1. Setting up the development environment:
   Choose a programming language and framework:
   I chose Node.js with Express.js to create endpoints /hello and /world.

   Setting up development environment:
   1. Downloaded and installed Node.js, which indeed includes npm.
   2. Created a new directory (hello-world)
   3. Opened the directory using VS Code
   4. Initialized npm using npm init -y
   5. Installed express using npm install express
   6. Initialized git using git init

2. Creating the Microservices:
   - Created hello.js in the root directory with endpoint /hello and set it to port 3000
   - Created world.js in the root directory with endpoint /world and set it to port 3001

3. Containerizing the Microservices with Docker
   Create Dockerfiles:
   - Created Dockerfile.hello in the root directory to create a docker image of hello.js microservice
   - Created Dockerfile.world in the root directory to create a docker image of world.js micro service

   Built Docker images:
   - Made sure that the Docker desktop is running and login into docker using "Docker login" command in the terminal

   Executed following command to build the docker images of both micro services
   - docker build -t hello:latest -f Dockerfile.hello .
   - docker build -t world:latest -f Dockerfile.world .

   Tagged and pushed the images to Docker hub
   - docker tag hello:latest harshakata/hello:latest
   - docker tag world:latest harshakata/world:latest

   - docker push harshakata/hello:latest
   - docker push harshakata/world:latest

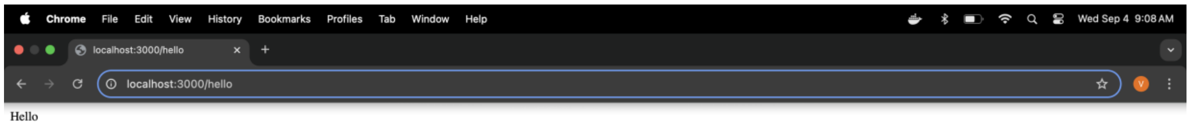
Note: harshakata is my docker hub username

Docker image links:
world:
https://hub.docker.com/layers/harshakata/world/latest/images/sha256:7d63bf16f63c1c11b7bf037b6f48b86d569513101e94fb1f83197dbb0e67f7fd?uuid=8f6ea230-7cc4-4730-9a1e-0317ed314bbe%0A

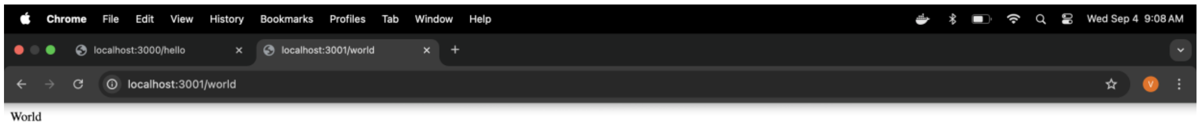
Hello:
https://hub.docker.com/layers/harshakata/hello/latest/images/sha256:3c5400bd3940300cc1b829541cd2700868b44664db7b5ca9dd03b18d2f41ff89?uuid=8f6ea230-7cc4-4730-9a1e-0317ed314bbe%0A

Run the services locally:

- Checked if the services are working independently by executing following commands:
  - docker run -p 3000:3000 harshakata/hello


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Wed Sep 4 9:08 AM
localhost:3000/hello
localhost:3000/hello
Hello

  - docker run -p 3001:3001 harshakata/world



World

4. Deploying the application on Kubernetes:

Set up a Kubernetes cluster:

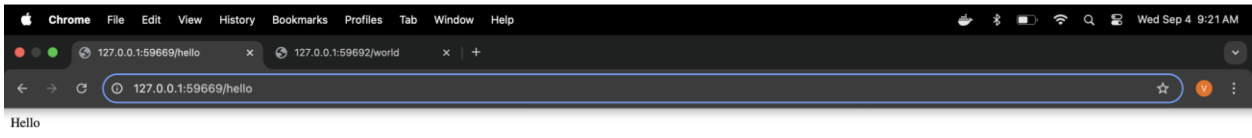
- Started Minikube using minikube start

Create Kubernetes Manifests:

- Created hello.yaml and world.yaml file defining deployments and services for each micro service
- Deployed the services using below commands:
  - kubectl apply -f hello.yaml
  - kubectl apply -f world.yaml

Ensured both services are accessible through Kubernetes services by executing following commands.

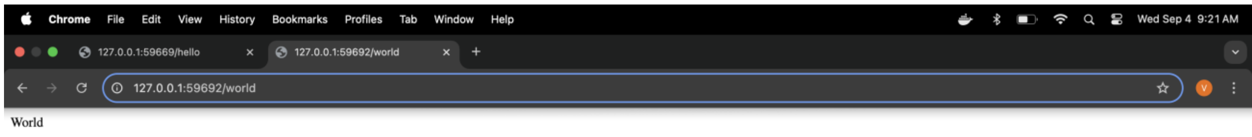
  - minikube service hello


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Wed Sep 4 9:21 AM
127.0.0.1:59669/hello
127.0.0.1:59692/world

127.0.0.1:59669/hello

Hello

  - minikube service world

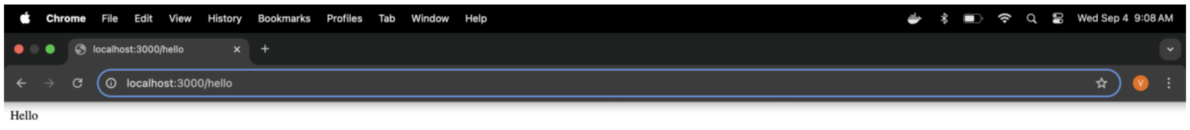

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Wed Sep 4 9:21 AM
127.0.0.1:59669/hello
127.0.0.1:59692/world

127.0.0.1:59692/world

World

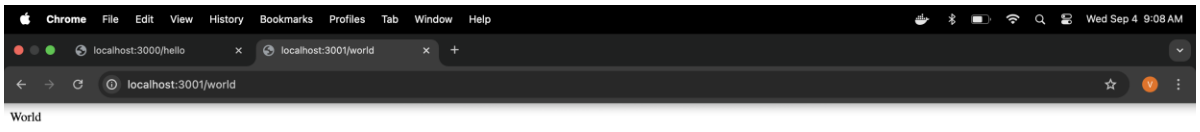
5. Testing and Integration

Tested the Hello and World services individually to confirm they return the correct responses.


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Wed Sep 4 9:08 AM
localhost:3000/hello

localhost:3000/hello

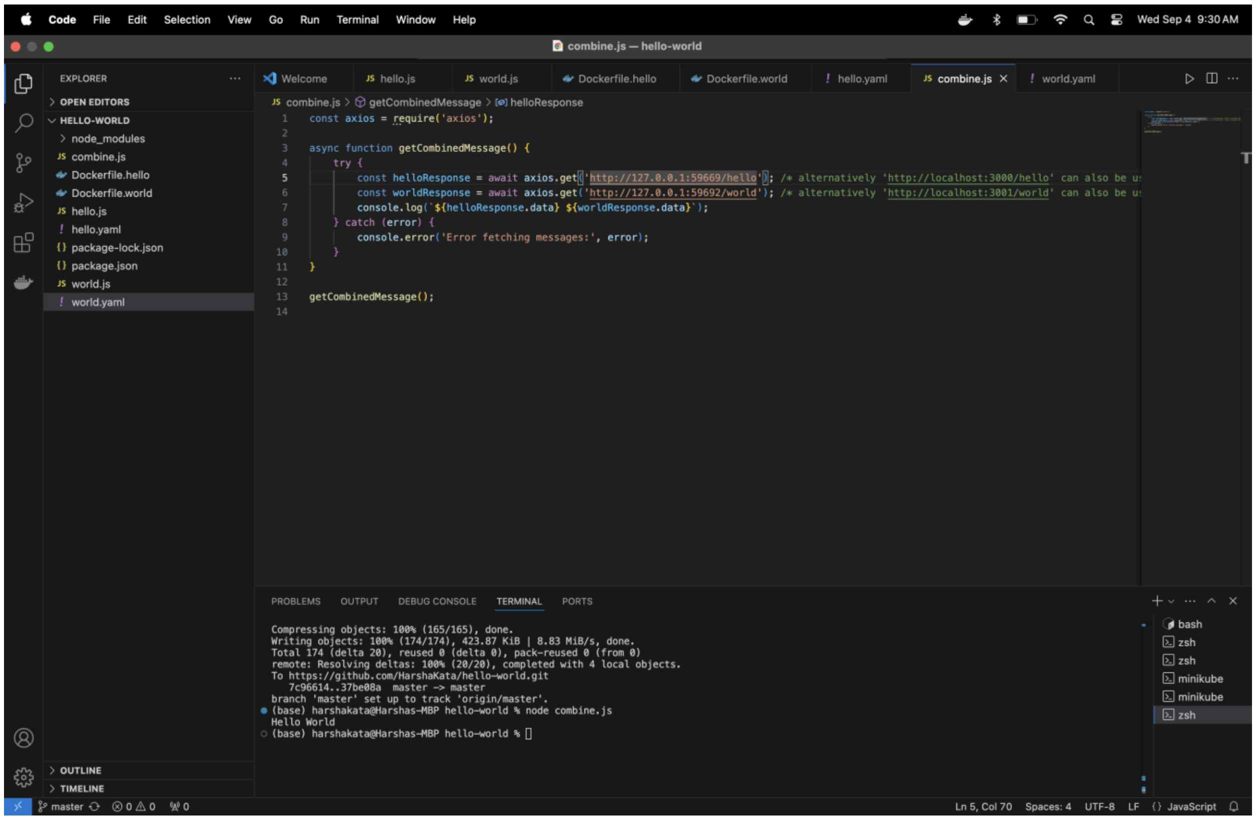
Hello



World

Create a simple script or service:

- Created a simple script "combine.js" to read response from individual services and to print "Hello World"
- Run both minikube services in separate terminals
- Executed "node combine.js" in another terminal to print the output in console



```javascript
const axios = require('axios');

async function getCombinedMessage() {
    try {
        const helloResponse = await axios.get('http://127.0.0.1:59669/hello'); /* alternatively 'http://localhost:3000/hello' can also be us
        const worldResponse = await axios.get('http://127.0.0.1:59692/world'); /* alternatively 'http://localhost:3001/world' can also be us
        console.log(`${helloResponse.data} ${worldResponse.data}`);
    } catch (error) {
        console.error('Error fetching messages:', error);
    }
}

getCombinedMessage();
```

```
Compressing objects: 100% (165/165), done.
Writing objects: 100% (174/174), 423.87 KiB | 8.83 MiB/s, done.
Total 174 (delta 20), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (20/20), completed with 4 local objects.
To https://github.com/HarshaKata/hello-world.git
   7c96614..37be08a  master -> master
branch 'master' set up to track 'origin/master'.
(base) harshakata@Harshas-MBP hello-world % node combine.js
Hello World
(base) harshakata@Harshas-MBP hello-world %
```

Used following git commands to timely push the code into git repository:

Git add .

Git commit –m "commit message"

Git push –u origin master