogDate}}</td>

<td>{{x.blogLikes}}</td>

<!--<td>{{x.blogContent}}</td>-->

</tr>

</table>

</div>

</div>

8.Add html,singleBlog.html

<div class="row">

<div class="col-md-2"></div>

<div class="col-md-8">

<center>

<h3 style="text-decoration:underline">{{demo.blogHeader}}</h3>

</center>

<div class="well">

<p style="font-family:fantasy;font-size:20pt;color:midnightblue">

{{demo.blogContent}}

</p>

<div class="alert alert-info">

<button ng-click="blogCtrl.likeBlog(demo.blogId)">

<img src="./assets/images/like.jpg" height="30px" width="80px" alt="like button"/>

</button>

<span class="badge">{{demo.blogLikes}}</span>

</div>

</div>

<br/>

<div class="col-md-2"></div>

</div>

</div>