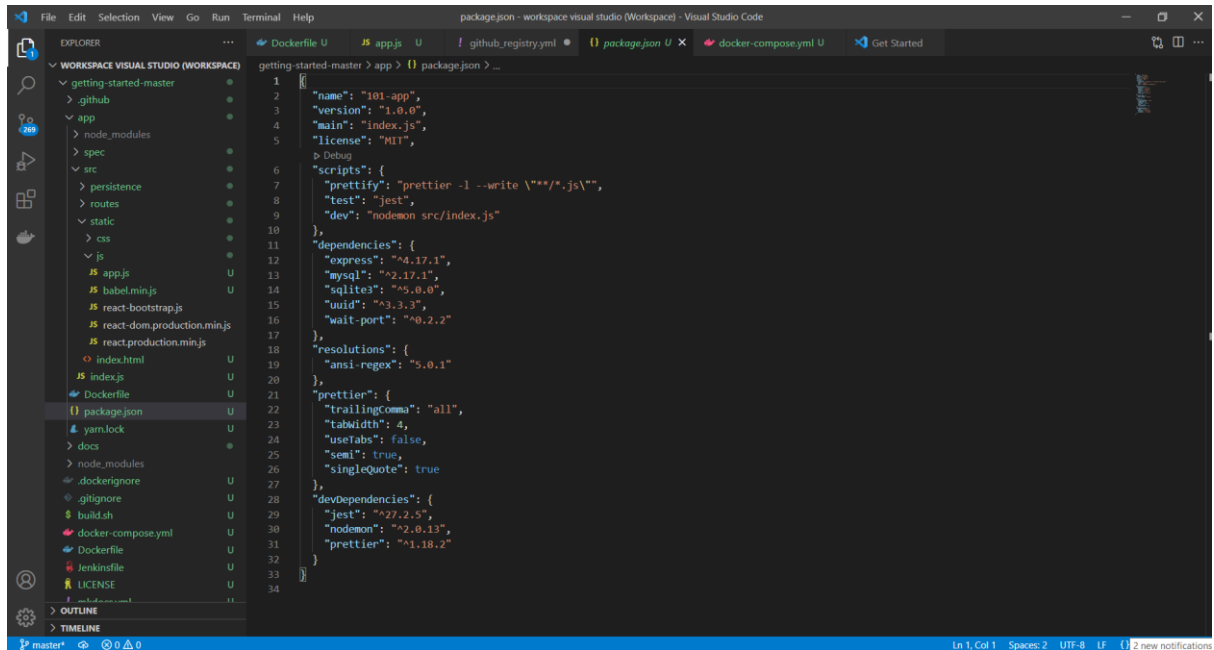


Downloaded a sample docker application from git and working with it in Visual studio code.

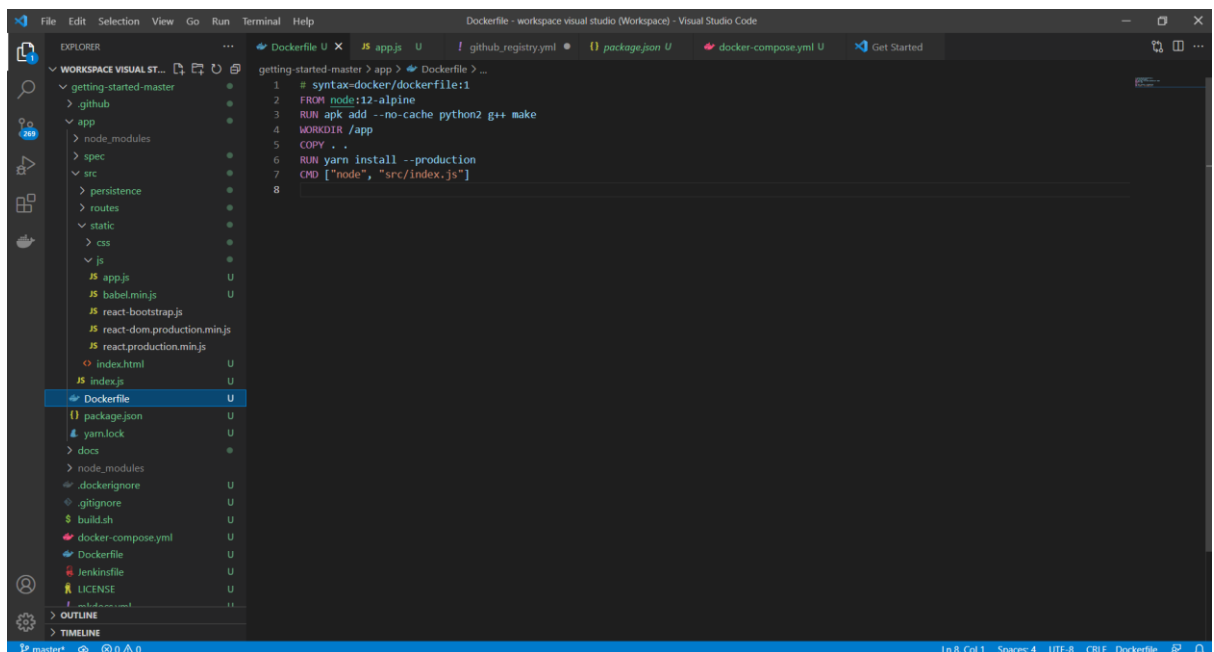
## Pckage.json file



The screenshot shows the Visual Studio Code interface with the `package.json` file open in the editor. The Explorer sidebar on the left shows the project structure, including `getting-started-master`, `github`, `app`, `node_modules`, `spec`, `src`, `persistence`, `routes`, `static`, `css`, `js`, `app.js`, `babel.min.js`, `react-bootstrap.js`, `react-dom.production.min.js`, `react.production.min.js`, `index.html`, `index.js`, `Dockerfile`, `package.json`, `yarn.lock`, `docs`, `node_modules`, `.dockerignore`, `.gitignore`, `build.sh`, `docker-compose.yml`, `Dockerfile`, `Jenkinsfile`, `LICENSE`, `OUTLINE`, and `TIMELINE`. The `package.json` file is selected in the Explorer. The editor shows the following content:

```
1 {
2   "name": "101-app",
3   "version": "1.0.0",
4   "main": "index.js",
5   "license": "MIT",
6   "scripts": {
7     "prettify": "prettier -l --write \"**/*.js\"",
8     "test": "jest",
9     "dev": "nodemon src/index.js"
10  },
11  "dependencies": {
12    "express": "^4.17.1",
13    "mysql": "^2.17.1",
14    "sqlite3": "^5.0.0",
15    "uuid": "^3.3.3",
16    "wait-port": "^0.2.2"
17  },
18  "resolutions": {
19    "ansi-regex": "5.0.1"
20  },
21  "prettier": {
22    "trailingComma": "all",
23    "tabWidth": 4,
24    "useTabs": false,
25    "semi": true,
26    "singleQuote": true
27  },
28  "devDependencies": {
29    "jest": "^27.2.5",
30    "nodemon": "^2.0.13",
31    "prettier": "^1.18.2"
32  }
33 }
```

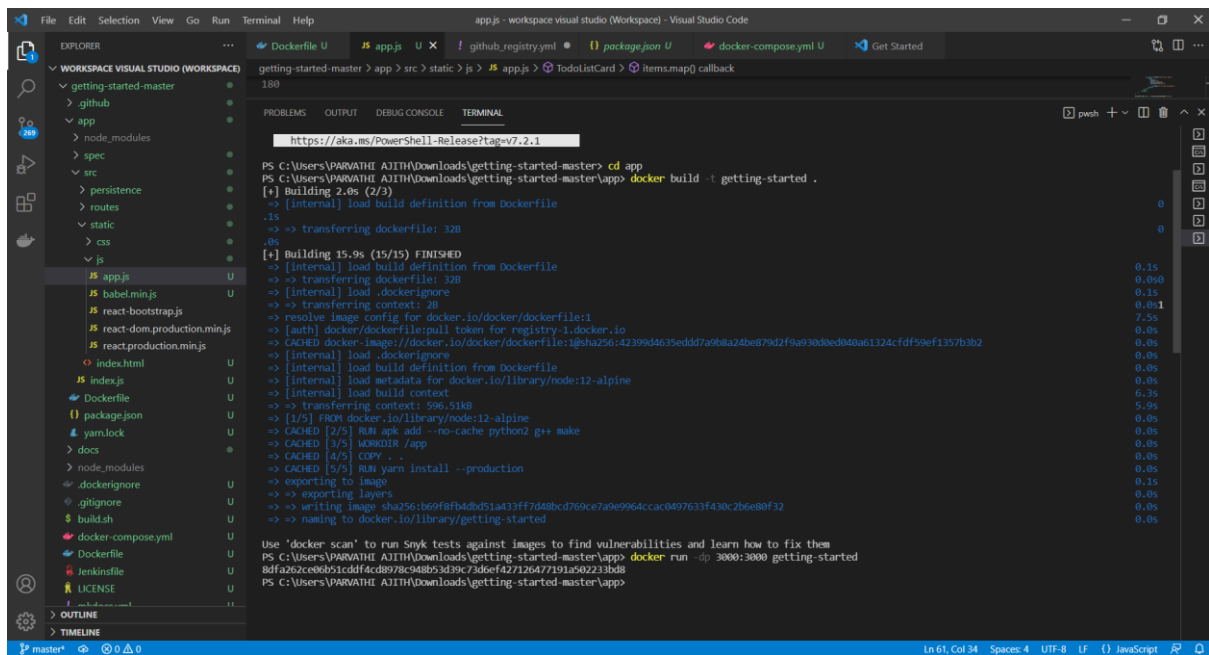
## Dockerfile



The screenshot shows the Visual Studio Code interface with the `Dockerfile` file open in the editor. The Explorer sidebar on the left shows the project structure, including `getting-started-master`, `github`, `app`, `node_modules`, `spec`, `src`, `persistence`, `routes`, `static`, `css`, `js`, `app.js`, `babel.min.js`, `react-bootstrap.js`, `react-dom.production.min.js`, `react.production.min.js`, `index.html`, `index.js`, `Dockerfile`, `package.json`, `yarn.lock`, `docs`, `node_modules`, `.dockerignore`, `.gitignore`, `build.sh`, `docker-compose.yml`, `Dockerfile`, `Jenkinsfile`, `LICENSE`, `OUTLINE`, and `TIMELINE`. The `Dockerfile` file is selected in the Explorer. The editor shows the following content:

```
1 # syntax=docker/dockerfile:1
2 FROM node:12-alpine
3 RUN apk add --no-cache python2 g++ make
4 WORKDIR /app
5 COPY . .
6 RUN yarn install --production
7 CMD ["node", "src/index.js"]
8
```

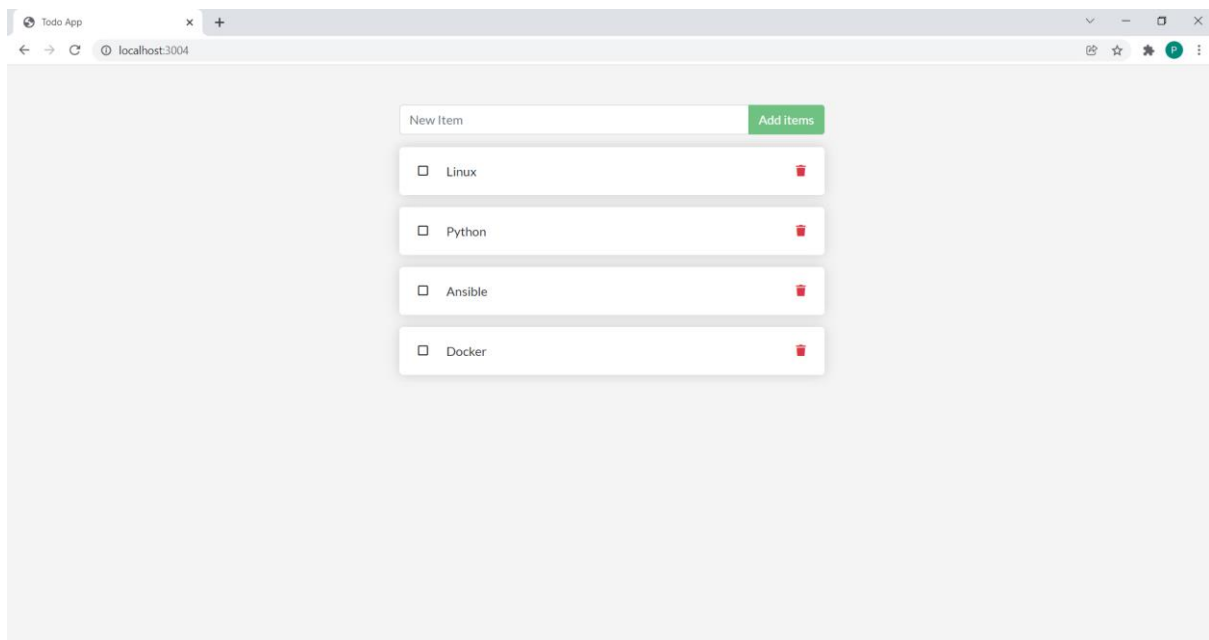
## Building and running the app's container image



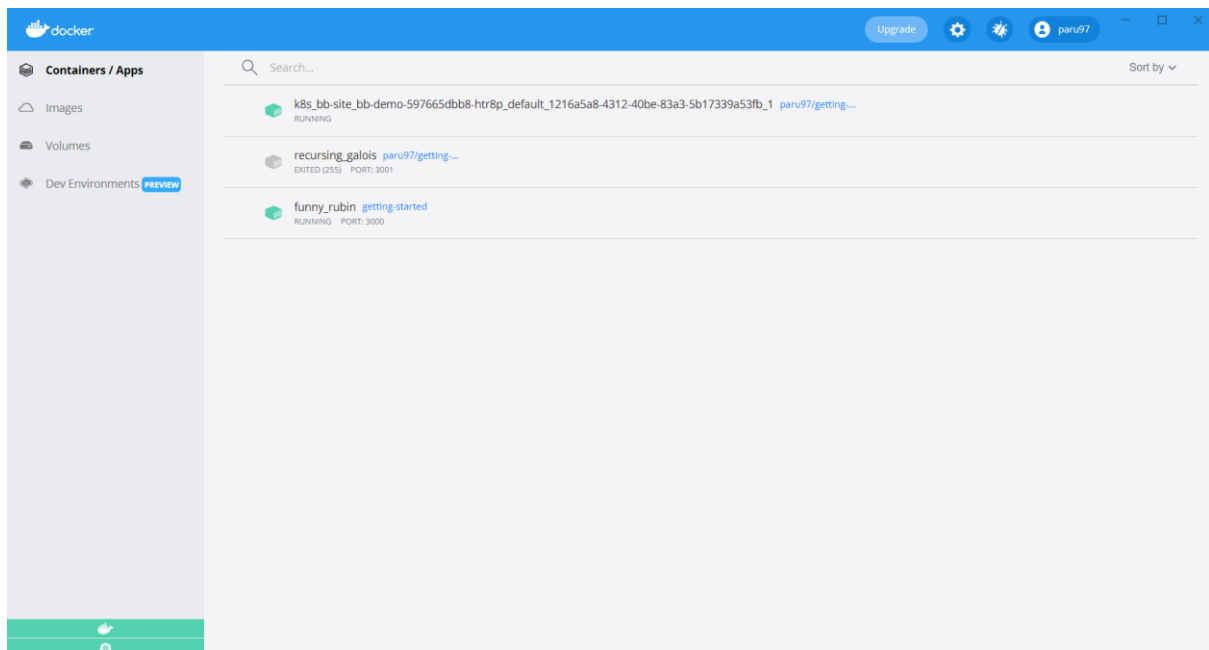
The screenshot shows the Visual Studio Code interface with the Docker build process running in the terminal. The Explorer panel on the left shows the project structure, including files like `app.js`, `babel.min.js`, `react-bootstrap.js`, `react-dom.production.min.js`, `react.production.min.js`, `index.html`, `index.js`, `Dockerfile`, `package.json`, `yarn.lock`, `docs`, `node_modules`, `.dockerignore`, `.gitignore`, `build.sh`, `docker-compose.yml`, `Dockerfile`, `Jenkinsfile`, and `LICENSE`. The terminal output shows the following steps:

```
PS C:\Users\PARVATHI AJITH\Downloads\getting-started-master> cd app
PS C:\Users\PARVATHI AJITH\Downloads\getting-started-master\app> docker build -t getting-started .
[+] Building 2.0s (2/3)
=> [internal] load build definition from Dockerfile
=> [internal] load build context
=> transferring dockerfile: 328
    .0s
[+] Building 15.9s (15/15) FINISHED
=> [internal] load build definition from Dockerfile
=> [internal] load build context
=> transferring dockerfile: 328
    .0s
=> [internal] load .dockerignore
=> transferring context: 28
    .0s
=> resolve image config for docker.io/docker/dockerfile:1
    .0s
=> [auth] docker/dockerfile:pull token for registry-1.docker.io
    .0s
=> CACHED docker-image://docker.io/docker/dockerfile:1@sha256:42399d4635edd7a98a24be875d2f9a030d8ed840a61324cfd95ef1357b3b2
    .0s
=> [internal] load .dockerignore
=> [internal] load build definition from Dockerfile
=> [internal] load metadata for docker.io/library/node:12-alpine
    .0s
=> [internal] load build context
=> transferring context: 596.5kB
    .0s
=> [1/5] FROM docker.io/library/node:12-alpine
    .0s
=> CACHED [2/5] RUN apk add --no-cache python2 g++ make
    .0s
=> CACHED [3/5] WORKDIR /app
    .0s
=> CACHED [4/5] COPY . .
    .0s
=> CACHED [5/5] RUN yarn install --production
    .0s
=> exporting layers
    .0s
=> exporting image sha256:b99f8f6d4d51a433f7d480cd769ce7a9e9964ccac0497633f430c2b6e80f32
    .0s
=> writing image sha256:b99f8f6d4d51a433f7d480cd769ce7a9e9964ccac0497633f430c2b6e80f32
    .0s
=> naming to docker.io/library/getting-started
    .0s
Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS C:\Users\PARVATHI AJITH\Downloads\getting-started-master\app> docker run -dp 3000:3000 getting-started
PS C:\Users\PARVATHI AJITH\Downloads\getting-started-master\app>
```

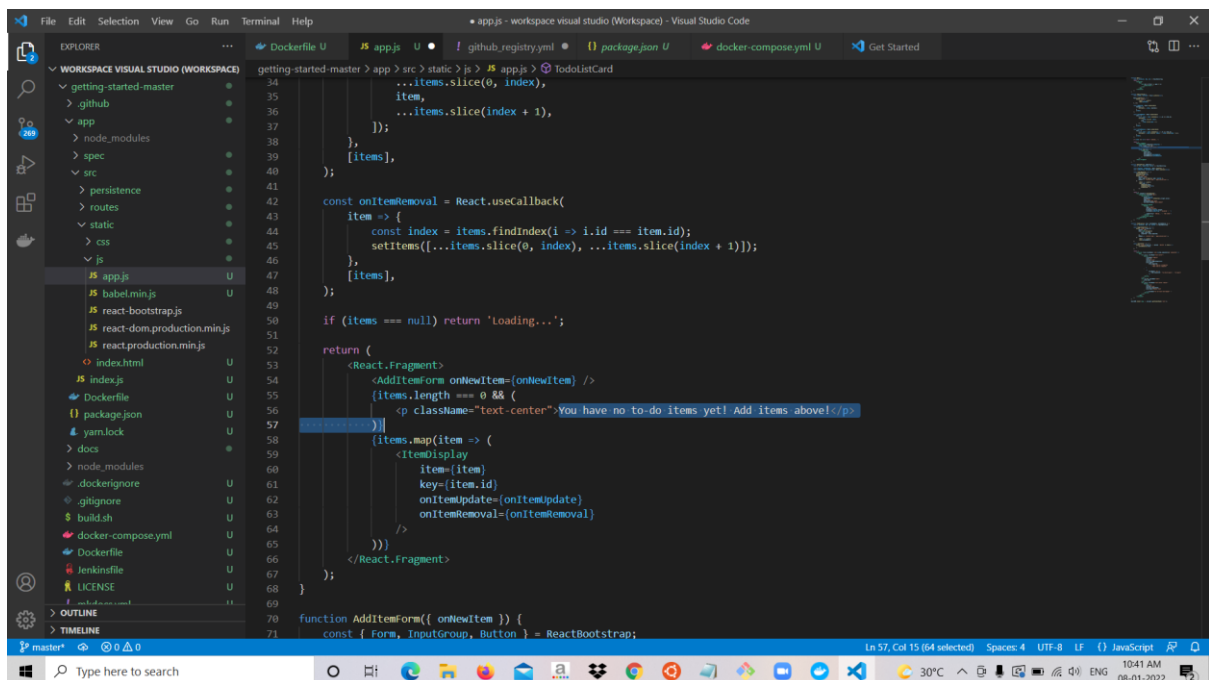
## Opening the To-do app in the browser and adding items to it.



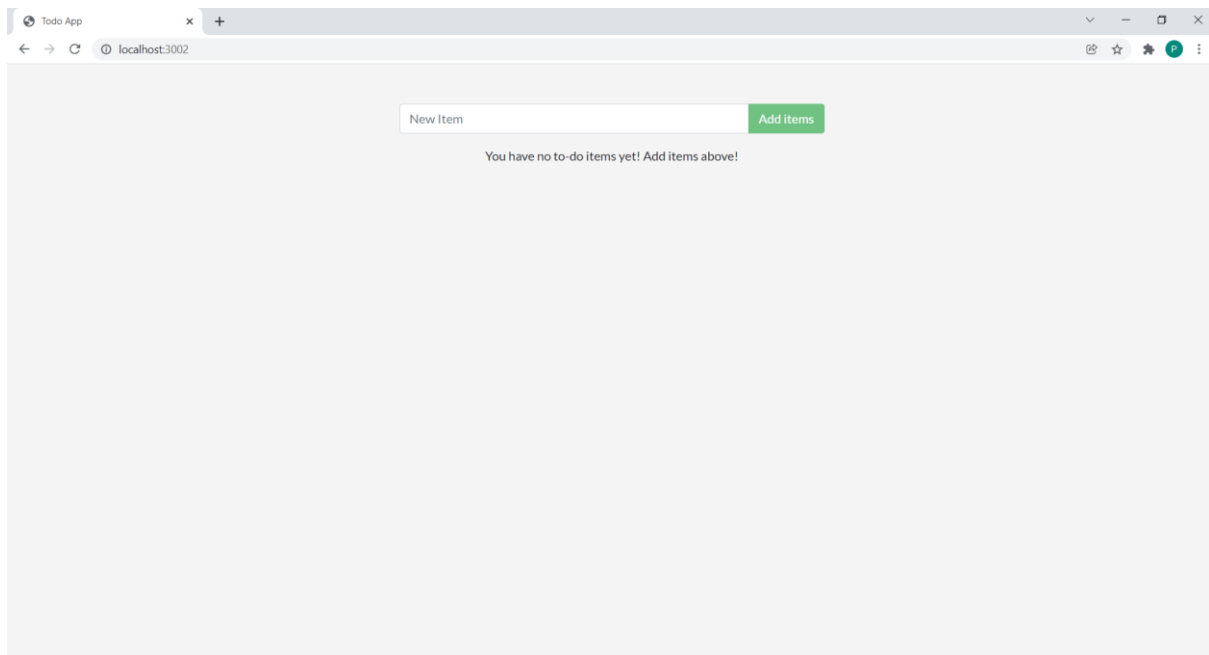
# The container running in the docker desktop



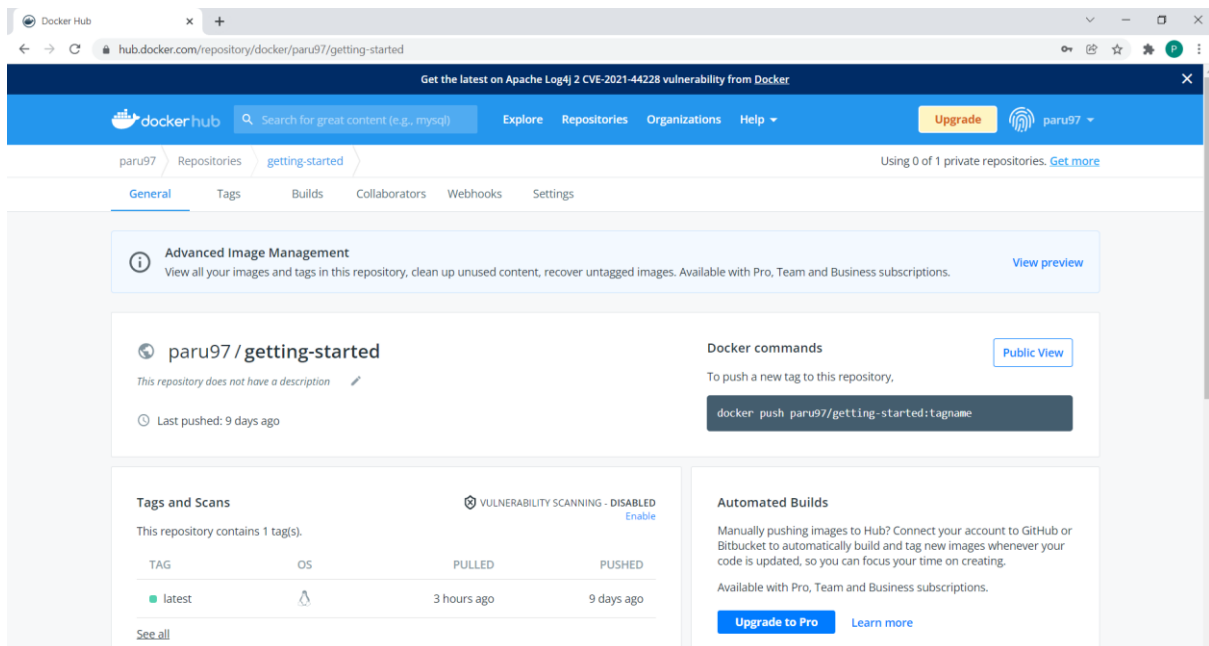
# Making changes to the app



## Viewing those changes after running it again



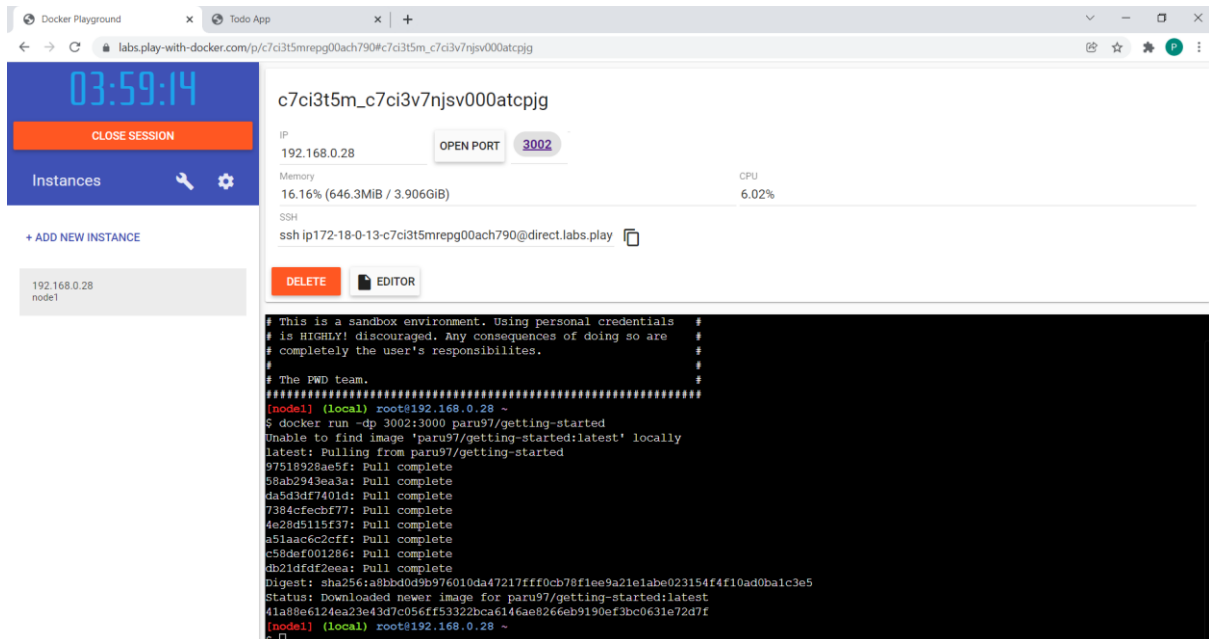
## Sharing the app by creating a repo in dockerhub



## Pushing the app to the hub

`docker push paru97/getting-started`

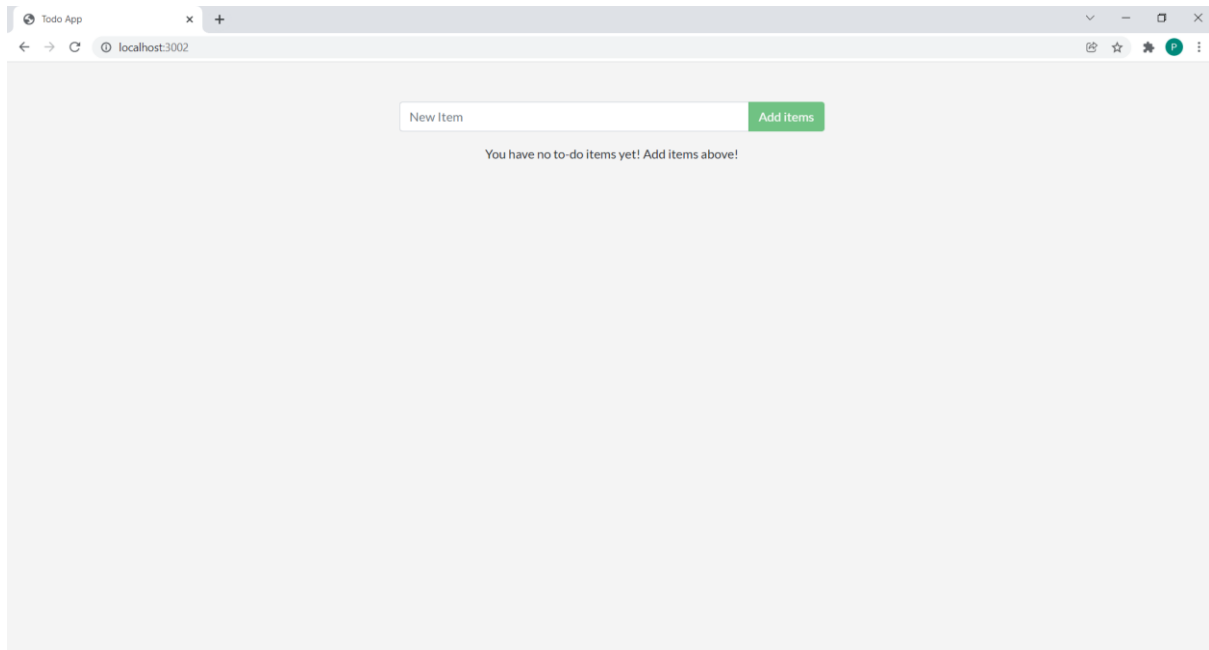
# Running the image in labs.play-with-docker



The screenshot shows the Docker Playground interface. On the left, there's a sidebar with a clock showing 03:59:14, a 'CLOSE SESSION' button, and a list of instances. The main area displays details for a container named 'c7ci3t5m\_c7ci3v7njsv000atcpjg'. It shows the IP address 192.168.0.28, memory usage of 16.16% (646.3MiB / 3.906GiB), and CPU usage of 6.02%. There's an 'OPEN PORT' button with '3002' selected. Below this, there's a 'DELETE' button and an 'EDITOR' button. The terminal window shows the following output:

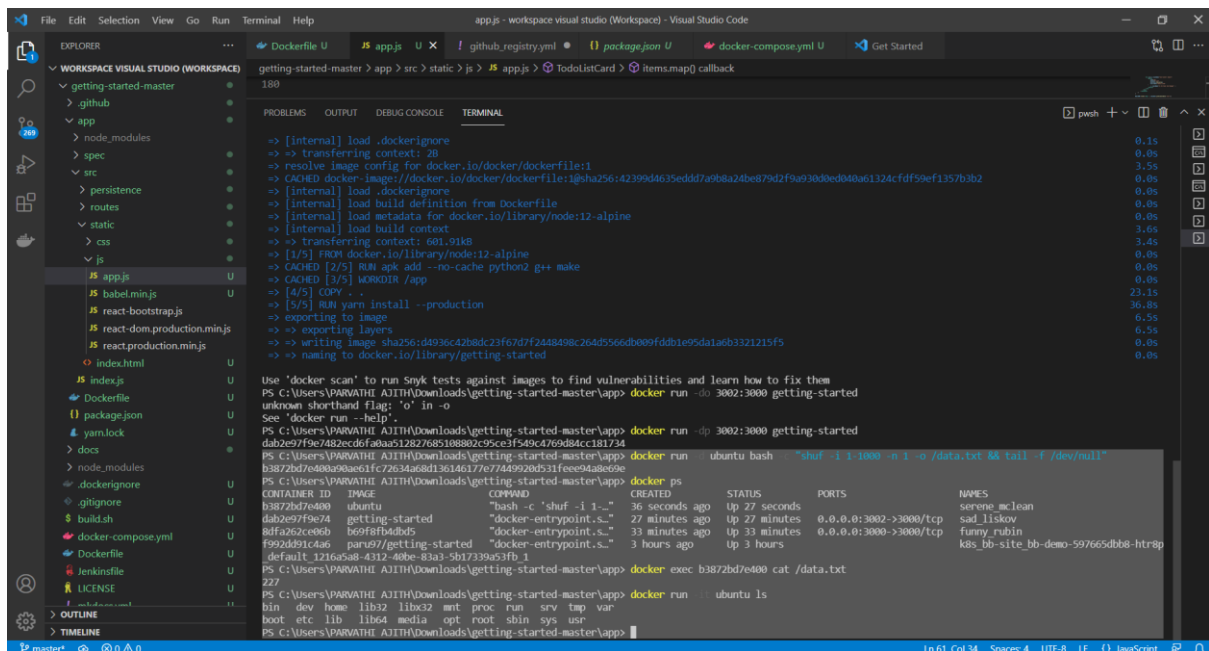
```
# This is a sandbox environment. Using personal credentials #
# is HIGHLY discouraged. Any consequences of doing so are #
# completely the user's responsibilities. #
# The PWD team. #
#####
[node1] (local) root@192.168.0.28 ~
$ docker run -dp 3002:3000 paru97/getting-started
Unable to find image 'paru97/getting-started:latest' locally
latest: Pulling from paru97/getting-started
97518928ae5f: Pull complete
58ab2943ea3a: Pull complete
da5d3df7401d: Pull complete
7384cfebf77: Pull complete
4e28d5115f37: Pull complete
a51aac6c2c6f: Pull complete
c58deff0012b6: Pull complete
8b21dfdf2eaa: Pull complete
Digest: sha256:a8bbd0d9b976010da47217fff0cb78f1ee9a21e1abe023154f4f10ad0ba1c3e5
Status: Downloaded newer image for paru97/getting-started:latest
41a88e6124ea23e43d7c056ff53322bca6146ae826eab9190ef3bc0631e72d7f
[node1] (local) root@192.168.0.28 ~
$
```

## Opening it in browser again through the port in this lab



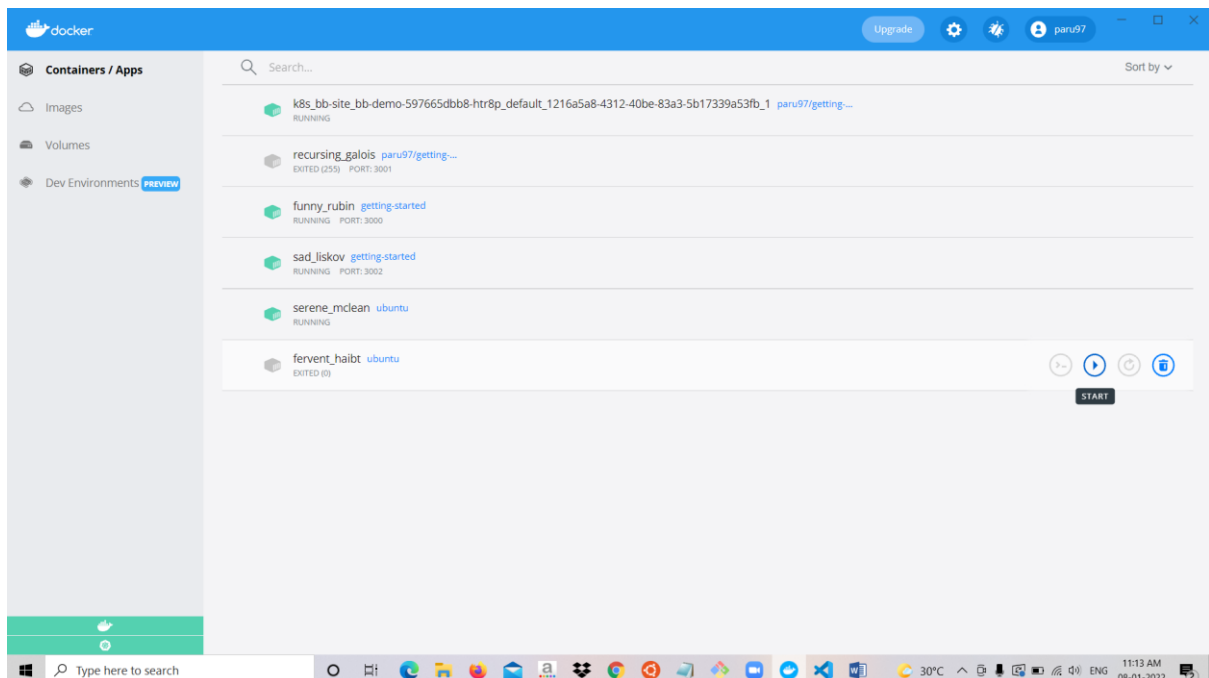
The screenshot shows a web browser with the address bar set to 'localhost:3002'. The page displays a simple 'Todo App' interface. At the top, there's a text input field labeled 'New Item' and a green 'Add Items' button. Below this, a message reads: 'You have no to-do items yet! Add items above!'. The background is a light gray color.

Running same images on two containers to show each container is independent.



The screenshot shows the Visual Studio Code interface with a Dockerfile open in the editor. The terminal displays the output of the `docker build` command, showing the build process for the `getting-started-master` app. The build process includes steps like `FROM docker.io/library/node:12-alpine`, `WORKDIR /app`, `ADD . .`, `RUN yarn install --production`, and `EXPOSE 3000`. The final image is named `getting-started` with a sha256 digest. The terminal also shows the `docker run` command being executed, which starts a container named `getting-started` with the `getting-started` image. The container is running on port 3000. The terminal output shows the container's IP address and the `npm start` command being executed.

When the new container is listed out the old image is not present, which shows containers are independent.

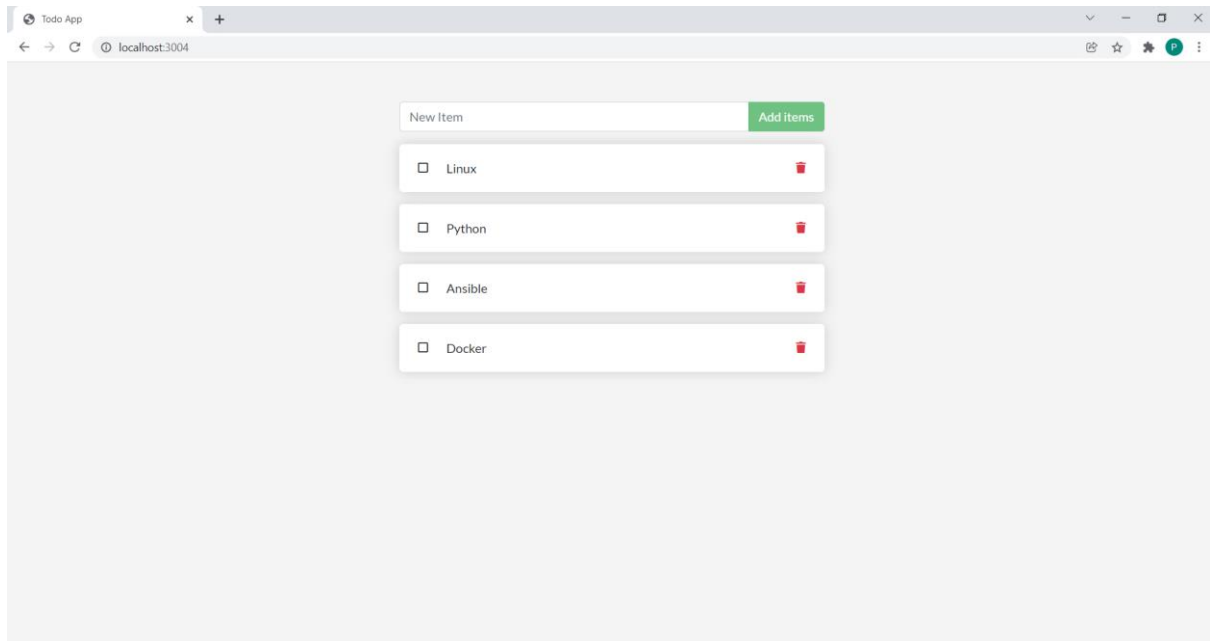


For persisting the database we can use named volumes.

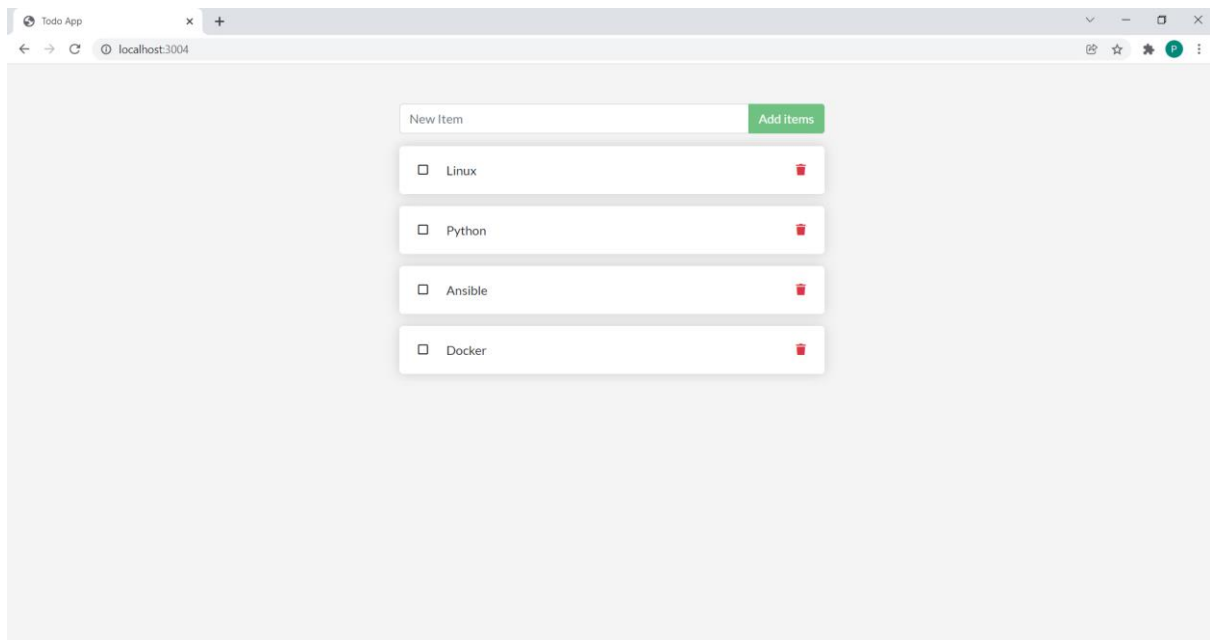
The screenshot shows the Visual Studio Code interface with a workspace named 'app.js - workspace visual studio (Workspace)'. The Explorer panel on the left shows the project structure, including files like 'getting-started-master', 'github', 'app', 'node\_modules', 'spec', 'src', 'persistence', 'routes', 'static', 'css', 'js', 'app.js', 'babel.min.js', 'react-bootstrap.js', 'react-dom.production.min.js', 'react.production.min.js', 'index.html', 'index.js', 'Dockerfile', 'package.json', 'yam.lock', 'docs', 'node\_modules', '.dockerignore', '.gitignore', 'build.sh', 'docker-compose.yml', 'Dockerfile', 'Jenkinsfile', and 'LICENSE'. The Dockerfile is open in the editor, showing the build process for a 'getting-started' application. The terminal output shows the build process, including the use of 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them. The output also shows the creation of a 'todo-db' volume and the execution of the 'docker run' command to start the container.

This screenshot shows the same Visual Studio Code interface, but with the terminal output scrolled down to show the 'docker run' command and the 'todo-db' volume. The output shows the container 'getting-started' being created and started. The 'docker run' command is: `docker run -dp 3002:3000 getting-started`. The output also shows the creation of the 'todo-db' volume: `docker volume create todo-db`. The 'docker run' command is then executed with the 'todo-db' volume: `docker run -dp 3004:3000 -v todo-db:/etc/todos getting-started`. The output shows the container 'getting-started' being created and started, and the 'todo-db' volume being inspected: `docker volume inspect todo-db`. The output shows the volume details, including the 'CreatedAt', 'Driver', 'Labels', 'Mountpoint', 'Name', 'Options', and 'Scope'.

## Opening it in browser



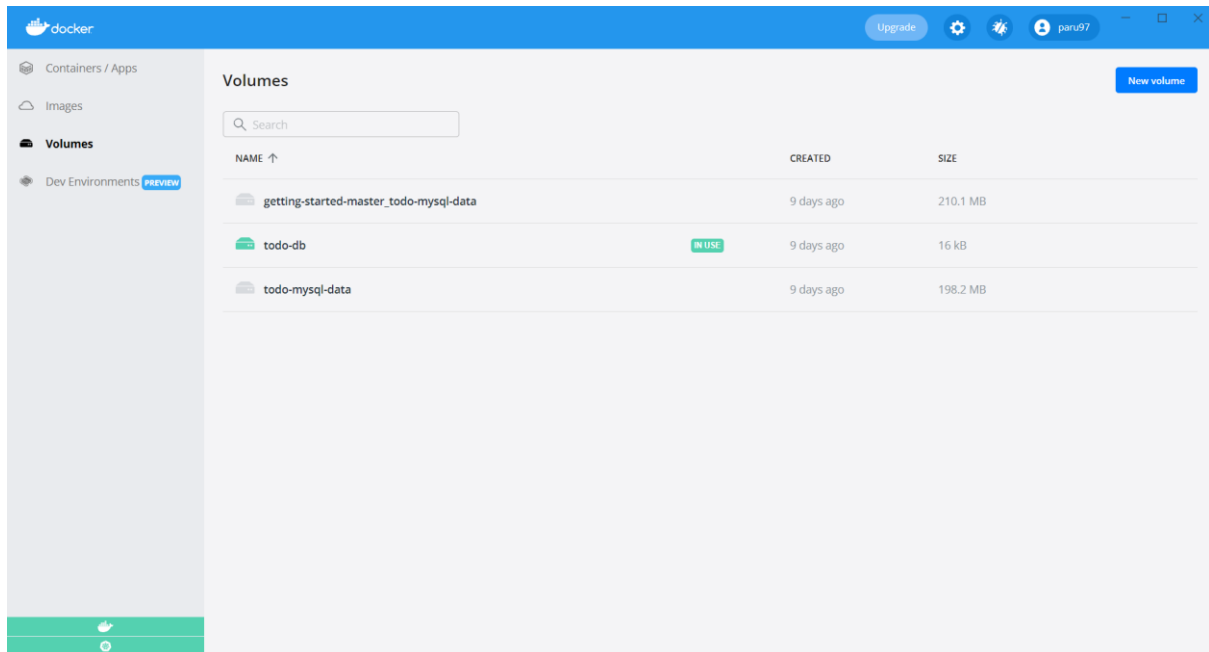
## Cross checking to see whether the data is persisting



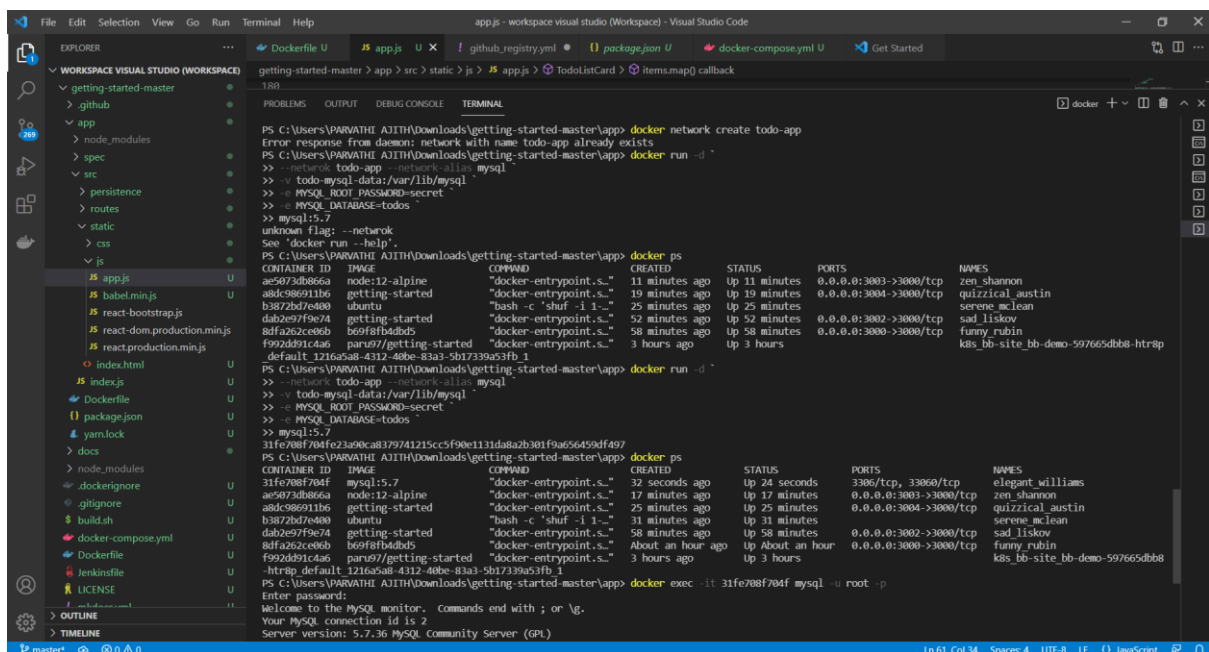
## Data persists



# Volumes in docker desktop



## Starting a multi-container app using container networking.



```
File Edit Selection View Go Run Terminal Help
app.js - workspace visual studio (Workspace) - Visual Studio Code

EXPLORER
WORKSPACE VISUAL STUDIO (WORKSPACE)
  > getting-started-master
  > .github
  > app
  > node_modules
  > spec
  > src
  > persistence
  > routes
  > static
  > css
  > js
  > app.js
  > babel.min.js
  > react-bootstrap.js
  > react-dom.production.min.js
  > react.production.min.js
  > index.html
  > index.js
  > Dockerfile
  > package.json
  > yarn.lock
  > docs
  > node_modules
  > .dockerignore
  > .gitignore
  > build.sh
  > docker-compose.yml
  > Dockerfile
  > Jenkinsfile
  > LICENSE
  > OUTLINE
  > TIMELINE

TERMINAL
getting-started-master > app > src > static > js > app.js > docker ps
188
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
31fe708f704fe23a90ca8379741215cc5f90e1131d48a2b301f9a656459df497  31fe708f704f       "docker-entrypoint.s..." 32 seconds ago      Up 24 seconds      3306/tcp, 33060/tcp  elegant-williams
ae5073db866a       node:12-alpine     "docker-entrypoint.s..." 17 minutes ago      Up 17 minutes      0.0.0.0:3003->3000/tcp zen-shannon
a8dc986911b6       mysql:5.7           "docker-entrypoint.s..." 25 minutes ago      Up 25 minutes      0.0.0.0:3004->3000/tcp quizzical-austin
b2872bd7e489       ubuntu             "bash -c 'shuf -i 1-...' 31 minutes ago      Up 31 minutes      0.0.0.0:3002->3000/tcp serene-mclean
da02e97f9074       getting-started     "docker-entrypoint.s..." 58 minutes ago      Up 58 minutes      0.0.0.0:3000->3000/tcp sad-11skov
8df2a262ce0b       b69f8fb4dd5        "docker-entrypoint.s..." About an hour ago    Up About an hour    0.0.0.0:3000->3000/tcp funny-rubin
f992dd91c4a6       paru97/getting-started "docker-entrypoint.s..." 3 hours ago          Up 3 hours          0.0.0.0:3000->3000/tcp k8s_b6-site_b6-demo-597665dbb8

PS C:\Users\PARVATHI_AJITH\Downloads\getting-started-master\app> docker exec -it 31fe708f704f mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.36 MySQL Community Server (GPL)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or 'h' for help. Type 'c' to clear the current input statement.

mysql> SHOW DATABASES;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'SHOW
DATABASES' at line 1
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| todos |
+-----+
5 rows in set (0.09 sec)

mysql>
```

```
File Edit Selection View Go Run Terminal Help
app.js - workspace visual studio (Workspace) - Visual Studio Code

EXPLORER
WORKSPACE VISUAL STUDIO (WORKSPACE)
  > getting-started-master
  > .github
  > app
  > node_modules
  > spec
  > src
  > persistence
  > routes
  > static
  > css
  > js
  > app.js
  > babel.min.js
  > react-bootstrap.js
  > react-dom.production.min.js
  > react.production.min.js
  > index.html
  > index.js
  > Dockerfile
  > package.json
  > yarn.lock
  > docs
  > node_modules
  > .dockerignore
  > .gitignore
  > build.sh
  > docker-compose.yml
  > Dockerfile
  > Jenkinsfile
  > LICENSE
  > OUTLINE
  > TIMELINE

TERMINAL
getting-started-master > app > src > static > js > app.js > docker run --network todo-app nicolaka/netshoot
188
mysql> ^C
mysql>
mysql> q
-> exit
mysql> exit
Bye
PS C:\Users\PARVATHI_AJITH\Downloads\getting-started-master\app> docker run --network todo-app nicolaka/netshoot
88d888b. .d8888b. d8888p .d8888b. 88d888b. .d8888b. .d8888b. d8888p
88" 88 880000d8 88 Y800000. 88" 88 88" 88 88" 88 88
88 88 88. ... 88 88 88 88 88 88. 88 88. 88 88
dp dp "88888p" dp "88888p" dp dp "88888p" "88888p" dp

Welcome to Netshoot! (github.com/nicolaka/netshoot)

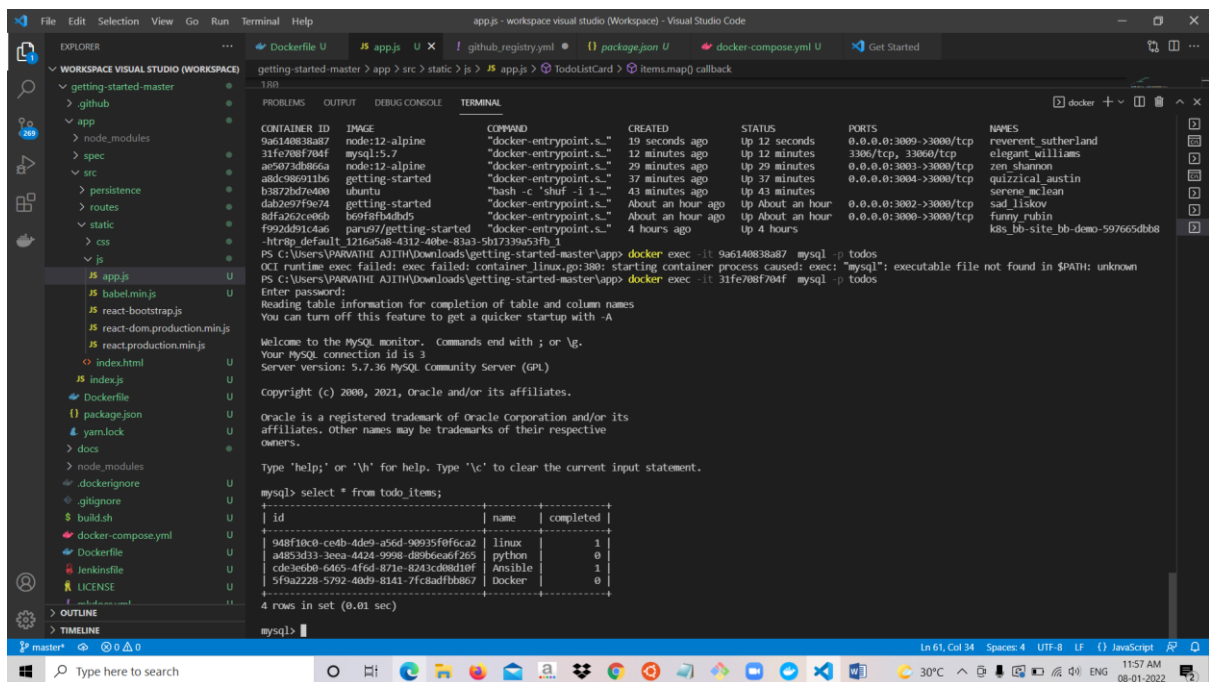
1c4a4762a77f 3: dig mysql
; <<> Dig 9.16.22 <<> mysql
;; global options: +cmd
;; Got answer:
;; -->HEADER<<- opcode: QUERY, status: NOERROR, id: 45134
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;mysql.                IN      A

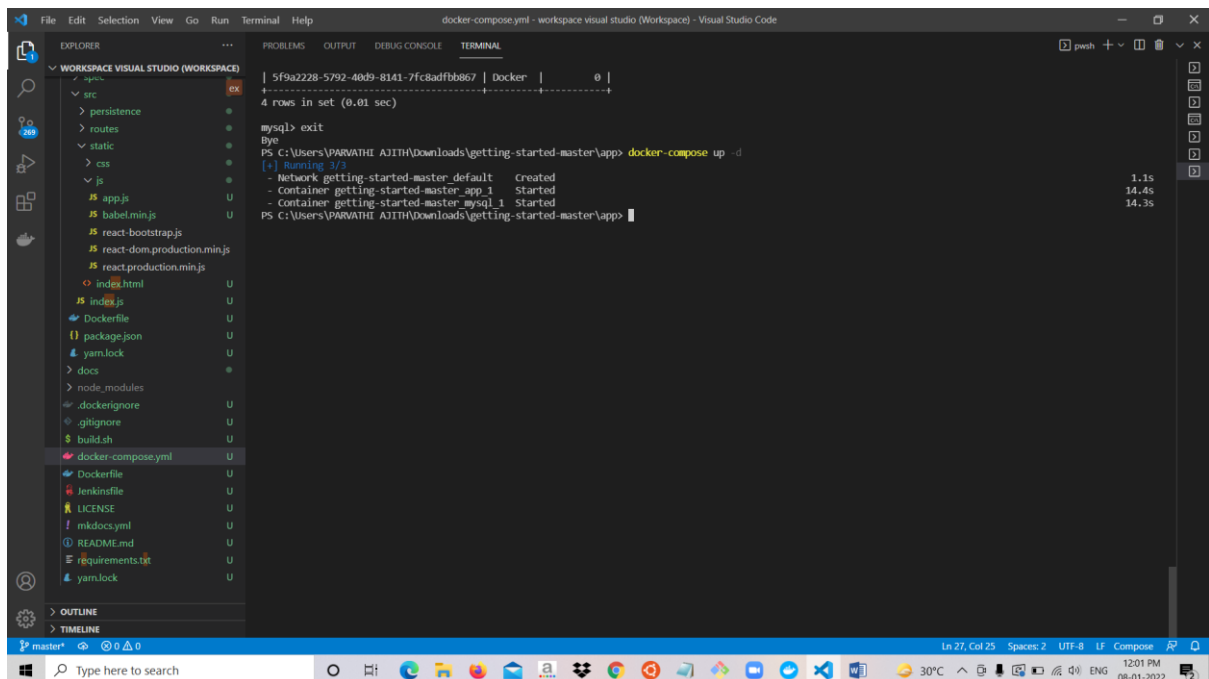
;; ANSWER SECTION:
mysql.                 600     IN      A      172.18.0.2

;; Query time: 10 msec
;; SERVER: 127.0.0.11#53(127.0.0.11)
;; WHEN: Sat Jan 08 06:19:38 UTC 2022
;; MSG SIZE rcvd: 44

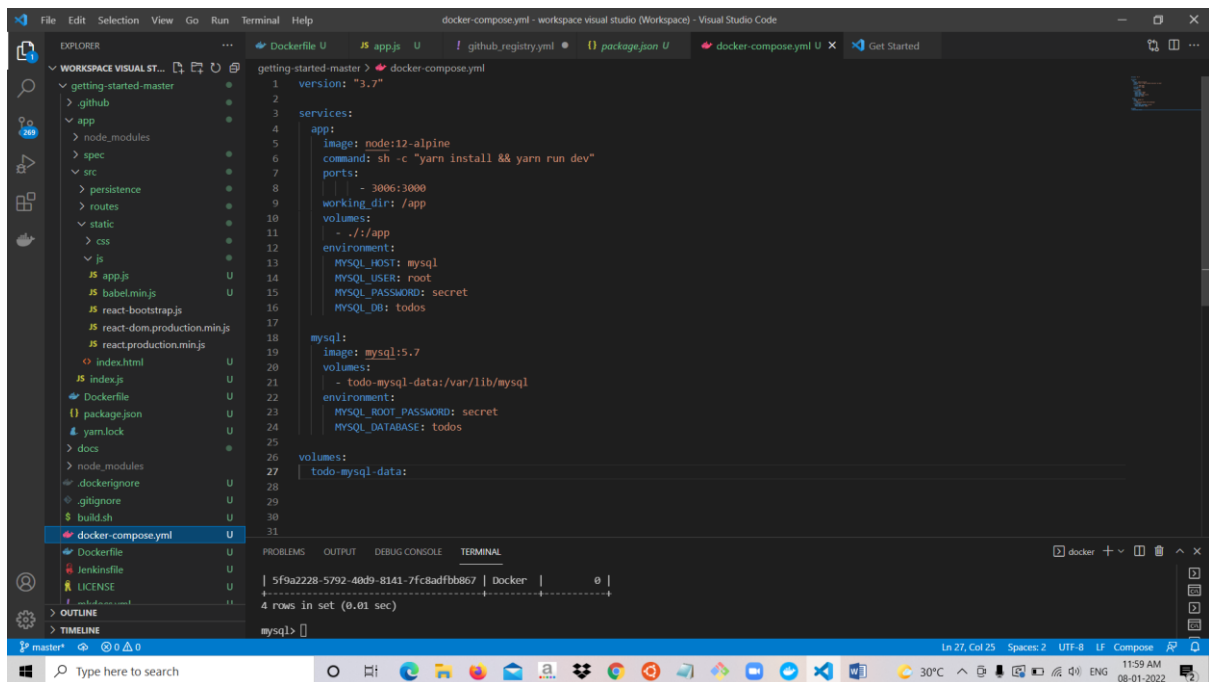
1c4a4762a77f 3: 
```



## Running multi container apps using docker compose



# Docker compose.yml



## Running the app as a stack more efficiently

