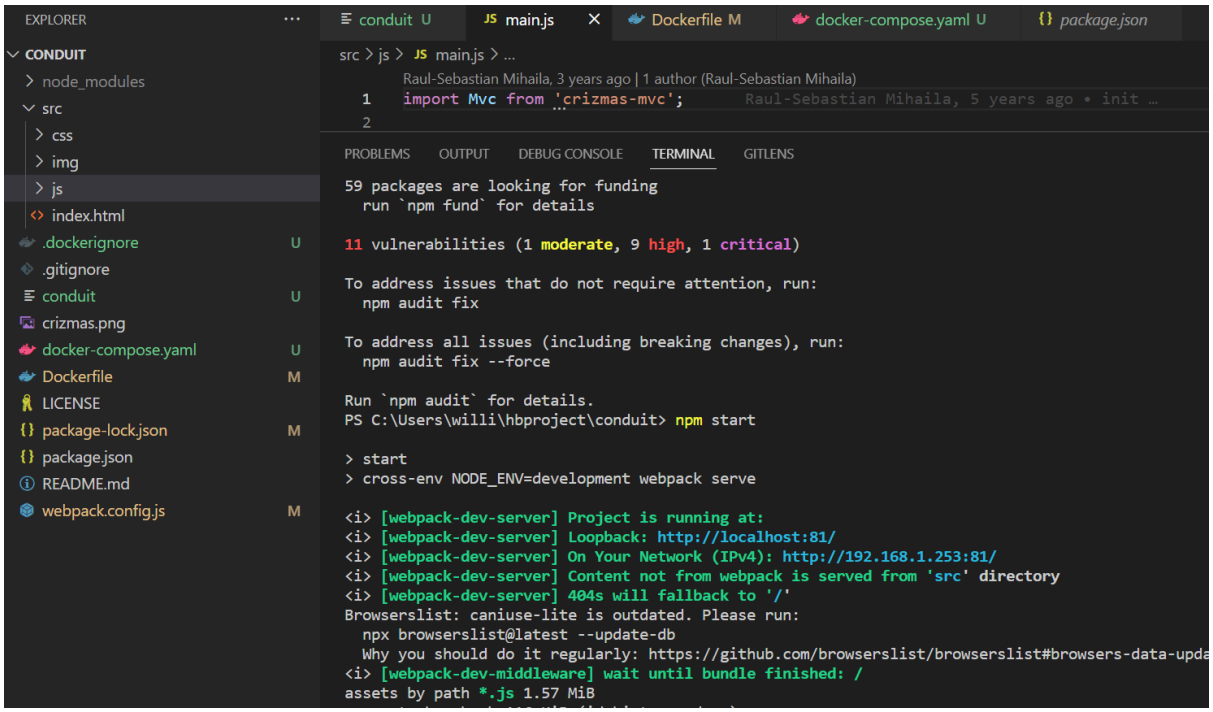
Documentation and Project Script Files

Building the Front End on a local machine using Node Js

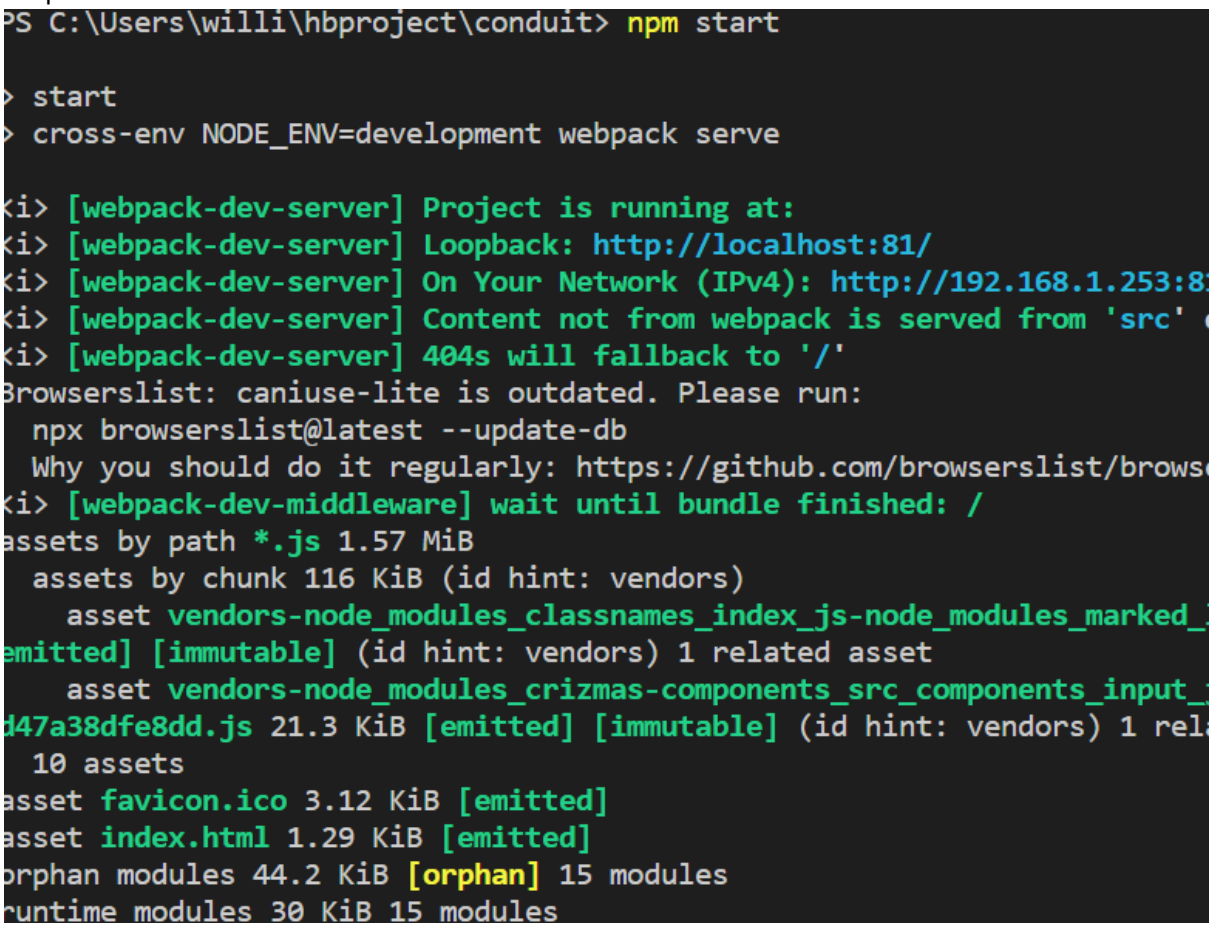
- Clone the repo to your c drive or any location on your pc.

```
willi@LAPTOP-G8DMH5MV MINGW64 ~/hbpro
$ git clone https://github.com/gothinkster/crizmas-mvc-realworld-example-app.gi
t
Cloning into 'crizmas-mvc-realworld-example-app'...
remote: Enumerating objects: 473, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (16/16), done.
remote: Total 473 (delta 8), reused 0 (delta 0), pack-reused 457R
Receiving objects: 100% (473/473), 1.46 MiB | 467.00 KiB/s, done.
Resolving deltas: 100% (253/253), done.
```

- Install all dependencies using npm install

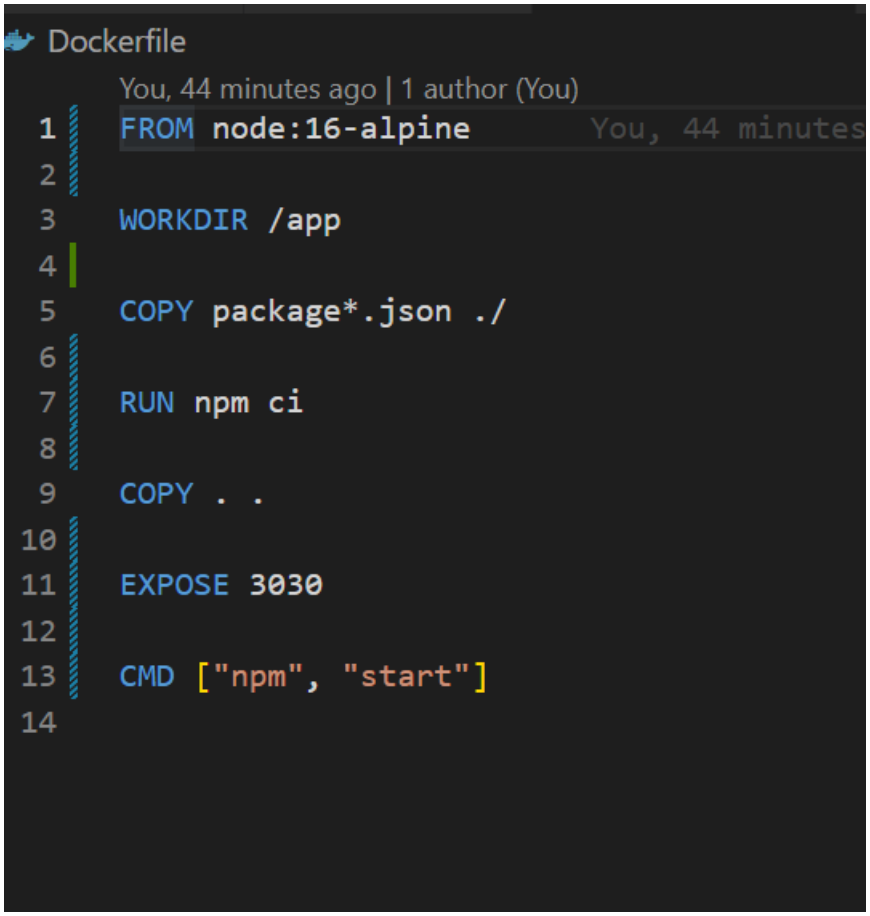


- Start the app using npm start, use nodemon to avoid running npm all the time. After a successful compilation, run your application from the local port you've configured in your script file.



Building and Starting Application Docker

Dockerization



- Step 1:Create Dockerfile

Create a new file in your project and call it Dockerfile, this will create image template contain script that will export and run our project in Docker.

- Step 2: Scripting

Using FROM key word to specify the base image to build the application on. Image is specify based on the type of platform you are running your application on.

WORKDIR to specify working directory for docker

Using COPY to copy the source code into the image

The EXPOSE key word to expose ports to be use by our application

The CMD command keyword to run our script using npm

-Step 3: Building

```
PS C:\Users\willi\hbproject\conduit> docker build -t conduit .
[+] Building 73.9s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 162B
 => [internal] load .dockerignore
 => => transferring context: 34B
 => [internal] load metadata for docker.io/library/node:16-alpine
 => [1/5] FROM docker.io/library/node:16-alpine@sha256:1908564153449b1c46b32
 => => resolve docker.io/library/node:16-alpine@sha256:1908564153449b1c46b32
 => => sha256:9a2db0008b592cfde1b98aa8972a8051ccb80bb51dde8b06c3a2e3762794e8
 => => sha256:b0cbdedc1b9dc2bb5c7e026b134a51116d1c6c33a75ed304c876d34e07bc61
 => => sha256:95ca4541bf43fa13f7c3ef511d8b0f1a156a738645582a3c7e27a7b3fdebf4
 => => sha256:b831dd745300d07e75a46ae8805ad39a4164cf888686f4a45ed782a26fd0d5
 => => sha256:86937321b535acf01ac5d7f108ead204d1c6a71f9a1c4402e8317ece72d1f6
 => => sha256:1908564153449b1c46b329e6ce2307e226bc566294f822f11c5a8bcef4eeaa
```

Use docker build –t [appname] . to run a build the app, the will build what we specify on the dockerfile step by step.

-Step 4: Starting

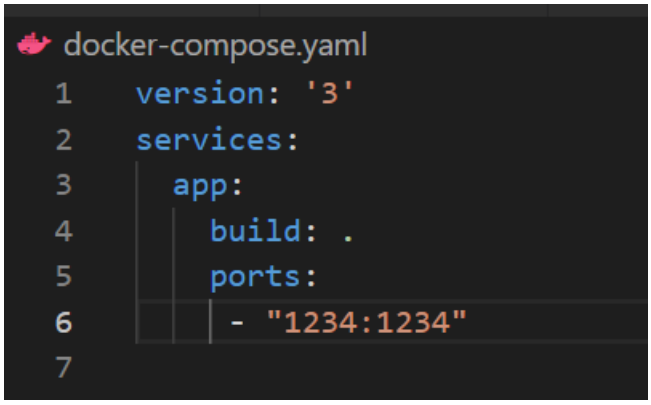
```
PS C:\Users\willi\hbproject\conduit> docker run -p 81:81 conduit

> start
> cross-env NODE_ENV=development webpack serve

<i> [webpack-dev-server] Project is running at:
<i> [webpack-dev-server] Loopback: http://localhost:81/
<i> [webpack-dev-server] On Your Network (IPv4): http://172.17.0.3:81/
<i> [webpack-dev-server] Content not from webpack is served from 'src' directory
<i> [webpack-dev-server] 404s will fallback to '/'
Browserslist: caniuse-lite is outdated. Please run:
  npx browserslist@latest --update-db
```

Finally we can start our app using **docker run –p port:port [appname]** and specify the port to run locally using docker.

Using Docker Compose



Docker compose is able to run all the containers that you need to run an app at once instead of running them individually.

Using docker-compose we can avoid using different commands to build and start the container every time like was done using the dockerfile above, instead we can create a docker-compose.yml file and specify the version of docker-compose format, the services which is app in this case, specify what to build and the port that will be use.

Because only the front-end is running we have only one service in the docker-compose file above but multiple services can be run here such as the database, server and other services that will complete your application.

After setting up the docker-compose file, run **docker compose up** this will build and start the project all at ounce.