

Module 6 Activity 2: Deploying Node.JS application into Docker Using Visual Studio

Brandon Trinkle

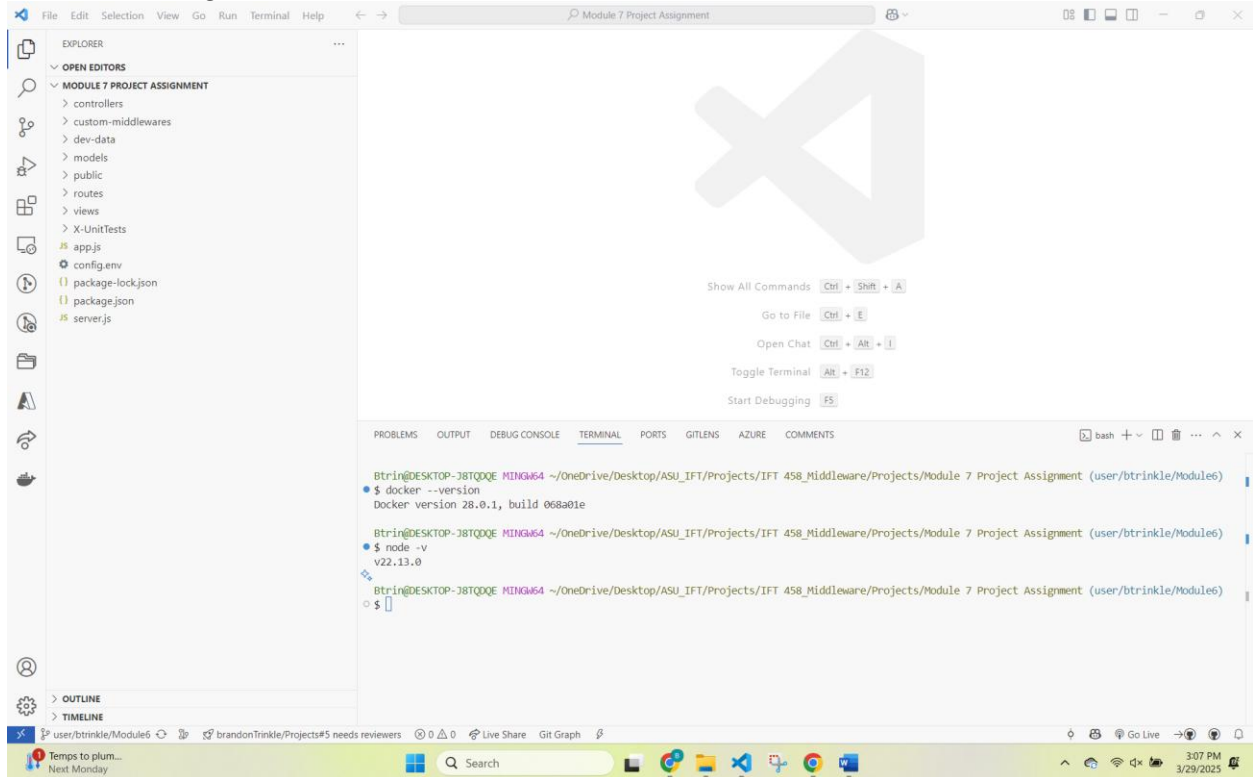
IFT 458: Middleware Programming

Professor Dinesh Sthapit

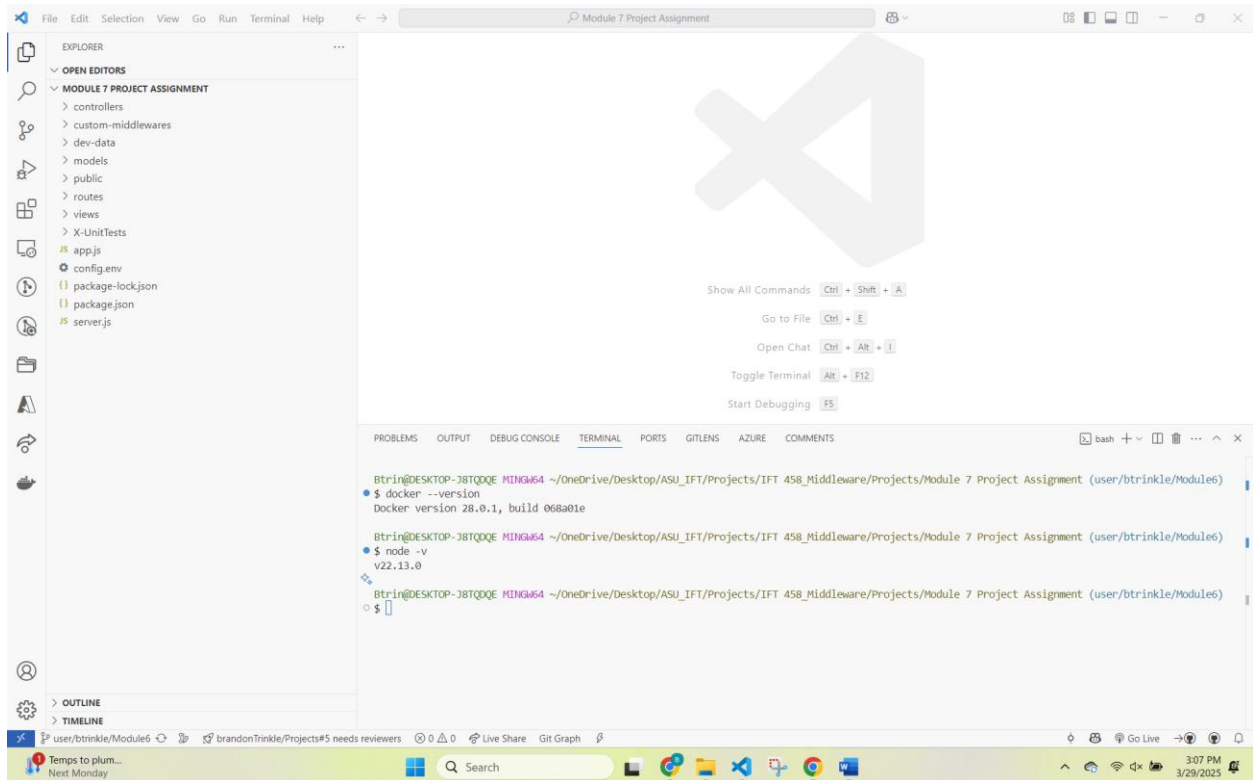
March 29, 2025

Module 6 Activity 2: Deploying Node.JS application into Docker Using Visual Studio

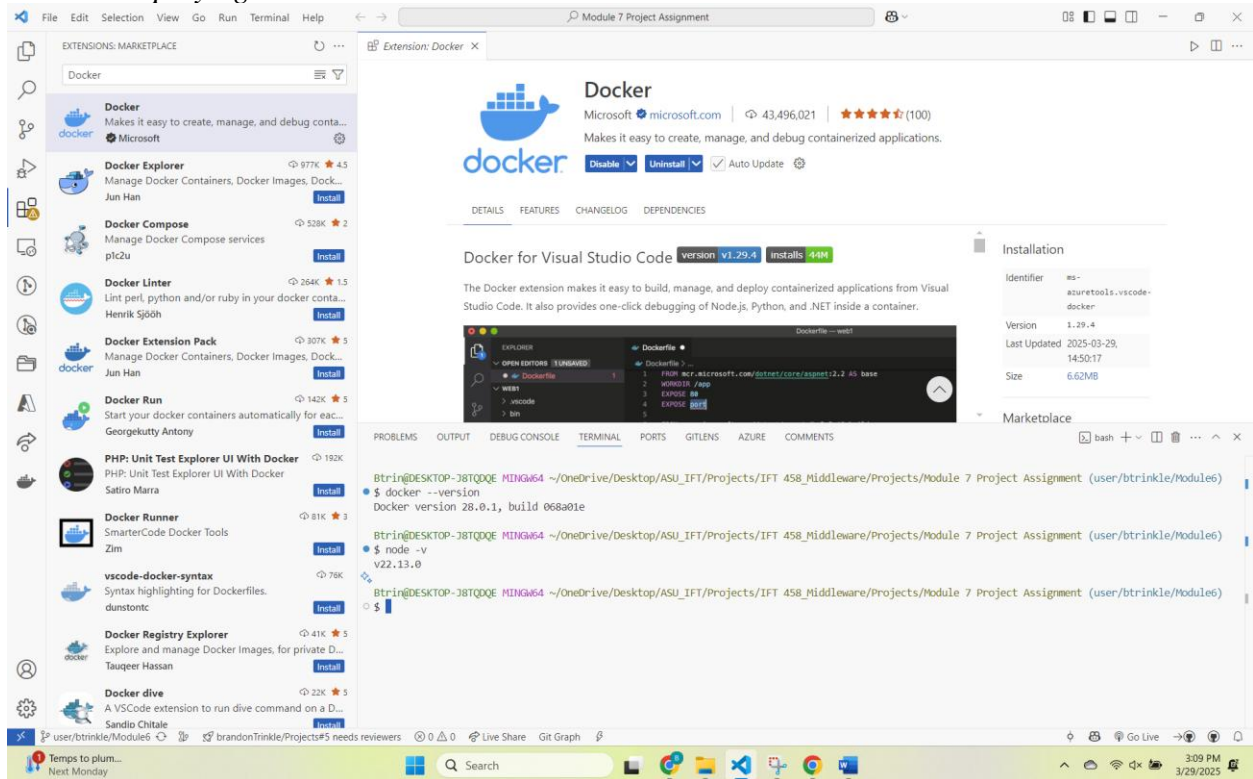
Task 1: Showing Docker version in the terminal.



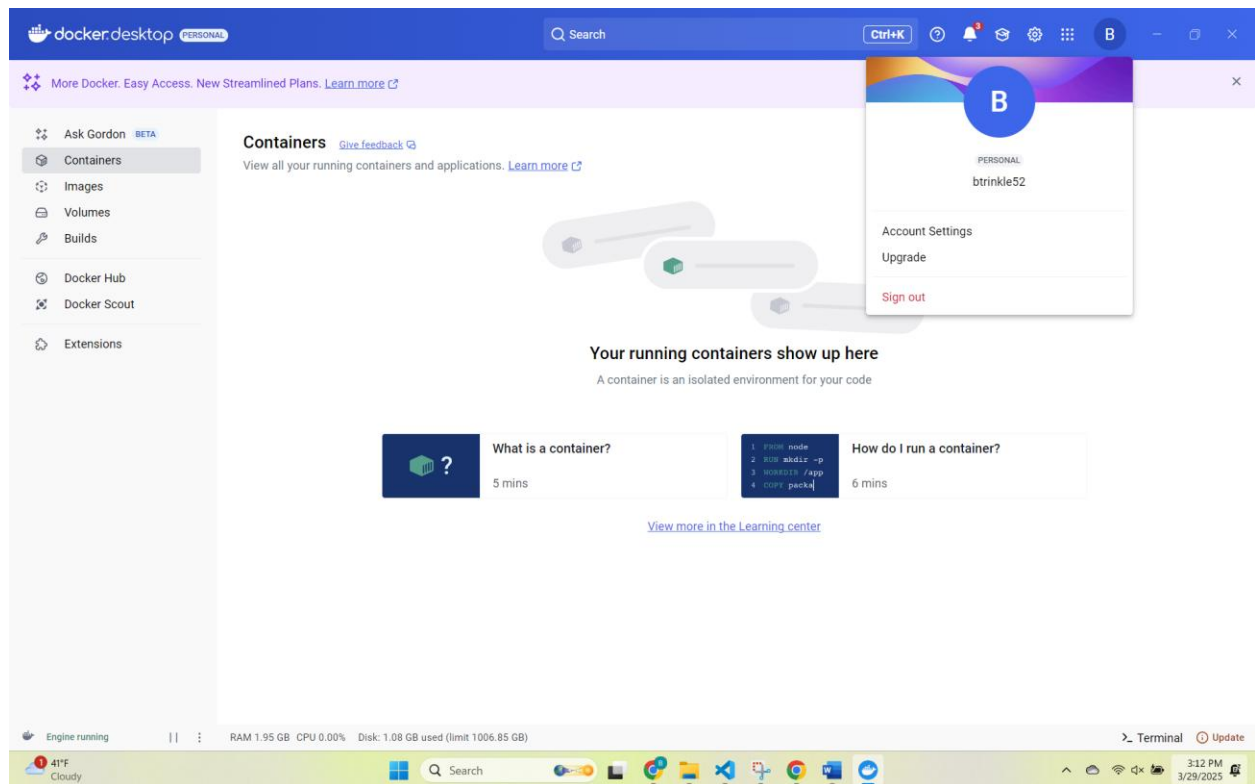
Task 2: Showing Node.js version in the terminal.



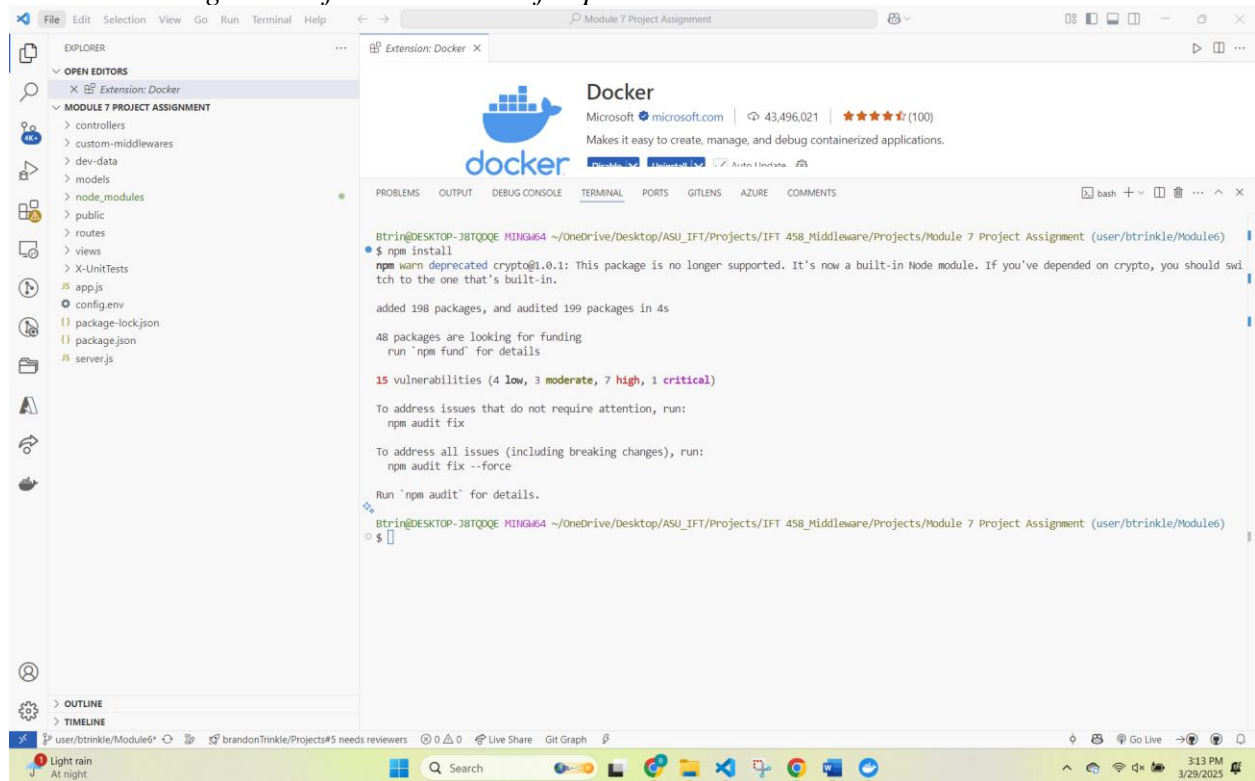
Task 3: Displaying Docker extension in VS Code.



Task 4: Of the running Docker Desktop application.



Task 5: Showing successful installation of dependencies in the terminal.



Task 6: Of the Dockerfile with content in VS Code.

The screenshot shows the VS Code interface with the Explorer on the left, the Dockerfile open in the editor, and the terminal at the bottom. The Dockerfile contains the following instructions:

```

1 # Use the official Node.js runtime as base image :18 is the version of node
2 FROM node:18
3
4
5 # Set the working directory in the image
6 WORKDIR /usr/src/app
7
8
9 # Install app dependencies by copying package.json and package-lock.json
10 COPY package*.json ./
11
12
13 # Install Node.js dependencies
14 RUN npm install
15
16
17 # Bundle app source code
18 COPY . .
19
20
21 # Expose the port the app will run on 4000
22 # as we have configured it on the config.env file
23 EXPOSE 4000
24
25
26 # Command to run the application
27 CMD ["npm", "start"]

```

The terminal output shows the results of running `npm audit`:

```

48 packages are looking for funding
  run `npm fund` for details

15 vulnerabilities (4 low, 3 moderate, 7 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

```

Task 7: Showing the Docker image build process in VS Code.

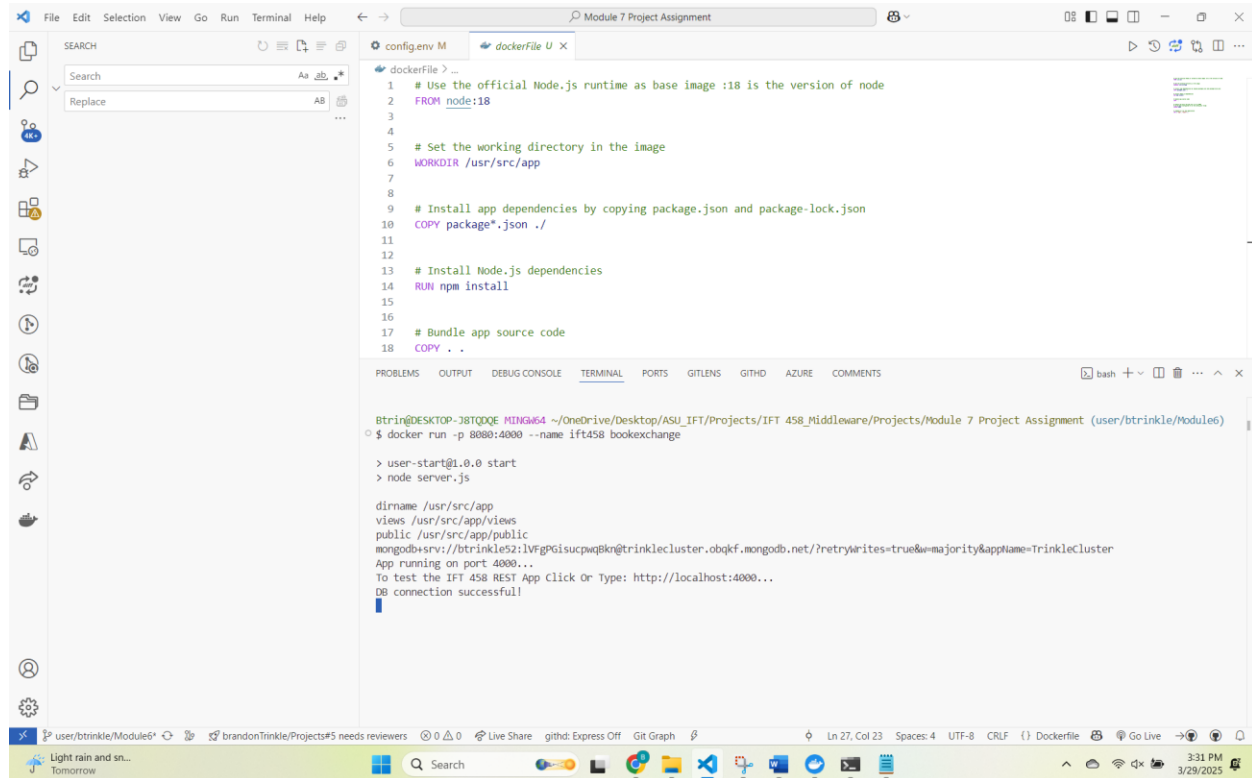
The screenshot shows the VS Code interface with the Dockerfile open in the editor and the terminal at the bottom. The terminal output shows the results of running `docker build -t module7 .`:

```

=> extracting sha256:7cd785773db44407e20a679ce5439222e505475eed5b99f1910eb2cda51729ab 2.7s
=> sha256:5aa9b40886c859f533da11496ce0f8fac54f61a2202d8141b35d58a13f5f286 45.68MB / 45.68MB 38.0s
=> extracting sha256:091eb8249475f42de217265c501e0186f0a3ea7490ef7f51458c30db91fb3cac 0.7s
=> extracting sha256:255774e0027b72d2327719e78dbad5ad8c9cf4466d055e45be7fc149418470bae 3.0s
=> sha256:7202769db681ba7a16091acab6a0741f0f210f4860639f164ab182934e210b6c3 1.25MB / 1.25MB 42.2s
=> sha256:2e36833b68b10fbec7cb863e3bc418f9f064cd930edc071df9f33e8bd4d0a8883 4478 / 4478 41.9s
=> extracting sha256:353e14e5cca7664fba714a7da288001d90427c789494847ac773f5cc08199451 6.6s
=> extracting sha256:d28ef26f74e7b12fddcf44897b6afca5a8f691567ec2a743a7a624921c81ef1 0.0s
=> extracting sha256:5aa9b40886c859f533da11496ce0f8fac54f61a2202d8141b35d58a13f5f286 2.2s
=> extracting sha256:7202769db681ba7a16091acab6a0741f0f210f4860639f164ab182934e210b6c3 0.0s
=> extracting sha256:2e36833b68b10fbec7cb863e3bc418f9f064cd930edc071df9f33e8bd4d0a8883 0.0s
=> [internal] load build context 0.3s
=> transferring context: 726.05kB 0.3s
=> [2/5] WORKDIR /usr/src/app 0.4s
=> [3/5] COPY package*.json ./ 0.0s
=> [4/5] RUN npm install 14.9s
=> [5/5] COPY . . 0.0s
=> exporting to image 0.6s
=> exporting layers 0.6s
=> writing image sha256:cb2400a09788fe253d358d8273f8ef2a22f81315e52369985fb11c9d713a561 0.0s
=> naming to docker.io/library/module7:latest 0.0s
Terminal will be reused by tasks, press any key to close it.

```

Task 8: Showing Docker container configuration



The screenshot shows the Visual Studio Code interface with a Dockerfile open in the editor and its execution output in the terminal.

Dockerfile Content:

```

1 # Use the official Node.js runtime as base image :18 is the version of node
2 FROM node:18
3
4
5 # Set the working directory in the image
6 WORKDIR /usr/src/app
7
8
9 # Install app dependencies by copying package.json and package-lock.json
10 COPY package*.json ./
11
12
13 # Install Node.js dependencies
14 RUN npm install
15
16
17 # Bundle app source code
18 COPY . .

```

Terminal Output:

```

Btrink@DESKTOP-38TQDQ6 MINGW64 ~/OneDrive/Desktop/ASU_IFT/Projects/IFT_458_Middleware/Projects/Module 7 Project Assignment (user/btrinkle/Module6)
$ docker run -p 8080:4000 --name ift458 bookexchange

> user-start@1.0.0 start
> node server.js

dirname /usr/src/app
views /usr/src/app/views
public /usr/src/app/public
mongodb+srv://btrinkle52:1VfgPgisucpwqBkn@trinklecluster.obqkf.mongodb.net/?retryWrites=true&majority=&appName=TrinkleCluster
App running on port 4000...
To test the IFT 458 REST App Click Or Type: http://localhost:4000...
DB connection successful!

```

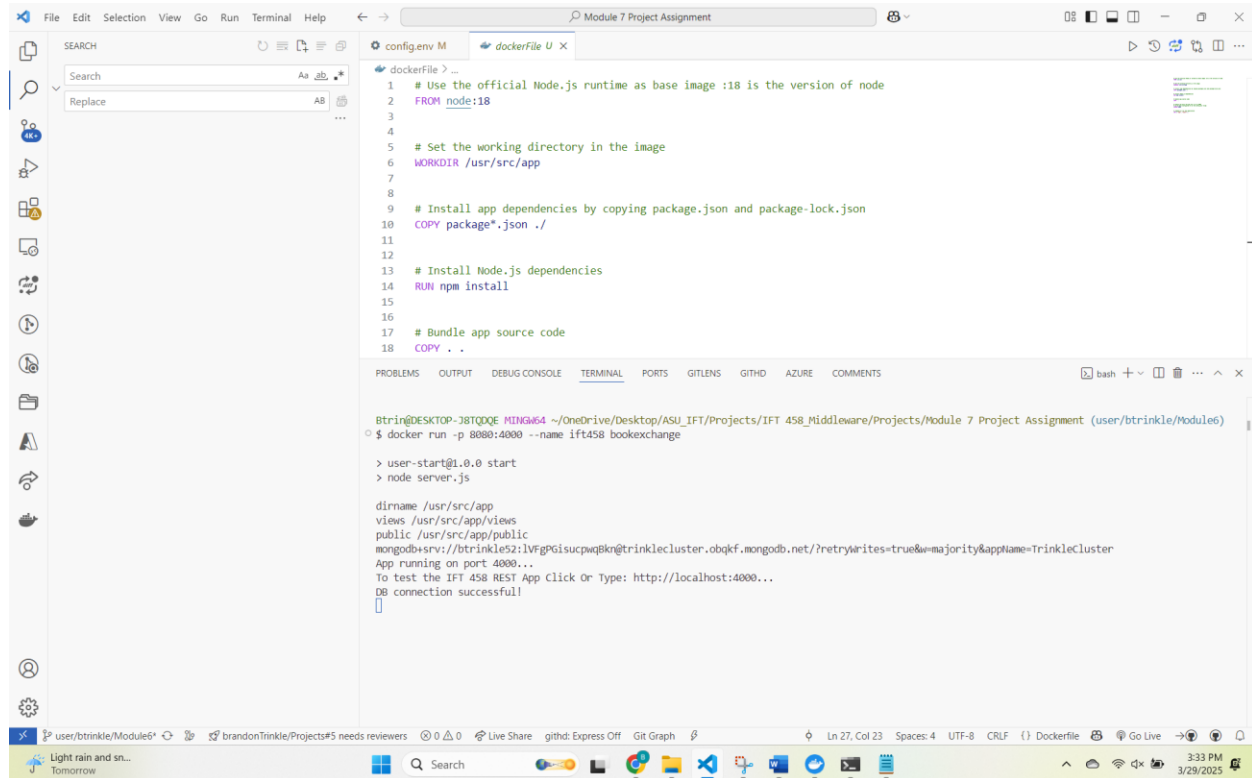
Task 9: Showing the running container in Docker Desktop.

The screenshot shows the Docker Desktop application window. The top bar is blue with the Docker logo and 'PERSONAL' label. A search bar and 'Ctrl+K' button are on the right. Below the top bar is a purple banner with the text 'More Docker. Easy Access. New Streamlined Plans. Learn more'. The left sidebar contains navigation links: 'Containers' (selected), 'Images', 'Volumes', 'Builds', 'Docker Hub', 'Docker Scout', and 'Extensions'. The main area is titled 'Containers' and shows a table of running containers. One container, 'ift458', is listed with ID 'f6d7fff5f888', image 'bookexchange', and port '8080:4000'. Below the table, there are 'Walkthroughs' for 'How do I run a container?' and 'Multi-container applications'. The bottom status bar shows 'Engine running', 'RAM 1.04 GB / CPU 0.08% / Disk: 2.23 GB used (limit 1006.85 GB)', and a 'Terminal' button. The Windows taskbar at the bottom shows the time as 3:23 PM on 3/29/2025.

Task 10: Of the running Node.js application in the browser.

The screenshot shows a web browser window with the address 'localhost:8080'. The page has a light blue background and contains the following text: 'Batch of IFT 458 Fall 2023', 'Server: 3/29/2025, 3:32:37 PM', 'Your IP address is: 98.5.196.243', 'Student Information', 'Student Image: [Avatar]', 'Student Name: John Doe', 'Student Email: johndoe@example.com', 'Student ID: 123456789', 'Course #: CS101', 'Welcome to Book Exchange!', 'Book Exchange', 'Register', 'Login', 'Available Books:', and 'Book Exchange Platform © IFT 510 - Fall 2023'. The browser's address bar shows 'localhost:8080' and the Windows taskbar at the bottom shows the time as 3:32 PM on 3/29/2025.

Task 11: Of the terminal after executing the Docker run command.



The screenshot shows the Visual Studio Code interface with a Dockerfile open in the editor and its execution output in the terminal. The Dockerfile is named 'dockerfile U' and contains the following instructions:

```
1 # Use the official Node.js runtime as base image :18 is the version of node
2 FROM node:18
3
4
5 # Set the working directory in the image
6 WORKDIR /usr/src/app
7
8
9 # Install app dependencies by copying package.json and package-lock.json
10 COPY package*.json ./
11
12
13 # Install Node.js dependencies
14 RUN npm install
15
16
17 # Bundle app source code
18 COPY . .
```

The terminal output shows the command being executed and the resulting output:

```
bt@DESKTOP-38TQDQ6 MINGW64 ~/OneDrive/Desktop/ASU_IFT/Projects/IFT_458_Middleware/Projects/Module 7 Project Assignment (user/btrinkle/Module6)
$ docker run -p 8080:4000 --name ift458 bookexchange

> user-start@1.0.0 start
> node server.js

dirname /usr/src/app
views /usr/src/app/views
public /usr/src/app/public
mongodb+srv://btrinkle52:1VfG6isucpwqBkn@trinklecluster.obqkf.mongodb.net/?retryWrites=true&majority=&appName=TrinkleCluster
App running on port 4000...
To test the IFT 458 REST App Click Or Type: http://localhost:4000...
DB connection successful!
```

Task 12: Of the interactive shell session with the running Docker container.

The screenshot shows the Visual Studio Code interface with a Dockerfile open in the editor and a terminal window at the bottom. The Dockerfile contains the following instructions:

```

1 # Use the official Node.js runtime as base image :18 is the version of node
2 FROM node:18
3
4
5 # Set the working directory in the image
6 WORKDIR /usr/src/app
7
8
9 # Install app dependencies by copying package.json and package-lock.json
10 COPY package*.json ./
11
12
13 # Install Node.js dependencies
14 RUN npm install
15
16
17 # Bundle app source code
18 COPY . .

```

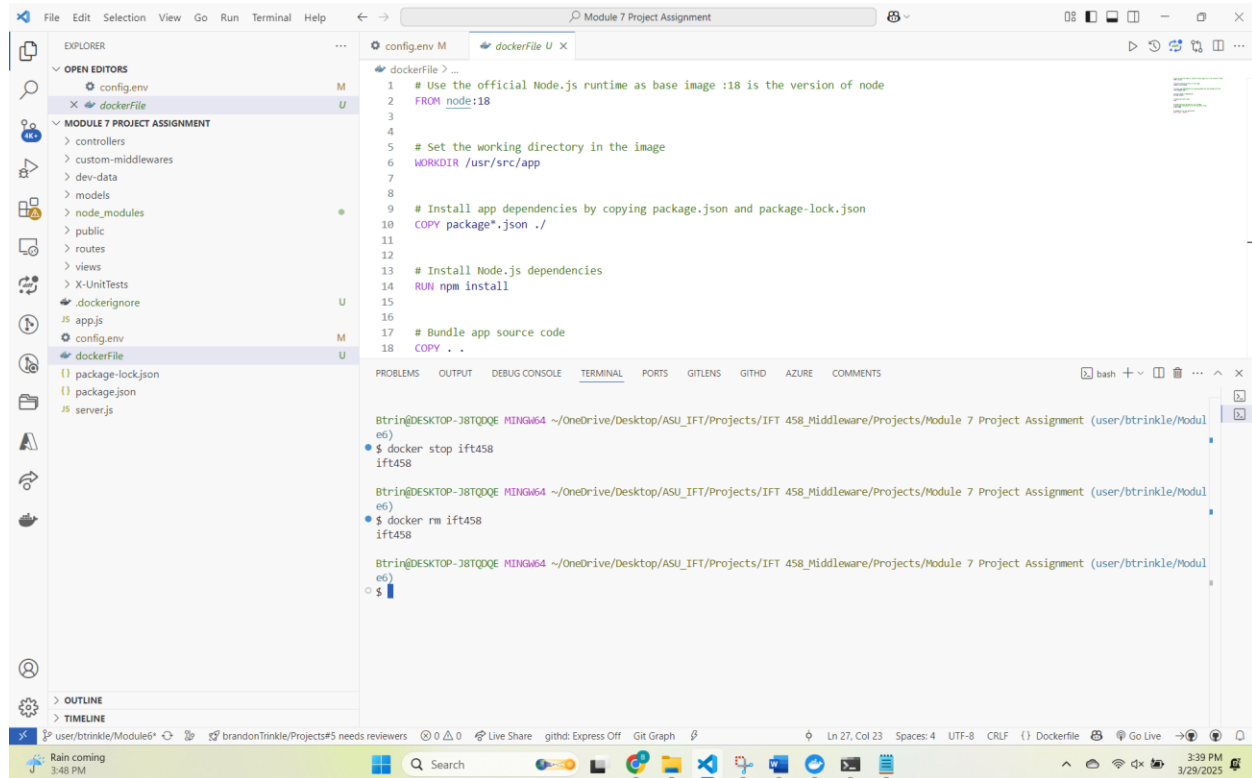
The terminal window shows the command `docker ps` being executed, resulting in the following output:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
b43d6261dc30	bookexchange	"docker-entrypoint.s..."	3 minutes ago	Up 3 minutes	0.0.0.0:8080->4000/tcp	ift458

Task 13: Showing the terminal after stopping the container.

The screenshot shows the Visual Studio Code interface with the same Dockerfile open in the editor and a terminal window at the bottom. The terminal window shows the command `docker stop ift458` being executed, followed by the prompt `$`.

Task 14: Showing the terminal after removing the container.



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left displays the file structure of a project named 'Module 7 Project Assignment'. The file 'dockerFile' is selected. The Editor pane shows the content of 'dockerFile', which is a Dockerfile for a Node.js application. The terminal pane at the bottom shows the execution of two Docker commands: 'docker stop ift458' and 'docker rm ift458', both of which were successful.

```
config.env M
dockerFile U

MODULE 7 PROJECT ASSIGNMENT
> controllers
> custom-middlewares
> dev-data
> models
> node_modules
> public
> routes
> views
> X-UnitTests
> dockerignore
> app.js
> config.env
> dockerFile
> package-lock.json
> package.json
> server.js

dockerFile > ...
1 # Use the official Node.js runtime as base image :18 is the version of node
2 FROM node:18
3
4
5 # Set the working directory in the image
6 WORKDIR /usr/src/app
7
8
9 # Install app dependencies by copying package.json and package-lock.json
10 COPY package*.json ./
11
12
13 # Install Node.js dependencies
14 RUN npm install
15
16
17 # Bundle app source code
18 COPY . .

BtrIn@DESKTOP-38TQQQE MINGW64 ~/OneDrive/Desktop/ASU_IFT/Projects/IFT 458_Middleware/Projects/Module 7 Project Assignment (user/btrinkle/Modul
e6)
$ docker stop ift458
ift458

BtrIn@DESKTOP-38TQQQE MINGW64 ~/OneDrive/Desktop/ASU_IFT/Projects/IFT 458_Middleware/Projects/Module 7 Project Assignment (user/btrinkle/Modul
e6)
$ docker rm ift458
ift458

BtrIn@DESKTOP-38TQQQE MINGW64 ~/OneDrive/Desktop/ASU_IFT/Projects/IFT 458_Middleware/Projects/Module 7 Project Assignment (user/btrinkle/Modul
e6)
$
```