Steps

1. I started off by making a dockerignore file in both, the client and the server folder, and added node modules to it.
2. I created dockerfiles in each folder, and added the first few commands.
3. To verify the axios api url, i went to the api folder inside the client folder and added a process.env.REACT_APP_API_URL in it and used that to define my in-container endpoints
4. the n i referenced it to my dockerfile.
5. Finally i listed all my services and their networks, build/images, and volumes in my docker file
6. Docker compose up, and everything was ready to go

```
PS D:\UNI\sixth\LAB SCD\i222633_F_LabMid\Social-Media-Web-App-Mern-Stack--master\Social-Media-Web-App-Mern-Stack--master> do
cker network create network-labmid
b28e2575cb7cdd28e0af0fddd4ba5318301b5a9bece6bcb2b0afd6dc3038e762
```

```
ker compose up
+] Running 1/1
√ mongo Pulled                                                                              2.3s
ompose now can delegate build to bake for better performances
ust set COMPOSE_BAKE=true
+] Building 9.3s (4/10)                                                           docker:desktop-linux
=> => transferring dockerfile: 153B                                                         0.0s
=> [server internal] load metadata for docker.io/library/node:18                            2.9s
=> [server auth] library/node:pull token for registry-1.docker.io                           0.0s
=> [server internal] load .dockerignore                                                     0.0s
=> => transferring context: 53B                                                             0.0s
=> [server 1/5] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d77  6.2s
=> => resolve docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e       0.1s
=> => sha256:3b3ca5178f3ece5f7e96b38b9c4b9c3a101a9d6777cf0f5c320f869512c80024 448B / 448B             0.3s
=> => sha256:04deb1529fda049f44f9be8d16ca833a53961813ce07eda3cefd50cd3fd74880 1.25MB / 1.25MB         0.7s
=> => sha256:058db40e534297246fb14bc4e107d2f6ddc140494a26793e1e90696cd6b2507a 7.34MB / 45.68MB        5.3s
```

```
] Building 44.1s (5/10)                                                           docker:desktop-linux
> [server internal] load .dockerignore                                                      0.0s
> => transferring context: 53B                                                              0.0s
> [server 1/5] FROM docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d7   41.0s
> => resolve docker.io/library/node:18@sha256:df9fa4e0e39c9b97e30240b5bb1d99bdb861573a82002b2c52ac7d6b8d6d773e        0.1s
> => sha256:3b3ca5178f3ece5f7e96b38b9c4b9c3a101a9d6777cf0f5c320f869512c80024 448B / 448B              0.3s
> => sha256:04deb1529fda049f44f9be8d16ca833a53961813ce07eda3cefd50cd3fd74880 1.25MB / 1.25MB          0.7s
> => sha256:058db40e534297246fb14bc4e107d2f6ddc140494a26793e1e90696cd6b2507a 45.68MB / 45.68MB        29.8s
> => sha256:ee496386c5de1ce84096ca486e1aabcdf7cb8f0afd5a9b4863dbf870b340744f 3.32kB / 3.32kB          1.0s
> => sha256:b617a119f8a27982374d94ec6eb3738ae3d38d6fc2c34c865813926cf596a621 128.97MB / 211.33MB      39.8s
> => sha256:1eb98adba0eb44a2e4facf9ca3626a4a66feedd0dd56d159cca90a35205744e7 64.40MB / 64.40MB        32.5s
> => sha256:07d1b5af933d2dfc3d0dd509d6e20534825e4a537f7b006a6cb5b8e5a1f20905 24.01MB / 24.01MB        10.4s
> => sha256:23b7d26ef1d294256da0d70ce374277b9aab5ca683015073316005cb63d33849 48.49MB / 48.49MB        20.1s
> => extracting sha256:23b7d26ef1d294256da0d70ce374277b9aab5ca683015073316005cb63d33849               8.3s
> => extracting sha256:07d1b5af933d2dfc3d0dd509d6e20534825e4a537f7b006a6cb5b8e5a1f20905               0.1s
> [server internal] load build context                                                      29.4s
> => transferring context: 48.47MB                                                          29.3s
```

```
[+] Building 102.9s (15/17)                                                          docker:d
=> => unpacking to docker.io/library/social-media-web-app-mern-stack--master-server:latest
=> [server] resolving provenance for metadata file
=> [client internal] load build definition from Dockerfile
=> => transferring dockerfile: 212B
=> [client internal] load .dockerignore
=> => transferring context: 55B
=> [client internal] load build context
=> => transferring context: 8.37MB
=> [client 3/5] COPY . .
=> [client 4/5] RUN npm install
=> => # Run `npm audit` for details.
=> => # npm notice
=> => # npm notice New major version of npm available! 10.8.2 -> 11.3.0
=> => # npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.3.0
=> => # npm notice To update run: npm install -g npm@11.3.0
=> => # npm notice
```

```
[+] Running 6/6
 ✓ client                                                      Built               0.0s
 ✓ server                                                      Built               0.0s
 ✓ Network social-media-web-app-mern-stack--master_network-labmid   Created        0.1s
 ✓ Container mongo_labmid                                      Created             5.7s
 ✓ Container server                                            Created             0.6s
 ✓ Container client                                            Created             0.9s
```

```
Attaching to client, mongo_labmid, server
server        |
server        | > server@1.0.0 start
server        | > nodemon index.js
server        |
server        | [nodemon] 2.0.22
server        | [nodemon] to restart at any time, enter `rs`
server        | [nodemon] watching path(s): *.*
server        | [nodemon] watching extensions: js,mjs,json
client        |
server        | [nodemon] starting `node index.js`
client        | > client@0.1.0 start
mongo_labmid  | {"t":{"$date":"2025-04-21T05:51:51.482+00:00"},"s":"I",  "c":"CONTROL",  "id":23285,    "ctx":"main","msg":"A
utomatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
client        | > react-scripts start
mongo_labmid  | {"t":{"$date":"2025-04-21T05:51:51.494+00:00"},"s":"I",  "c":"CONTROL",  "id":5945603, "ctx":"main","msg":"M
ulti threading initialized"}
client        |
```

```
PS D:\UNI\sixth\LAB SCD\i222633_F_LabMid\Social-Media-Web-App-Mern-Stack--master\Social-Media-Web-App-Mern-Stack--master> do
cker ps
CONTAINER ID    IMAGE                    COMMAND                CREATED              STATUS
     PORTS                    NAMES
9ac76493939b    social-media-web-app-mern-stack--master-client  "docker-entrypoint.s…"  About a minute ago   Up About a min
ute    0.0.0.0:4000->80/tcp        client
ba1a09f9a1d6    social-media-web-app-mern-stack--master-server  "docker-entrypoint.s…"  About a minute ago   Up About a min
ute    0.0.0.0:5000->5000/tcp      server
24cc0f7e06c5    mongo:latest             "docker-entrypoint.s…"  About a minute ago   Up About a min
ute    0.0.0.0:27017->27017/tcp  mongo_labmid
PS D:\UNI\sixth\LAB SCD\i222633_F_LabMid\Social-Media-Web-App-Mern-Stack--master\Social-Media-Web-App-Mern-Stack--master>
```

Implementing additional features

- I added a health check feature to my client service that runs health checks on my frontend endpoint with a 30s interval
- I added an nginx load balancing service that balances load usig docker swarm
- I then restarted my containers and everything was working as expected
- I already set up my environment variables in the dockerfiles, and the project root, but this time i aded environment to my client service in my docker compose file as well specifying the client url and the mongodb url

```
PS D:\UNI\sixth\LAB SCD\i222633_F_LabMid\Social-Media-Web-App-Mern-Stack--master\Social-Media-Web-App-Mern-Stack--master> do
cker compose up
[+] Running 4/4
 ✓ Container mongo_labmid    Created                                                                          0.0s
 ✓ Container server          Created                                                                          0.0s
 ✓ Container client          Recreated                                                                        0.5s
 ✓ Container load_balancer   Created                                                                          0.5s
Attaching to client, load_balancer, mongo_labmid, server
mongo_labmid   | {"t":{"$date":"2025-04-21T06:02:46.972+00:00"},"s":"I",  "c":"CONTROL",  "id":23285,   "ctx":"main","msg":"
Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
```

```
load_balancer  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
load_balancer  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
load_balancer  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
load_balancer  | 10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
load_balancer  | 10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
load_balancer  | /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
load_balancer  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
load_balancer  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
load_balancer  | /docker-entrypoint.sh: Configuration complete; ready for start up
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: using the "epoll" event method
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: nginx/1.27.4
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: OS: Linux 5.15.167.4-microsoft-standard-WSL2
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: start worker processes
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: start worker process 29
load_balancer  | 2025/04/21 06:02:49 [notice] 1#1: start worker process 30
```