

Web API Design with SpringBoot Week 1 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your Java project code, to the repository. Lastly, in the Learning Management System, click the "Add Submission" button and paste the URL to your GitHub repository.

Coding Steps:

Follow the tutorial in the homework section to build an API. Paste screenshots of your code and your postman requests and responses to show the API works. Push your project to GitHub and paste the link below.

Screenshots of Code:

```
CommentController.java
1 package com.promineotech.socialMediaApi.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5
6
7 @RestController
8 @RequestMapping("/users/{userId}/posts/{postId}/comments")
9 public class CommentController {
10
11     @Autowired
12     private CommentService service;
13
14     @PostMapping
15     public ResponseEntity<Object> createComment (@RequestBody Comment comment, @PathVariable Long userId, @PathVariable Long postId) {
16         try {
17             return new ResponseEntity<Object>(service.createComment(comment, userId, postId), HttpStatus.OK);
18         } catch (Exception e) {
19             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.BAD_REQUEST);
20         }
21     }
22
23     @RequestMapping (value =("/{commentId}", method=RequestMethod.DELETE)
24     public ResponseEntity<Object> deleteComment (@PathVariable Long commentId) {
25         service.deleteComment(commentId);
26         return new ResponseEntity<Object>("Deleted comment with id:" + commentId, HttpStatus.OK);
27     }
28 }
29
30
```

```
PostController.java
1 package com.promineotech.socialMediaApi.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5
6
7 @RestController
8 @RequestMapping("users/{userId}/posts")
9 public class PostController {
10
11     @Autowired
12     private PostService service;
13
14     @RequestMapping(method=RequestMethod.GET)
15     public ResponseEntity<Object> getAllPosts() {
16         return new ResponseEntity<Object>(service.getAllPosts(), HttpStatus.OK);
17     }
18
19     @RequestMapping(value="/{postId}", method=RequestMethod.GET)
20     public ResponseEntity<Object> getPost(@PathVariable Long postId) {
21         return new ResponseEntity<Object>(service.getPost(postId), HttpStatus.OK);
22     }
23
24     @RequestMapping(value="/{postId}", method=RequestMethod.PUT)
25     public ResponseEntity<Object> updatePost(@RequestBody Post post, @PathVariable Long postId) {
26         try {
27             return new ResponseEntity<Object>(service.updatePost(post, postId), HttpStatus.OK);
28         } catch (Exception e) {
29             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.BAD_REQUEST);
30         }
31     }
32
33     @RequestMapping(method=RequestMethod.POST)
34     public ResponseEntity<Object> createPost(@RequestBody Post post, @PathVariable Long userId) {
35         try {
36             return new ResponseEntity<Object>(service.createPost(post, userId), HttpStatus.OK);
37         } catch (Exception e) {
38             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.BAD_REQUEST);
39         }
40     }
41 }
42
43
44
45
46
47
48
49
50
51
52
53 }
54
```

```

UserController.java 88
1 package com.promineotech.socialMediaApi.controller;
2
3 import java.nio.file.Files;
4
5
6
7
8
9
10
11
12
13 @RestController
14 @RequestMapping("/users")
15 public class UserController {
16
17     private static String UPLOADED_FOLDER = "./pictures/";
18
19     @Autowired
20     private UserService service;
21
22     @RequestMapping(value = "/register", method = RequestMethod.POST)
23     public ResponseEntity<Object> register (@RequestBody User user) {
24         return new ResponseEntity<Object>(service.createUser(user), HttpStatus.CREATED);
25     }
26
27     @RequestMapping(value = "/login", method = RequestMethod.POST)
28     public ResponseEntity<Object> login(@RequestBody User user) {
29         try {
30             return new ResponseEntity<Object>(service.login(user), HttpStatus.OK);
31         } catch (Exception e) {
32             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.BAD_REQUEST);
33         }
34     }
35
36     @RequestMapping(value =("/{id}/follows")
37     public ResponseEntity<Object> showFollowedUsers(@PathVariable Long id) {
38         try {
39             return new ResponseEntity<Object>(service.getFollowedUsers(id), HttpStatus.CREATED);
40         } catch (Exception e) {
41             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.BAD_REQUEST);
42         }
43     }
44
45     @RequestMapping(value =("/{id}/follows/{followId}", method = RequestMethod.POST)
46     public ResponseEntity<Object> follow (@PathVariable Long id, @PathVariable Long followId) {
47         try {
48             return new ResponseEntity<Object>(service.follow(id, followId), HttpStatus.CREATED);
49         } catch (Exception e) {
50             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.BAD_REQUEST);
51         }
52     }
53
54     @RequestMapping(value =("/{id}/profilePicture", method = RequestMethod.POST)
55     public ResponseEntity<Object> singleFileUpload(@PathVariable Long id, @RequestParam("file") MultipartFile file) {
56         if (file.isEmpty()) {
57             return new ResponseEntity<Object>("Please upload a file.", HttpStatus.BAD_REQUEST);
58         }
59
60         try {
61             String url = UPLOADED_FOLDER + file.getOriginalFilename();
62             byte[] bytes = file.getBytes();
63             Path path = Paths.get(url);
64             Files.write(path, bytes);
65             return new ResponseEntity<Object>(service.updateProfilePicture(id, url), HttpStatus.CREATED);
66         } catch (Exception e) {
67             return new ResponseEntity<Object>(e.getMessage(), HttpStatus.INTERNAL_SERVER_ERROR);
68         }
69     }
70
71
72
73
74
75
76
77
78
79
80

```

```
Comment.java
1 package com.promineotech.socialMediaApi.entity;
2
3 import java.util.Date;
4
5
6
7
8
9
10
11
12
13
14
15
16 @Entity
17 public class Comment {
18
19     private Long id;
20     private String content;
21     private Date date;
22     private User user;
23
24     @JsonIgnore
25     private Post post;
26
27     @Id
28     @GeneratedValue(strategy = GenerationType.AUTO)
29     public Long getId() {
30         return id;
31     }
32     public void setId(Long id) {
33         this.id = id;
34     }
35     public String getContent() {
36         return content;
37     }
38     public void setContent(String content) {
39         this.content = content;
40     }
41     public Date getDate() {
42         return date;
43     }
44     public void setDate(Date date) {
45         this.date = date;
46     }
47
48     @ManyToOne
49     @JoinColumn(name = "userId")
50     public User getUser() {
51         return user;
52     }
53     public void setUser(User user) {
54         this.user = user;
55     }
56
57     @ManyToOne
58     @JoinColumn(name = "postId")
59     public Post getPost() {
60         return post;
61     }
62
63     public void setPost(Post post) {
64         this.post = post;
65     }
66
67 }
68
```

Post.java

```
1 package com.promineotech.socialMediaApi.entity;
2
3 import java.util.Date;
4
5 @Entity
6 public class Post {
7     private Long id;
8     private String content;
9     private Date date;
10    private User user;
11    private Set<Comment> comments;
12
13    @Id
14    @GeneratedValue(strategy=GenerationType.AUTO)
15    public Long getId() {
16        return id;
17    }
18    public void setId(Long id) {
19        this.id = id;
20    }
21    public String getContent() {
22        return content;
23    }
24    public void setContent(String content) {
25        this.content = content;
26    }
27    public Date getDate() {
28        return date;
29    }
30    public void setDate(Date date) {
31        this.date = date;
32    }
33
34    @ManyToOne
35    @JoinColumn(name = "userId")
36    public User getUser() {
37        return user;
38    }
39    public void setUser(User user) {
40        this.user = user;
41    }
42
43    @OneToMany(mappedBy = "post")
44    public Set<Comment> getComments() {
45        return comments;
46    }
47    public void setComments(Set<Comment> comments) {
48        this.comments = comments;
49    }
50 }
51
52 }
```

```

1 package com.promineotech.socialMediaApi.entity;
2
3 import java.util.Set;
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18 @Entity
19 public class User {
20
21     private Long id;
22     private String username;
23     private String profilePictureUrl;
24
25     @JsonIgnore
26     private Set<User> following;
27
28     private String password;
29
30     @JsonIgnore
31     private Set<Post> posts;
32
33     @JsonIgnore
34     private Set<Comment> comments;
35
36     @Id
37     @GeneratedValue(strategy = GenerationType.AUTO)
38     public Long getId() {
39         return id;
40     }
41
42     public void setId(Long id) {
43         this.id=id;
44     }
45
46     public String getUsername() {
47         return username;
48     }
49
50     public void setUsername(String username) {
51         this.username = username;
52     }
53
54     @JsonIgnore
55     public String getPassword() {
56         return password;
57     }
58
59     @JsonProperty
60     public void setPassword(String password) {
61         this.password = password;
62     }
63
64     @OneToMany(mappedBy = "user")
65     public Set<Post> getPosts() {
66         return posts;
67     }
68
69     public void setPosts(Set<Post> posts) {
70         this.posts = posts;
71     }
72
73     @OneToMany(mappedBy = "user")
74     public Set<Comment> getComments() {
75         return comments;
76     }
77
78     public void setComments (Set<Comment> comments) {
79         this.comments= comments;
80     }
81
82     @ManyToMany (cascade = CascadeType.ALL)
83     @JoinTable(name = "following",
84         joinColumns = @JoinColumn(name = "userId",referencedColumnName = "id"),
85         inverseJoinColumns = @JoinColumn(name = "followingId", referencedColumnName = "id"))
86     public Set<User> getFollowing() {
87         return following;
88     }
89
90     public void setFollowing(Set <User> following) {
91         this.following = following;
92     }
93
94     public String getProfilePictureUrl() {
95         return profilePictureUrl;
96     }
97
98     public void setProfilePictureUrl(String profilePictureUrl) {
99         this.profilePictureUrl = profilePictureUrl;
100     }
101
102 }
103

```

```

1 package com.promineotech.socialMediaApi.repository;
2
3 import org.springframework.data.repository.CrudRepository;
4
5 public interface CommentRepository extends CrudRepository<Comment, Long> {
6
7 }
8
9
10

```

```

1 package com.promineotech.socialMediaApi.repository;
2
3 import org.springframework.data.repository.CrudRepository;
4
5 public interface PostRepository extends CrudRepository<Post, Long> {
6
7 }
8
9
10

```

```

1 package com.promineotech.socialMediaApi.repository;
2
3 import org.springframework.data.repository.CrudRepository;
4
5 public interface UserRepository extends CrudRepository<User, Long> {
6
7     public User findByUsername(String username);
8
9 }
10
11
12

```

```

1 package com.promineotech.socialMediaApi.service;
2
3 import java.util.Date;
4
5 @Service
6 public class CommentService {
7
8     @Autowired
9     private CommentRepository repo;
10
11     @Autowired
12     private PostRepository postRepo;
13
14     @Autowired
15     private UserRepository userRepo;
16
17     public Comment createComment(Comment comment, Long userId, Long postId) throws Exception {
18         User user = userRepo.findById(userId).orElseThrow();
19         Post post = postRepo.findById(postId).orElseThrow();
20         if (user == null || post == null) {
21             throw new Exception("User or Post does not exists.");
22         }
23
24         comment.setDate(new Date());
25         comment.setUser(user);
26         comment.setPost(post);
27         return repo.save(comment);
28     }
29
30     public void deleteComment(Long commentId) {
31         repo.deleteById(commentId);
32     }
33 }
34
35
36

```

```

PostService.java
1 package com.promineotech.socialMediaApi.service;
2
3 import java.util.Date; |
4
5
6
7
8
9
10
11
12
13
14
15 @Service
16 public class PostService {
17
18     @Autowired
19     private PostRepository repo;
20
21     @Autowired
22     private UserRepository userRepo;
23
24     public Iterable<Post> getAllPosts() {
25         return repo.findAll();
26     }
27
28     public Post getPost (Long id) {
29         return repo.findById(id).orElseThrow();
30     }
31
32     public Post updatePost(Post post, Long id) throws Exception {
33         Post foundPost = repo.findById(id).orElseThrow();
34         if (foundPost == null) {
35             throw new Exception("Post not found.");
36         }
37         foundPost.setContent(post.getContent());
38         return repo.save(foundPost);
39     }
40
41     public Post createPost (Post post, Long userId) throws Exception {
42         User user = userRepo.findById(userId).orElseThrow();
43         if (user == null) {
44             throw new Exception("User not found.");
45         }
46         post.setDate(new Date());
47         post.setUser(user);
48         return repo.save(post);
49     }
50 }
51
52

```

```

UserService.java
1 package com.promineotech.socialMediaApi.service;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4
5
6
7
8
9
10 @Service
11 public class UserService {
12
13     @Autowired
14     private UserRepository repo;
15
16     public User createUser(User user) {
17         return repo.save(user);
18     }
19
20     public User login(User user) throws Exception {
21         User foundUser = repo.findByUsername(user.getUsername());
22         if (foundUser != null && foundUser.getPassword().equals(user.getPassword())) {
23             return foundUser;
24         } else {
25             throw new Exception("Invalid username or password.");
26         }
27     }
28
29     public Following follow(Long userId, Long followId) throws Exception {
30         User user = repo.findById(userId).orElseThrow();
31         User follow = repo.findById(followId).orElseThrow();
32         if (user == null || follow == null) {
33             throw new Exception("User does not exist.");
34         }
35         user.getFollowing().add(follow);
36         repo.save(user);
37         return new Following(user);
38     }
39
40     public Following getFollowedUsers(Long userId) throws Exception {
41         User user = repo.findById(userId).orElseThrow();
42         if (user == null) {
43             throw new Exception("User does not exist.");
44         }
45         return new Following(user);
46     }
47
48     public User updateProfilePicture (Long userId, String url) throws Exception {
49         User user = repo.findById(userId).orElseThrow();
50         if (user == null) {
51             throw new Exception("User does not exist.");
52         }
53         user.setProfilePictureUrl(url);
54         return repo.save(user);
55     }
56 }
57
58

```



```
Following.java 83
1 package com.promineotech.socialMediaApi.view;
2
3 import java.util.Set;
4
5
6
7 public class Following {
8
9     private Set<User> following;
10
11     public Following(User user) {
12         following = user.getFollowing();
13     }
14
15     public Set<User> getFollowing() {
16         return following;
17     }
18
19     public void setFollowing(Set<User> following) {
20         this.following = following;
21     }
22 }
23
24
```

```
App.java 83
1 package com.promineotech.socialMediaApi;
2
3 import org.springframework.boot.SpringApplication;
4
5
6
7 @ComponentScan("com.promineotech.socialMediaApi")
8 @SpringBootApplication
9
10 public class App
11 {
12     public static void main ( String[] args )
13     {
14         SpringApplication.run(App.class, args);
15     }
16 }
17
```

Postman Screenshots:

http://localhost:8080/users/register

POST http://localhost:8080/users/register

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "username": "Tim",
3   ... "password": "12345"
4 }
```

Body Cookies Headers (5) Test Results Status: 201

Pretty Raw Preview Visualize JSON

```
1 {
2   ... "id": 1,
3   ... "username": "Tim",
4   ... "profilePictureUrl": null
5 }
```

http://localhost:8080/users/login

POST http://localhost:8080/users/login

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "username": "Tim",
3   ... "password": "12345"
4 }
```

Body Cookies Headers (5) Test Results Status

Pretty Raw Preview Visualize JSON

```
1 {
2   ... "id": 1,
3   ... "username": "Tim",
4   ... "profilePictureUrl": null
5 }
```

http://localhost:8080/users/register

POST http://localhost:8080/users/register

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "username": "Sally",
3   ... "password": "12345"
4 }
```

Body Cookies Headers (5) Test Results

Status: 201 Created

Pretty Raw Preview Visualize JSON

```
1 {
2   ... "id": 2,
3   ... "username": "Sally",
4   ... "profilePictureUrl": null
5 }
```

http://localhost:8080/users/1/follows/2

POST http://localhost:8080/users/1/follows/2

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   ... "username": "Sally",
3   ... "password": "12345"
4 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   ... "following": [
3     ... {
4       ... "id": 1,
5       ... "username": "Tim",
6       ... "profilePictureUrl": null
7     }
8   ]
9 }
```

http://localhost:8080/users/1/profilePicture

POST

http://localhost:8080/users/1/profilePicture

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

	KEY	VALUE
<input checked="" type="checkbox"/>	file	IMG_5690.jpg ×
	Key	Value

Body Cookies Headers (5) Test Results

Pretty

Raw

Preview

Visualize

JSON



```
1 {
2   ... "id": 1,
3   ... "username": "Tim",
4   ... "profilePictureUrl": "./pictures/IMG_5690.jpg"
5 }
```

http://localhost:8080/users/1/posts

POST

http://localhost:8080/users/1/posts

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON**

```
1 {
2   ... "content": "This is my frist post!"
3 }
```

Body Cookies Headers (5) Test Results

Pretty

Raw

Preview

Visualize

JSON



```
1 {
2   ... "id": 3,
3   ... "content": "This is my frist post!",
4   ... "date": "2021-01-29T17:10:25.691+00:00",
5   ... "user": {
6     ... "id": 1,
7     ... "username": "Tim",
8     ... "profilePictureUrl": "./pictures/IMG_5690.jpg"
9   },
10  ... "comments": null
11 }
```

http://localhost:8080/users/1/posts

GET http://localhost:8080/users/1/posts

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1
2  ...."content": "This is my frist post!"
3
```

Body Cookies Headers (5) Test Results

Status: 200

Pretty Raw Preview Visualize JSON

```
1 {
2   ....{
3     ...."id": 3,
4     ...."content": "This is my frist post!",
5     ...."date": "2021-01-29T17:10:25.691+00:00",
6     ...."user": {
7       ...."id": 1,
8       ...."username": "Tim",
9       ...."profilePictureUrl": "./pictures/IMG_5690.jpg"
10    ....},
11    ...."comments": []
12  ....}
13 }
```

http://localhost:8080/users/1/posts/3

PUT http://localhost:8080/users/1/posts/3

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1
2  ...."content": "I am updating this post!"
3
```

Body Cookies Headers (5) Test Results



Pretty Raw Preview Visualize JSON

```
1 {
2   ...."id": 3,
3   ...."content": "I am updating this post!",
4   ...."date": "2021-01-29T17:10:25.691+00:00",
5   ...."user": {
6     ...."id": 1,
7     ...."username": "Tim",
8     ...."profilePictureUrl": "./pictures/IMG_5690.jpg"
9   ....},
10  ...."comments": []
11 }
```

http://localhost:8080/users/2/posts/3/comments

POST http://localhost:8080/users/2/posts/3/comments

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "content": "first comment on ths post"
3 }
```

Body Cookies Headers (5) Test Results

Sta

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 4,
3   "content": "first comment on ths post",
4   "date": "2021-01-29T17:11:45.643+00:00",
5   "user": {
6     "id": 2,
7     "username": "Sally",
8     "profilePictureUrl": null
9   }
10 }
```

http://localhost:8080/users/1/posts/

GET http://localhost:8080/users/1/posts/

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "content": "first comment on ths post"
3 }
```

Body Cookies Headers (5) Test Results

Stat

Pretty Raw Preview Visualize JSON

```
1 {
2   {
3     "id": 3,
4     "content": "I am updating this post!",
5     "date": "2021-01-29T17:10:25.691+00:00",
6     "user": {
7       "id": 1,
8       "username": "Tim",
9       "profilePictureUrl": "./pictures/IMG_5690.jpg"
10    },
11    "comments": [
12      {
13        "id": 4,
14        "content": "first comment on ths post",
15        "date": "2021-01-29T17:11:45.643+00:00",
16        "user": {
17          "id": 2,
18          "username": "Sally",
19          "profilePictureUrl": null
20        }
21      }
22    ]
23  }
24 }
```

http://localhost:8080/users/1/posts/3/comments/4


DELETE http://localhost:8080/users/1/posts/3/comments/4

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON** ▼

```
1 {
2   ... "content": "first comment on ths post"
3 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize Text ▼ 

1 Deleted comment with id:4

http://localhost:8080/users/1/posts/3/


GET http://localhost:8080/users/1/posts/3/

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON**

```
1 {
2   ... "content": "first comment on ths post"
3 }
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize **JSON** ▼ 

```
1 {
2   ... "id": 3,
3   ... "content": "I am updating this post!",
4   ... "date": "2021-01-29T17:10:25.691+00:00",
5   ... "user": {
6     ... "id": 1,
7     ... "username": "Tim",
8     ... "profilePictureUrl": ". /pictures/IMG_5690.jpg"
9   },
10  ... "comments": []
11 }
```

<https://github.com/Christinalytle/SocialMediaApi.git>