INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY, BANGALORE

SOFTWARE PRODUCTION ENGINEERING - FINAL PROJECT

SOCIAL MEDIA WEBSITE

Stay connected with others

In the supervision of :- Prof B.Thangaraju
Teaching Assistant :- Vishal Rai
Team Member Details :-

Name	Roll No.
Prafull Pandey	MT2022150
Himanshu Digrase	MT2022155

20.04.2023

Index
toc: yes

1. ABSTRACT

Our social media website is a platform where people can connect, share information, and engage with each other in meaningful ways. Users can create profiles, share updates, photos, and follow the activities of other users. Our website is designed to foster positive interactions and provide a safe, inclusive space for people of all backgrounds and identities. With powerful privacy and security features. Our social media website aims to bring people together and create a vibrant online community where individuals can express themselves, learn from others, and build meaningful relationships.

2. INTRODUCTION

a. OVERVIEW A social media website is an online platform where people can connect, share information, and engage with each other in a variety of ways. Social media websites typically allow users to create a profile, post updates, photos, and videos, join groups and communities, and follow the activities of other users.

One of the primary features of social media websites is the ability to interact with other users through comments, likes, and shares. This creates a sense of community and encourages users to engage with each other in positive and meaningful ways.

Social media websites can be used for a wide range of purposes, from personal communication and entertainment to business marketing and professional networking. They can also be used to share news and information, discuss social issues, and mobilize for social causes.

b. FEATURES

c. Why DevOps? DevOps is a set of tools that automate software development and IT operations. It focuses on shortening the systems development life cycle and providing continuous delivery with high software quality. DevOps is one up on Agile software development; multiple DevOps aspects came from the Agile methodology.
1. It optimizes the overall business by increasing efficiency through automation.
2. Improves software development and deployment speed and stability.
3. Deployment failures, rollbacks, and recovery time are all reduced.
4. Improved collaboration and communication; lower costs and IT headcount

3. TOOLS USED

1. Version Control System : Git and GitHub https://github.com/himanshudigrase/SocialWeb

2. Continuous Integration/Continuous Delivery : Github Actions
3. Building Tool : NPM (Node Package Manager)
4. Testing : Mocha framework
5. Containerization : Docker
   https://hub.docker.com/r/hims0301/backend
   https://hub.docker.com/r/hims0301/frontend
6. Deployment : Ansible
7. Log creation, monitoring and visualization : ELK (Elasticsearch,Logstash, Kibana)
8. Frontend Development : React.js, Vite
9. Backend Development : Node.js, Express.js
10. Database : MongoDB

4. SYSTEM CONFIGRATION

1. Operating System - Linux Ubuntu 22
2. CPU & RAM - 4 core processor and 8GB RAM
3. Kernel Version - 5.4.0-89-generic
4. Database - MongoDB 4.4.5

5. SOFTWARE DEVELOPEMENT LIFECYCLE- SDLC

i. INSTALLATIONS The frontend(client) of the project uses React.js and the backend(server) is built using Node.js, Express.js and the database is using MongoDB. To begin with, install nodejs on the system.

ii. NODEJS INSTALLATION Node.js is a javascript programming package management. It is the JavaScript runtime environment Node.js' default package manager. It comes pre-installed with Node.js. The package.json file contains the definitions for all npm packages. Package.json's content must be in JSON format. The definition file must have at least two fields. The names and variants are as follows. It is capable of managing dependencies. It installs all of the project's dependencies in a single command line. The package.json file also defines dependencies.

Inorder to run React, node environment shall be installed before starting React app

```
sudo apt-get install nodejs
sudo apt-get install npm
```

Run npm install command inside the frontend and backend folder to install all the necessary dependencies of the project.

```
sudo npm install
```

```
● himanshu@pop-os:/media/himanshu/New Volume/SocialMediaWebsite/SocialFrontEnd$ cd socialfrontend/
● himanshu@pop-os:/media/himanshu/New Volume/SocialMediaWebsite/SocialFrontEnd/socialfrontend$ npm install

 changed 12 packages, and audited 167 packages in 3s
```

3

Figure: Frontend npm install


```
● himanshu@pop-os:/media/himanshu/New Volume/SocialMediaWebsite$ cd SocialBackend/
● himanshu@pop-os:/media/himanshu/New Volume/SocialMediaWebsite/SocialBackend$ npm install

up to date, audited 872 packages in 9s
```

```json
{} package.json > ...
        You, last week | 1 author (You)
  1     {
  2       "type": "module",
  3       "dependencies": {
  4         "bcrypt": "^5.1.0",
  5         "body-parser": "^1.20.2",
  6         "cors": "^2.8.5",
  7         "dotenv": "^16.0.3",
  8         "express": "^4.18.2",
  9         "gridfs-stream": "^1.1.1",
 10         "helmet": "^6.0.1",
 11         "jsonwebtoken": "^9.0.0",
 12         "mongoose": "^7.0.3",
 13         "morgan": "^1.10.0",
 14         "multer": "^1.3.0",
 15         "multer-gridfs-storage": "^5.0.2"
 16       }
 17     }
```

Figure: Backend npm install

```
FrontEnd > socialfrontend > {} package.json > {} dependencies > 🔤 yup
{
    "name": "socialfrontend",
    "private": true,
    "version": "0.0.0",
    "type": "module",
    ▷ Debug
    "scripts": {
        "dev": "vite",
        "start": "run dev --host",
        "build": "vite build",
        "preview": "vite preview"
    },
    "dependencies": {
        "@emotion/react": "^11.10.6",
        "@emotion/styled": "^11.10.6",
        "@mui/icons-material": "^5.11.16",
        "@mui/material": "^5.12.0",
        "@reduxjs/toolkit": "^1.9.3",
        "dotenv": "^16.0.3",
        "formik": "^2.2.9",
        "react": "^18.2.0",
        "react-dom": "^18.2.0",
        "react-dropzone": "^14.2.3",
        "react-redux": "^8.0.5",
        "react-router-dom": "^6.10.0",
        "redux-persist": "^6.0.0",
        "yup": "^1.1.1"              Prafull Pandey, 4 weeks ago • state pe
    },
    "devDependencies": {
        "@types/react": "^18.0.28",
        "@types/react-dom": "^18.0.11",
        "@vitejs/plugin-react": "^3.1.0",
        "vite": "^4.2.0"
    }
}
```

Figure: Backend package.json
Figure: Frontend package.json

iii. MONGODB   We create a mongodb database on atlas to store all the user and posts information. To set this database with our project use create URI and store it in the .env file

```
⚙ .env
  1   MONGO_URL = 'mongodb+srv://<username>:<password>@socialmediacluster.1vyzwup.mongodb.net/?retryWrites=true&w=majority'
  2   PORT = 3001
```

Figure 1: .env SS

5

Database Deployments

Load sample datasets to SocialMediaCLuster.
Atlas provides sample data you can load into your Atlas clusters. You can use this data to quickly get started exploring with data in MongoDB.

● SocialMediaCLuster   Connect   View Monitoring   Browse Collections   ...

💧 Enhance Your Experience
For production throughput and richer metrics, upgrade to a dedicated cluster now!
Upgrade

● R 0
● W 0
Last 4 hours
100.0s

Connections 0
Last 4 hours
17.0

● In 0.0 B/s
● Out 0.0 B/s
Last 4 hours
1.3 KB/s

VERSION	REGION	CLUSTER TIER	TYPE	BACKUPS	LINKED APP SERVICES	ATLAS SEARCH
6.0.5	AWS / Mumbai (ap-south-1)	M0 Sandbox (General)	Replica Set - 3 nodes	Inactive	None Linked	Create Index

iv. Create a cluster on MONGO-DB Atlas

Network Access

IP Access List   Peering   Private Endpoint

＋ ADD IP ADDRESS

You will only be able to connect to your cluster from the following list of IP Addresses:

IP Address	Comment	Status	Actions
103.156.19.229/32	My IP Address	● Active	⚙ EDIT 🗑 DELETE
119.161.98.68/32 (includes your current IP address)		● Active	⚙ EDIT 🗑 DELETE

Set Network Access to 0.0.0.0/0 so that the website can be accessed from all IP addresses. This is ideal for real time development project, for development we can set a few specific addresses.

v. Connect app to database   Add the above URL to .env to connect to the database. Replace the username with mongodb's username and set the password for the same. Change the database name at myFirstDatabase, to your specific database name.

Database Entries from - Browse Collections

vi. Store MongoDB URL in .env file

By creating a.env file in the api (backend) subdirectory, you can save the cluster's URL. In that URL, provide all of the MongoDB Atlas' credentials (password). This.env file is part of.gitignore. Furthermore, Mongoose uses the variable used to store this URL to connect to MongoDB.

vii. JWT Authentication

JSON Web Tokens (JWT) are an RFC 7519 open industry standard for representing claims between two parties. For example, you can use jwt.io to decode, verify, and produce JWT.

JWT defines a concise and self-contained way for transmitting information between two parties as a JSON object. This information may be reviewed and trusted because it is signed. A secret (using the HMAC algorithm) or an RSA or ECDSA public/private key pair can be used to sign JWTs.
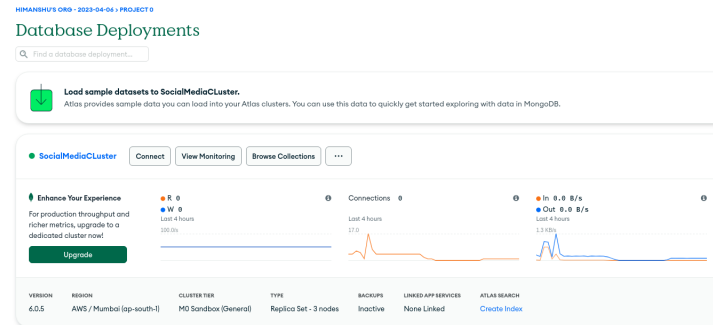
Connect to SocialMediaCLuster

Set up connection security ✓ — Choose a connection method ✓ — ③ Connect

Connecting with MongoDB Driver

1. Select your driver and version

We recommend installing and using the latest driver version.

Driver

Node.js ▼

Version

4.1 or later ▼

2. Install your driver

Run the following on the command line

```
npm install mongodb@4.1
```

View MongoDB Node.js Driver installation instructions. ⧉

3. Add your connection string into your application code

⬤ View full code sample

```
mongodb+srv://himanshudigrase:<password>@socialmediacluster.1vyzwup.mongodb.net/?
retryWrites=true&w=majority
```

Replace **<password>** with the password for the **himanshudigrase** user. Ensure any option params are URL encoded.

Figure 2: Another ss of DB

Figure 3: DB collection SS
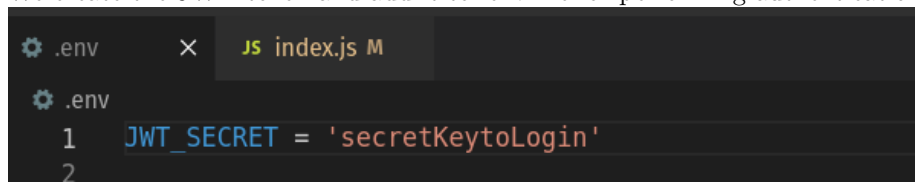
```
mongoose.connect(process.env.MONGO_URL,{
    useNewUrlParser: true,
    useUnifiedTopology: true,
})
.then(()=>{
    app.listen(PORT,()=>console.log(`Server Port: ${PORT}`));

}).catch((e)=>console.log(`${e}`));
```

Figure 4: Mongo COnnect method SS

We create the JWT token and add it to .env file for performing authentication.

```
.env          ×    JS index.js M

.env
1    JWT_SECRET = 'secretKeytoLogin'
2
```

viii. Source Code Management Source code management (SCM) is used to keep a track of all the modifications done to a source code repository. SCM tracks a current history of changes to a code base and helps resolve conflicts when merging updates from multiple contributors. SCM is very similar to Version control.

himanshudigrase / **SocialWeb** Public

<> Code   Issues   Pull requests   Actions   Projects   Wiki   Security   Insights   Settings

master   1 branch   0 tags   Go to file   Add file   <> Code

himanshudigrase Added Dummy Images for Data   2e9ea3d 2 hours ago   12 commits

SocialBackend   Added Dummy Images for Data   2 hours ago

SocialFrontEnd   Initial Commit   last week
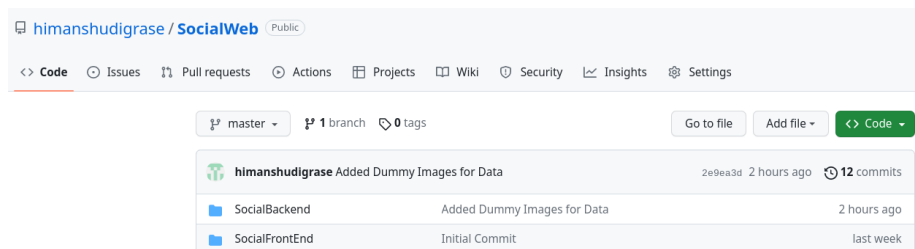
Figure 5: GITHUB SS

To integrate project with GitHub we do the following steps :
1. git init - Initializes the project as a github repository locally.
2. git remote add origin - Add the details of the remote branch.
3. git add - This command stages all the changes of local repo.
4. git commit - Commit command commits the changes to the respective branch in the current repository.
5. git push - Push command will push the changes to the remote repository on GitHub.

Figure 6: git init SS

ix. Testing

To make sure the project is running alright we provide a set of test cases. We use a library called 'Jest'. Jest is a JavaScript test framework running on Node. js and in the browser. Jest is a testing framework developed by Facebook. Originally designed to make UI testing easier for React developers, it's now a full standalone suite of tools for any type of JavaScript project (including Node.js) and includes features such as a built-in assertion library, code coverage, and mocking. Jest also runs multiple test suites concurrently, which can speed up the overall testing process. The downside of parallel execution is it can make debugging tests more difficult.

ix. Testing

For anyone coming from a BDD-style of Mocha, Jest tests are pretty familiar looking. Jest adds several global functions to help with setting up and running tests, such as describe, it, expect, and the jest object (used mostly for mocking). Backend Testing SS Backend Testing SS2

x. Containerization

Docker is an open source platform for developing, shipping, and running applications. Docker enables users to separate your applications from your infrastructure so you can deliver software quickly. This allows us to deploy products directly to users' computers without installing each software one-by-one.

We need to create an account on DockerHub, which is a public registry.

We then push our created image on this repository, this is publicly available and can be pulled by any user and deployed on a local machine.
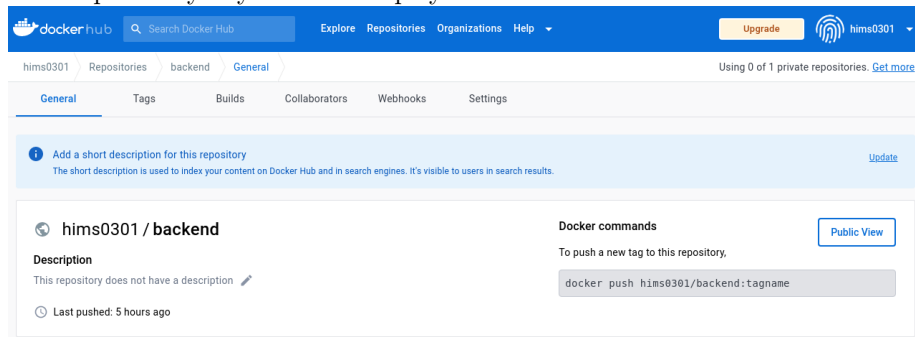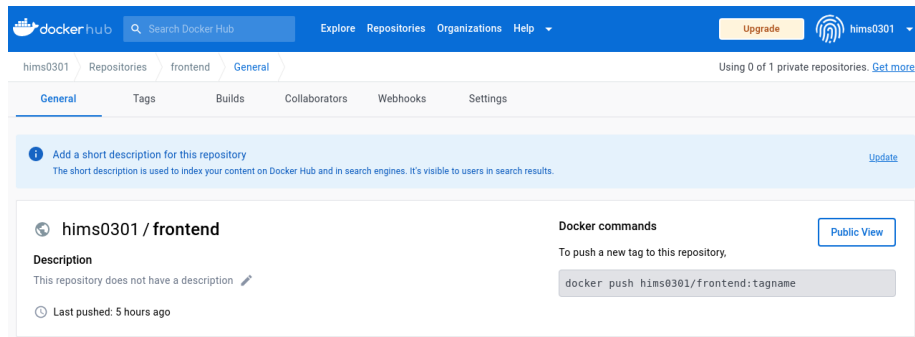
xi. Create repository on docker-hub

Link :
a. https://hub.docker.com/r/hims0301/backend
b. https://hub.docker.com/r/hims0301/frontend
c. https://hub.docker.com/repositories

```
      You, 2 weeks ago | 1 author (You)
  1   FROM node:17-alpine
  2   ARG MONGO_URL
  3   ARG PORT
  4   ARG JWT_SECRET
  5   RUN npm install -g nodemon
  6
  7   WORKDIR /app
  8   COPY package.json .
  9   RUN npm install
 10
 11   COPY . .
 12   ENV MONGO_URL=$MONGO_URL
 13   ENV PORT=$PORT
 14   ENV JWT_SECRET=$JWT_SECRET
 15   EXPOSE 3001
 16
 17   CMD ["nodemon", "index.js"]
```

```
SocialFrontEnd > socialfrontend > 🐳 Dockerfile > ...
          You, 1 second ago | 1 author (You)
   1    FROM node:17-alpine
   2
   3    ARG PORT
   4    WORKDIR /app
   5    COPY package.json .
   6
   7    RUN apk add --no-cache xdg-utils curl && \
   8        npm install
   9
  10    COPY . .
  11    ENV PORT=$PORT
  12
  13    EXPOSE 6002
  14    CMD ["npm", "run", "dev"]
  15
```

Docker holds a set of all the commands a user could call on the command line
to assemble an image. For our node project we set to copy the package.json and
package-lock.json to the container and then run npm install to install all the
packages there.

xii. CI/CD Pipeline

Continuous Integration : Continuous integration (CI) is the practice of automat-
ing the integration of code changes from multiple contributors into a single
software project.

Continuous Delivery : Continuous Delivery is the ability to get changes of
all types such as including new features, configuration changes, bug fixes and
experiments into production, or into the hands of users, safely and quickly in a
sustainable way. We use Github Actions to build our CI/CD pipeline.

CI/CD Pipeline Continuous Integration : Continuous integration (CI) is the
practice of automating the integration of code changes from multiple contributors
into a single software project.

Continuous Delivery : Continuous Delivery is the ability to get changes of
all types such as including new features, configuration changes, bug fixes and
experiments into production, or into the hands of users, safely and quickly in a
sustainable way. We use Github Actions to build our CI/CD pipeline.

First launched in 2018 as a platform-native automation tool, GitHub Actions has evolved to give developers powerful automation and CI/CD (continuous integration/continuous deployment) capabilities right next to your code in GitHub. At its core, GitHub Actions is designed to help simplify workflows with 16flexible automation and offer easy-to-use CI/CD capabilities built by developers for developers.
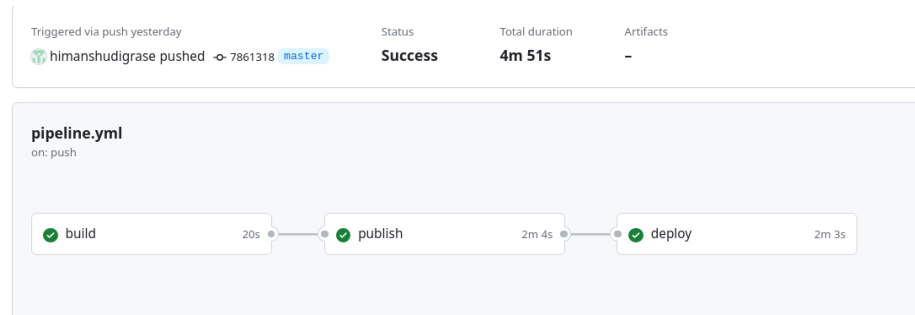
xiii. Getting started with github actions

Github actions is a famous platform in recent times to automate developer workflows. CI/CD pipelines is one of the many workflows offered by github to automate work processes.

xiv. Setting up github actions

All the required YAML files are placed in the .github/ directory which will be triggered on github action basis. Workflow is a collection of jobs and these jobs will run on the trigger of an event.

xv. Github Workflows:(Build Job)

Triggered via push yesterday	Status	Total duration	Artifacts
🐙 himanshudigrase pushed -o- 7861318 `master`	**Success**	**4m 51s**	–

pipeline.yml
on: push

✅ build	20s	✅ publish	2m 4s	✅ deploy	2m 3s

build
succeeded 20 hours ago in 20s

> ✓ Set up job

> ✓ Checkout code

> ✓ Install and Test Client

> ✓ Install and Test Server

> ✓ Post Checkout code

> ✓ Complete job

publish
succeeded 20 hours ago in 2m 4s

> ✓ Set up job

> ✓ Checkout code

> ✓ Build Server Docker Image

> ✓ Build Client Docker Image

> ✓ Login to Docker Hub

> ✓ Push Docker Images to Docker Hub

> ✓ Post Login to Docker Hub

> ✓ Post Checkout code

> ✓ Complete job
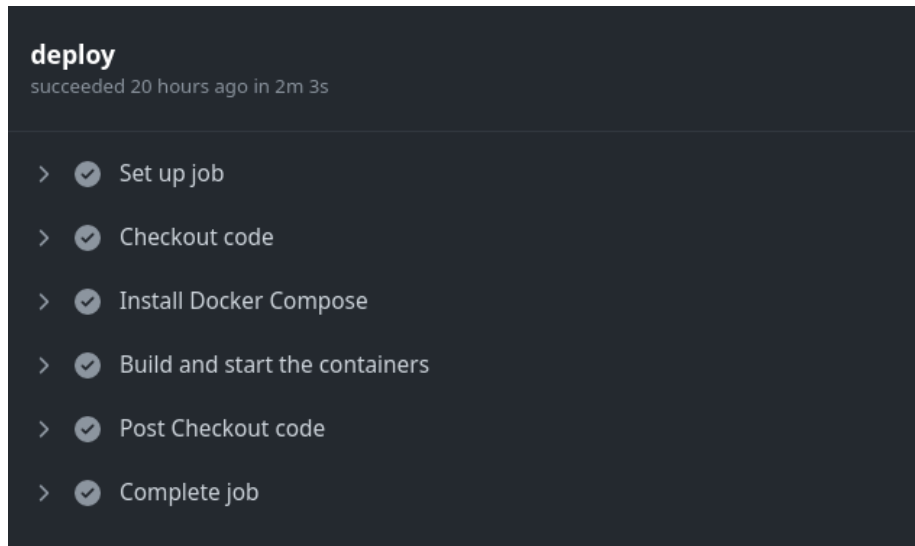
Pipeline Script in Github Actions   Before we set up the pipeline we declare environment variables to use in the pipeline in the future. We add variables for server and client image and docker credentials. We are storing MONGO_URL, JWT_TOKEN, PORT(Backend), PORT(Frontend) as secrets in Github. This worflow will run whenever user pushes/ commits code in Github.

```
1   name: Build and Deploy
2
3   # Run the workflow when code is pushed to the main branch
4   on:
5     push:
6       branches:
7         - master
8
9   # Set environment variables
10  env:
11    MONGO_URL: ${{ secrets.MONGO_URL }}
12    JWT_SECRET: ${{ secrets.JWT_SECRET }}
13    PORT: 3001
14    PORTF: 6002
```

```
16  jobs:
17    build:
18      # This is telling GitHub to run the workflow on the latest version of Ubuntu.
19      runs-on: ubuntu-latest
20      env:
21        MONGO_URL: ${{ secrets.MONGO_URL }}
22        JWT_SECRET: ${{ secrets.JWT_SECRET }}
23        PORT: 3001
24        PORTF: 6002
25      steps:
26        # Checkout the code from the GitHub repository
27        - name: Checkout code
28          uses: actions/checkout@v3
29
30        # Install dependencies and run tests for the client application
31        - name: Install and Test Client
32          working-directory: ./SocialFrontEnd/socialfrontend
33          run: |
34            npm install
35
36        # Install dependencies, export environment variables to be used by application and run tests for the server application
37        - name: Install and Test Server
38          working-directory: ./SocialBackend
39          run: |
40            npm install
41            export JWT_SECRET=$JWT_SECRET
42            export PORT=3001
```

Build job worflow where we are checking out the code from Github Repo to the
server hosted by Github which runs ubuntu-latest. We are installlling required
dependencies for frontend and backend by running npm install and providing
required env variables.

```
44    publish:
45      needs: build
46      runs-on: ubuntu-latest
47      steps:
48      # Checkout the code from the GitHub repository
49        - name: Checkout code
50          uses: actions/checkout@v3
51
52        # Build a Docker image for the server application
53        - name: Build Server Docker Image
54          working-directory:
55            ./SocialBackend
56
57          run: |
58            docker build --network host \
59            --build-arg MONGO_URL=$MONGO_URL \
60            --build-arg PORT=$PORT \
61            --build-arg JWT_SECRET=$JWT_SECRET \
62            -t hims0301/backend:latest .
63
64        - name: Build Client Docker Image
65          working-directory: ./SocialFrontEnd/socialfrontend
66
67          run: |
68            docker build --network host \
69            --build-arg PORT=$PORTF \
70            -t hims0301/frontend:latest .
71
72        # Login to Docker Hub using credentials from repository secrets
73        - name: Login to Docker Hub
74          uses: docker/login-action@v2
75          with:
76            username: ${{ secrets.DOCKER_USERNAME }}
77            password: ${{ secrets.DOCKER_PASSWORD }}
78
79        # Push the Docker images to Docker Hub
80        - name: Push Docker Images to Docker Hub
81          run: |
82            docker push hims0301/frontend:latest
83            docker push hims0301/backend:latest
```

Publish job publishes the images for frontend and backend in docker repository.It builds images of frontend and backend on server hosted by github. While building docker images we are passing the necessary env variables as arguments to docker. By logging into docker with credentials specified as secret variables we are pushing our built images to dockerHub.

```
85    deploy:
86      needs: publish
87      runs-on: self-hosted
88      steps:
89        # Checkout the code from the GitHub repository
90        - name: Checkout code
91          uses: actions/checkout@v2
92
93        # Install Docker Compose
94        - name: Install Docker Compose
95          run: |
96            sudo apt-get update
97            sudo apt-get install -y docker-compose
98
99        # Build and start the containers
100       - name: Build and start the containers
101         run: |
102           docker-compose -f docker-compose.yaml up -d
```

Deploy job deploys the docker images into conatiners on the localhost. We can
notice that server has been specified as *self-hosted* which means our localhost is
acting as server. Docker images will be deployed on our localhost with the help
of docker compose file.

Docker Compose File

```yaml
 docker-compose.yaml
      You, yesterday | 2 authors (You and others)
 1    version: "3.9"
 2    services:
 3      sociopedia-frontend:
 4        image: hims0301/frontend:latest
 5        container_name: sociopedia-frontend
 6        stdin_open: true
 7        # ports:
 8        #   - "6002:6002"
 9        network_mode: host
10        # - sociopedia
11
12      sociopedia-backend:
13        image: hims0301/backend:latest
14        container_name: sociopedia-backend
15        ports:
16          - "3001:3001"
17        networks:
18          - sociopedia
19        volumes:
20          - /home/himanshu/Downloads/socialLogs/logs.log:/app/logs.log
21
```

```yaml
 docker-compose.yaml
23      Elasticsearch:
24        image: elasticsearch:7.16.2
25        container_name: elasticsearch
26        restart: always
27        volumes:
28        - elastic_data:/usr/share/elasticsearch/data/
29        environment:
30          ES_JAVA_OPTS: "-Xmx256m -Xms256m"
31          discovery.type: single-node
32        ports:
33        - '9200:9200'
34        - '9300:9300'
35        networks:
36          - sociopedia
37
38      Logstash:
39        image: logstash:7.16.2
40        container_name: logstash
41        restart: always
42        volumes:
43        - /home/himanshu/Downloads/socialLogs/:/logstash_dir
44        command: logstash -f /logstash_dir/logstash.conf
45        depends_on:
46          - Elasticsearch
47        ports:
48        - '9600:9600'
49        environment:
50          LS_JAVA_OPTS: "-Xmx256m -Xms256m"
51        networks:
52          - sociopedia
```

```yaml
docker-compose.yaml
54    Kibana:
55      image: kibana:7.16.2
56      container_name: kibana
57      restart: always
58      ports:
59      - '5601:5601'
60      environment:
61        - ELASTICSEARCH_URL=http://elasticsearch:9200
62      depends_on:
63        - Elasticsearch
64      networks:
65        - sociopedia
66
67  networks:
68    sociopedia:
69      ipam:
70        driver: default
71        config:
72        - subnet: 127.0.0.1/8
73  volumes:
74    elastic_data:
75      driver: local
```