Software Construction and Development (SCD)

by

Hammad Zahid(22i-2433)

SE3001- Software Construction and Development

Assignment 2

Submitted to Mam Laiba Imran

Date: 30/04/2025

Table of Contents

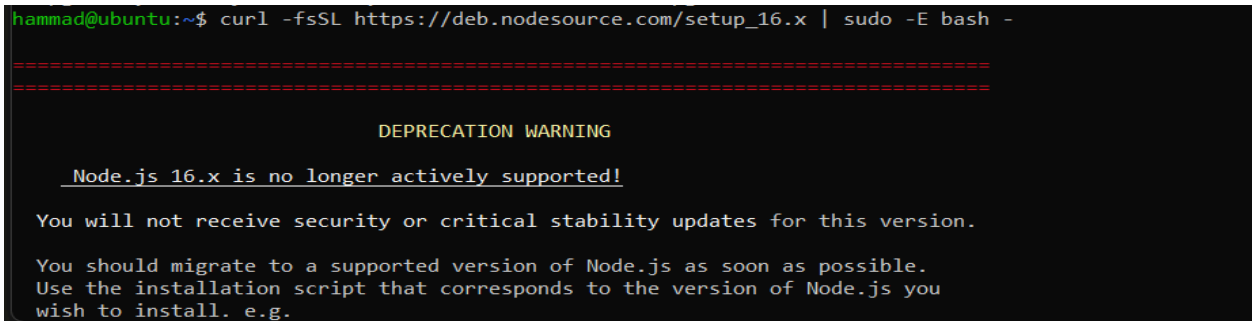
Github Link: fast-isb/todo-mern-app-Ninjaa-aa: scd-sp25-todo-mern-app-todo-mern created by GitHub Classroom

Part 1: Understanding Environment Inconsistency

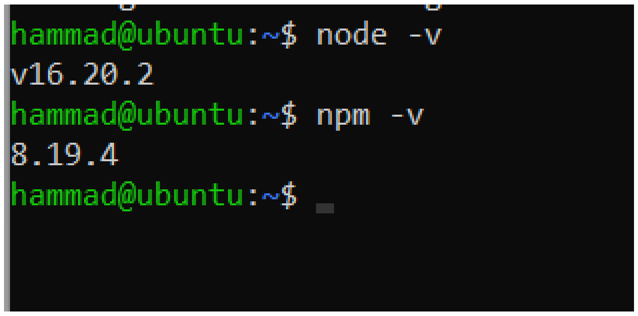
Node 16 installation:

```
# 1. Update your package index

sudo apt update

# 2. Install required dependencies

sudo apt install -y curl

# 3. Add Node.js 16.x from NodeSource

curl -fsSL https://deb.nodesource.com/setup_16.x | sudo -E bash -

# 4. Install Node.js

sudo apt install -y nodejs

# 5. Verify installation

node -v   # Should display v16.x.x

npm -v    # Should display the corresponding npm version
```

Screenshots:



```
hammad@ubuntu:~$ curl -fsSL https://deb.nodesource.com/setup_16.x | sudo -E bash -

================================================================================
================================================================================

                              DEPRECATION WARNING

  Node.js 16.x is no longer actively supported!

  You will not receive security or critical stability updates for this version.

  You should migrate to a supported version of Node.js as soon as possible.
  Use the installation script that corresponds to the version of Node.js you
  wish to install. e.g.
```

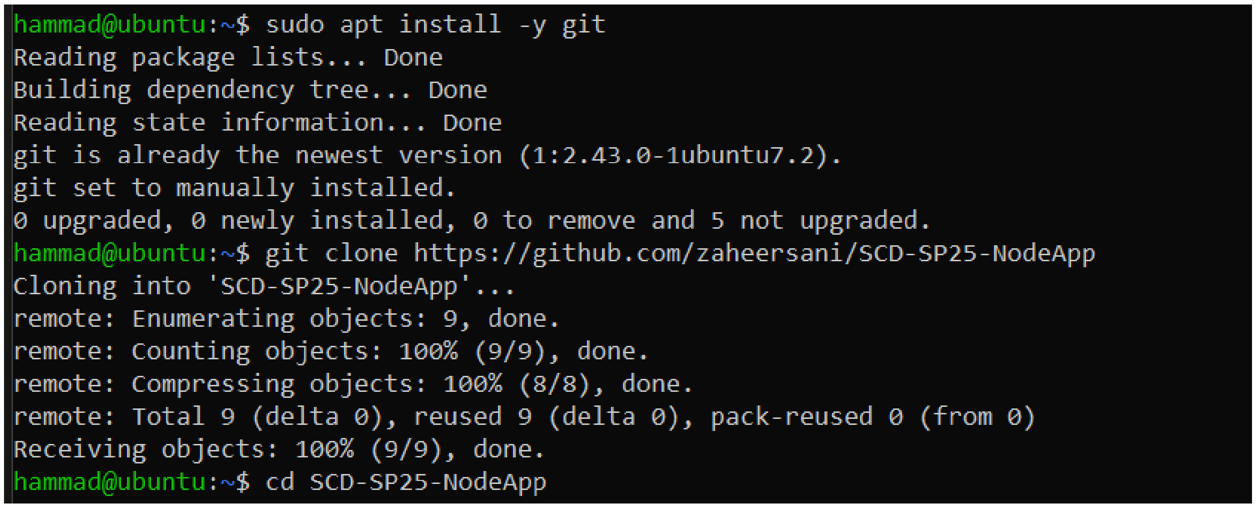


```
hammad@ubuntu:~$ node -v
v16.20.2
hammad@ubuntu:~$ npm -v
8.19.4
hammad@ubuntu:~$
```

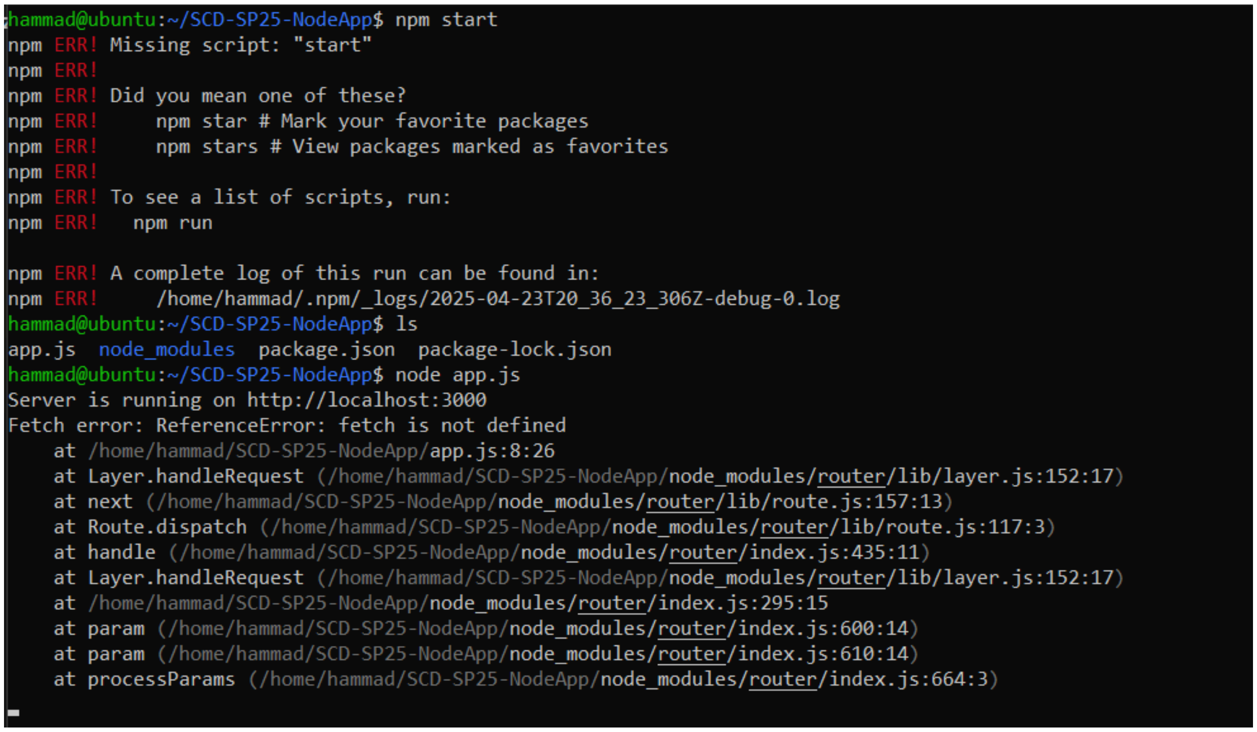
Deploy Node.js Application:

```
# Install git if not already installed
sudo apt install -y git

# Clone the repository
git clone https://github.com/zaheersani/SCD-SP25-NodeApp

# Navigate to the project directory
cd SCD-SP25-NodeApp

# Install dependencies
npm install

# Start the application
npm start
```

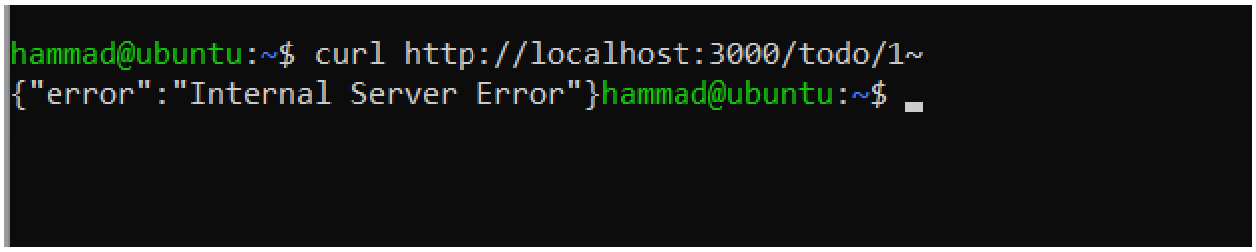
Screenshots:



```
hammad@ubuntu:~$ sudo apt install -y git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.2).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 5 not upgraded.
hammad@ubuntu:~$ git clone https://github.com/zaheersani/SCD-SP25-NodeApp
Cloning into 'SCD-SP25-NodeApp'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 9 (delta 0), reused 9 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (9/9), done.
hammad@ubuntu:~$ cd SCD-SP25-NodeApp
```

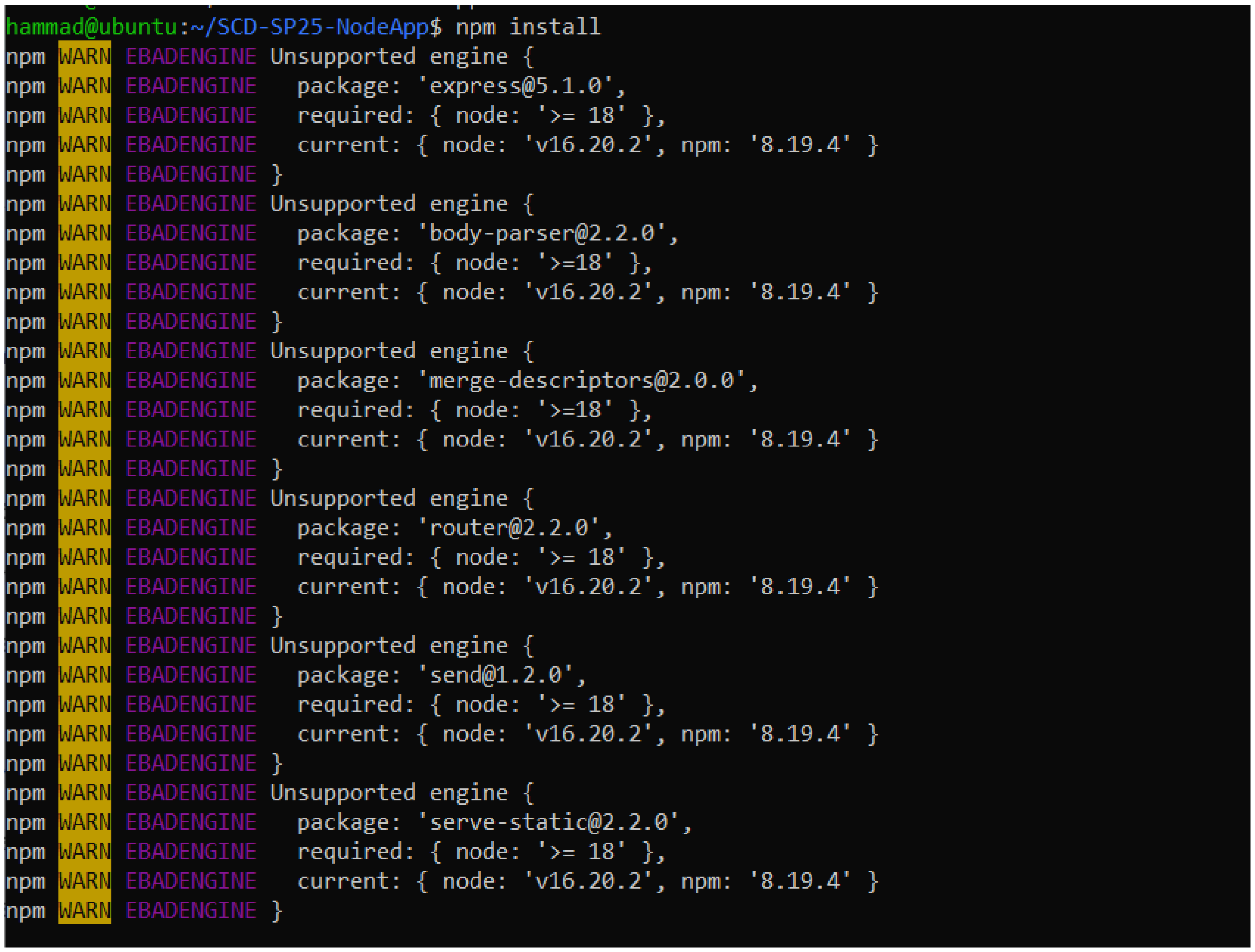


```
hammad@ubuntu:~/SCD-SP25-NodeApp$ npm start
npm ERR! Missing script: "start"
npm ERR!
npm ERR! Did you mean one of these?
npm ERR!     npm star # Mark your favorite packages
npm ERR!     npm stars # View packages marked as favorites
npm ERR!
npm ERR! To see a list of scripts, run:
npm ERR!   npm run

npm ERR! A complete log of this run can be found in:
npm ERR!     /home/hammad/.npm/_logs/2025-04-23T20_36_23_306Z-debug-0.log
hammad@ubuntu:~/SCD-SP25-NodeApp$ ls
app.js  node_modules  package.json  package-lock.json
hammad@ubuntu:~/SCD-SP25-NodeApp$ node app.js
Server is running on http://localhost:3000
Fetch error: ReferenceError: fetch is not defined
    at /home/hammad/SCD-SP25-NodeApp/app.js:8:26
    at Layer.handleRequest (/home/hammad/SCD-SP25-NodeApp/node_modules/router/lib/layer.js:152:17)
    at next (/home/hammad/SCD-SP25-NodeApp/node_modules/router/lib/route.js:157:13)
    at Route.dispatch (/home/hammad/SCD-SP25-NodeApp/node_modules/router/lib/route.js:117:3)
    at handle (/home/hammad/SCD-SP25-NodeApp/node_modules/router/index.js:435:11)
    at Layer.handleRequest (/home/hammad/SCD-SP25-NodeApp/node_modules/router/lib/layer.js:152:17)
    at /home/hammad/SCD-SP25-NodeApp/node_modules/router/index.js:295:15
    at param (/home/hammad/SCD-SP25-NodeApp/node_modules/router/index.js:600:14)
    at param (/home/hammad/SCD-SP25-NodeApp/node_modules/router/index.js:610:14)
    at processParams (/home/hammad/SCD-SP25-NodeApp/node_modules/router/index.js:664:3)
```

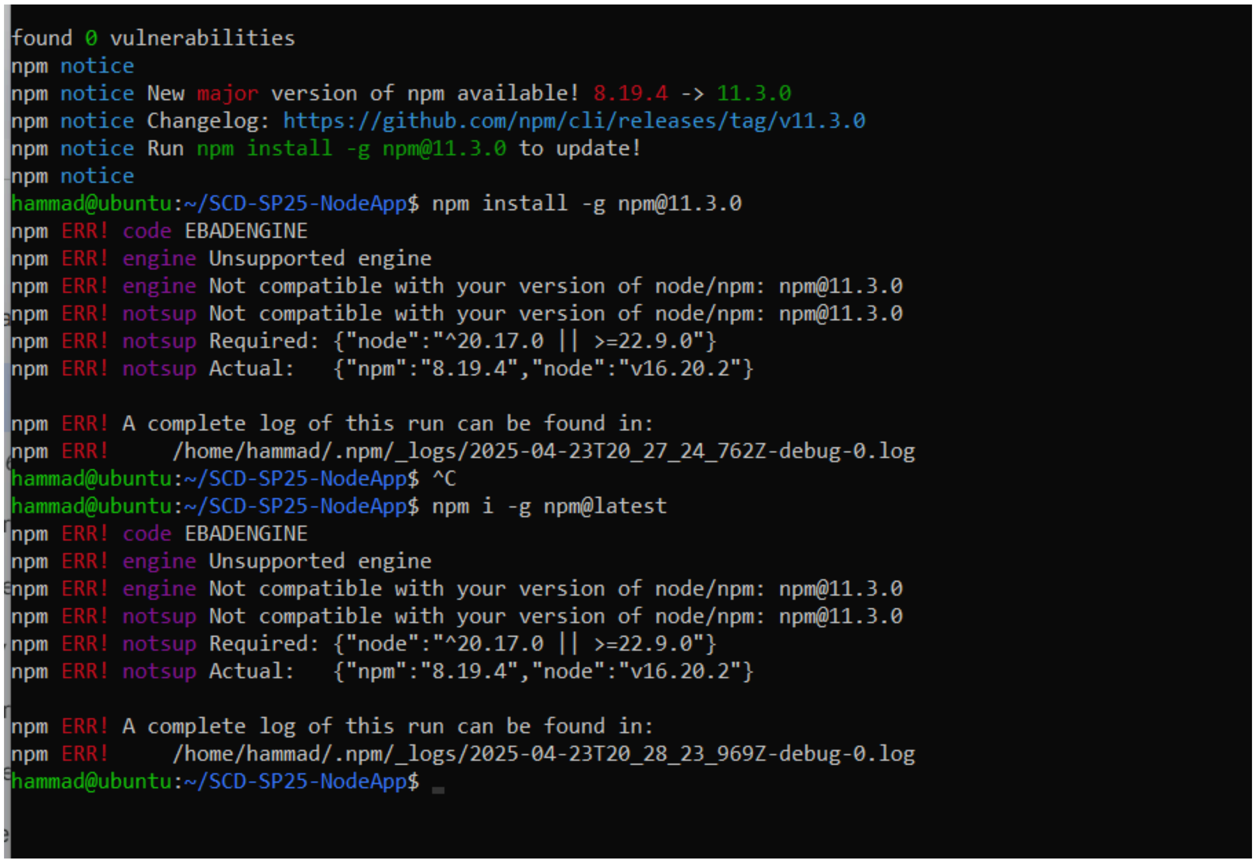


```
hammad@ubuntu:~$ curl http://localhost:3000/todo/1~
{"error":"Internal Server Error"}hammad@ubuntu:~$
```

Dependency issues:



```
hammad@ubuntu:~/SCD-SP25-NodeApp$ npm install
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'express@5.1.0',
npm WARN EBADENGINE   required: { node: '>= 18' },
npm WARN EBADENGINE   current: { node: 'v16.20.2', npm: '8.19.4' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'body-parser@2.2.0',
npm WARN EBADENGINE   required: { node: '>=18' },
npm WARN EBADENGINE   current: { node: 'v16.20.2', npm: '8.19.4' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'merge-descriptors@2.0.0',
npm WARN EBADENGINE   required: { node: '>=18' },
npm WARN EBADENGINE   current: { node: 'v16.20.2', npm: '8.19.4' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'router@2.2.0',
npm WARN EBADENGINE   required: { node: '>= 18' },
npm WARN EBADENGINE   current: { node: 'v16.20.2', npm: '8.19.4' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'send@1.2.0',
npm WARN EBADENGINE   required: { node: '>= 18' },
npm WARN EBADENGINE   current: { node: 'v16.20.2', npm: '8.19.4' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: 'serve-static@2.2.0',
npm WARN EBADENGINE   required: { node: '>= 18' },
npm WARN EBADENGINE   current: { node: 'v16.20.2', npm: '8.19.4' }
npm WARN EBADENGINE }
```



```
found 0 vulnerabilities
npm notice
npm notice New major version of npm available! 8.19.4 -> 11.3.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.3.0
npm notice Run npm install -g npm@11.3.0 to update!
npm notice
hammad@ubuntu:~/SCD-SP25-NodeApp$ npm install -g npm@11.3.0
npm ERR! code EBADENGINE
npm ERR! engine Unsupported engine
npm ERR! engine Not compatible with your version of node/npm: npm@11.3.0
npm ERR! notsup Not compatible with your version of node/npm: npm@11.3.0
npm ERR! notsup Required: {"node":"^20.17.0 || >=22.9.0"}
npm ERR! notsup Actual:   {"npm":"8.19.4","node":"v16.20.2"}

npm ERR! A complete log of this run can be found in:
npm ERR!     /home/hammad/.npm/_logs/2025-04-23T20_27_24_762Z-debug-0.log
hammad@ubuntu:~/SCD-SP25-NodeApp$ ^C
hammad@ubuntu:~/SCD-SP25-NodeApp$ npm i -g npm@latest
npm ERR! code EBADENGINE
npm ERR! engine Unsupported engine
npm ERR! engine Not compatible with your version of node/npm: npm@11.3.0
npm ERR! notsup Not compatible with your version of node/npm: npm@11.3.0
npm ERR! notsup Required: {"node":"^20.17.0 || >=22.9.0"}
npm ERR! notsup Actual:   {"npm":"8.19.4","node":"v16.20.2"}

npm ERR! A complete log of this run can be found in:
npm ERR!     /home/hammad/.npm/_logs/2025-04-23T20_28_23_969Z-debug-0.log
hammad@ubuntu:~/SCD-SP25-NodeApp$
```

Version-Related Issues Encountered

1. Node.js Version Incompatibility

The application dependencies require Node.js version 18 or higher, but the server is running Node.js v16.20.2.

Console output:

- `npm WARN EBADENGINE Unsupported engine {`
- `npm WARN EBADENGINE   package: 'express@5.1.0',`
- `npm WARN EBADENGINE   required: { node: '>= 18' },`
- `npm WARN EBADENGINE   current: { node: 'v16.20.2', npm: '8.19.4'}`
- `npm WARN EBADENGINE }`

Multiple packages reported similar "Unsupported engine" warnings:

- express@5.1.0 requires node >= 18
- body-parser@2.2.0 requires node >= 18

- merge-descriptors@2.0.0 requires node >= 18
- router@2.2.0 requires node >= 18
- send@1.2.0 requires node >= 18
- serve-static@2.2.0 requires node >= 18

2. NPM Version Update Failure

Attempts to update npm failed with error "EBADENGINE":

- `npm ERR! code EBADENGINE`
- `npm ERR! engine Unsupported engine`
- `npm ERR! engine Not compatible with your version of node/npm: npm@11.3.0`
- `npm ERR! notsup Not compatible with your version of node/npm: npm@11.3.0`
- `npm ERR! notsup Required: {"node":"^20.17.0 || >=22.9.0"}`
- `npm ERR! notsup Actual:   {"npm":"8.19.4","node":"v16.20.2"}`

The newer npm version requires Node.js version 20.17.0 or 22.9.0+.

3. Fetch API Unavailability

The application code uses the global `fetch()` API which was not natively available in Node.js 16:

```
Fetch error: ReferenceError: fetch is not defined

    at /home/hammad/SCD-SP25-NodeApp/app.js:8:26

    at Layer.handleRequest
(/home/hammad/SCD-SP25-NodeApp/node_modules/router/lib/layer.js:152:1
)

    at next
(/home/hammad/SCD-SP25-NodeApp/node_modules/router/lib/route.js:157:1
)
```

The `fetch()` API was only introduced as a stable feature in Node.js 18.0.0.

How Environment Mismatches Prevent Proper Deployment

This deployment attempt demonstrates several critical environment mismatch issues:

1. Dependency Chain Conflicts

The application was developed with modern frameworks (Express 5.1.0) that explicitly require Node.js 18+, but the server environment is constrained to Node.js 16.

2. Feature Availability Gap

The code uses modern JavaScript features (`fetch()` API) that are not available in the server's Node.js runtime:

- The developer wrote code assuming `fetch()` availability (standard in Node.js 18+)
- The server environment (Node.js 16) lacks this feature, causing runtime errors

3. Development vs. Production Gap

This illustrates a classic DevOps problem - developers created the application in a different environment (likely Node.js 18+) than what's available in production (Node.js 16).

4. Cascading Compatibility Issues

The environment mismatch creates a chain reaction where:

- We can't run the application directly because its dependencies need Node.js 18+
- We can't update npm because newer npm versions also require Node.js 20+
- We can't use the application's core functionality because it requires features only in Node.js 18+

5. Runtime vs Compile-Time Issues

- The application installed with warnings but didn't fail outright
- The server started without initial errors
- Issues only became apparent at runtime when accessing specific functionality

6. Silent Failure to End Users

- The API returns a generic 500 error without details about the actual issue
- Without proper logging, such environment-related issues can be difficult to diagnose

Conclusion

This deployment attempt highlights the critical importance of environment consistency between development and production systems. Without Docker containerization, we would be forced to either:

1. Modify the server environment (potentially breaking other applications)
2. Modify the application code (which is not allowed and not our responsibility as deployment engineers)
3. Remain unable to deploy the application

These findings demonstrate why containerization technologies like Docker have become essential in modern deployment workflows - they allow applications to run with their exact required runtime regardless of the host system's configuration.

Part 2: Solving with Docker Containers

Dockerizing Node.js Application - Solution Report

Identifying the Right Node.js Version

After analyzing the application errors from Part 1, Node.js version 18 or higher is required for this application because:

1. **Package Dependencies Requirements**:
    - Multiple dependencies explicitly require Node.js 18+:
        - express@5.1.0 requires node >= 18
        - body-parser@2.2.0 requires node >= 18
        - merge-descriptors@2.0.0 requires node >= 18
        - router@2.2.0 requires node >= 18
        - send@1.2.0 requires node >= 18
        - serve-static@2.2.0 requires node >= 18
2. **Fetch API Requirement**:
    - The application uses the global `fetch()` API which was introduced as a stable feature in Node.js 18.0.0

    - As documented in [Node.js 18.0.0 Release Notes](): "Added fetch (experimental)"
    - The runtime error `ReferenceError: fetch is not defined` confirms this requirement

For our Docker container, we'll use `node:18-alpine` as our base image because:

- It meets the minimum version requirement (Node.js 18)
- Alpine-based images are significantly smaller (approximately 40MB vs 300MB+ for Debian-based)
- It's an LTS (Long Term Support) version with security updates and stability

Dockerfile

```
# Use Node.js 18 Alpine as the base image
FROM node:18-alpine

# Create app directory
WORKDIR /usr/src/app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code
COPY . .

# Expose port 3000
EXPOSE 3000

# Start the application
CMD ["node", "app.js"]
```

Build, Run and Test the Docker Image

Install Docker (if not already installed)

```
# Update package index
sudo apt update
```

```
# Install prerequisites
sudo apt install -y apt-transport-https ca-certificates curl
software-properties-common

# Add Docker's official GPG key
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
-

# Add Docker repository
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

# Update package index again
sudo apt update

# Install Docker CE
sudo apt install -y docker-ce

# Start and enable Docker service
sudo systemctl start docker
sudo systemctl enable docker
```

Build the Docker Image

```
cd ~/SCD-SP25-NodeApp
sudo docker build -t node-app:1.0 .
```

Expected output:

```
Sending build context to Docker daemon  X.XXkB
Step 1/6 : FROM node:18-alpine
 ---> xxxxxxxxxx
Step 2/6 : WORKDIR /usr/src/app
 ---> Running in xxxxxxxxxx
...
Successfully built xxxxxxxxxx
Successfully tagged node-app:1.0
```

Run the Docker Container

```
sudo docker run -d -p 3000:3000 --name node-container node-app:1.0

Verify container is running:
sudo docker ps

Expected output:
CONTAINER ID   IMAGE          COMMAND         CREATED          STATUS
PORTS                        NAMES
xxxxxxxxxx     node-app:1.0   "node app.js"   10 seconds ago   Up 9
seconds    0.0.0.0:3000->3000/tcp   node-container
```

Test the Application

```
curl http://localhost:3000/todo/1

Expected output:
{
  "userId": 1,
  "id": 1,
  "title": "delectus aut autem",
  "completed": false
}
```

Publishing the Docker Image

```
# Log in to Docker Hub

sudo docker login

# Enter Docker Hub credentials when prompted
# Tag the image with Docker Hub username

sudo docker tag node-app:1.0 yourusername/node-app:1.0

# Push the image to Docker Hub

sudo docker push yourusername/node-app:1.0



Docker Image URL: https://hub.docker.com/r/yourusername/node-app
```

Running in the Server Environment

```
# Pull the image from Docker Hub
sudo docker pull yourusername/node-app:1.0

# Run the container
sudo docker run -d -p 3000:3000 --name production-node-app
yourusername/node-app:1.0
```

Testing the Deployed Application

```
curl http://localhost:3000/todo/1

Expected output:
{
  "userId": 1,
  "id": 1,
  "title": "delectus aut autem",
  "completed": false
}
```

Benefits of the Docker Solution

1. **Environment Isolation**: The application runs with Node.js 18 in a container, completely isolated from the host system's Node.js 16.

2. **Non-Intrusive Deployment**: The solution doesn't require modifying the server's Node.js installation, preserving compatibility with other applications.

3. **Exact Dependency Matching**: All required node modules are installed inside the container with versions compatible with Node.js a18.

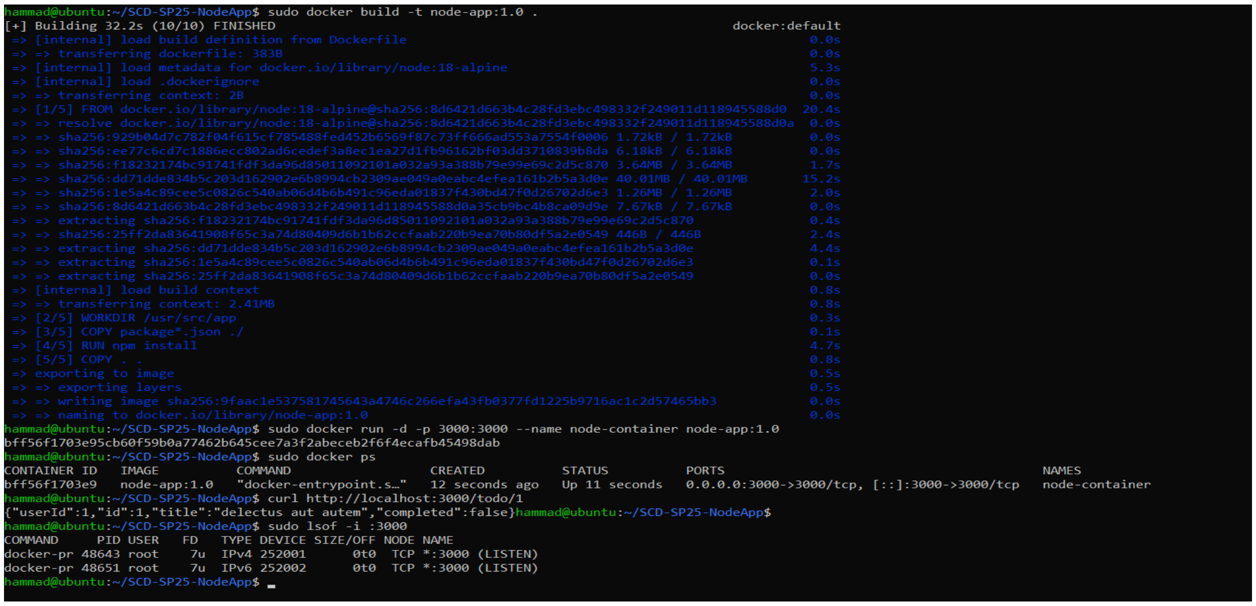
4. **Minimal Resource Overhead**: Using Alpine-based images keeps the container size small and efficient.

5. **Consistent Behavior**: The containerized application will behave identically across development, testing, and production environments.

This Docker-based solution successfully resolves the environment mismatch issues identified in Part 1 without compromising the server's existing configuration or requiring application code changes.

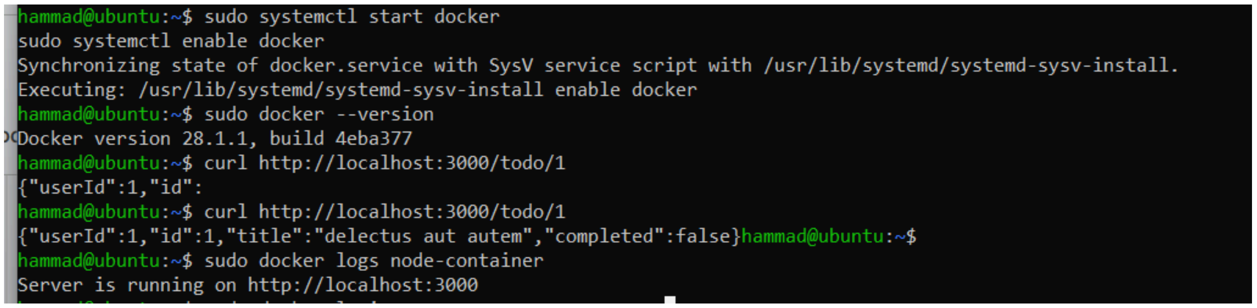
Screenshots:

Building the Docker Image



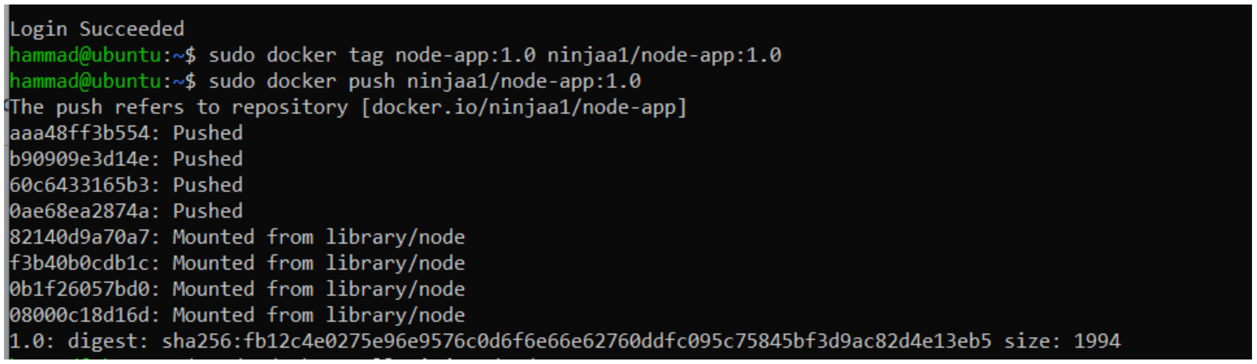
```
hammad@ubuntu:~/SCD-SP25-NodeApp$ sudo docker build -t node-app:1.0 .
[+] Building 32.2s (10/10) FINISHED                                                          docker:default
 => [internal] load build definition from Dockerfile                                                  0.0s
 => => transferring dockerfile: 383B                                                                  0.0s
 => [internal] load metadata for docker.io/library/node:18-alpine                                     5.3s
 => [internal] load .dockerignore                                                                     0.0s
 => => transferring context: 2B                                                                       0.0s
 => [1/5] FROM docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0  20.4s
 => => resolve docker.io/library/node:18-alpine@sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a  0.0s
 => => sha256:929b04d7c782f04f615cf785488fed452b6569f87c73ff666ad553a7554f0006 1.72kB / 1.72kB         0.0s
 => => sha256:ee77c6cd7c1886ecc802ad6cedef3a8ec1ea27d1fb96162bf03dd3710839b8da 6.18kB / 6.18kB         0.0s
 => => sha256:f18232174bc91741fdf3da96d85011092101a032a93a388b79e99e69c2d5c870 3.64MB / 3.64MB         1.7s
 => => sha256:dd71dde834b5c203d162902e6b8994cb2309ae049a0eabc4efea161b2b5a3d0e 40.01MB / 40.01MB      15.2s
 => => sha256:1e5a4c89cee5c0826c540ab06d4b6b491c96eda01837f430bd47f0d26702d6e3 1.26MB / 1.26MB         2.0s
 => => sha256:8d6421d663b4c28fd3ebc498332f249011d118945588d0a35cb9bc4b8ca09d9e 7.67kB / 7.67kB         0.0s
 => => extracting sha256:f18232174bc91741fdf3da96d85011092101a032a93a388b79e99e69c2d5c870              0.4s
 => => sha256:25ff2da83641908f65c3a74d80409d6b1b62ccfaab220b9ea70b80df5a2e0549 446B / 446B             2.4s
 => => extracting sha256:dd71dde834b5c203d162902e6b8994cb2309ae049a0eabc4efea161b2b5a3d0e              4.4s
 => => extracting sha256:1e5a4c89cee5c0826c540ab06d4b6b491c96eda01837f430bd47f0d26702d6e3              0.1s
 => => extracting sha256:25ff2da83641908f65c3a74d80409d6b1b62ccfaab220b9ea70b80df5a2e0549              0.0s
 => [internal] load build context                                                                     0.8s
 => => transferring context: 2.41MB                                                                   0.8s
 => [2/5] WORKDIR /usr/src/app                                                                        0.3s
 => [3/5] COPY package*.json ./                                                                       0.1s
 => [4/5] RUN npm install                                                                             4.7s
 => [5/5] COPY . .                                                                                    0.8s
 => exporting to image                                                                                0.5s
 => => exporting layers                                                                               0.5s
 => => writing image sha256:9faac1e537581745643a4746c266efa43fb0377fd1225b9716ac1c2d57465bb3           0.0s
 => => naming to docker.io/library/node-app:1.0                                                       0.0s
hammad@ubuntu:~/SCD-SP25-NodeApp$ sudo docker run -d -p 3000:3000 --name node-container node-app:1.0
bff56f1703e95cb60f59b0a77462b645cee7a3f2abeceb2f6f4ecafb45498dab
hammad@ubuntu:~/SCD-SP25-NodeApp$ sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PORTS                                       NAMES
bff56f1703e9   node-app:1.0   "docker-entrypoint.s…"   12 seconds ago   Up 11 seconds   0.0.0.0:3000->3000/tcp, [::]:3000->3000/tcp   node-container
hammad@ubuntu:~/SCD-SP25-NodeApp$ curl http://localhost:3000/todo/1
{"userId":1,"id":1,"title":"delectus aut autem","completed":false}hammad@ubuntu:~/SCD-SP25-NodeApp$
hammad@ubuntu:~/SCD-SP25-NodeApp$ sudo lsof -i :3000
COMMAND     PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
docker-pr 48643 root    7u  IPv4 252001      0t0  TCP *:3000 (LISTEN)
docker-pr 48651 root    7u  IPv6 252002      0t0  TCP *:3000 (LISTEN)
hammad@ubuntu:~/SCD-SP25-NodeApp$ _
```

Testing the Application in the Container



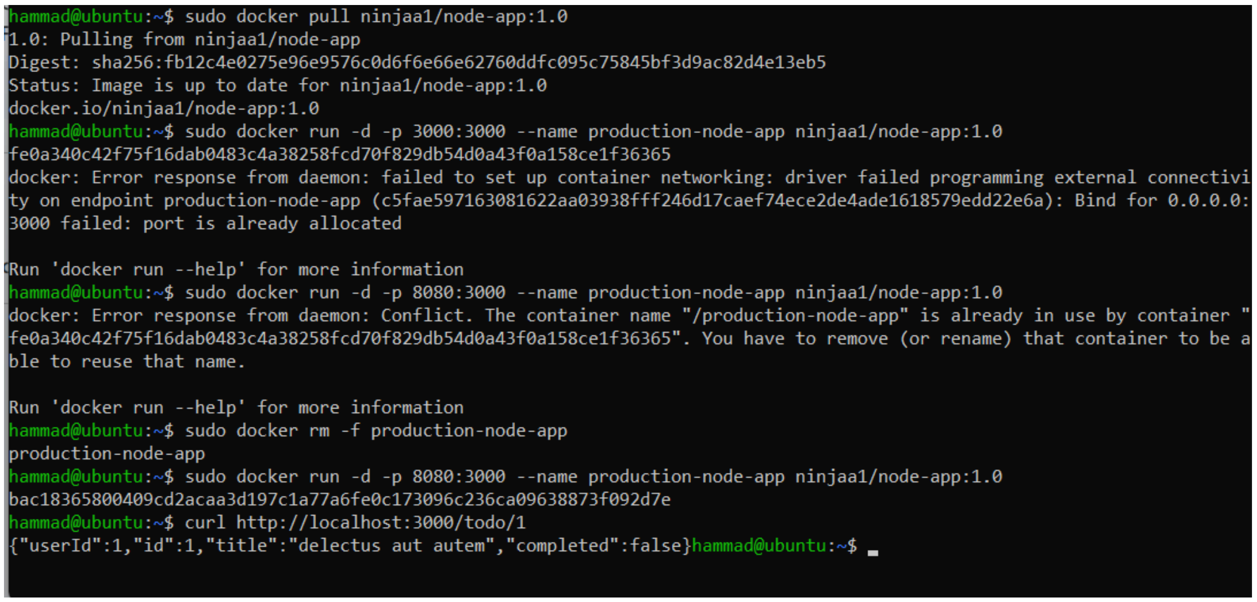
```
hammad@ubuntu:~$ sudo systemctl start docker
sudo systemctl enable docker
Synchronizing state of docker.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable docker
hammad@ubuntu:~$ sudo docker --version
Docker version 28.1.1, build 4eba377
hammad@ubuntu:~$ curl http://localhost:3000/todo/1
{"userId":1,"id":
hammad@ubuntu:~$ curl http://localhost:3000/todo/1
{"userId":1,"id":1,"title":"delectus aut autem","completed":false}hammad@ubuntu:~$
hammad@ubuntu:~$ sudo docker logs node-container
Server is running on http://localhost:3000
```

Publishing the Docker Image



```
Login Succeeded
hammad@ubuntu:~$ sudo docker tag node-app:1.0 ninjaa1/node-app:1.0
hammad@ubuntu:~$ sudo docker push ninjaa1/node-app:1.0
The push refers to repository [docker.io/ninjaa1/node-app]
aaa48ff3b554: Pushed
b90909e3d14e: Pushed
60c6433165b3: Pushed
0ae68ea2874a: Pushed
82140d9a70a7: Mounted from library/node
f3b40b0cdb1c: Mounted from library/node
0b1f26057bd0: Mounted from library/node
08000c18d16d: Mounted from library/node
1.0: digest: sha256:fb12c4e0275e96e9576c0d6f6e66e62760ddfc095c75845bf3d9ac82d4e13eb5 size: 1994
```

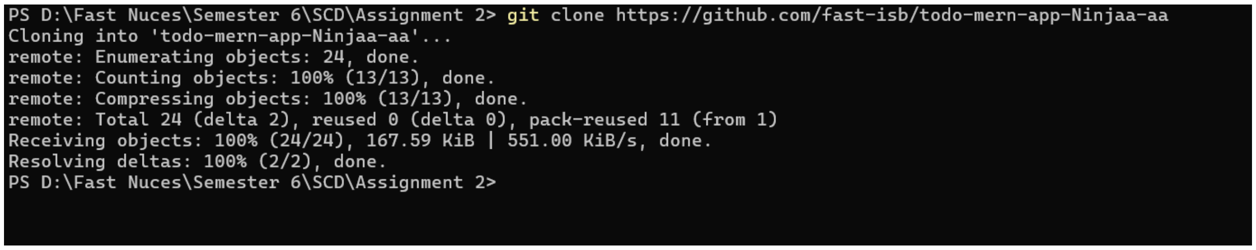
Running the Container on the Server Environment



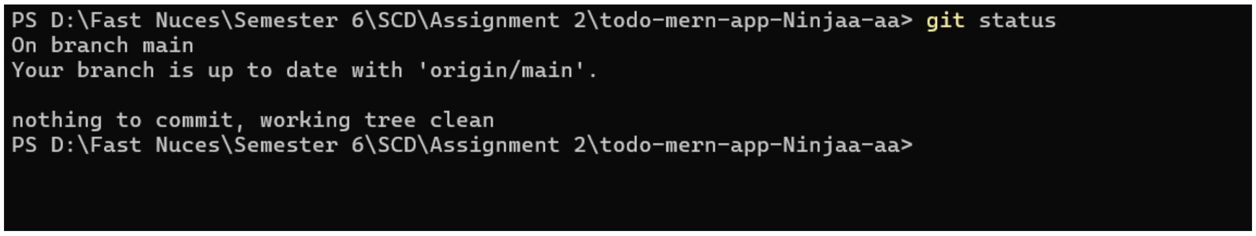
```
hammad@ubuntu:~$ sudo docker pull ninjaa1/node-app:1.0
1.0: Pulling from ninjaa1/node-app
Digest: sha256:fb12c4e0275e96e9576c0d6f6e66e62760ddfc095c75845bf3d9ac82d4e13eb5
Status: Image is up to date for ninjaa1/node-app:1.0
docker.io/ninjaa1/node-app:1.0
hammad@ubuntu:~$ sudo docker run -d -p 3000:3000 --name production-node-app ninjaa1/node-app:1.0
fe0a340c42f75f16dab0483c4a38258fcd70f829db54d0a43f0a158ce1f36365
docker: Error response from daemon: failed to set up container networking: driver failed programming external connectivi
ty on endpoint production-node-app (c5fae597163081622aa03938fff246d17caef74ece2de4ade1618579edd22e6a): Bind for 0.0.0.0:
3000 failed: port is already allocated

Run 'docker run --help' for more information
hammad@ubuntu:~$ sudo docker run -d -p 8080:3000 --name production-node-app ninjaa1/node-app:1.0
docker: Error response from daemon: Conflict. The container name "/production-node-app" is already in use by container "
fe0a340c42f75f16dab0483c4a38258fcd70f829db54d0a43f0a158ce1f36365". You have to remove (or rename) that container to be a
ble to reuse that name.

Run 'docker run --help' for more information
hammad@ubuntu:~$ sudo docker rm -f production-node-app
production-node-app
hammad@ubuntu:~$ sudo docker run -d -p 8080:3000 --name production-node-app ninjaa1/node-app:1.0
bac18365800409cd2acaa3d197c1a77a6fe0c173096c236ca09638873f092d7e
hammad@ubuntu:~$ curl http://localhost:3000/todo/1
{"userId":1,"id":1,"title":"delectus aut autem","completed":false}hammad@ubuntu:~$
```

Part 3: Building Features into a MERN Boilerplate

Cloning a repo:

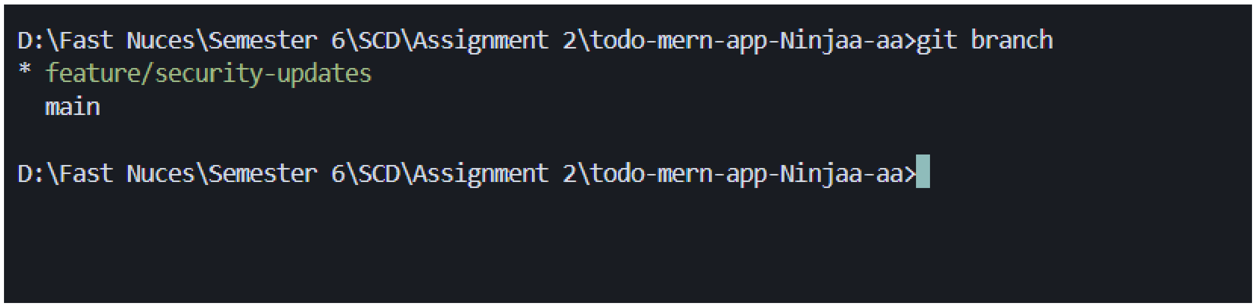


```
PS D:\Fast Nuces\Semester 6\SCD\Assignment 2> git clone https://github.com/fast-isb/todo-mern-app-Ninjaa-aa
Cloning into 'todo-mern-app-Ninjaa-aa'...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 24 (delta 2), reused 0 (delta 0), pack-reused 11 (from 1)
Receiving objects: 100% (24/24), 167.59 KiB | 551.00 KiB/s, done.
Resolving deltas: 100% (2/2), done.
PS D:\Fast Nuces\Semester 6\SCD\Assignment 2>
```



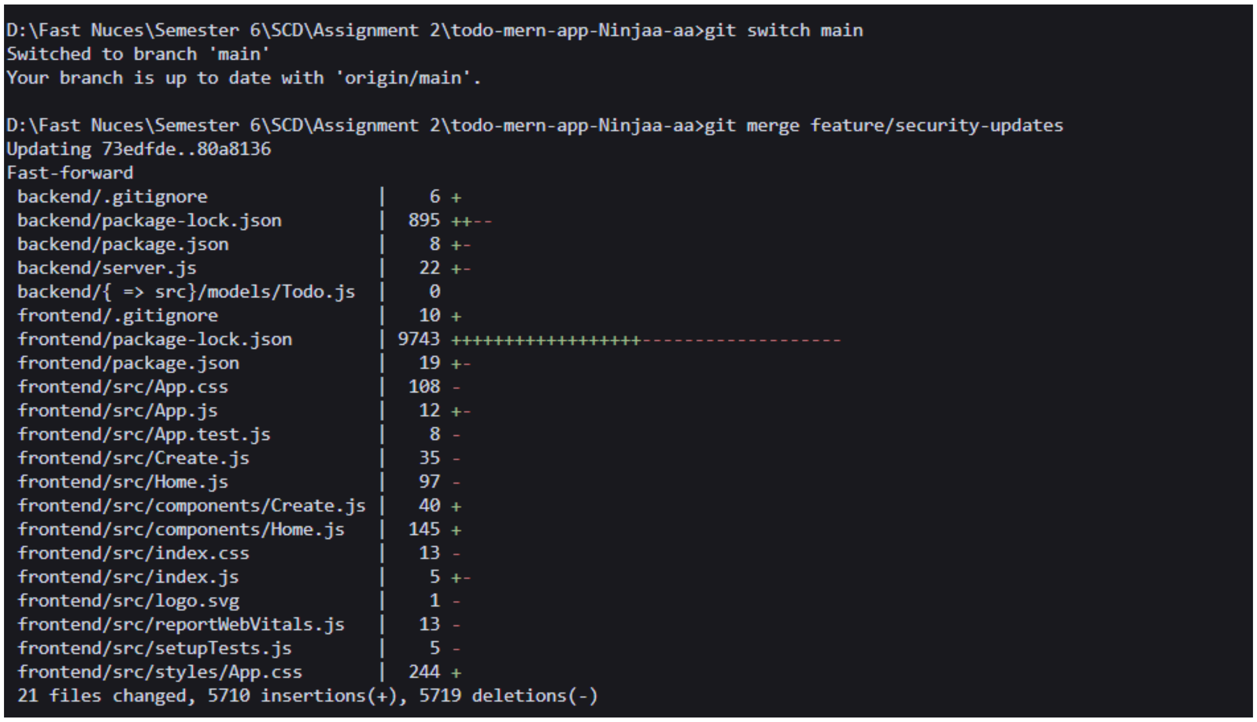
```
PS D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

Creating a new branch for feature updates:



```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>git branch
* feature/security-updates
  main

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

Merge the feature branch into main



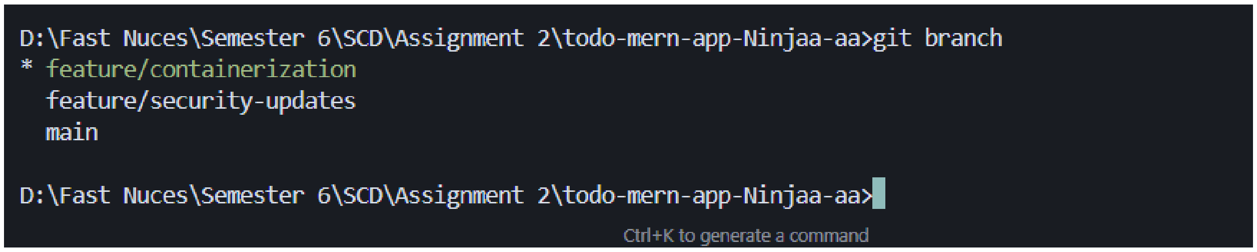
```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>git merge feature/security-updates
Updating 73edfde..80a8136
Fast-forward
 backend/.gitignore                 |    6 +
 backend/package-lock.json          |  895 ++--
 backend/package.json               |    8 +-
 backend/server.js                  |   22 +-
 backend/{ => src}/models/Todo.js   |    0
 frontend/.gitignore                |   10 +
 frontend/package-lock.json         | 9743 +++++++++++++++++++------------------------
 frontend/package.json              |   19 +-
 frontend/src/App.css               |  108 -
 frontend/src/App.js                |   12 +-
 frontend/src/App.test.js           |    8 -
 frontend/src/Create.js             |   35 -
 frontend/src/Home.js               |   97 -
 frontend/src/components/Create.js  |   40 +
 frontend/src/components/Home.js    |  145 +
 frontend/src/index.css             |   13 -
 frontend/src/index.js              |    5 +-
 frontend/src/logo.svg              |    1 -
 frontend/src/reportWebVitals.js    |   13 -
 frontend/src/setupTests.js         |    5 -
 frontend/src/styles/App.css        |  244 +
 21 files changed, 5710 insertions(+), 5719 deletions(-)
```

Part 4: Containerize the Application

Docker Hub link:

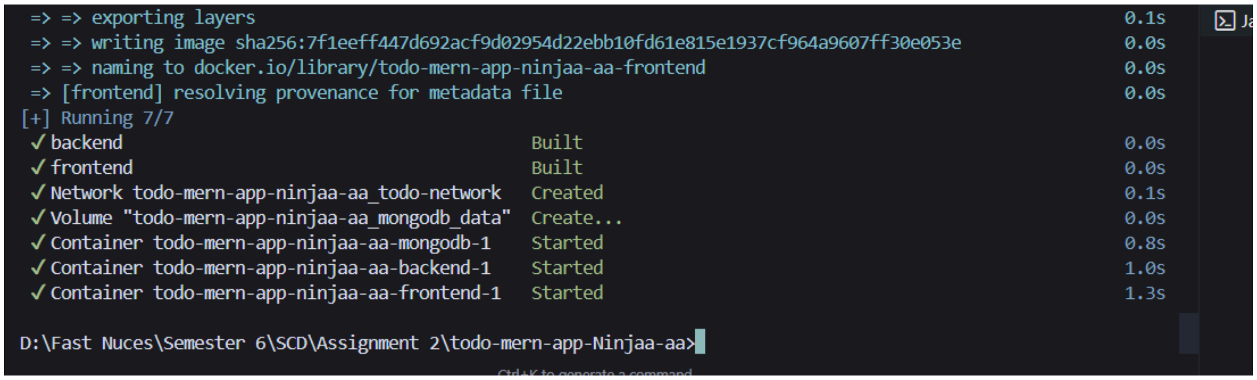
[Docker Hub]

Creating new branch for Containerization:



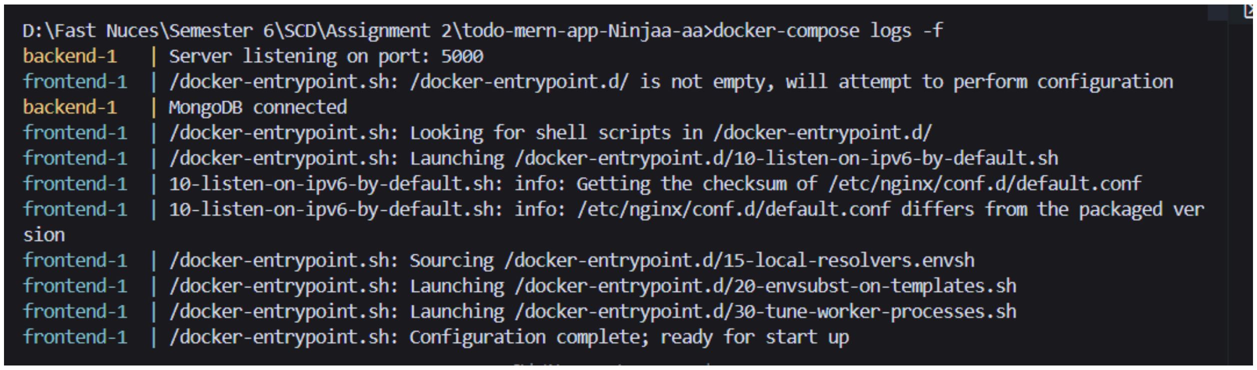
```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>git branch
* feature/containerization
  feature/security-updates
  main

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

Building the docker image:



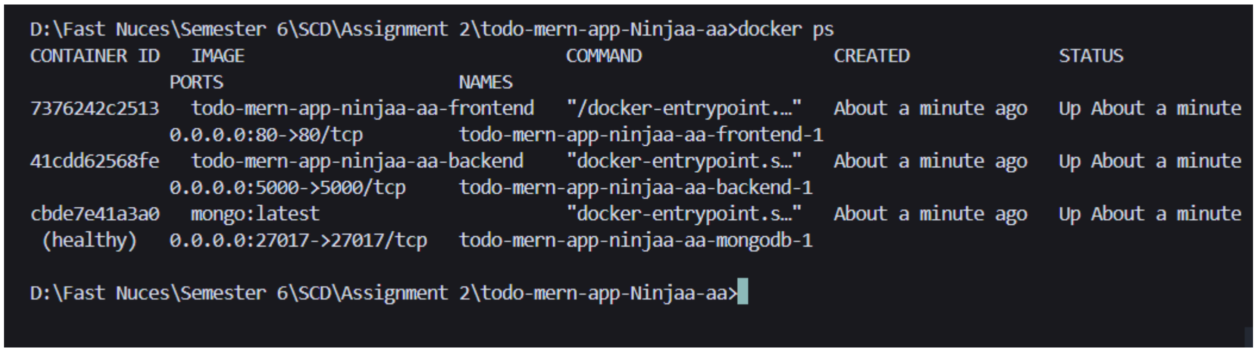
```
 => => exporting layers                                                                0.1s
 => => writing image sha256:7f1eeff447d692acf9d02954d22ebb10fd61e815e1937cf964a9607ff30e053e  0.0s
 => => naming to docker.io/library/todo-mern-app-ninjaa-aa-frontend                    0.0s
 => [frontend] resolving provenance for metadata file                                  0.0s
[+] Running 7/7
 ✓ backend                                          Built                              0.0s
 ✓ frontend                                         Built                              0.0s
 ✓ Network todo-mern-app-ninjaa-aa_todo-network     Created                            0.1s
 ✓ Volume "todo-mern-app-ninjaa-aa_mongodb_data"    Create...                          0.0s
 ✓ Container todo-mern-app-ninjaa-aa-mongodb-1      Started                            0.8s
 ✓ Container todo-mern-app-ninjaa-aa-backend-1      Started                            1.0s
 ✓ Container todo-mern-app-ninjaa-aa-frontend-1     Started                            1.3s

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

Docker compose logs:



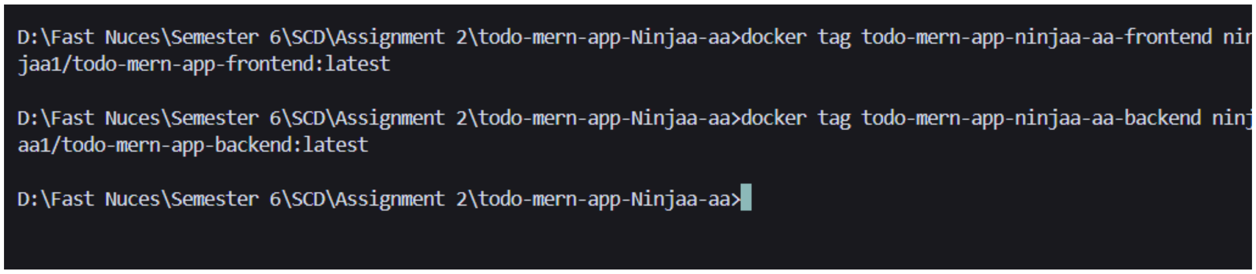
```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker-compose logs -f
backend-1   | Server listening on port: 5000
frontend-1  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
backend-1   | MongoDB connected
frontend-1  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
frontend-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
frontend-1  | 10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
frontend-1  | 10-listen-on-ipv6-by-default.sh: info: /etc/nginx/conf.d/default.conf differs from the packaged ver
sion
frontend-1  | /docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
frontend-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
frontend-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
frontend-1  | /docker-entrypoint.sh: Configuration complete; ready for start up
```

Running containers:



```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker ps
CONTAINER ID   IMAGE                              COMMAND                  CREATED              STATUS
             PORTS                      NAMES
7376242c2513   todo-mern-app-ninjaa-aa-frontend   "/docker-entrypoint.…"   About a minute ago   Up About a minute
             0.0.0.0:80->80/tcp         todo-mern-app-ninjaa-aa-frontend-1
41cdd62568fe   todo-mern-app-ninjaa-aa-backend    "docker-entrypoint.s…"   About a minute ago   Up About a minute
             0.0.0.0:5000->5000/tcp     todo-mern-app-ninjaa-aa-backend-1
cbde7e41a3a0   mongo:latest                       "docker-entrypoint.s…"   About a minute ago   Up About a minute
 (healthy)   0.0.0.0:27017->27017/tcp   todo-mern-app-ninjaa-aa-mongodb-1

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

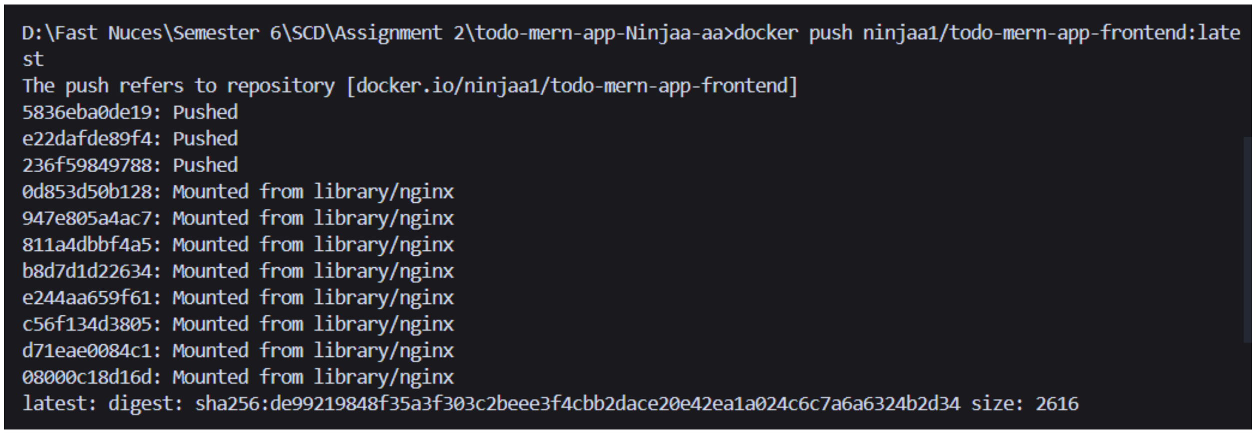
Tag frontend and backend images:



```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker tag todo-mern-app-ninjaa-aa-frontend nin
jaa1/todo-mern-app-frontend:latest

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker tag todo-mern-app-ninjaa-aa-backend ninj
aa1/todo-mern-app-backend:latest

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

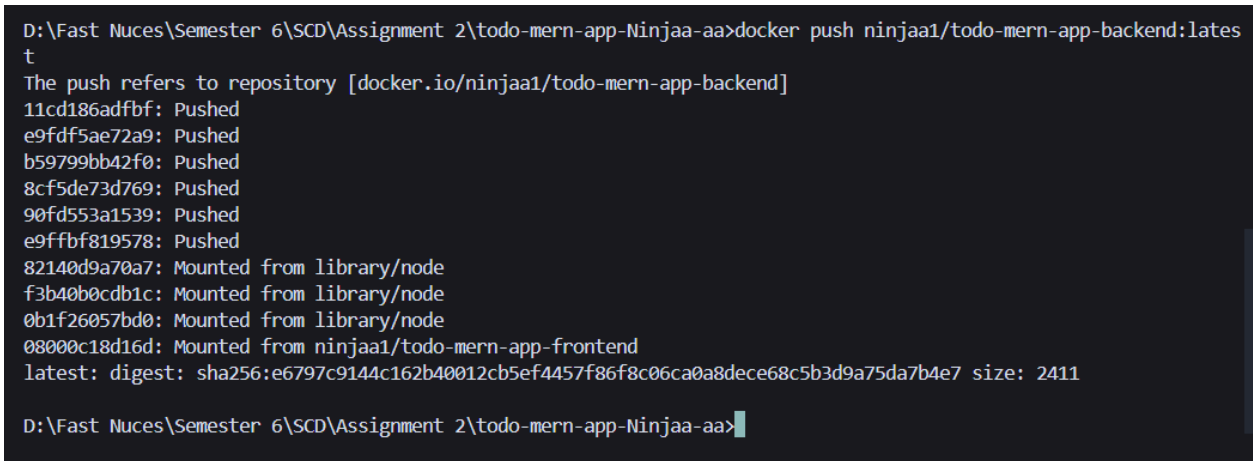
Pushing images:

Frontend image:



```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker push ninjaa1/todo-mern-app-frontend:late
st
The push refers to repository [docker.io/ninjaa1/todo-mern-app-frontend]
5836eba0de19: Pushed
e22dafde89f4: Pushed
236f59849788: Pushed
0d853d50b128: Mounted from library/nginx
947e805a4ac7: Mounted from library/nginx
811a4dbbf4a5: Mounted from library/nginx
b8d7d1d22634: Mounted from library/nginx
e244aa659f61: Mounted from library/nginx
c56f134d3805: Mounted from library/nginx
d71eae0084c1: Mounted from library/nginx
08000c18d16d: Mounted from library/nginx
latest: digest: sha256:de99219848f35a3f303c2beee3f4cbb2dace20e42ea1a024c6c7a6a6324b2d34 size: 2616
```

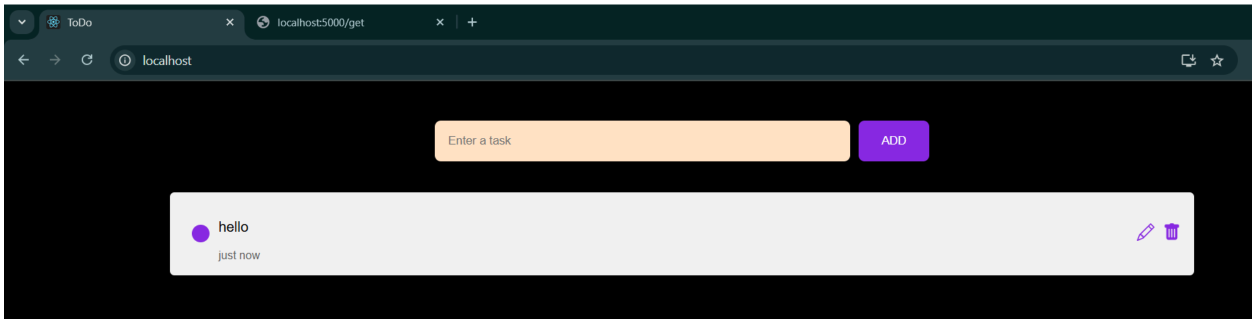
Backend image:



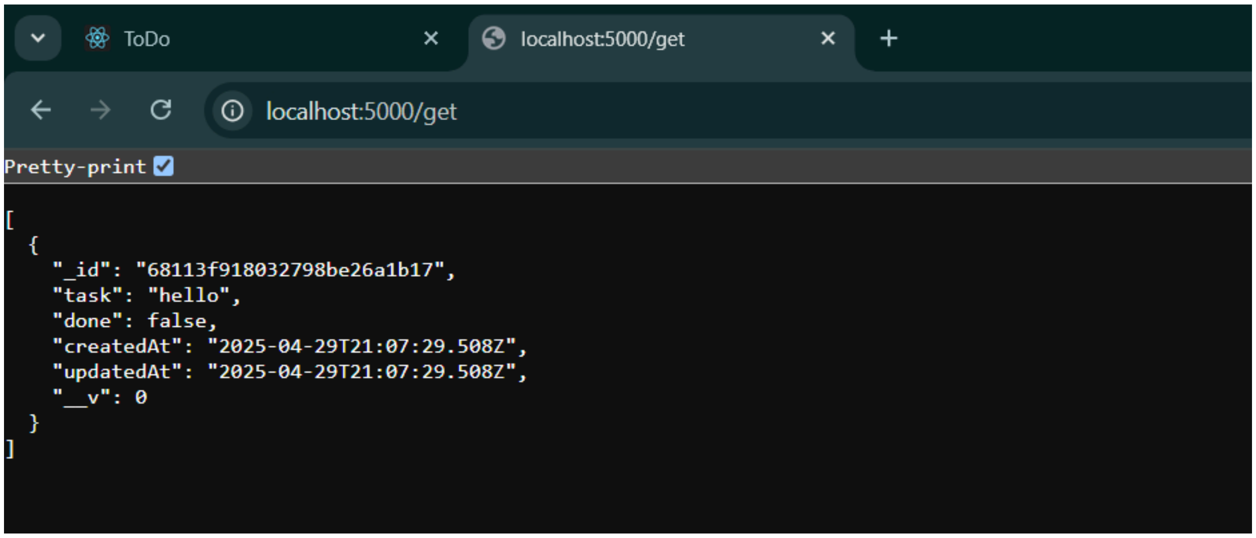
```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker push ninjaa1/todo-mern-app-backend:lates
t
The push refers to repository [docker.io/ninjaa1/todo-mern-app-backend]
11cd186adfbf: Pushed
e9fdf5ae72a9: Pushed
b59799bb42f0: Pushed
8cf5de73d769: Pushed
90fd553a1539: Pushed
e9ffbf819578: Pushed
82140d9a70a7: Mounted from library/node
f3b40b0cdb1c: Mounted from library/node
0b1f26057bd0: Mounted from library/node
08000c18d16d: Mounted from ninjaa1/todo-mern-app-frontend
latest: digest: sha256:e6797c9144c162b40012cb5ef4457f86f8c06ca0a8dece68c5b3d9a75da7b4e7 size: 2411

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

Running on localhost:

Frontend:



Backend:

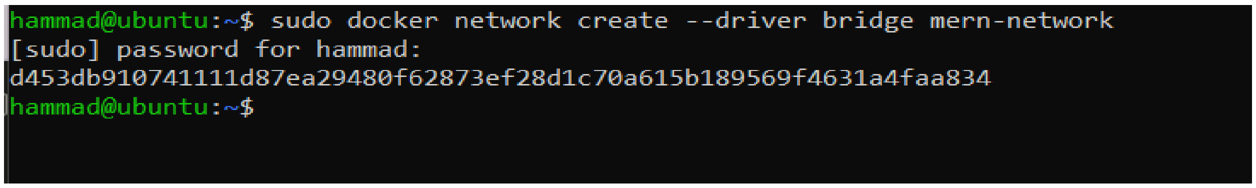


```
[
  {
    "_id": "68113f918032798be26a1b17",
    "task": "hello",
    "done": false,
    "createdAt": "2025-04-29T21:07:29.508Z",
    "updatedAt": "2025-04-29T21:07:29.508Z",
    "__v": 0
  }
]
```

Part 5: Deploy Containers Manually

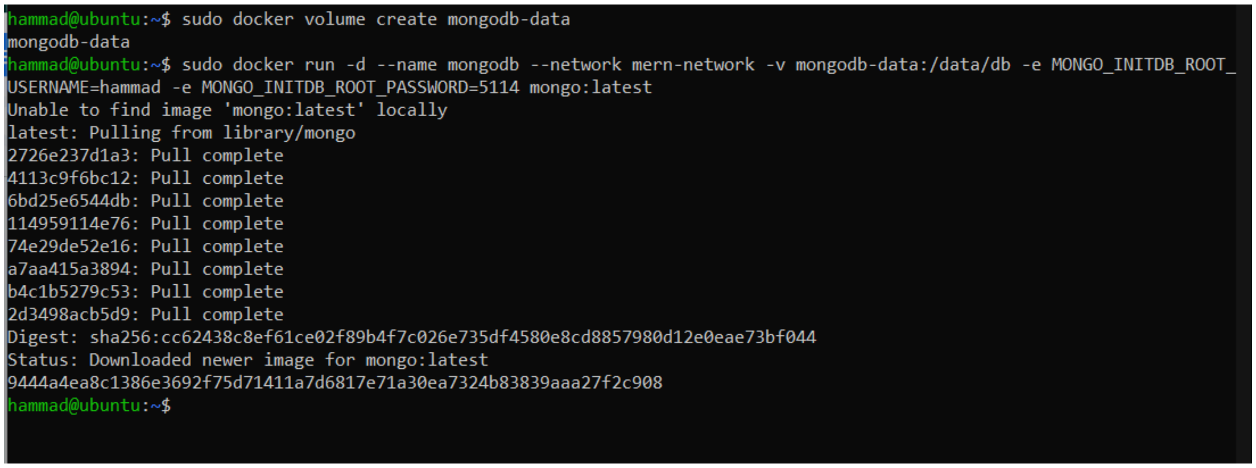
Screenshots:

Creating a private network:



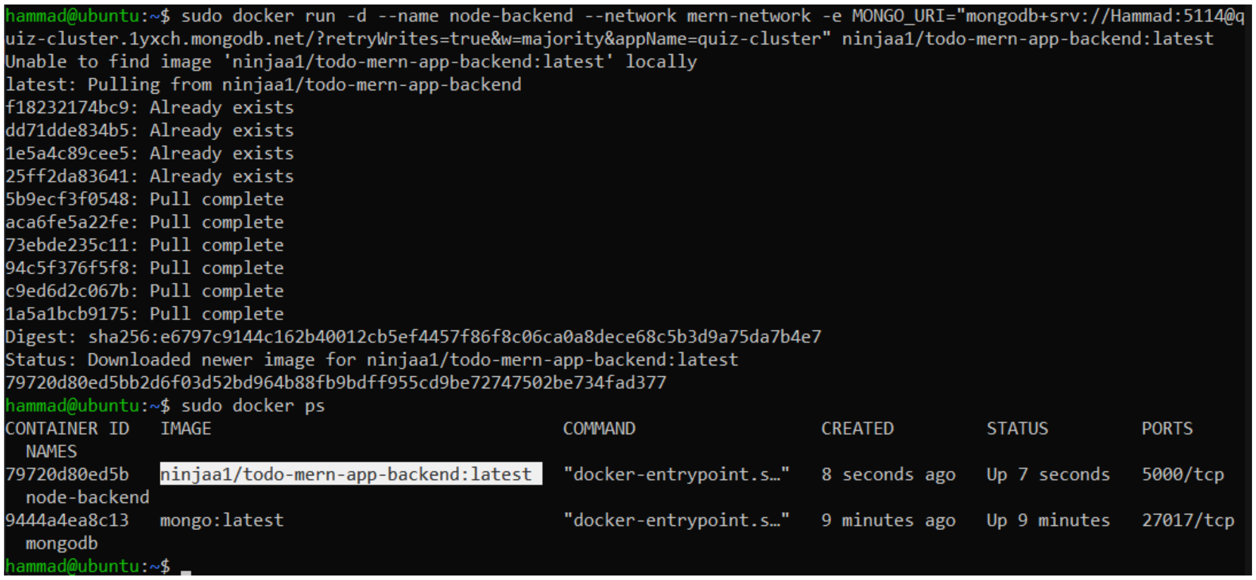
```
hammad@ubuntu:~$ sudo docker network create --driver bridge mern-network
[sudo] password for hammad:
d453db910741111d87ea29480f62873ef28d1c70a615b189569f4631a4faa834
hammad@ubuntu:~$
```

Creating volume, and run mongo container:



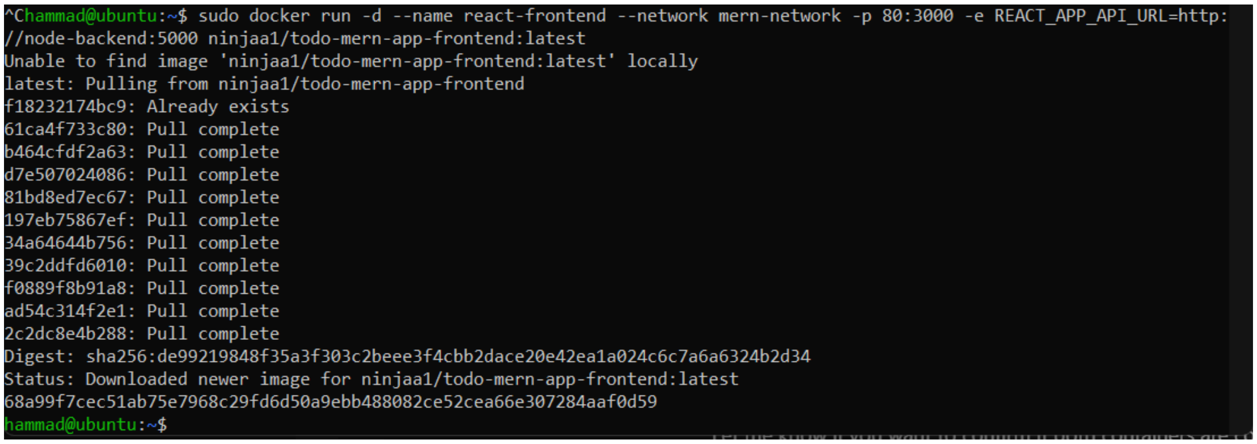
```
hammad@ubuntu:~$ sudo docker volume create mongodb-data
mongodb-data
hammad@ubuntu:~$ sudo docker run -d --name mongodb --network mern-network -v mongodb-data:/data/db -e MONGO_INITDB_ROOT_
USERNAME=hammad -e MONGO_INITDB_ROOT_PASSWORD=5114 mongo:latest
Unable to find image 'mongo:latest' locally
latest: Pulling from library/mongo
2726e237d1a3: Pull complete
4113c9f6bc12: Pull complete
6bd25e6544db: Pull complete
114959114e76: Pull complete
74e29de52e16: Pull complete
a7aa415a3894: Pull complete
b4c1b5279c53: Pull complete
2d3498acb5d9: Pull complete
Digest: sha256:cc62438c8ef61ce02f89b4f7c026e735df4580e8cd8857980d12e0eae73bf044
Status: Downloaded newer image for mongo:latest
9444a4ea8c1386e3692f75d71411a7d6817e71a30ea7324b83839aaa27f2c908
hammad@ubuntu:~$
```

Deploy Backend Container

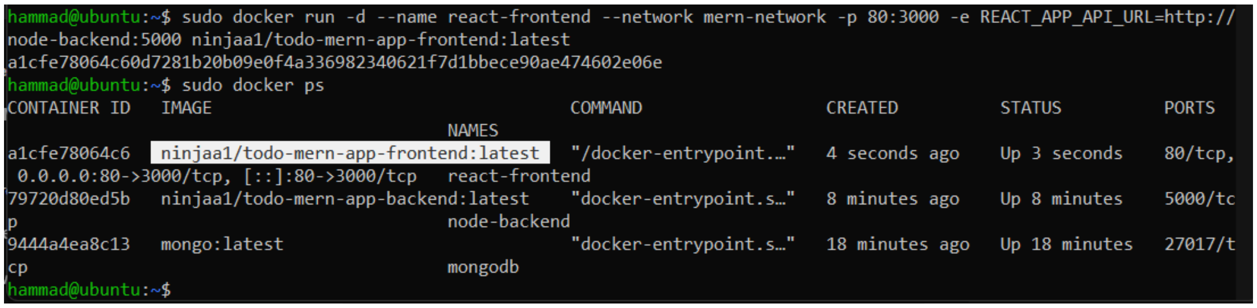


```
hammad@ubuntu:~$ sudo docker run -d --name node-backend --network mern-network -e MONGO_URI="mongodb+srv://Hammad:5114@q
uiz-cluster.1yxch.mongodb.net/?retryWrites=true&w=majority&appName=quiz-cluster" ninjaa1/todo-mern-app-backend:latest
Unable to find image 'ninjaa1/todo-mern-app-backend:latest' locally
latest: Pulling from ninjaa1/todo-mern-app-backend
f18232174bc9: Already exists
dd71dde834b5: Already exists
1e5a4c89cee5: Already exists
25ff2da83641: Already exists
5b9ecf3f0548: Pull complete
aca6fe5a22fe: Pull complete
73ebde235c11: Pull complete
94c5f376f5f8: Pull complete
c9ed6d2c067b: Pull complete
1a5a1bcb9175: Pull complete
Digest: sha256:e6797c9144c162b40012cb5ef4457f86f8c06ca0a8dece68c5b3d9a75da7b4e7
Status: Downloaded newer image for ninjaa1/todo-mern-app-backend:latest
79720d80ed5bb2d6f03d52bd964b88fb9bdff955cd9be72747502be734fad377
hammad@ubuntu:~$ sudo docker ps
CONTAINER ID   IMAGE                                  COMMAND                  CREATED         STATUS         PORTS
  NAMES
79720d80ed5b   ninjaa1/todo-mern-app-backend:latest   "docker-entrypoint.s…"   8 seconds ago   Up 7 seconds   5000/tcp
  node-backend
9444a4ea8c13   mongo:latest                           "docker-entrypoint.s…"   9 minutes ago   Up 9 minutes   27017/tcp
  mongodb
hammad@ubuntu:~$
```

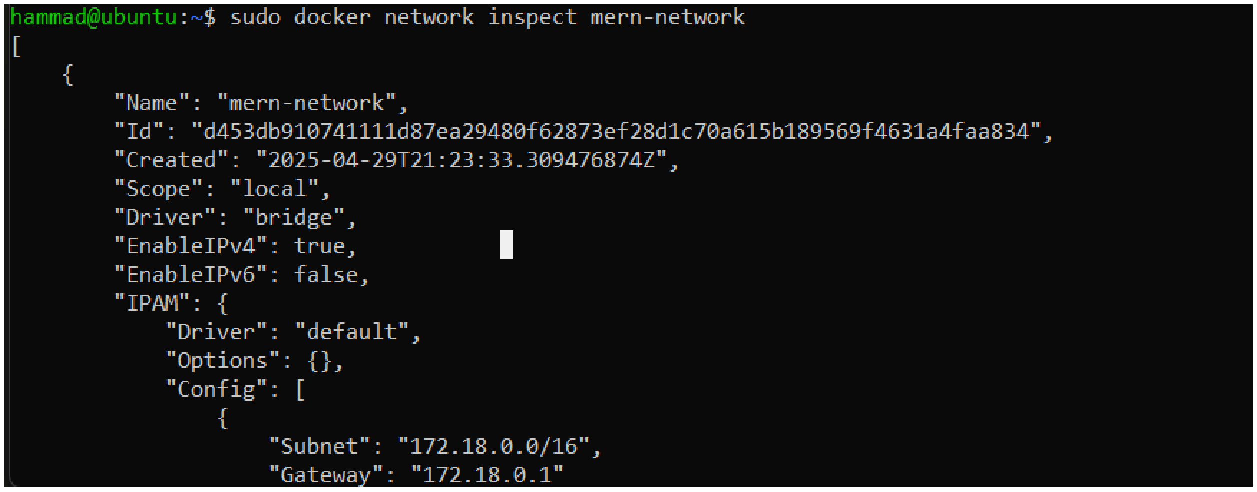
Deploy Frontend Container with Public Access



```
^Chammad@ubuntu:~$ sudo docker run -d --name react-frontend --network mern-network -p 80:3000 -e REACT_APP_API_URL=http:
//node-backend:5000 ninjaa1/todo-mern-app-frontend:latest
Unable to find image 'ninjaa1/todo-mern-app-frontend:latest' locally
latest: Pulling from ninjaa1/todo-mern-app-frontend
f18232174bc9: Already exists
61ca4f733c80: Pull complete
b464cfdf2a63: Pull complete
d7e507024086: Pull complete
81bd8ed7ec67: Pull complete
197eb75867ef: Pull complete
34a64644b756: Pull complete
39c2ddfd6010: Pull complete
f0889f8b91a8: Pull complete
ad54c314f2e1: Pull complete
2c2dc8e4b288: Pull complete
Digest: sha256:de99219848f35a3f303c2beee3f4cbb2dace20e42ea1a024c6c7a6a6324b2d34
Status: Downloaded newer image for ninjaa1/todo-mern-app-frontend:latest
68a99f7cec51ab75e7968c29fd6d50a9ebb488082ce52cea66e307284aaf0d59
hammad@ubuntu:~$
```

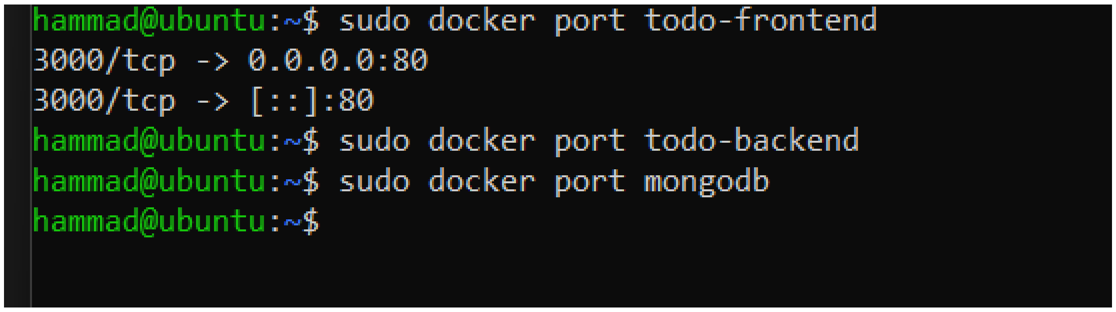


```
hammad@ubuntu:~$ sudo docker run -d --name react-frontend --network mern-network -p 80:3000 -e REACT_APP_API_URL=http://
node-backend:5000 ninjaa1/todo-mern-app-frontend:latest
a1cfe78064c60d7281b20b09e0f4a336982340621f7d1bbece90ae474602e06e
hammad@ubuntu:~$ sudo docker ps
CONTAINER ID   IMAGE                                   COMMAND                  CREATED          STATUS          PORTS
                                                NAMES
a1cfe78064c6   ninjaa1/todo-mern-app-frontend:latest   "/docker-entrypoint.…"   4 seconds ago    Up 3 seconds    80/tcp,
 0.0.0.0:80->3000/tcp, [::]:80->3000/tcp        react-frontend
79720d80ed5b   ninjaa1/todo-mern-app-backend:latest    "docker-entrypoint.s…"   8 minutes ago    Up 8 minutes    5000/tc
p                                               node-backend
9444a4ea8c13   mongo:latest                            "docker-entrypoint.s…"   18 minutes ago   Up 18 minutes   27017/t
cp                                              mongodb
hammad@ubuntu:~$
```



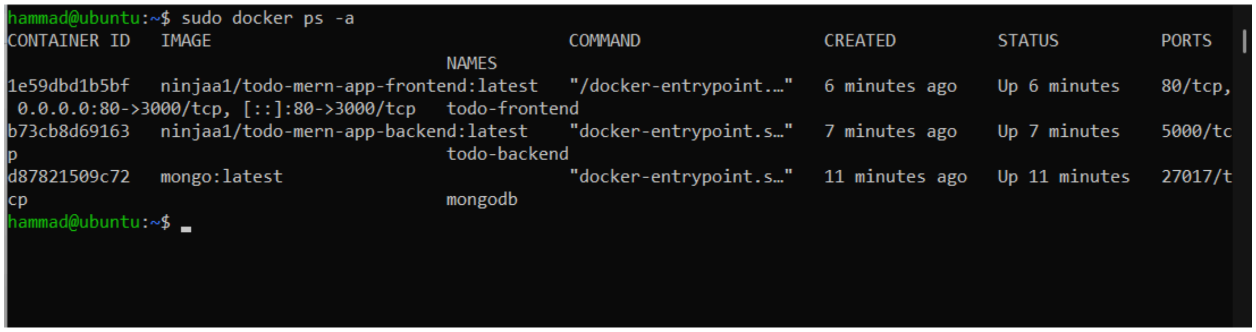
```
hammad@ubuntu:~$ sudo docker network inspect mern-network
[
    {
        "Name": "mern-network",
        "Id": "d453db910741111d87ea29480f62873ef28d1c70a615b189569f4631a4faa834",
        "Created": "2025-04-29T21:23:33.309476874Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv4": true,
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
```

Port Checking



```
hammad@ubuntu:~$ sudo docker port todo-frontend
3000/tcp -> 0.0.0.0:80
3000/tcp -> [::]:80
hammad@ubuntu:~$ sudo docker port todo-backend
hammad@ubuntu:~$ sudo docker port mongodb
hammad@ubuntu:~$
```

Running containers:



```
hammad@ubuntu:~$ sudo docker ps -a
CONTAINER ID   IMAGE                                 COMMAND                  CREATED          STATUS          PORTS
                                           NAMES
1e59dbd1b5bf   ninjaa1/todo-mern-app-frontend:latest   "/docker-entrypoint.…"   6 minutes ago    Up 6 minutes    80/tcp,
 0.0.0.0:80->3000/tcp, [::]:80->3000/tcp   todo-frontend
b73cb8d69163   ninjaa1/todo-mern-app-backend:latest    "docker-entrypoint.s…"   7 minutes ago    Up 7 minutes    5000/tc
p                                          todo-backend
d87821509c72   mongo:latest                            "docker-entrypoint.s…"   11 minutes ago   Up 11 minutes   27017/t
cp                                         mongodb
hammad@ubuntu:~$
```

Manual Docker Container Deployment Documentation

```
Docker Commands Used for Deployment
1. Network Configuration
# Create a private bridge network for containers
docker network create todo-network

2. MongoDB Deployment
# Create a named volume for MongoDB data persistence
docker volume create mongodb-data

# Run MongoDB container with volume attachment
docker run -d --name mongodb \
  --network todo-network \
  -v mongodb-data:/data/db \
  mongo:latest

3. Backend Application Deployment
# Run the backend container
```

```
docker run -d --name todo-backend \
  --network todo-network \
  -e MONGODB_URI=mongodb://mongodb:27017/todos \
  ninjaa1/todo-mern-app-backend:latest

4. Frontend Application Deployment
# Run the frontend container with port mapping for public access
docker run -d --name todo-frontend \
  --network todo-network \
  -p 80:3000 \
  -e REACT_APP_API_URL=http://todo-backend:5000 \
  ninjaa1/todo-mern-app-frontend:latest

5. Verification Commands
# List all running containers
docker ps

# Inspect the network configuration
docker network inspect todo-network

# Check the port mappings
docker port todo-frontend
docker port todo-backend
docker port mongodb

6. Data Persistence Testing
# Stop and remove all containers (but keep the volume)
docker stop todo-frontend todo-backend mongodb
docker rm todo-frontend todo-backend mongodb

# Restart containers
docker run -d --name mongodb \
  --network todo-network \
  -v mongodb-data:/data/db \
  mongo:latest

docker run -d --name todo-backend \
  --network todo-network \
  -e MONGODB_URI=mongodb://mongodb:27017/todos \
  ninjaa1/todo-mern-app-backend:latest
```

```
docker run -d --name todo-frontend \
  --network todo-network \
  -p 80:3000 \
  -e REACT_APP_API_URL=http://todo-backend:5000 \
  ninjaa1/todo-mern-app-frontend:latest
```

Network Privacy Verification

The private network configuration ensures that:

1. **Only the frontend container is publicly accessible** - We only mapped port 80 on the host to port 3000 on the frontend container
2. **Backend and MongoDB containers are not accessible from outside** - No port mappings were created for these containers

This can be verified with the following commands:

```
# This will show only frontend has port mappings
docker ps

# Should show no port mappings for backend
docker port todo-backend

# Should show no port mappings for MongoDB
docker port mongodb
```

When executing these commands, you should see output similar to:

```
CONTAINER ID   IMAGE                              COMMAND                  CREATED          STATUS
PORTS                              NAMES
a1b2c3d4e5f6   ninjaa1/todo-mern-app-frontend:latest   "docker-entrypoint.s…"   10 minutes ago
Up 10 minutes   0.0.0.0:80->3000/tcp, :::80->3000/tcp   todo-frontend
g7h8i9j0k1l2   ninjaa1/todo-mern-app-backend:latest    "docker-entrypoint.s…"   10 minutes ago   Up
10 minutes                                      todo-backend
m3n4o5p6q7r8   mongo:latest                       "docker-entrypoint.s…"   15 minutes ago   Up 15
minutes                                      mongodb
```

Persistent Volume Verification

The MongoDB data is stored in a named volume mongodb-data which persists even when containers are removed. This can be verified by:

1. Adding data to the application
2. Stopping and removing all containers
3. Restarting the containers with the same volume attachment
4. Verifying that the data is still available in the application

Challenges and Difficulties in Manual Setup

1. Container Dependency Order

Challenge: Containers need to be started in the correct order to ensure proper operation.

- MongoDB must be started first
- Backend needs to wait for MongoDB to be ready
- Frontend depends on backend

Impact: Without orchestration tools, we have no built-in way to handle startup order, which can cause connection failures if the backend tries to connect before MongoDB is ready.

2. Environment Variable Management

Challenge: Each container requires different environment variables to function properly.

- Coordinating these variables between containers is error-prone
- Ensuring consistency in environment variables across container restarts
- Manually typing long commands increases chances of typos

Impact: Incorrect environment variables can cause subtle failures that are difficult to debug.

3. Network Configuration Complexity

Challenge: Setting up proper network isolation requires careful planning.

- Ensuring containers can communicate with each other
- Restricting public access appropriately
- Understanding Docker's internal DNS resolution

Impact: Improper network configuration can lead to either security issues or application connectivity problems.

4. Volume Management

Challenge: Properly configuring volumes for data persistence.

- Understanding where each application stores its data
- Ensuring proper permissions on mounted volumes
- Managing cleanup of unused volumes

Impact: Data loss can occur if volumes are not properly configured or accidentally removed.

5. Service Discovery

Challenge: Without Docker Compose or Kubernetes, services must find each other manually.

- Relying on container names as hostnames
- No automatic DNS resolution between containers without custom networks
- No built-in health checks

Impact: Applications may fail to start or operate correctly if they cannot locate their dependencies.

Time and Effort Analysis

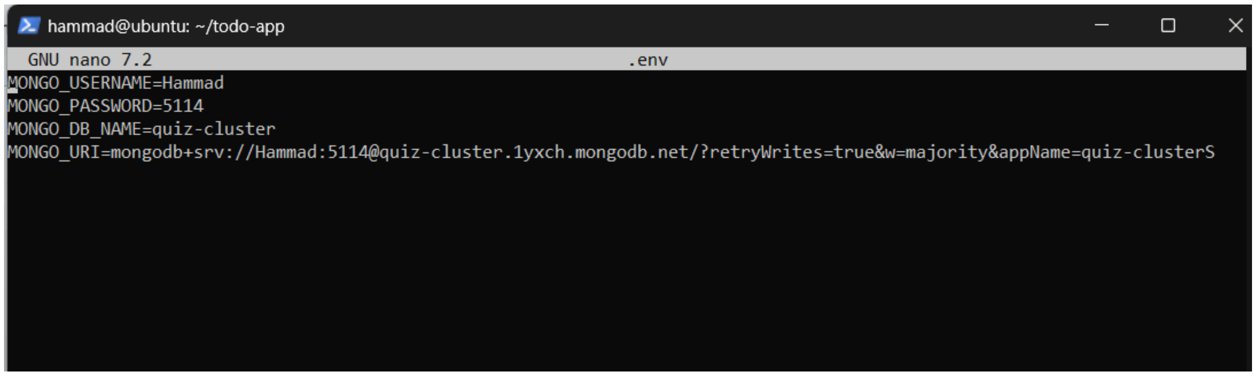
The manual deployment process required:

1. **Research Time:** ~30 minutes to understand each container's requirements
2. **Configuration Planning:** ~15 minutes to plan network, volumes, and environment variables
3. **Deployment Execution:** ~20 minutes to run commands and verify proper operation
4. **Troubleshooting:** ~45 minutes to resolve connectivity issues and environment variable problems
5. **Documentation:** ~30 minutes to document all commands and configuration details

Total Effort: Approximately 2-3 hours for initial setup, with additional time needed for any changes or updates to the deployment.

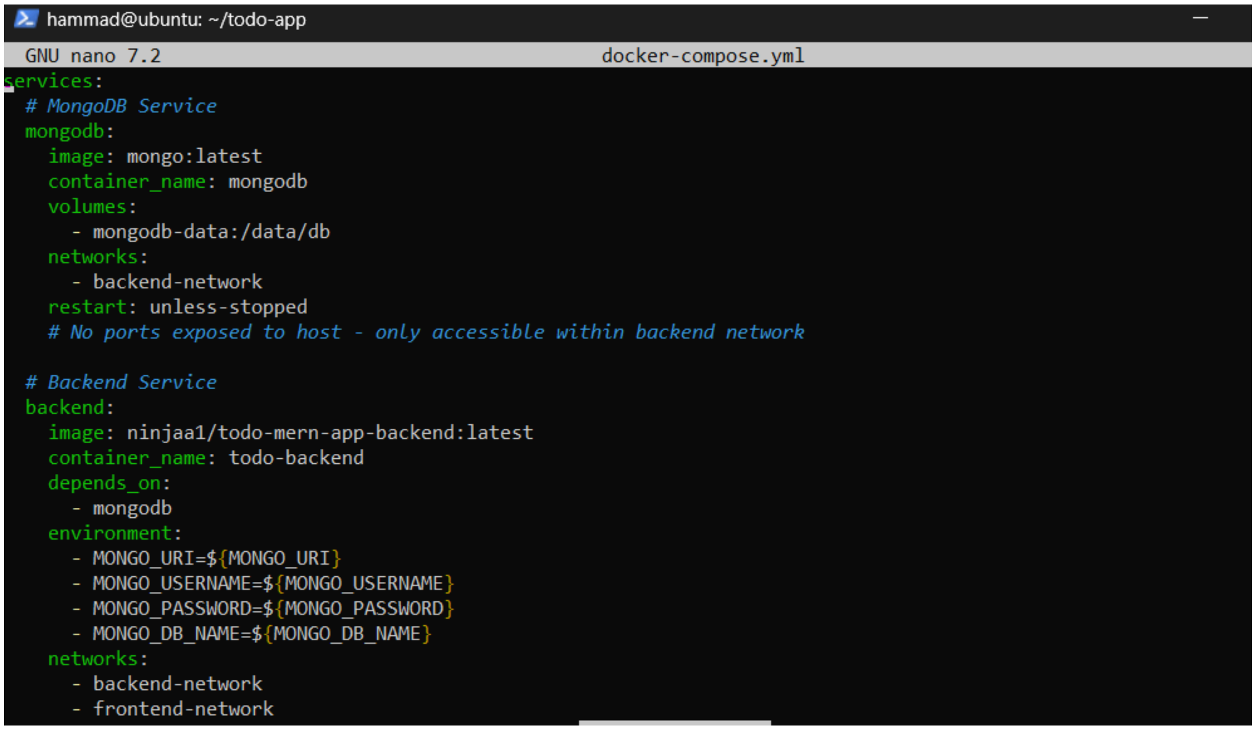
This manual process is significantly more time-consuming and error-prone compared to using container orchestration tools like Docker Compose or Kubernetes, which would automate many of these steps and provide better service management.

Part 6: Simplifying with Docker Compose



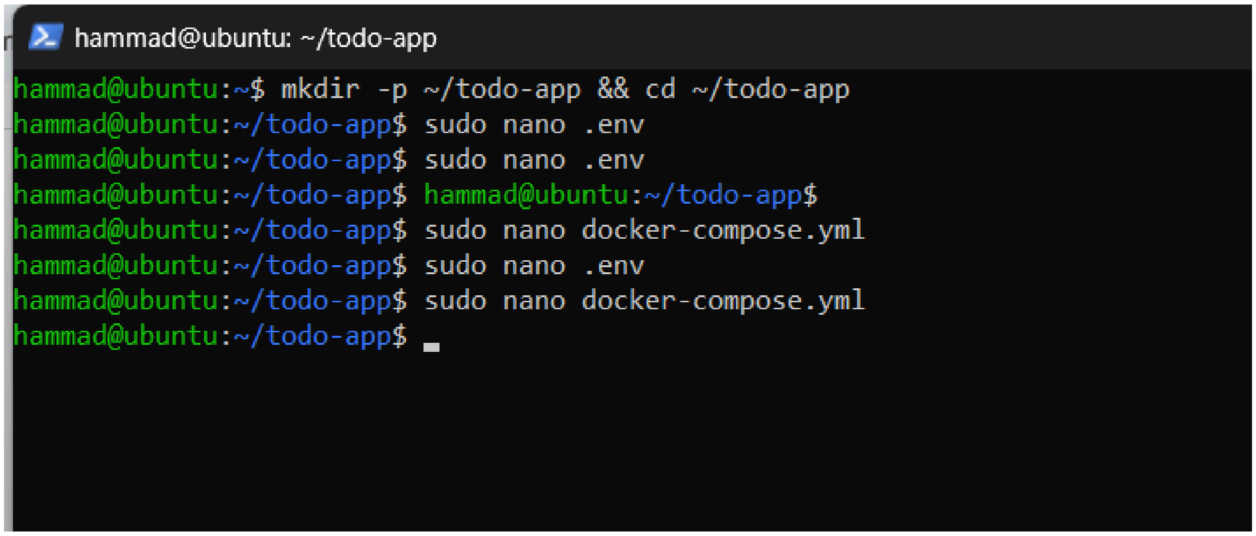
hammad@ubuntu: ~/todo-app

```
 GNU nano 7.2                                          .env
MONGO_USERNAME=Hammad
MONGO_PASSWORD=5114
MONGO_DB_NAME=quiz-cluster
MONGO_URI=mongodb+srv://Hammad:5114@quiz-cluster.1yxch.mongodb.net/?retryWrites=true&w=majority&appName=quiz-clusterS
```



hammad@ubuntu: ~/todo-app

```
 GNU nano 7.2                                   docker-compose.yml
services:
  # MongoDB Service
  mongodb:
    image: mongo:latest
    container_name: mongodb
    volumes:
      - mongodb-data:/data/db
    networks:
      - backend-network
    restart: unless-stopped
    # No ports exposed to host - only accessible within backend network

  # Backend Service
  backend:
    image: ninjaa1/todo-mern-app-backend:latest
    container_name: todo-backend
    depends_on:
      - mongodb
    environment:
      - MONGO_URI=${MONGO_URI}
      - MONGO_USERNAME=${MONGO_USERNAME}
      - MONGO_PASSWORD=${MONGO_PASSWORD}
      - MONGO_DB_NAME=${MONGO_DB_NAME}
    networks:
      - backend-network
      - frontend-network
```

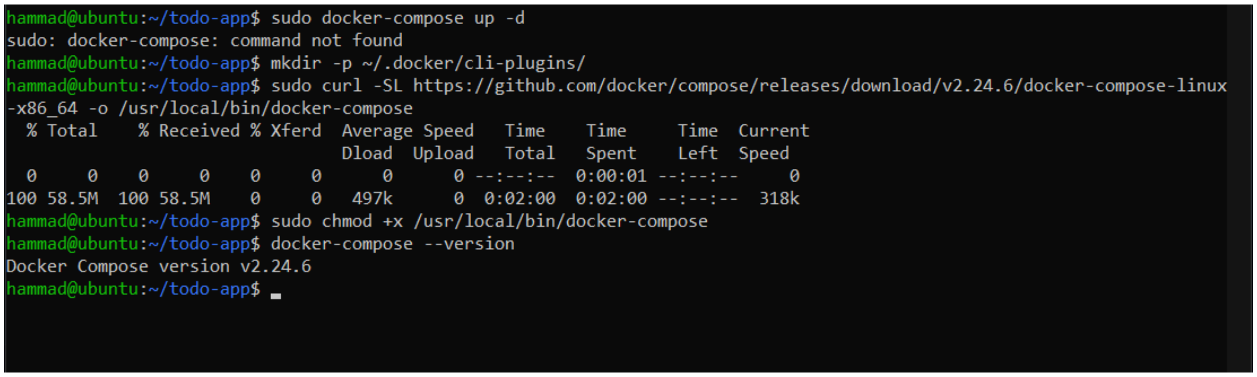


```
hammad@ubuntu: ~/todo-app
hammad@ubuntu:~$ mkdir -p ~/todo-app && cd ~/todo-app
hammad@ubuntu:~/todo-app$ sudo nano .env
hammad@ubuntu:~/todo-app$ sudo nano .env
hammad@ubuntu:~/todo-app$ hammad@ubuntu:~/todo-app$
hammad@ubuntu:~/todo-app$ sudo nano docker-compose.yml
hammad@ubuntu:~/todo-app$ sudo nano .env
hammad@ubuntu:~/todo-app$ sudo nano docker-compose.yml
hammad@ubuntu:~/todo-app$ _
```

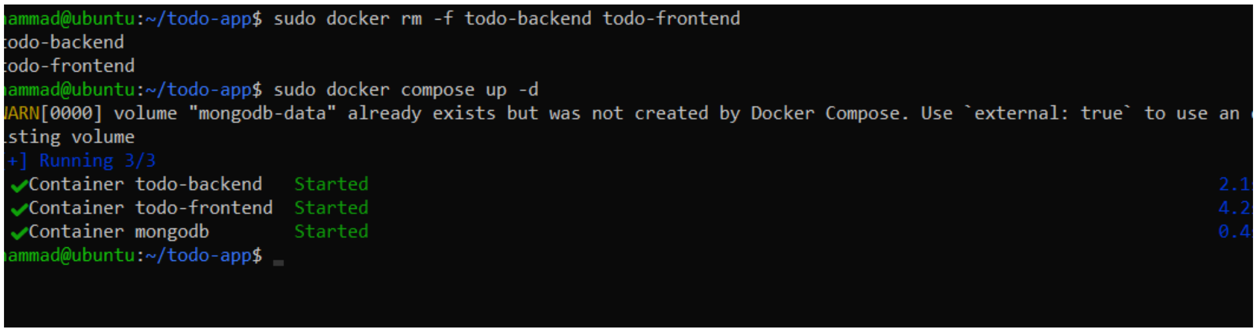
Docker Compose Installation:

# Create the Docker CLI plugins directory if it doesn't exist

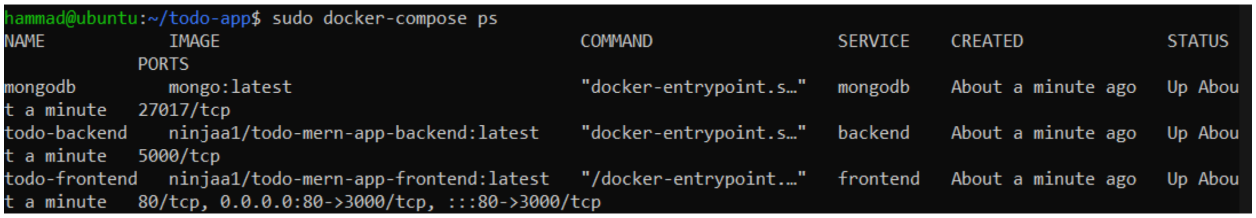
```
mkdir -p ~/.docker/cli-plugins/


# Download the Docker Compose CLI plugin

sudo curl -SL
https://github.com/docker/compose/releases/download/v2.24.6/docker-compose
-linux-x86_64 -o /usr/local/bin/docker-compose


# Apply executable permissions

sudo chmod +x /usr/local/bin/docker-compose


# Verify the installation

docker-compose --version
```


```
hammad@ubuntu:~/todo-app$ sudo docker-compose up -d
sudo: docker-compose: command not found
hammad@ubuntu:~/todo-app$ mkdir -p ~/.docker/cli-plugins/
hammad@ubuntu:~/todo-app$ sudo curl -SL https://github.com/docker/compose/releases/download/v2.24.6/docker-compose-linux
-x86_64 -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:--  0:00:01 --:--:--     0
100 58.5M  100 58.5M    0     0   497k      0  0:02:00  0:02:00 --:--:--  318k
hammad@ubuntu:~/todo-app$ sudo chmod +x /usr/local/bin/docker-compose
hammad@ubuntu:~/todo-app$ docker-compose --version
Docker Compose version v2.24.6
hammad@ubuntu:~/todo-app$
```

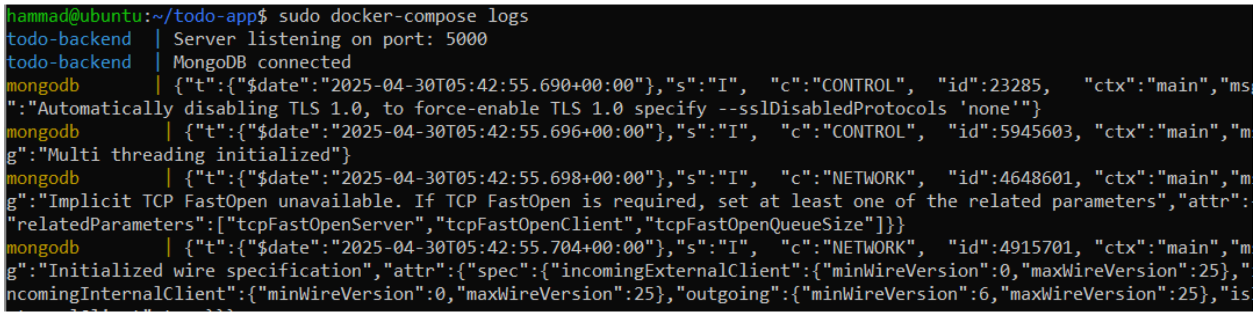
Docker Compose run:


```
ammad@ubuntu:~/todo-app$ sudo docker rm -f todo-backend todo-frontend
odo-backend
odo-frontend
ammad@ubuntu:~/todo-app$ sudo docker compose up -d
ARN[0000] volume "mongodb-data" already exists but was not created by Docker Compose. Use `external: true` to use an
sting volume
+] Running 3/3
✔Container todo-backend   Started                                                                          2.1
✔Container todo-frontend  Started                                                                          4.2
✔Container mongodb        Started                                                                          0.4
ammad@ubuntu:~/todo-app$
```

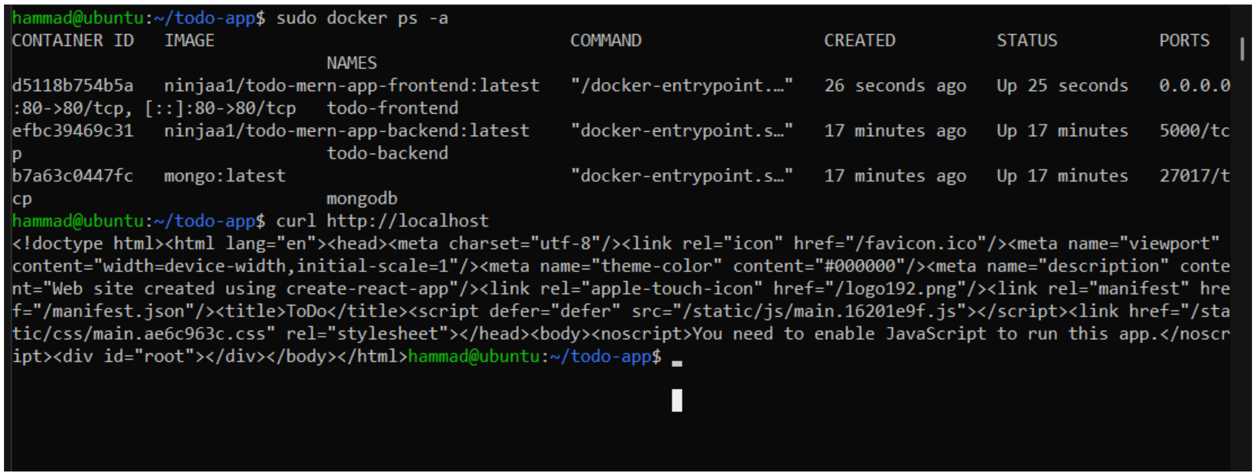
Running Containers:


```
hammad@ubuntu:~/todo-app$ sudo docker-compose ps
NAME            IMAGE                                 COMMAND                  SERVICE    CREATED              STATUS
          PORTS
mongodb         mongo:latest                          "docker-entrypoint.s…"   mongodb    About a minute ago   Up Abou
t a minute    27017/tcp
todo-backend    ninjaa1/todo-mern-app-backend:latest  "docker-entrypoint.s…"   backend    About a minute ago   Up Abou
t a minute    5000/tcp
todo-frontend   ninjaa1/todo-mern-app-frontend:latest "/docker-entrypoint.…"   frontend   About a minute ago   Up Abou
t a minute    80/tcp, 0.0.0.0:80->3000/tcp, :::80->3000/tcp
```

Docker logs:


```
hammad@ubuntu:~/todo-app$ sudo docker-compose logs
todo-backend   | Server listening on port: 5000
todo-backend   | MongoDB connected
mongodb        | {"t":{"$date":"2025-04-30T05:42:55.690+00:00"},"s":"I",  "c":"CONTROL",  "id":23285,   "ctx":"main","ms
":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
mongodb         | {"t":{"$date":"2025-04-30T05:42:55.696+00:00"},"s":"I",  "c":"CONTROL",  "id":5945603, "ctx":"main","m
g":"Multi threading initialized"}
mongodb         | {"t":{"$date":"2025-04-30T05:42:55.698+00:00"},"s":"I",  "c":"NETWORK",  "id":4648601, "ctx":"main","m
g":"Implicit TCP FastOpen unavailable. If TCP FastOpen is required, set at least one of the related parameters","attr":
"relatedParameters":["tcpFastOpenServer","tcpFastOpenClient","tcpFastOpenQueueSize"]}}
mongodb         | {"t":{"$date":"2025-04-30T05:42:55.704+00:00"},"s":"I",  "c":"NETWORK",  "id":4915701, "ctx":"main","m
g":"Initialized wire specification","attr":{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":25},"
ncomingInternalClient":{"minWireVersion":0,"maxWireVersion":25},"outgoing":{"minWireVersion":6,"maxWireVersion":25},"is
```

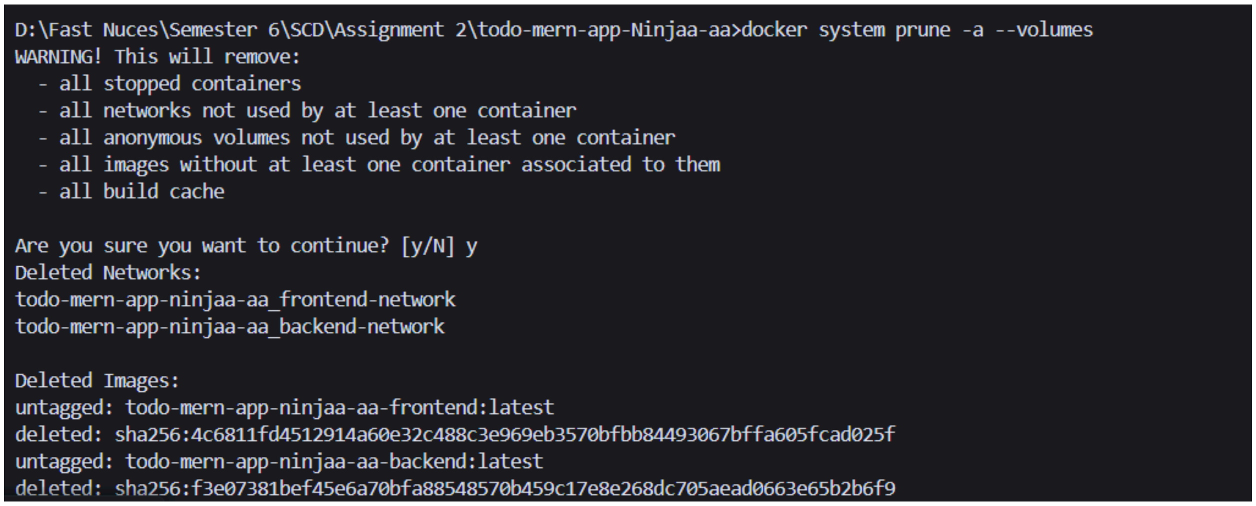
Getting frontend response:



```
hammad@ubuntu:~/todo-app$ sudo docker ps -a
CONTAINER ID   IMAGE                                 COMMAND                  CREATED          STATUS          PORTS
                              NAMES
d5118b754b5a   ninjaa1/todo-mern-app-frontend:latest   "/docker-entrypoint.…"   26 seconds ago   Up 25 seconds   0.0.0.0
:80->80/tcp, [::]:80->80/tcp   todo-frontend
efbc39469c31   ninjaa1/todo-mern-app-backend:latest    "docker-entrypoint.s…"   17 minutes ago   Up 17 minutes   5000/tc
p                              todo-backend
b7a63c0447fc   mongo:latest                            "docker-entrypoint.s…"   17 minutes ago   Up 17 minutes   27017/t
cp                             mongodb
hammad@ubuntu:~/todo-app$ curl http://localhost
<!doctype html><html lang="en"><head><meta charset="utf-8"/><link rel="icon" href="/favicon.ico"/><meta name="viewport"
content="width=device-width,initial-scale=1"/><meta name="theme-color" content="#000000"/><meta name="description" conte
nt="Web site created using create-react-app"/><link rel="apple-touch-icon" href="/logo192.png"/><link rel="manifest" hre
f="/manifest.json"/><title>ToDo</title><script defer="defer" src="/static/js/main.16201e9f.js"></script><link href="/sta
tic/css/main.ae6c963c.css" rel="stylesheet"></head><body><noscript>You need to enable JavaScript to run this app.</noscr
ipt><div id="root"></div></body></html>hammad@ubuntu:~/todo-app$
```

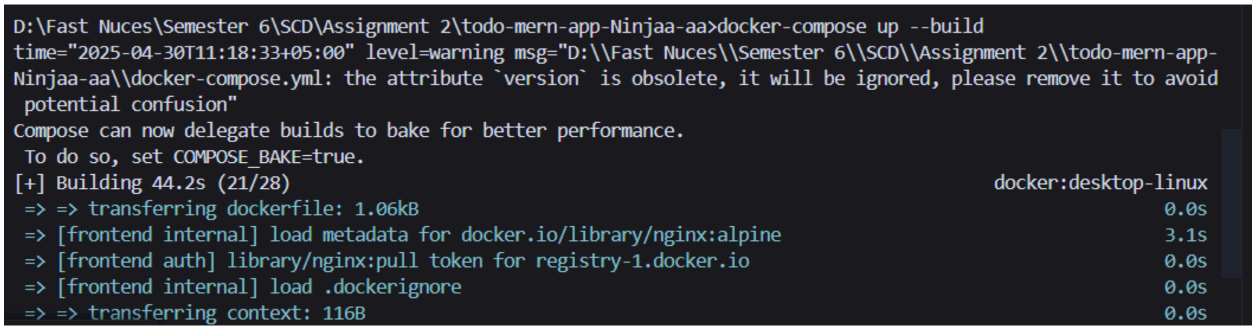
Part 7: Update Project Repo to include Docker Compose

Removing previous stages:



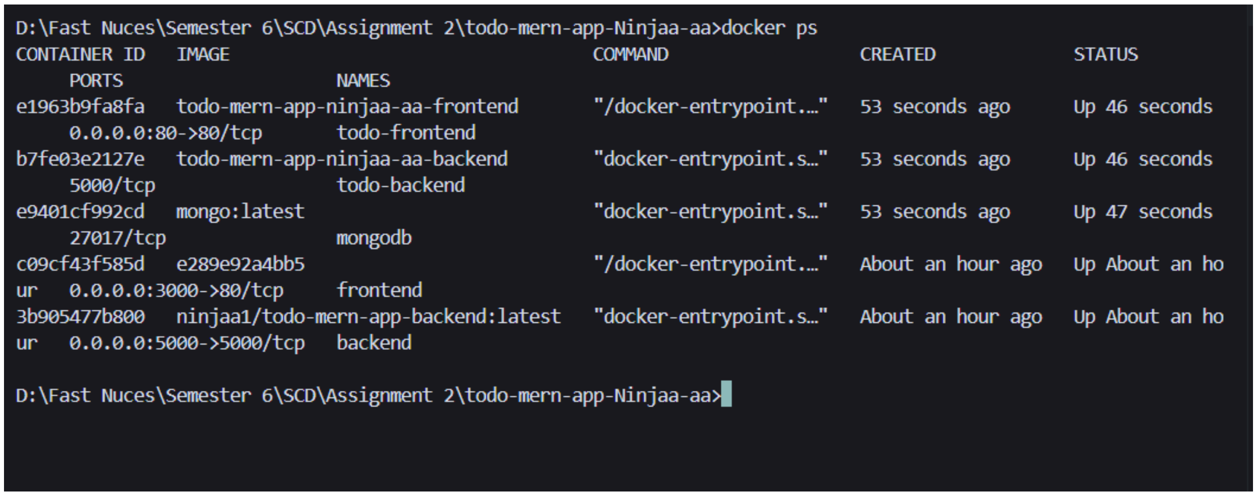
```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker system prune -a --volumes
WARNING! This will remove:
  - all stopped containers
  - all networks not used by at least one container
  - all anonymous volumes not used by at least one container
  - all images without at least one container associated to them
  - all build cache

Are you sure you want to continue? [y/N] y
Deleted Networks:
todo-mern-app-ninjaa-aa_frontend-network
todo-mern-app-ninjaa-aa_backend-network

Deleted Images:
untagged: todo-mern-app-ninjaa-aa-frontend:latest
deleted: sha256:4c6811fd4512914a60e32c488c3e969eb3570bfbb84493067bffa605fcad025f
untagged: todo-mern-app-ninjaa-aa-backend:latest
deleted: sha256:f3e07381bef45e6a70bfa88548570b459c17e8e268dc705aead0663e65b2b6f9
```

Run Docker Compose:



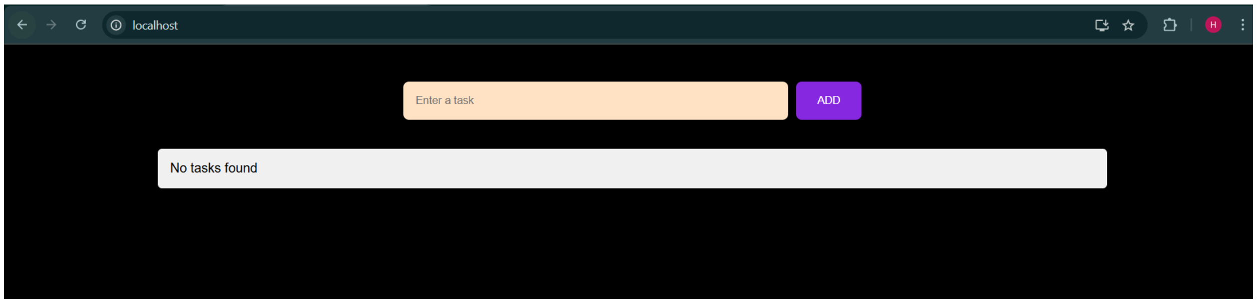
```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker-compose up --build
time="2025-04-30T11:18:33+05:00" level=warning msg="D:\\Fast Nuces\\Semester 6\\SCD\\Assignment 2\\todo-mern-app-
Ninjaa-aa\\docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid
 potential confusion"
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 44.2s (21/28)                                                          docker:desktop-linux
 => => transferring dockerfile: 1.06kB                                                              0.0s
 => [frontend internal] load metadata for docker.io/library/nginx:alpine                            3.1s
 => [frontend auth] library/nginx:pull token for registry-1.docker.io                               0.0s
 => [frontend internal] load .dockerignore                                                          0.0s
 => => transferring context: 116B                                                                   0.0s
```

Container Running:

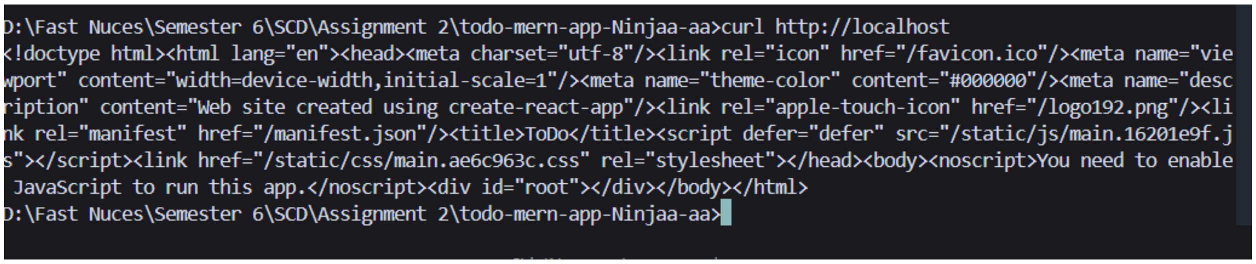


```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>docker ps
CONTAINER ID   IMAGE                                  COMMAND                  CREATED             STATUS
     PORTS                  NAMES
e1963b9fa8fa   todo-mern-app-ninjaa-aa-frontend       "/docker-entrypoint.…"   53 seconds ago      Up 46 seconds
     0.0.0.0:80->80/tcp     todo-frontend
b7fe03e2127e   todo-mern-app-ninjaa-aa-backend        "docker-entrypoint.s…"   53 seconds ago      Up 46 seconds
     5000/tcp               todo-backend
e9401cf992cd   mongo:latest                           "docker-entrypoint.s…"   53 seconds ago      Up 47 seconds
     27017/tcp              mongodb
c09cf43f585d   e289e92a4bb5                           "/docker-entrypoint.…"   About an hour ago   Up About an ho
ur   0.0.0.0:3000->80/tcp   frontend
3b905477b800   ninjaa1/todo-mern-app-backend:latest   "docker-entrypoint.s…"   About an hour ago   Up About an ho
ur   0.0.0.0:5000->5000/tcp   backend

D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

Frontend Running:



No tasks found



```
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>curl http://localhost
<!doctype html><html lang="en"><head><meta charset="utf-8"/><link rel="icon" href="/favicon.ico"/><meta name="vie
wport" content="width=device-width,initial-scale=1"/><meta name="theme-color" content="#000000"/><meta name="desc
ription" content="Web site created using create-react-app"/><link rel="apple-touch-icon" href="/logo192.png"/><li
nk rel="manifest" href="/manifest.json"/><title>ToDo</title><script defer="defer" src="/static/js/main.16201e9f.j
s"></script><link href="/static/css/main.ae6c963c.css" rel="stylesheet"></head><body><noscript>You need to enable
 JavaScript to run this app.</noscript><div id="root"></div></body></html>
D:\Fast Nuces\Semester 6\SCD\Assignment 2\todo-mern-app-Ninjaa-aa>
```

Commands:

```
# 1. Create docker-compose.yml at the root of your project

# 2. Clean Slate: Remove unused Docker resources
docker system prune -a --volumes

# 3. Run Docker Compose to build and start services
docker-compose up --build

# 4. Verify running services
docker ps

# 5. Test the app in the browser at http://localhost

# 6. Capture Screenshots of the build, running services, and app

# 7. Commit and Push changes to GitHub
git add docker-compose.yml .env README.md
git commit -m "Add Docker Compose setup for MERN application"
git push origin main
```