

# Social Media Application

*Submitted for the course :*

*SmartBridge- Modern Application Development (Java Spring Boot)*

*Submitted By:*

*Tirthankar Chakraborty(20MIS0206)*

*S Pooja (20MIS0037)*

*Piyush Rana(20MIS0410)*

*Anjali Jha( 20MIS0124)*

# **CONTENTS**

## 1.Introduction

### 1.1) Overview

### 1.2)Purpose

## 2.Literature Survey

### 2.1)Existing Problem

### 2.2)Proposed Solution

## 3.Theoretical Analysis

### 3.1)Block Diagram

### 3.2)Hardware and Software Specification

## 4.Experimental Investigations

## 5. Flowchart

## 6.Result

## 7.Advantages and Disadvantages

## 8.Applications

## 9.Conclusion

## 10.Future Scope

## 11.Bibliography

# Introduction

## Overview

This project provides a simple solution to create a platform for connecting people all over the world with ease and securely in the form of a simple social media application. Users of the application would be able to post about their interests and other users would be able to show their appreciation to the post by sharing likes and comments. Similarly stories about the user can also be shared on the app by the user. Users can connect to other users by means of “following” them, which would be reflected in the users’ accounts as following and other users who are following the user would be reflected in the users’ account as followers.

It follows a loosely coupled architecture, combining Java Spring Boot for the backend API and React for the frontend web application. MySQL serves as the database to store movie data and user reviews. Future deployment using Docker and Kubernetes ensures scalability and reliability.

## Purpose

The purpose of a social media app like our social media app is to provide a platform for users to share and discover visual content, primarily through photos and videos. It allows users to create a profile, connect with others, and engage with content by liking, commenting, and sharing. Our Social media app aims to foster creativity, self-expression, and social interaction, enabling users to showcase their lives, interests, and talents, while also staying connected with friends, celebrities, brands, and communities of shared interests. It serves as a medium for communication, entertainment, inspiration, and building relationships through visual storytelling in a visually-centric and engaging environment.

# Literature Review

## Existing problems

As the user base grows, social media platforms need to ensure their infrastructure can handle the increasing load of users, data, and interactions. Scaling the app's servers, databases, and networking systems to maintain performance and responsiveness can be a significant technological challenge.

User Privacy and Data security is also major concern that revolves around the usage of social media apps

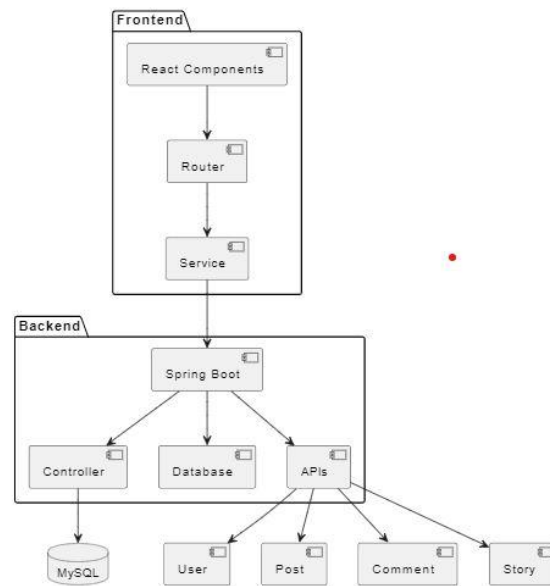
## Proposed Solution

To address the existing problem, our project proposes the development of a social media application. Our solution aims to provide a centralized platform that offers the users to connect with their peers and loved ones in a digital form enabling self-expression and social interaction . By combining Java Spring Boot for the backend API, React for the frontend web application, and MySQL as the database, our proposed solution provides a seamless and user-friendly experience.

Through our proposed solution we intend to diminish the existing problems by promoting scalability and security using spring security and Docker and Kubernetes.

# Theoretical Analysis

## Block Diagram



---

## Hardware / Software designing

### Hardware Requirements

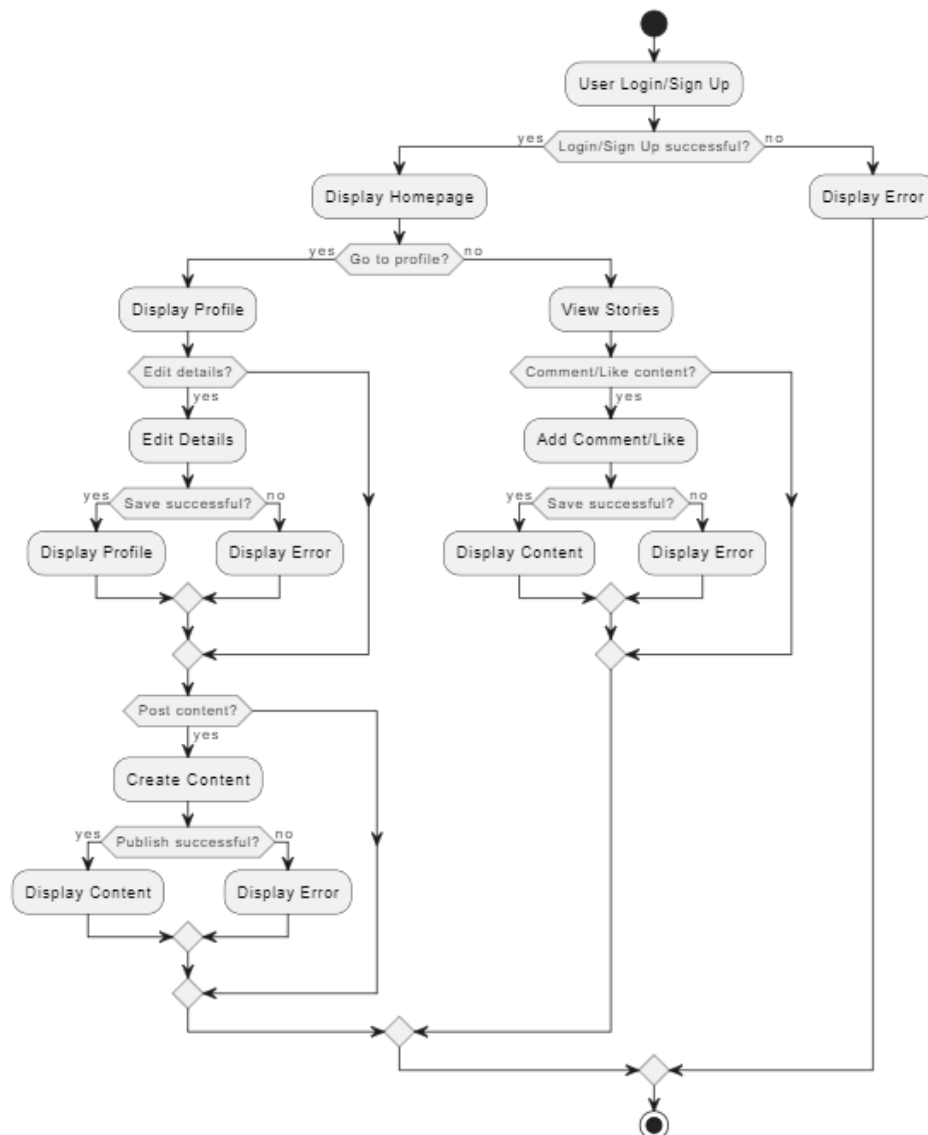
### Software Requirements

1. JDK
2. Maven
3. spring tool suite
4. mysql
5. Bootstrap
6. React
7. Axios

# Experimental Investigations

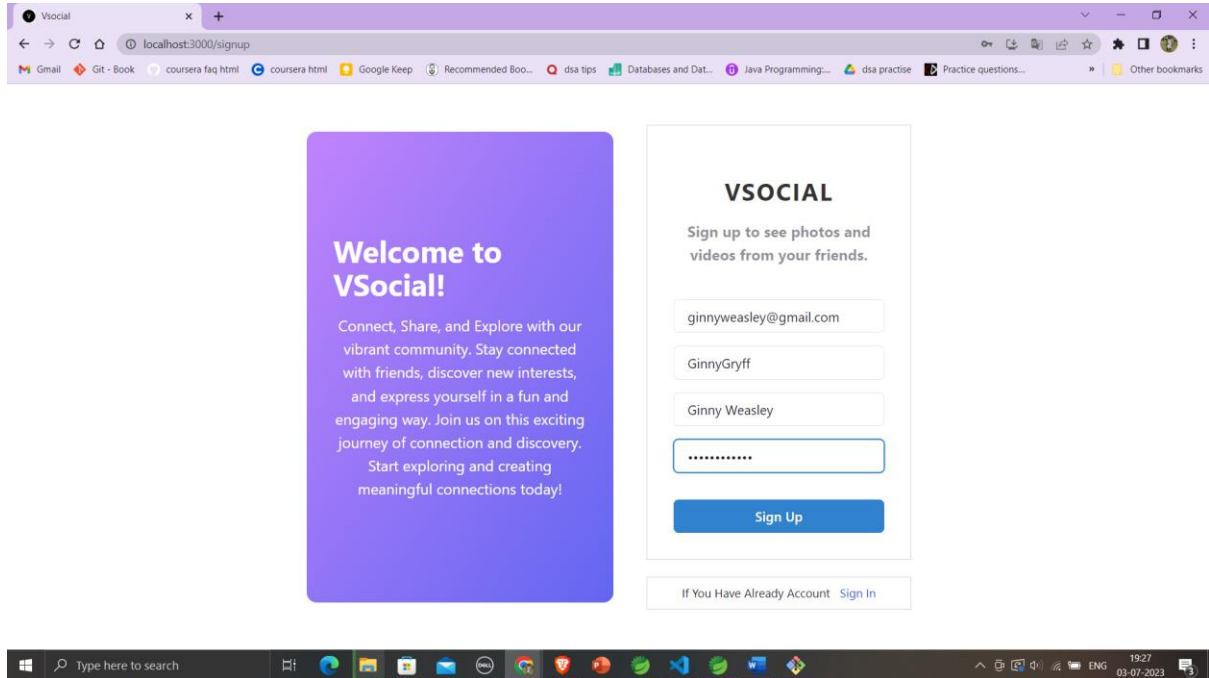
Analysis or the investigation made while working on the solution.

## Flowchart

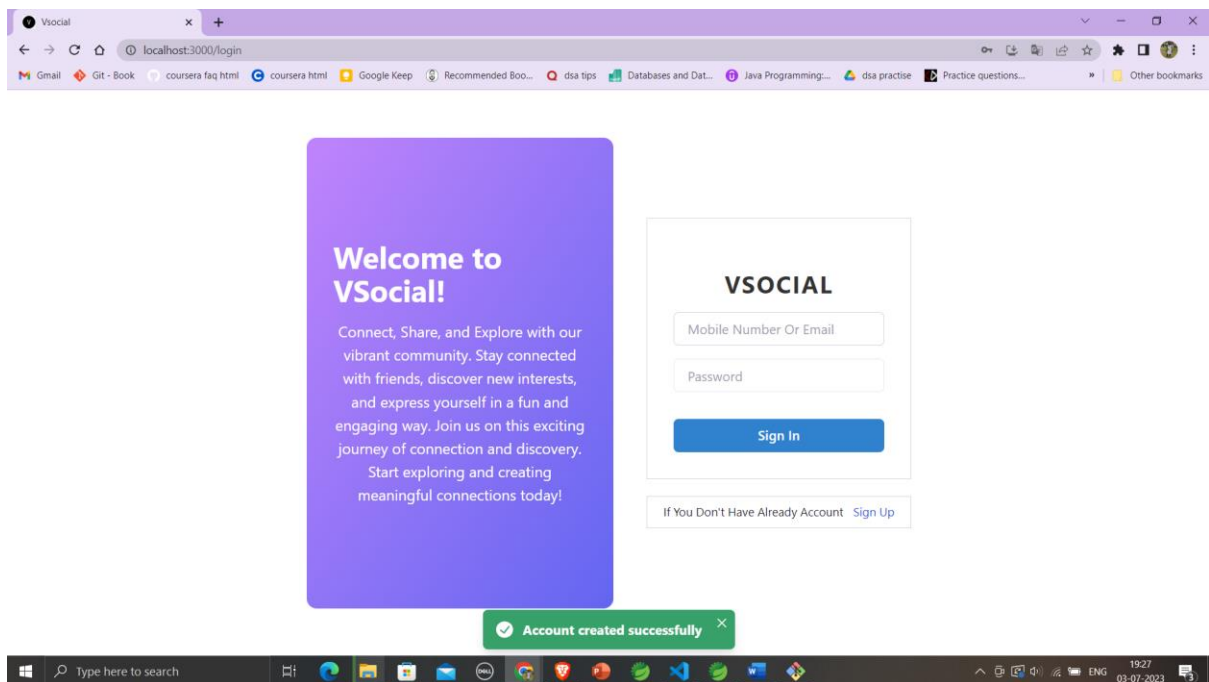


# Reults

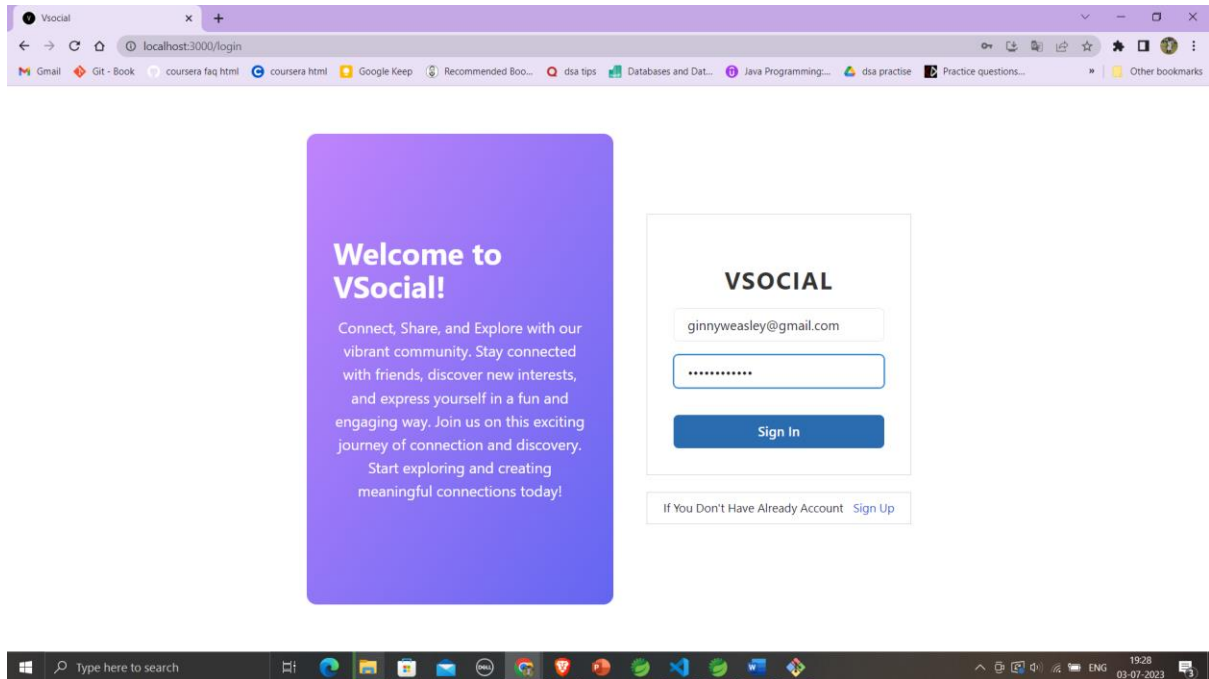
## Signup:



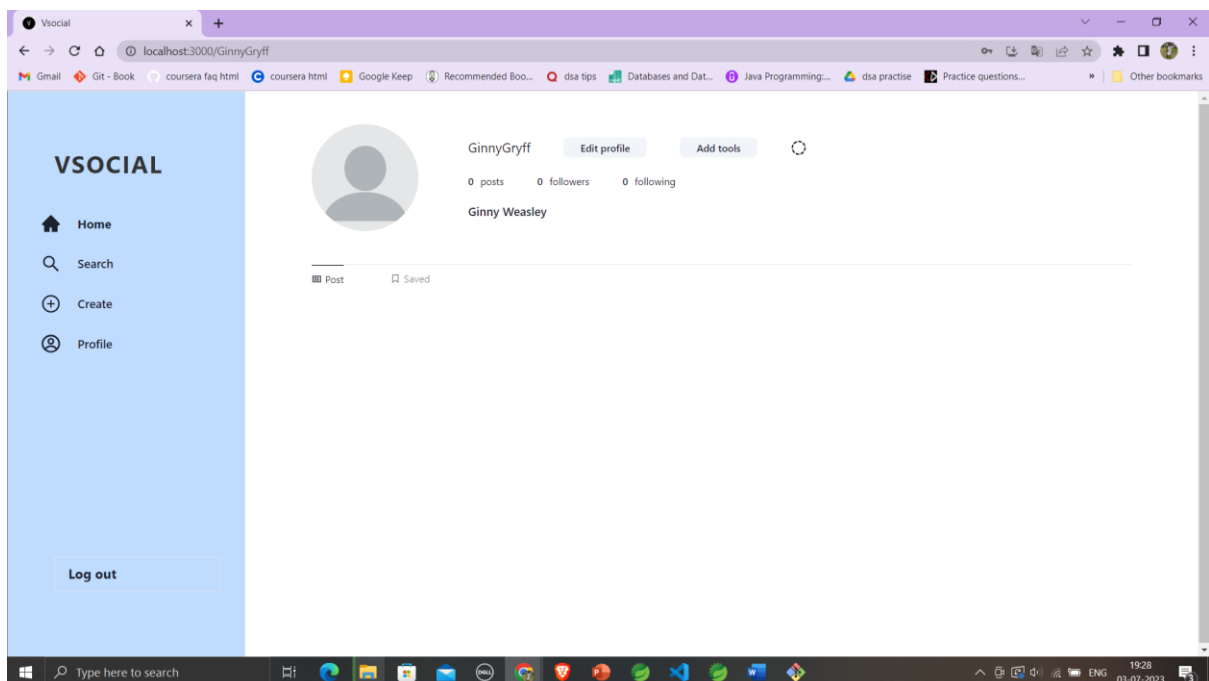
## On successful creation of account:



## Signin:

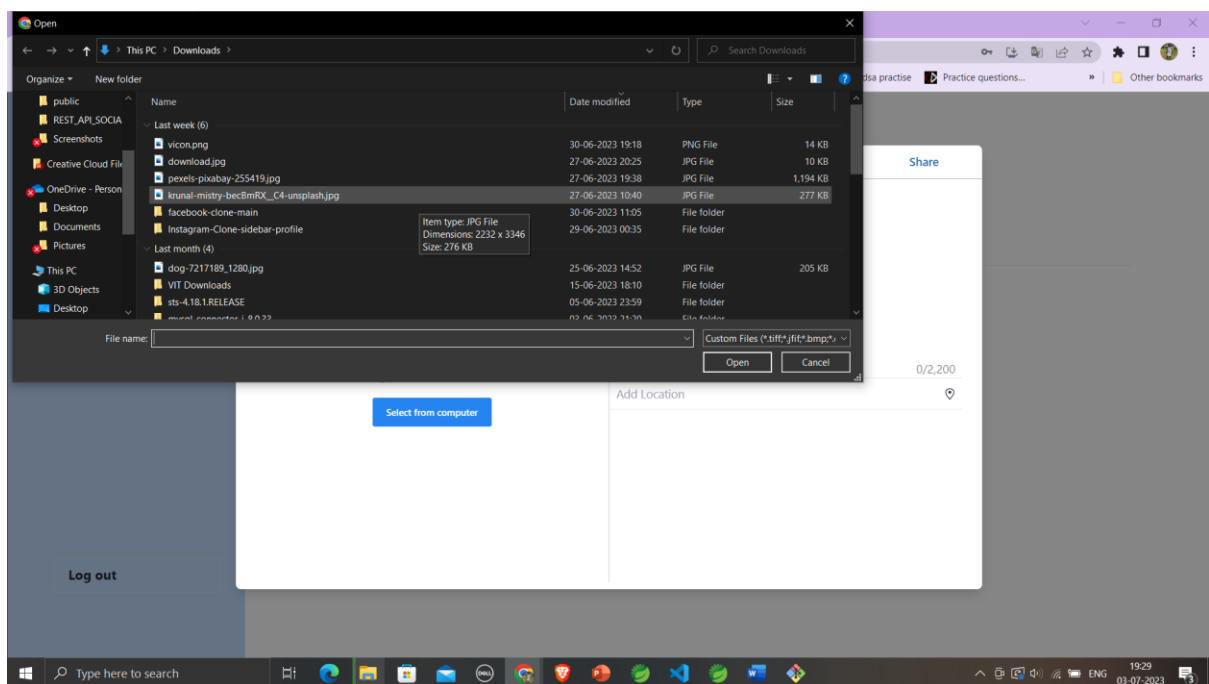
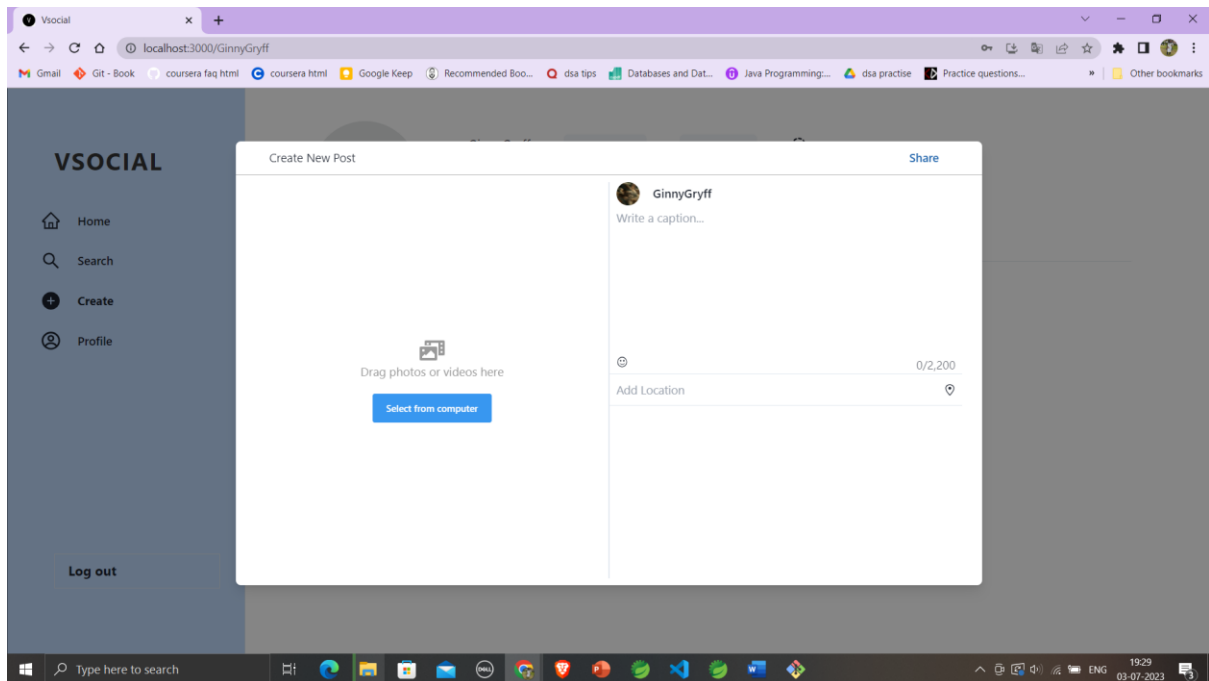


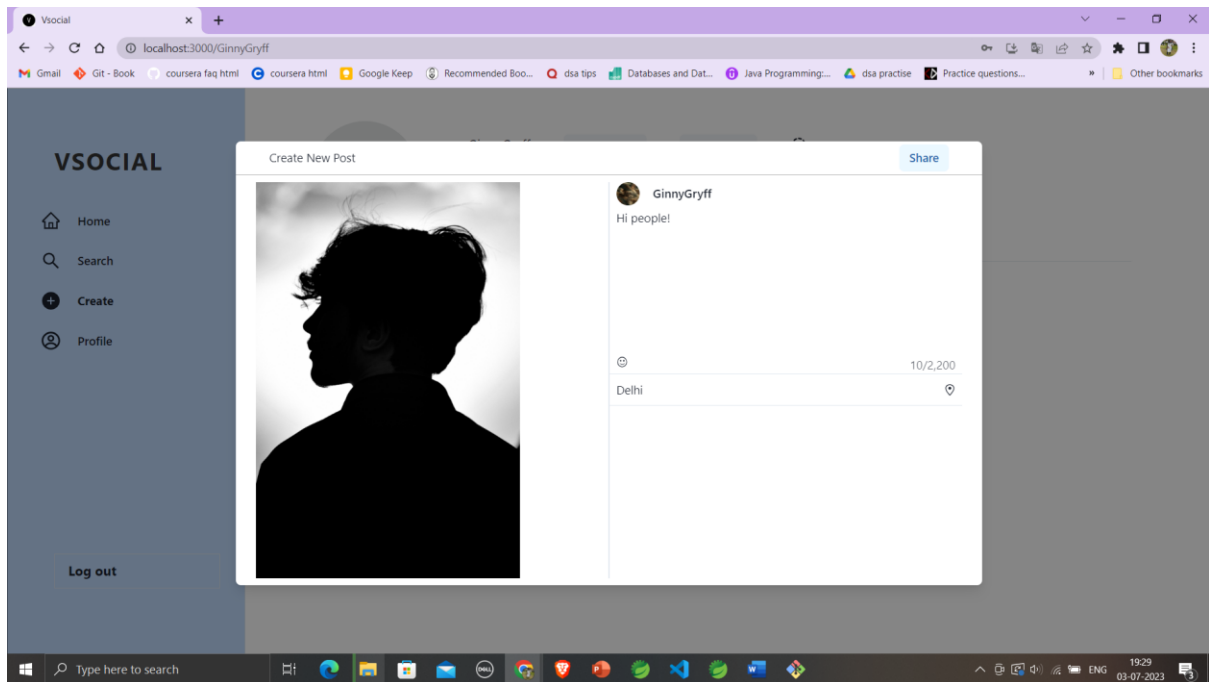
## Initial profile page:



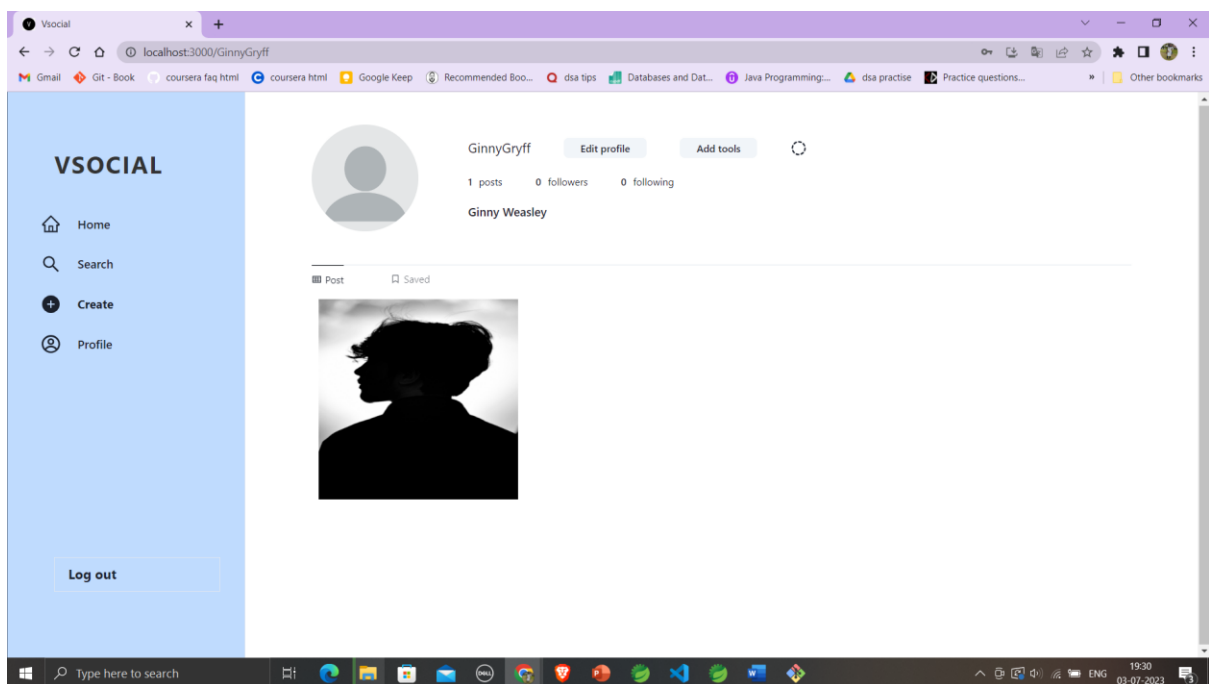


# Posting a new photo/video:

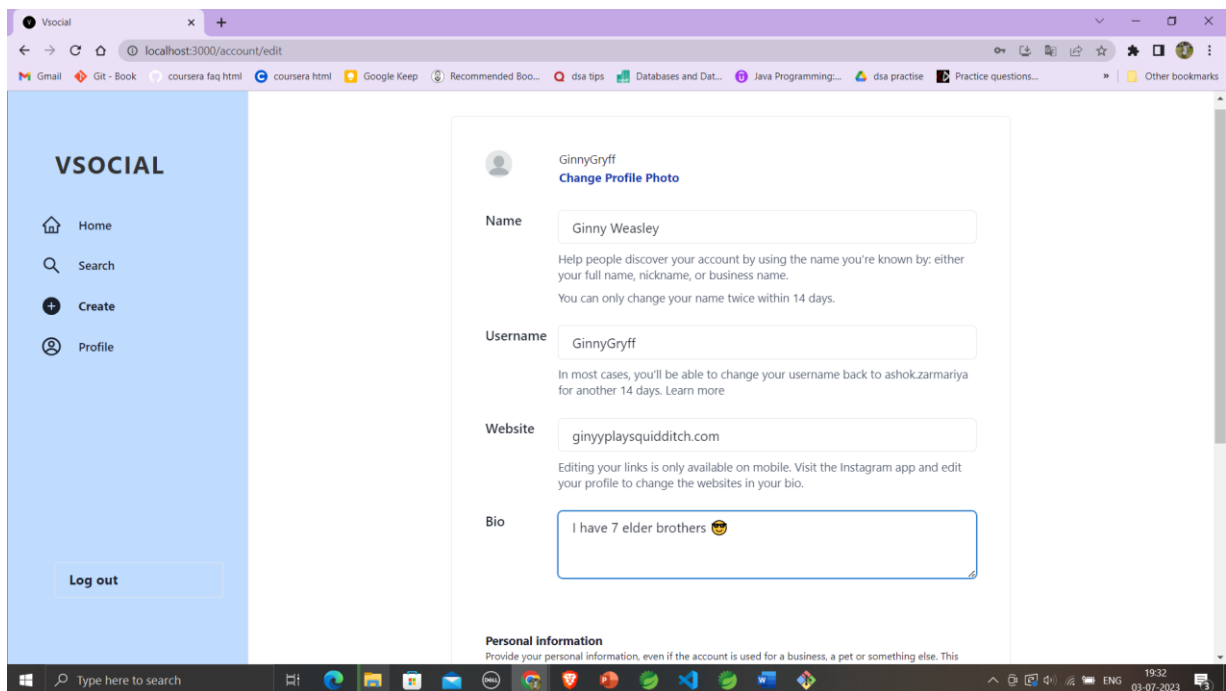
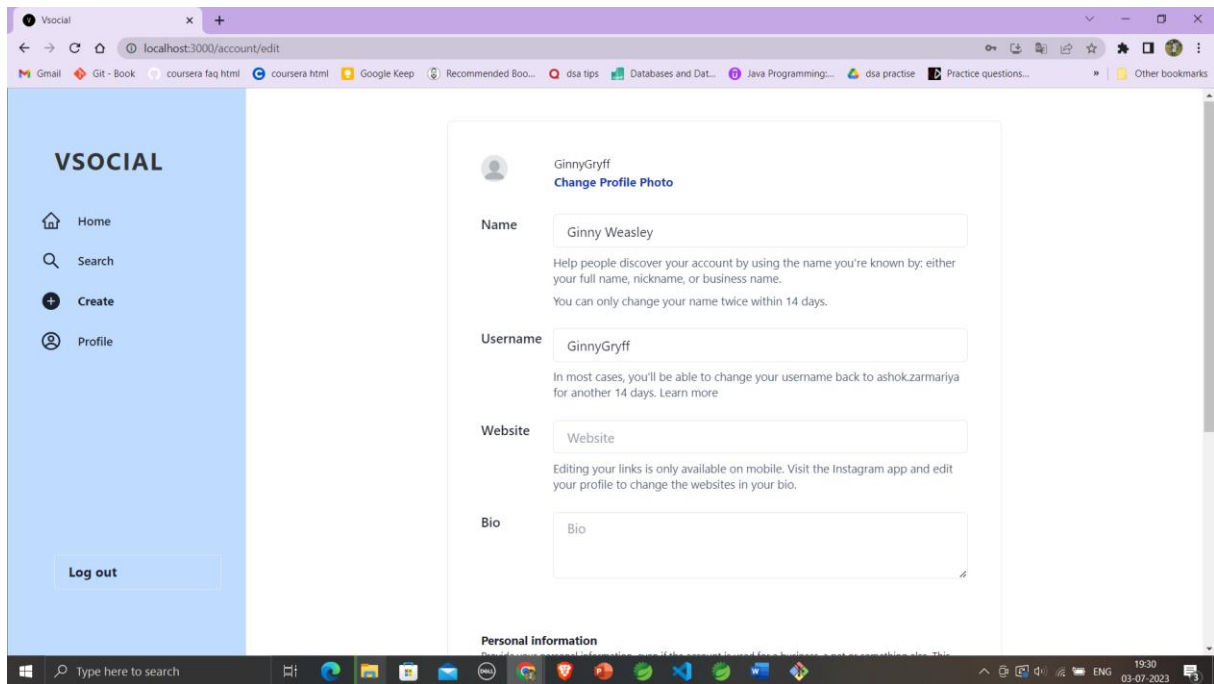


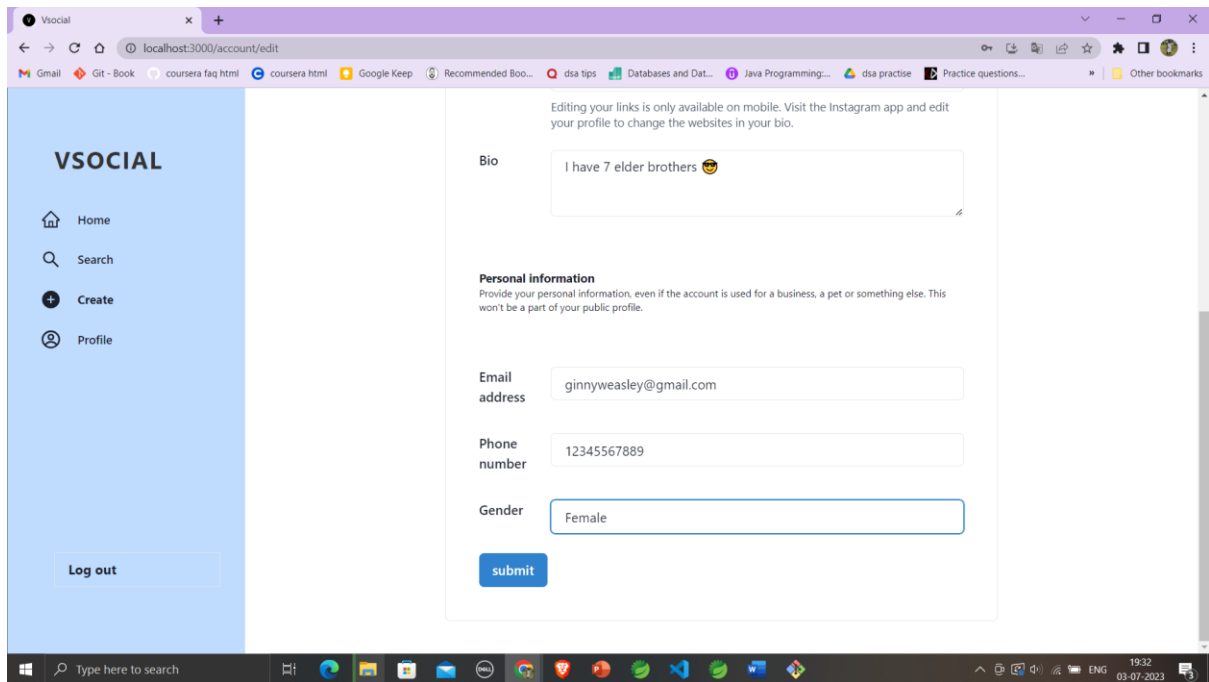


Updated profile page:

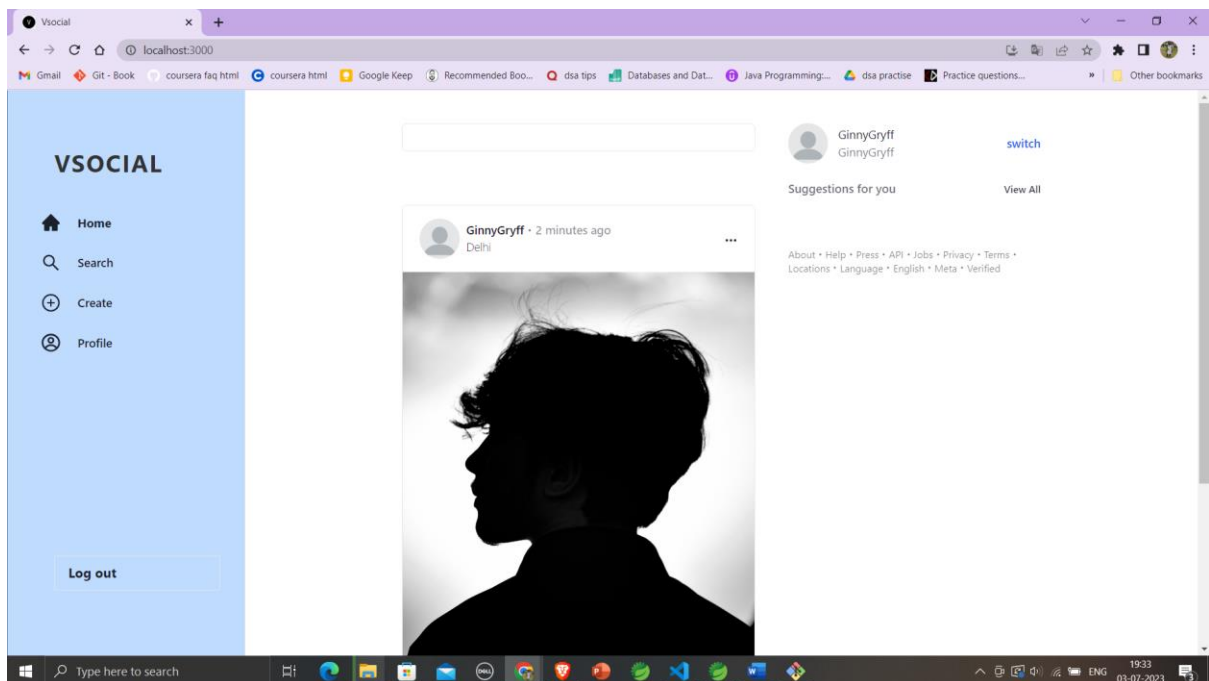


Editing profile:

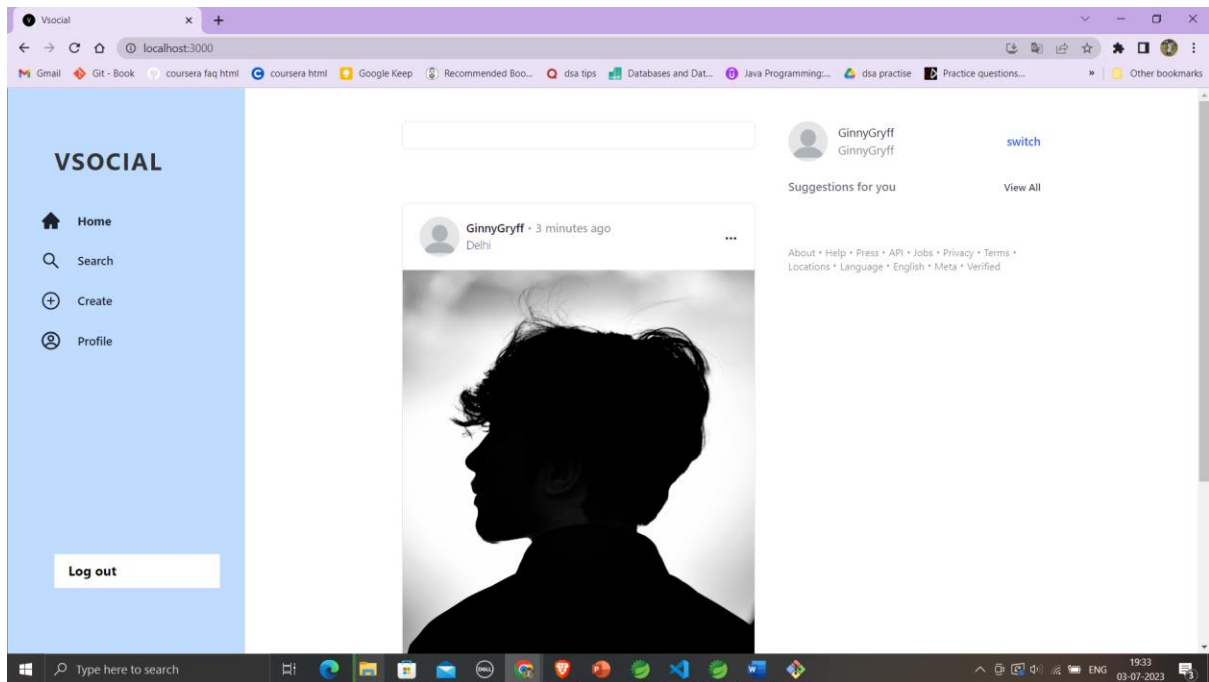




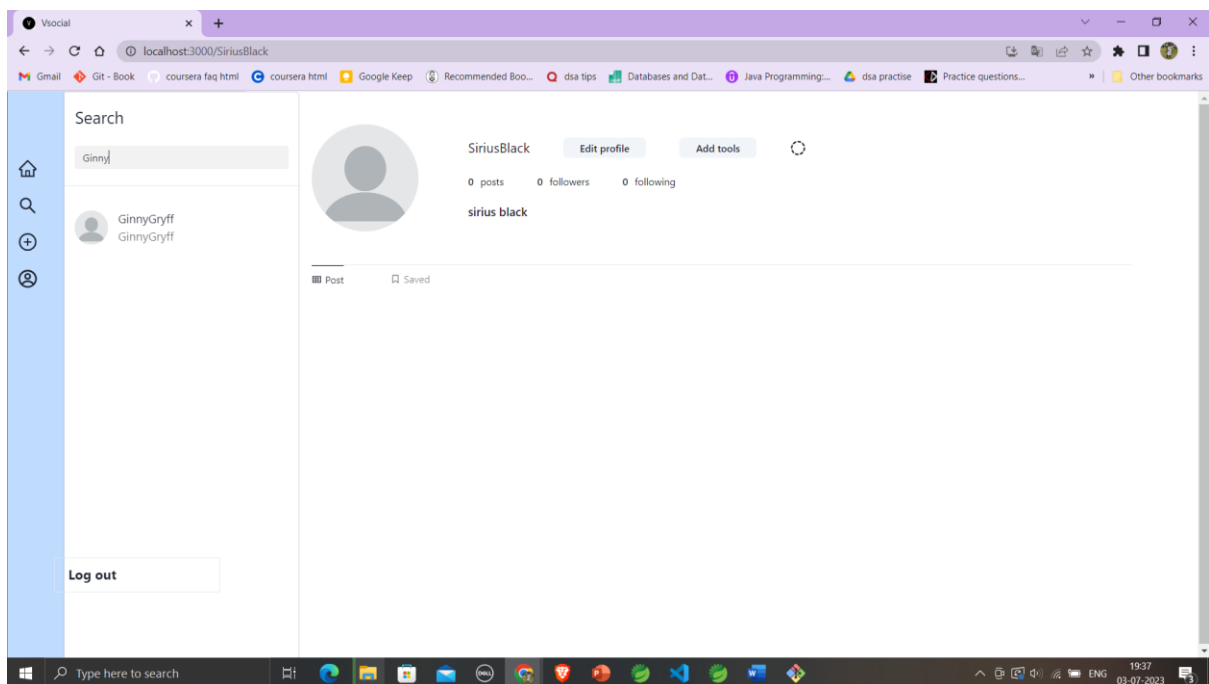
Home page:



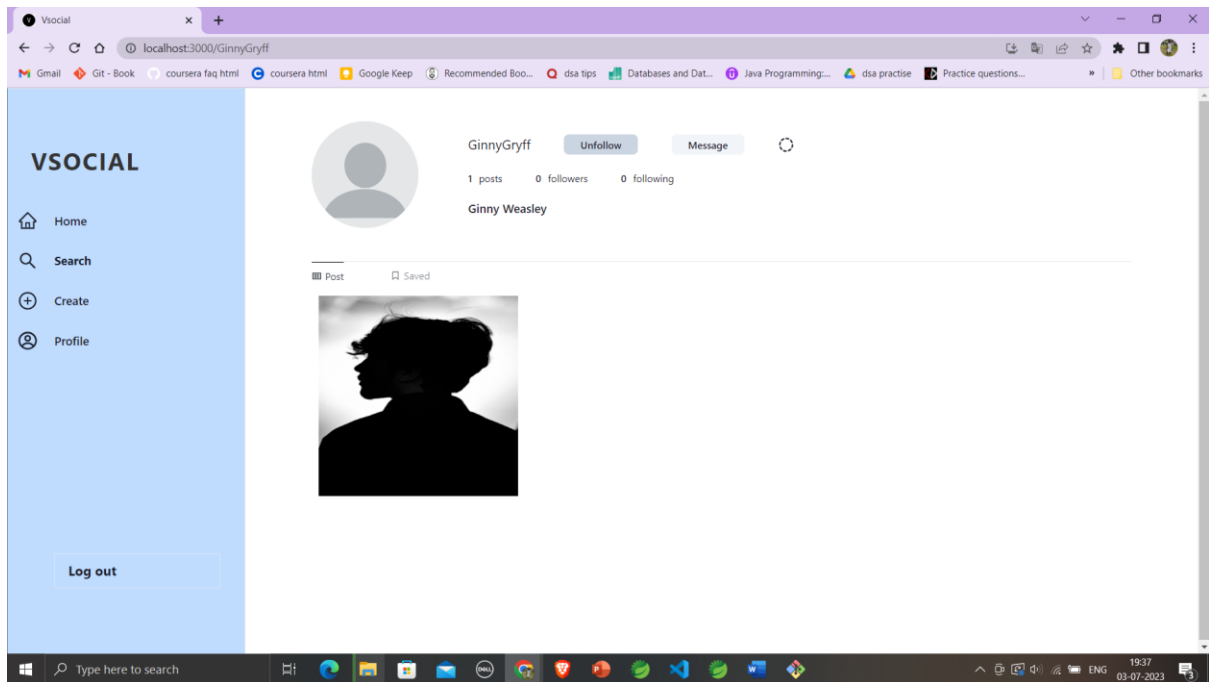
Logging out:



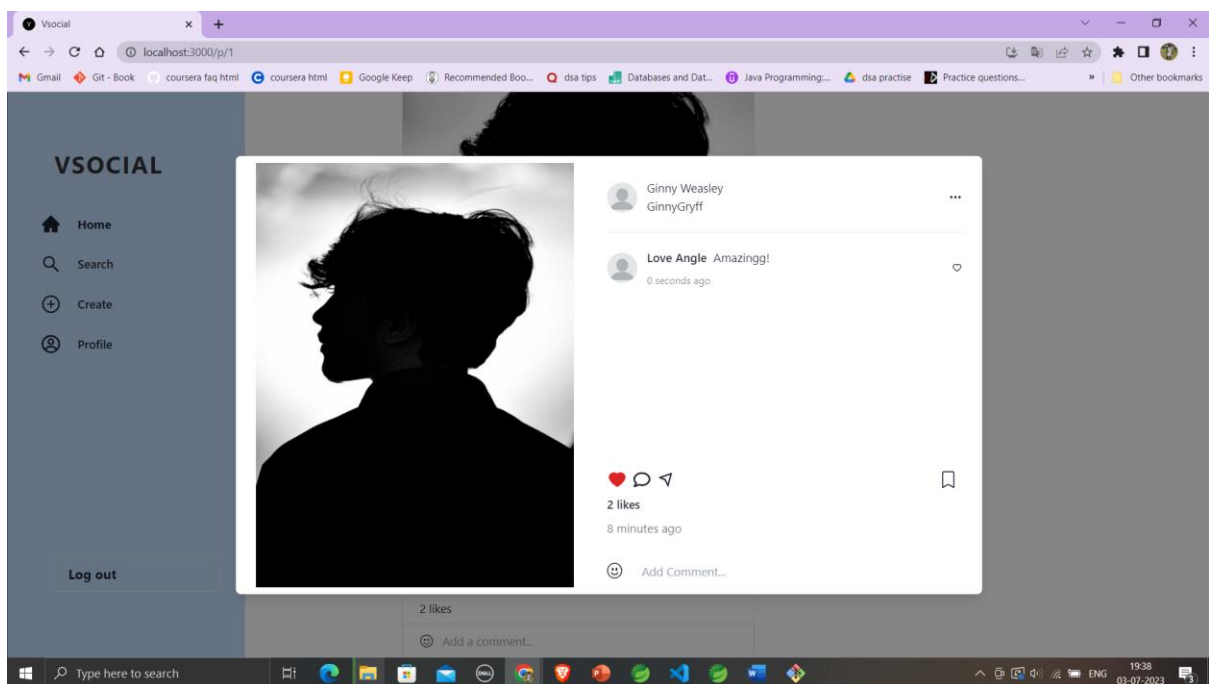
Searching other profiles:



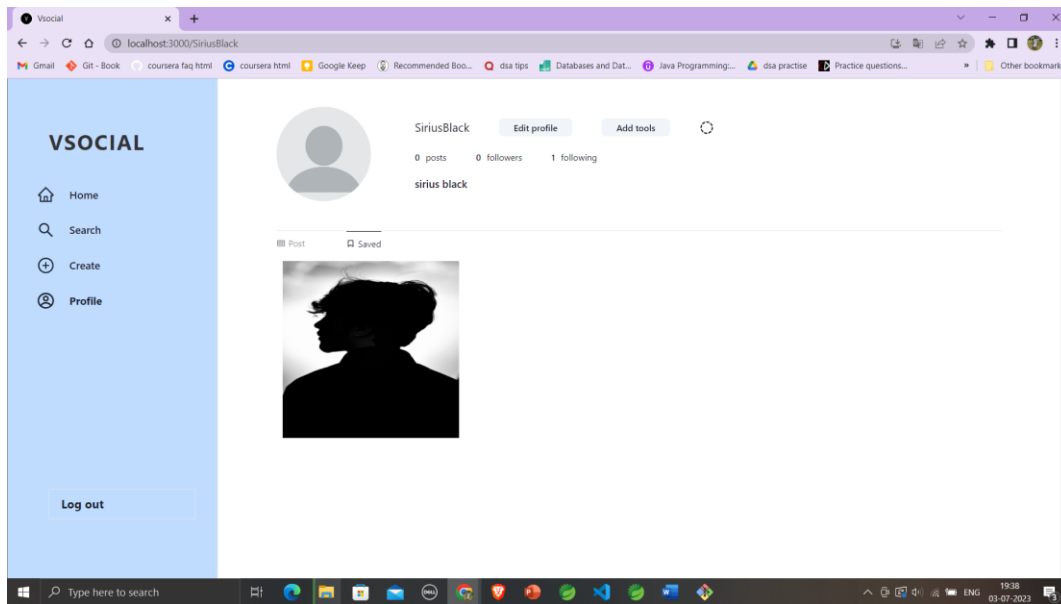
Following other accounts:



Liking and commenting:



Viewing saved posts:



Backend:

Files structure:

instagram (in server) [boot] [devtools]

src/main/java

com.zos

InstagramApplication.java

com.zos.config

AppConfig.java

JwtGenratorFilter.java

JwtValidationFilter.java

SecurityContest.java

com.zos.controller

AuthController.java

CommentController.java

HomeController.java

PostController.java

StoryController.java

UserController.java

com.zos.dto

UserDto.java

com.zos.exception

CommentException.java

ErrorDetails.java

GlobleException.java

PostException.java

StoryException.java

UserException.java

com.zos.model

Comments.java

Post.java

Story.java

User.java

com.zos.repository

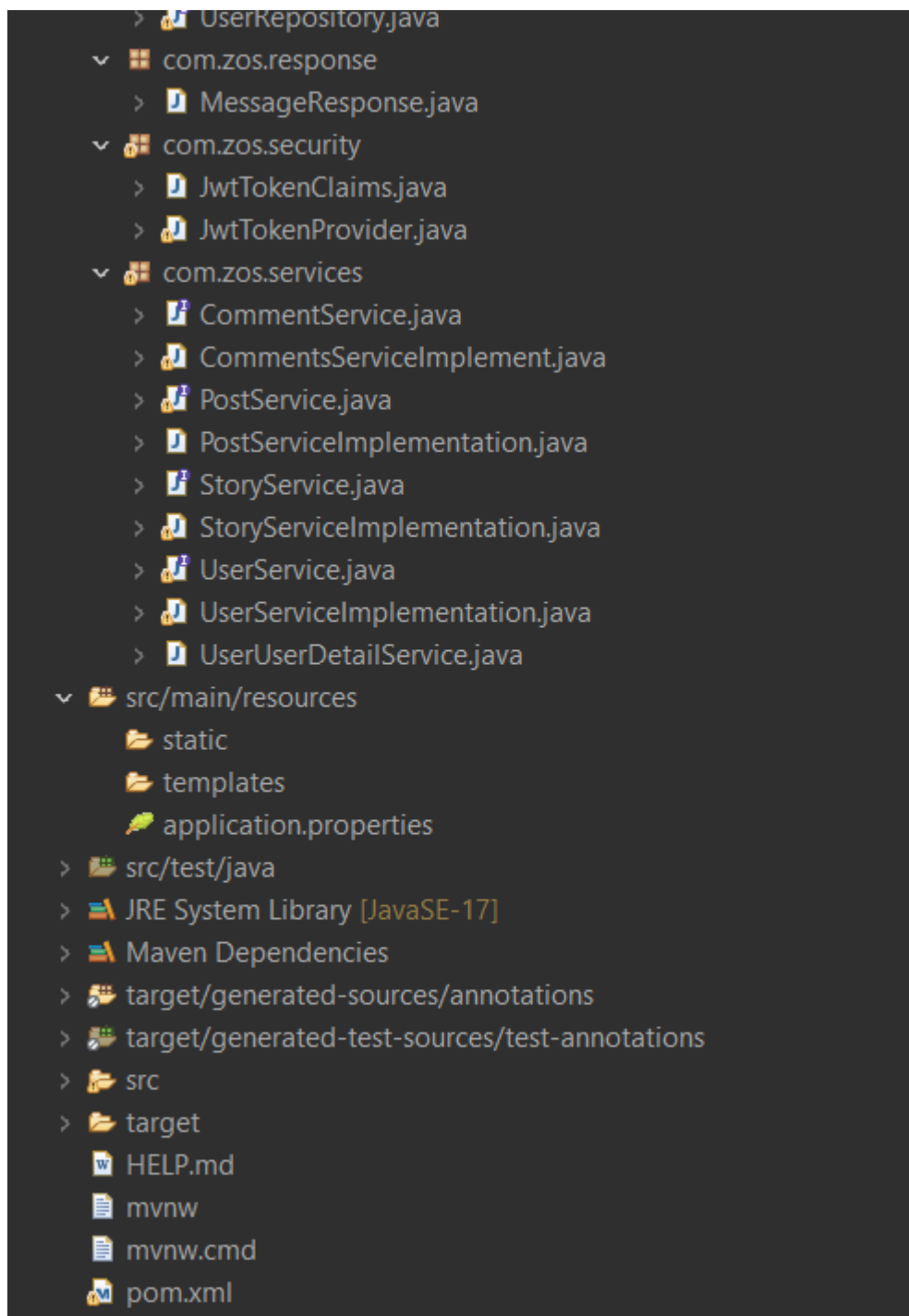
CommentRepository.java

PostRepository.java

StoryRepository.java

UserRepository.java



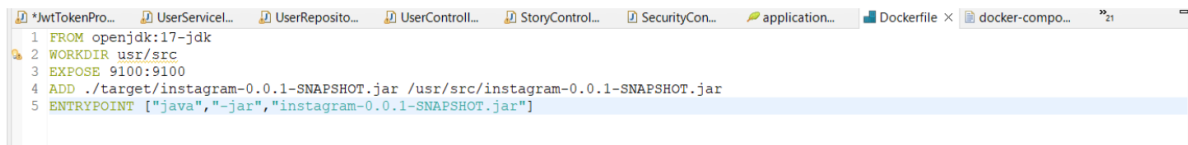


Usercontroller:



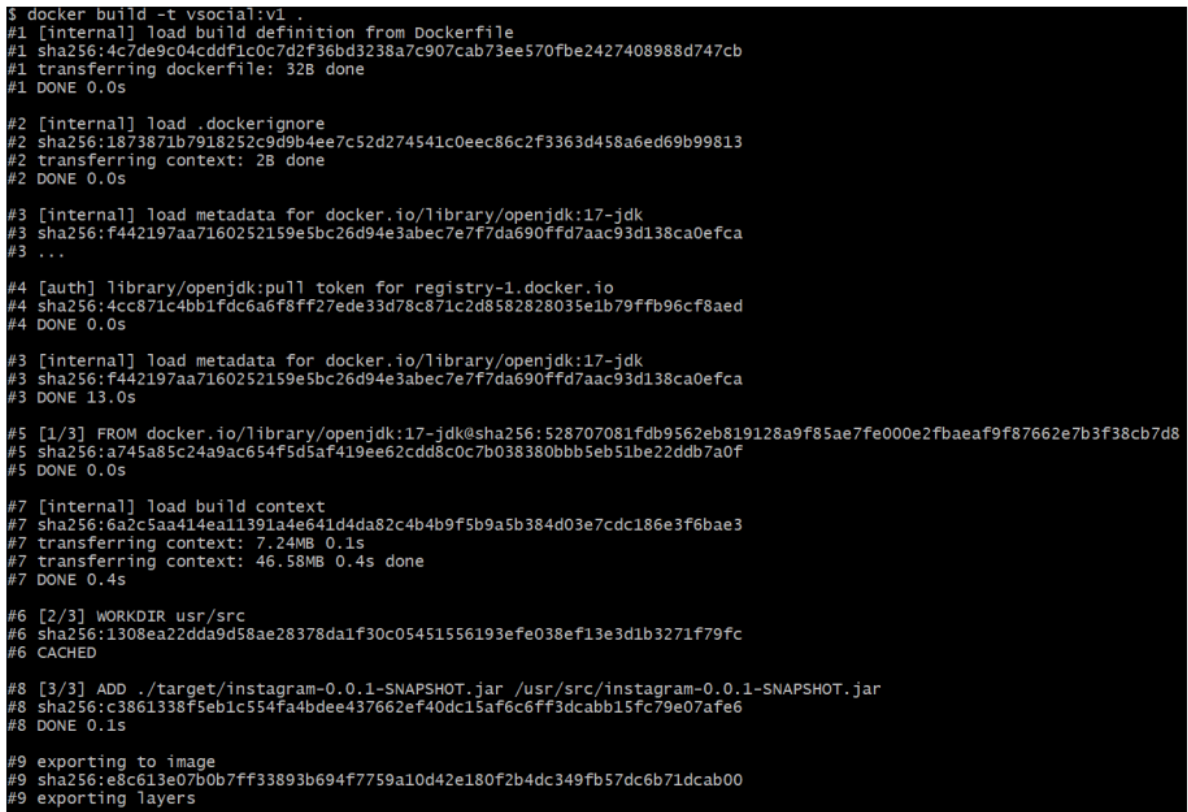
# Docker:

## Creating dockerfile:

A screenshot of a code editor showing a Dockerfile. The tabs at the top include \*JwtTokenPro..., UserServicel..., UserReposito..., UserControll..., StoryControl..., SecurityCon..., application..., Dockerfile x, and docker-compo... The Dockerfile content is as follows:

```
1 FROM openjdk:17-jdk
2 WORKDIR usr/src
3 EXPOSE 9100:9100
4 ADD ./target/instagram-0.0.1-SNAPSHOT.jar /usr/src/instagram-0.0.1-SNAPSHOT.jar
5 ENTRYPOINT ["java", "-jar", "instagram-0.0.1-SNAPSHOT.jar"]
```

docker build -t vsocial:v1 command:

A terminal window showing the output of the command 'docker build -t vsocial:v1 .'. The output is as follows:

```
$ docker build -t vsocial:v1 .
#1 [internal] load build definition from Dockerfile
#1 sha256:4c7de9c04cddf1c0c7d2f36bd3238a7c907cab73ee570fbe2427408988d747cb
#1 transferring dockerfile: 32B done
#1 DONE 0.0s

#2 [internal] load .dockerignore
#2 sha256:1873871b7918252c9d9b4ee7c52d274541c0eec86c2f3363d458a6ed69b99813
#2 transferring context: 2B done
#2 DONE 0.0s

#3 [internal] load metadata for docker.io/library/openjdk:17-jdk
#3 sha256:f442197aa7160252159e5bc26d94e3abec7e7f7da690ffd7aac93d138ca0efca
#3 ...

#4 [auth] library/openjdk:pull token for registry-1.docker.io
#4 sha256:4cc871c4bb1fdc6a6f8ff27ede33d78c871c2d8582828035e1b79ffb96cf8aed
#4 DONE 0.0s

#3 [internal] load metadata for docker.io/library/openjdk:17-jdk
#3 sha256:f442197aa7160252159e5bc26d94e3abec7e7f7da690ffd7aac93d138ca0efca
#3 DONE 13.0s

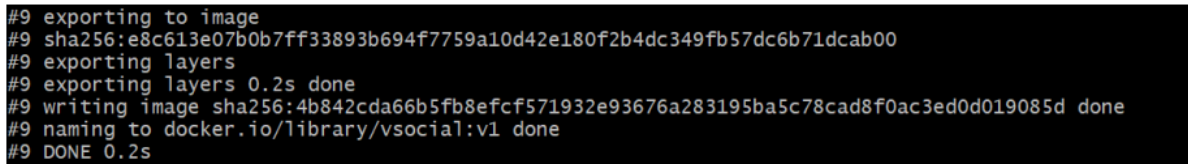
#5 [1/3] FROM docker.io/library/openjdk:17-jdk@sha256:528707081fdb9562eb819128a9f85ae7fe000e2fbaea9f987662e7b3f38cb7d8
#5 sha256:a745a85c24a9ac654f5d5af419ee62cdd8c0c7b038380bbb5eb51be22ddb7a0f
#5 DONE 0.0s

#7 [internal] load build context
#7 sha256:6a2c5aa414ea11391a4e641d4da82c4b4b9f5b9a5b384d03e7cdc186e3f6bae3
#7 transferring context: 7.24MB 0.1s
#7 transferring context: 46.58MB 0.4s done
#7 DONE 0.4s

#6 [2/3] WORKDIR usr/src
#6 sha256:1308ea22dda9d58ae28378da1f30c05451556193efe038ef13e3d1b3271f79fc
#6 CACHED

#8 [3/3] ADD ./target/instagram-0.0.1-SNAPSHOT.jar /usr/src/instagram-0.0.1-SNAPSHOT.jar
#8 sha256:c3861338f5eb1c554fa4bdee437662ef40dc15af6c6ff3dcabb15fc79e07afe6
#8 DONE 0.1s

#9 exporting to image
#9 sha256:e8c613e07b0b7ff33893b694f7759a10d42e180f2b4dc349fb57dc6b71dcab00
#9 exporting layers
#9 exporting layers 0.2s done
#9 writing image sha256:4b842cda66b5fb8efcf571932e93676a283195ba5c78cad8f0ac3ed0d019085d done
#9 naming to docker.io/library/vsocial:v1 done
#9 DONE 0.2s
```

A continuation of the terminal output from the previous block, showing the final steps of the build process:

```
#9 exporting to image
#9 sha256:e8c613e07b0b7ff33893b694f7759a10d42e180f2b4dc349fb57dc6b71dcab00
#9 exporting layers
#9 exporting layers 0.2s done
#9 writing image sha256:4b842cda66b5fb8efcf571932e93676a283195ba5c78cad8f0ac3ed0d019085d done
#9 naming to docker.io/library/vsocial:v1 done
#9 DONE 0.2s
```

pull the mysql image from docker hub using the command `docker pull mysql:5.5`

```
$ docker pull mysql:5.5
5.5: Pulling from library/mysql
743f2d6c1f65: Pulling fs layer
3f0c413ee255: Pulling fs layer
aef1ef8f1aac: Pulling fs layer
f9ee573e34cb: Pulling fs layer
3f237e01f153: Pulling fs layer
03da1e065b16: Pulling fs layer
04087a801070: Pulling fs layer
7efd5395ab31: Pulling fs layer
1b5cc03aaac8: Pulling fs layer
2b7adaec9998: Pulling fs layer
385b8f96a9ba: Pulling fs layer
f9ee573e34cb: Waiting
03da1e065b16: Waiting
04087a801070: Waiting
3f237e01f153: Waiting
7efd5395ab31: Waiting
1b5cc03aaac8: Waiting
2b7adaec9998: Waiting
385b8f96a9ba: Waiting
3f0c413ee255: Verifying Checksum
3f0c413ee255: Download complete
aef1ef8f1aac: Verifying Checksum
aef1ef8f1aac: Download complete
3f237e01f153: Verifying Checksum
3f237e01f153: Download complete
f9ee573e34cb: Verifying Checksum
f9ee573e34cb: Download complete
743f2d6c1f65: Download complete
7efd5395ab31: Download complete
1b5cc03aaac8: Verifying Checksum
1b5cc03aaac8: Download complete
743f2d6c1f65: Pull complete
3f0c413ee255: Pull complete
aef1ef8f1aac: Pull complete
f9ee573e34cb: Pull complete
3f237e01f153: Pull complete
2b7adaec9998: Verifying Checksum
2b7adaec9998: Download complete
03da1e065b16: Verifying Checksum
04087a801070: Verifying Checksum
04087a801070: Download complete
385b8f96a9ba: Verifying Checksum
```

```

2b7adaec9998: Download complete
03dale065b16: Verifying Checksum
04087a801070: Verifying Checksum
04087a801070: Download complete
385b8f96a9ba: Verifying Checksum
385b8f96a9ba: Download complete
03dale065b16: Pull complete
04087a801070: Pull complete
7efd5395ab31: Pull complete
1b5cc03aaac8: Pull complete
2b7adaec9998: Pull complete
385b8f96a9ba: Pull complete
Digest: sha256:12da85ab88aedfdf39455872fb044f607c32fdc233cd59f1d26769fbf439b045
Status: Downloaded newer image for mysql:5.5
docker.io/library/mysql:5.5

```

cross check whether the images have been created and pulled successfully using the command `docker images`:

```

$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
vsocial             v1                 4b842cda66b5       7 minutes ago      518MB
docker/desktop-git-helper
mysql               5.5               352f04e41a19       8 months ago       45.8MB
mysql               5.5               d404d78aa797       4 years ago        205MB

```

run the MySQL image with the command `docker run -p 3306:3306 -e MYSQL_ROOT_PASSWORD=root mysql: 5.5` :

```

$ docker run -p 3306:3306 -e MYSQL_ROOT_PASSWORD=root mysql:5.5
Initializing database
230703 5:50:42 [Note] Ignoring --secure-file-priv value as server is running with --bootstrap.
230703 5:50:42 [Note] /usr/local/mysql/bin/mysqld (mysqld 5.5.62) starting as process 69 ...
230703 5:50:42 [Note] Ignoring --secure-file-priv value as server is running with --bootstrap.
230703 5:50:42 [Note] /usr/local/mysql/bin/mysqld (mysqld 5.5.62) starting as process 75 ...

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, start the server, then issue the following commands:

/usr/local/mysql/bin/mysqladmin -u root password 'new-password'
/usr/local/mysql/bin/mysqladmin -u root -h password 'new-password'

Alternatively you can run:
/usr/local/mysql/bin/mysql_secure_installation

which will also give you the option of removing the test
databases and anonymous user created by default. This is
strongly recommended for production servers.

See the manual for more instructions.

Please report any problems at http://bugs.mysql.com/

Database initialized

```

```

MySQL init process done. Ready for start up.

230703 5:50:47 [Note] --secure-file-priv is set to NULL. Operations related to importing and exporting data are disabled
230703 5:50:47 [Note] mysqld (mysqld 5.5.62) starting as process 1 ...
230703 5:50:47 [Note] Plugin 'FEDERATED' is disabled.
230703 5:50:47 InnoDB: The InnoDB memory heap is disabled
230703 5:50:47 InnoDB: Mutexes and rw_locks use GCC atomic builtins
230703 5:50:47 InnoDB: Compressed tables use zlib 1.2.11
230703 5:50:47 InnoDB: Using Linux native AIO
230703 5:50:47 InnoDB: Initializing buffer pool, size = 128.0M
230703 5:50:47 InnoDB: Completed initialization of buffer pool
230703 5:50:47 InnoDB: highest supported file format is Barracuda.
230703 5:50:47 InnoDB: waiting for the background threads to start
230703 5:50:48 InnoDB: 5.5.62 started; log sequence number 1595675
230703 5:50:48 [Note] Server hostname (bind-address): '0.0.0.0'; port: 3306
230703 5:50:48 [Note] - '0.0.0.0' resolves to '0.0.0.0';
230703 5:50:48 [Note] Server socket created on IP: '0.0.0.0'.
230703 5:50:48 [Warning] 'proxies_priv' entry '@ root@126f6adab8bc' ignored in --skip-name-resolve mode.
230703 5:50:48 [Note] Event Scheduler: Loaded 0 events
230703 5:50:48 [Note] mysqld: ready for connections.
Version: '5.5.62' socket: '/tmp/mysql.sock' port: 3306 MySQL Community Server (GPL)

```



```
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
7539e69ae204	vsocial:v1	"java -jar instagram..."	3 minutes ago	Exited (1) 2 minutes ago	
83f31d4b3667	mysql:5.5	"docker-entrypoint.s..."	3 minutes ago	Up 3 minutes	0.0.0.0:3306->3306/tcp

## Successful connection:

```
$ docker run -p 9100:5454 vsocial:v1
```



```

:: Spring Boot ::
(v3.0.2)

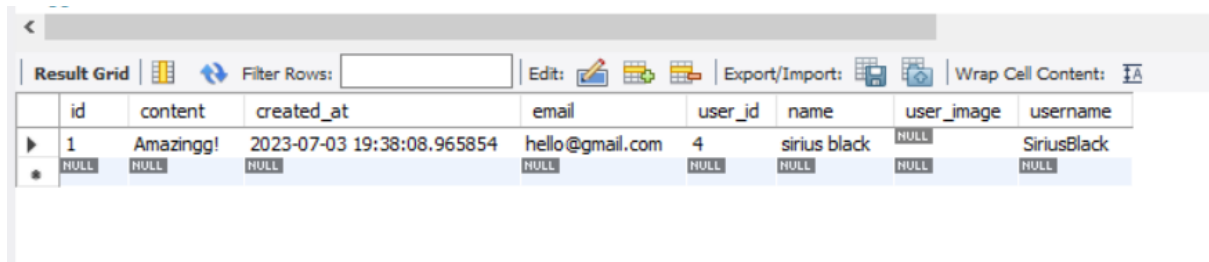
2023-07-03T06:06:49.688Z INFO 1 --- [main] com.zos.InstagramApplication : Starting InstagramApplication v0.0.1-SNAPSHOT using Java 17.0.2 with PID 1 (/usr/src/instagram-0.0.1-SNAPSHOT.jar started by root in /usr/src)
2023-07-03T06:06:49.691Z INFO 1 --- [main] com.zos.InstagramApplication : No active profile set, falling back to 1 default profile: "default"
2023-07-03T06:06:50.438Z INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2023-07-03T06:06:50.505Z INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 58 ms. Found 4 JPA repository interfaces.
2023-07-03T06:06:51.031Z INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 5454 (http)
2023-07-03T06:06:51.042Z INFO 1 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2023-07-03T06:06:51.043Z INFO 1 --- [main] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.5]
2023-07-03T06:06:51.123Z INFO 1 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2023-07-03T06:06:51.126Z INFO 1 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1381 ms
2023-07-03T06:06:51.289Z INFO 1 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
2023-07-03T06:06:51.338Z INFO 1 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 6.1.6.Final
2023-07-03T06:06:51.483Z WARN 1 --- [main] org.hibernate.orm.deprecation : HHH90000021: Encountered deprecated setting [javax.persistence.sharedCache.mode] instead [jakarta.persistence.sharedCache.mode]
2023-07-03T06:06:51.569Z INFO 1 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2023-07-03T06:06:51.752Z INFO 1 --- [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@1a1d3c1a
2023-07-03T06:06:51.754Z INFO 1 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2023-07-03T06:06:51.798Z INFO 1 --- [main] SQL dialect : HHH000400: using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: create table comments (id integer not null, content varchar(255) not null, created_at datetime, email varchar(255), user_id integer, name varchar(255), user_image varchar(255), use
Hibernate: create table comments_seq (next_val bigint) engine=InnoDB
Hibernate: insert into comments_seq values ( 1 )
Hibernate: create table like_by_users (user_id integer not null, email varchar(255), id integer, name varchar(255), user_image varchar(255), username varchar(255)) engine=InnoDB
Hibernate: create table users (id integer not null, bio varchar(255), email varchar(255), gender varchar(255), image varchar(255), mobile varchar(255), name varchar(255), password varchar(255), username varchar(255), website varchar(255), primary key (id)) engine=InnoDB
Hibernate: create table users_saved_post (user_id integer not null, saved_post_id integer not null) engine=InnoDB
Hibernate: insert into users_seq values ( 1 )
Hibernate: create table users_stories (user_id integer not null, stories_id integer not null) engine=InnoDB
Hibernate: alter table posts_comments drop index UK_sjeadiuvioecng9psjdcjqr
Hibernate: alter table posts_comments add constraint UK_sjeadiuvioecng9psjdcjqr unique (comments_id)
Hibernate: alter table users_stories drop index UK_xfqvpldcg0fyu5bvpv0maka
Hibernate: alter table users_stories add constraint UK_xfqvpldcg0fyu5bvpv0maka unique (stories_id)
Hibernate: alter table comments_liked_by_users add constraint FK6gna6c8qkdm215hk679asoyax foreign key (comments_id) references comments (id)
Hibernate: alter table like_by_users add constraint FK7di930sr48t6l138x75frbqno foreign key (user_id) references posts (id)
Hibernate: alter table posts_comments add constraint FKgfwb3n0tyt6x31ro7kavj8fu foreign key (comments_id) references comments (id)
Hibernate: alter table posts_comments add constraint FKbjdq8a62c5sivlmk27umswg9 foreign key (post_id) references posts (id)
Hibernate: alter table user_follower add constraint FK3lvprrct5cwjjo172deguk3y foreign key (user_id) references users (id)
Hibernate: alter table user_following add constraint FK1auj02dmroabw3nethnlnrye foreign key (user_id) references users (id)
Hibernate: alter table users_saved_post add constraint FKnlw7dr0j8swsex3cy5b7jcl foreign key (saved_post_id) references posts (id)
Hibernate: alter table users_saved_post add constraint FKqsrxvk32r6mgwffwylesk6lpf foreign key (user_id) references users (id)
Hibernate: alter table users_stories add constraint FKbugm5cf28mpgrljxcubduleq foreign key (stories_id) references stories (id)
Hibernate: alter table users_stories add constraint FKbrkpb5fknc6sgejyy06l3twk foreign key (user_id) references users (id)
2023-07-03T06:06:53.266Z INFO 1 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
2023-07-03T06:06:53.279Z INFO 1 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2023-07-03T06:06:54.230Z WARN 1 --- [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2023-07-03T06:06:54.782Z INFO 1 --- [main] o.s.w.s.DefaultSecurityFilterChain : Will secure any request with [org.springframework.security.web.session.DisableEncodeURIFilter@5939e24, org.springframework.security.web.context.request.async.WebAsyncManagerIntegrationFilter@595f9916, org.springframework.security.web.context.SecurityContextHolderFilter@6812c8cc, org.springframework.security.web.header.HeaderWriterFilter@32ae5f27, org.springframework.security.web.csrf.CorsFilter@205833f, org.springframework.security.web.authentication.logout.LogoutFilter@66634e9, org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter@93b0e0, com.zos.config.jwtValidationFilter@2768e25, org.springframework.security.web.authentication.www.BasicAuthenticationFilter@3dec3f87, com.zos.config.jwtGeneratorFilter@310d57b1, org.springframework.security.web.savedrequest.RequestCacheAwareFilter@7a66c35a, org.springframework.security.web.servletapi.SecurityContextHolderAwareRequestFilter@73e4387, org.springframework.security.web.authentication.AnonymousAuthenticationFilter@143fefaf, org.springframework.security.web.session.SessionManagementFilter@5b8853, org.springframework.security.web.access.ExceptionTranslationFilter@4fd74223, org.springframework.security.web.access.intercept.AuthorizationFilter@216c22ce]
2023-07-03T06:06:55.157Z INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 5454 (http) with context path ''
2023-07-03T06:06:55.188Z INFO 1 --- [main] com.zos.InstagramApplication : Started InstagramApplication in 5.891 seconds (process running for 6.42)

```

## Tables in backend:



## Comments table:



	id	content	created_at	email	user_id	name	user_image	username
▶	1	Amazingg!	2023-07-03 19:38:08.965854	hello@gmail.com	4	sirius black	NULL	SiriusBlack
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

## GitHub link for source code :

[https://github.com/Tirthankar2001/projectsmar  
tinternz](https://github.com/Tirthankar2001/projectsmartinternz)



# Advantages & Disadvantages

## Advantages

- 1.Responsive and Interactive User Interface: React's efficient rendering and virtual DOM enable a smooth and responsive user interface. Users can enjoy fast page load times, seamless transitions, and interactive features, enhancing their overall experience while using the app.
- 2.Real-Time Updates: React's ability to handle real-time updates makes it well-suited for social media apps. Users can receive instant notifications, updates, and new content without needing to manually refresh the page, keeping them engaged and informed.
- 3.Enhanced User Experience: The combination of React's component-based architecture and Spring Boot's rapid development capabilities can result in an app with a user-friendly and intuitive interface. Users can navigate easily, discover content, interact with posts, and connect with others, creating an enjoyable experience.
- 4.Security and Data Protection: With Spring Boot as the backend, user data can be stored securely, and appropriate measures can be implemented to protect user privacy. Robust authentication mechanisms and secure data handling practices can help instill user confidence and trust in the app.
- 5.Cross-Platform Accessibility: React's capability to build mobile-friendly interfaces ensures that the social media app can be accessed across various devices and platforms. Users can use the app on their desktops, smartphones, and tablets, providing flexibility and convenience.
- 6.Performance and Speed: React's efficient rendering and Spring Boot's optimized backend can contribute to improved performance and faster response times. Users can enjoy a snappy and lag-free experience while browsing, uploading content, or interacting with the app's features.

## Disadvantages

- 1.Learning Curve and Familiarity: If users are not accustomed to social media apps built with React and Spring Boot, they may require some time to understand the app's interface, navigation, and functionalities. This learning curve can initially present challenges for new users.
- 2.Compatibility and Technical Requirements: Users may need to ensure their devices meet the technical requirements for running the social media app smoothly. Compatibility issues or older devices may result in reduced performance or limited functionality.
- 3.Occasional Bugs and Updates: Like any software, social media apps can encounter bugs or require periodic updates to introduce new features or fix issues. Users may occasionally experience glitches, crashes, or temporary disruptions during updates.

4.Privacy Concerns: Users should remain vigilant about their privacy and data security while using any social media app. It is important to review and understand the app's privacy settings, data collection practices, and take appropriate measures to protect personal information.

5.Dependency on Internet Connectivity: Social media apps heavily rely on internet connectivity. Users may experience limitations in accessing or using the app in areas with poor network coverage or during network outages.

## Applications

1. Connecting with Friends and Family: Social media apps provide a platform for users to connect and stay in touch with their friends and family, regardless of geographical distances. It enables users to share updates, photos, and videos, and engage in real-time conversations, fostering stronger relationships and reducing feelings of isolation.

2. Sharing Life Moments: Social media apps allow users to share significant moments and experiences from their lives with their network. Users can post photos and videos, share their achievements, travel experiences, special occasions, and everyday moments, creating a digital diary of memories.

3. Discovering and Exploring Interests: Users can explore their interests and discover new content through social media apps. They can follow accounts, hashtags, and communities related to their hobbies, passions, or areas of interest. This facilitates the discovery of relevant and engaging content, such as articles, tutorials, art, music, and more.

4. Professional Networking and Career Growth: Social media apps offer opportunities for professional networking and career advancement. Users can connect with industry professionals, join relevant groups and communities, and showcase their skills and expertise. It can serve as a platform for job hunting, recruitment, and expanding professional networks.

5. Accessing News and Information: Social media apps can be a source of news and information. Users can follow news outlets, journalists, and influencers to stay updated on

current events, trends, and topics of interest. However, users should exercise critical thinking and verify the credibility of sources to avoid misinformation.

6. Support and Community Engagement: Social media apps enable users to find and join communities of shared interests, causes, or support groups. It offers a platform for users to engage in discussions, seek advice, provide support, and connect with like-minded individuals who can relate to their experiences.

7. Creative Expression and Inspiration: Social media apps provide a space for creative expression and inspiration. Users can share their artwork, photography, writing, and other forms of creativity, and receive feedback and encouragement from their peers. They can also explore and draw inspiration from the creative works of others.

8. Promoting Businesses and Entrepreneurship: Social media apps offer opportunities for businesses and entrepreneurs to promote their products, services, or personal brands. Users can create business profiles, showcase their offerings, engage with customers, and leverage the app's advertising and marketing features to reach a wider audience.

## Conclusion

We were able to successfully create a social media application by using react.js and spring boot in order to provide a smooth and fruitful user experience .

## Future Scope

Personalization and AI-driven Recommendations: The future of social media lies in personalized content and recommendations. By utilizing machine learning algorithms and artificial intelligence (AI), social media apps can analyze user preferences, behavior, and interactions to deliver more tailored and relevant content, ensuring a highly personalized user experience.

Improved Privacy and Data Security: With growing concerns around privacy and data security, social media apps will need to continue enhancing their privacy measures. Future iterations can implement advanced encryption techniques, granular privacy controls, and transparent data handling practices to provide users with greater control over their personal information.

Enhanced Social Features: Social media apps can continue to evolve by introducing new and innovative social features. For example, incorporating live streaming, group video calls, collaborative content creation, and interactive gaming can foster more meaningful connections and engagement among users

## Bibliography

1. "Pro Spring Boot 2: An Authoritative Guide to Building Microservices, Web, and Enterprise Applications" by Felipe Gutierrez
2. "Spring Boot in Action" by Craig Walls.
3. "React Up and Running: Building Web Applications" by Stoyan Stefanov
4. Official Documentation: The official documentation for Spring Boot (<https://spring.io/projects/spring-boot>) and React (<https://reactjs.org/docs/>)
5. Spring Boot and React tutorial series by Amigoscode
6. Spring Boot and React Tutorials by JavaGuides
7. Ma, Meng, et al. "Light-weight and scalable hierarchical-MVC architecture for cloud web applications." 2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom). IEEE, 2019.
8. Shah, Jay, and Dushyant Dubaria. "Building modern clouds: using docker, kubernetes & Google cloud platform." 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2019.