Links:

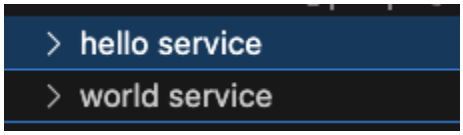
https://hub.docker.com/repositories/bmltera
https://github.com/bmltera/helloworld-microservices/tree/main

Step 1

I chose Node.js with Express.js. My git repository is initiated at
https://github.com/bmltera/helloworld-microservices/tree/main

Step 2

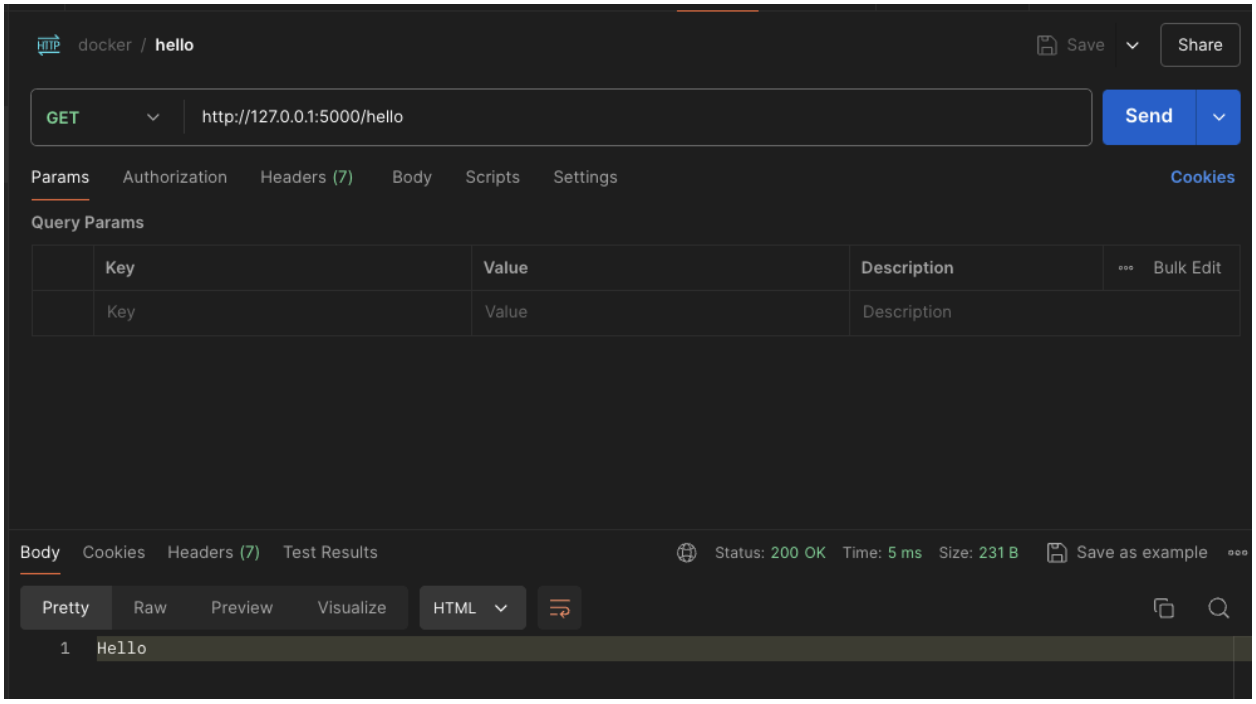
I created the hello service and world service each in their respective folders.



Each service creates a simple express client with a single endpoint which returns “hello” or “world”.
Here is the code of the hello service:

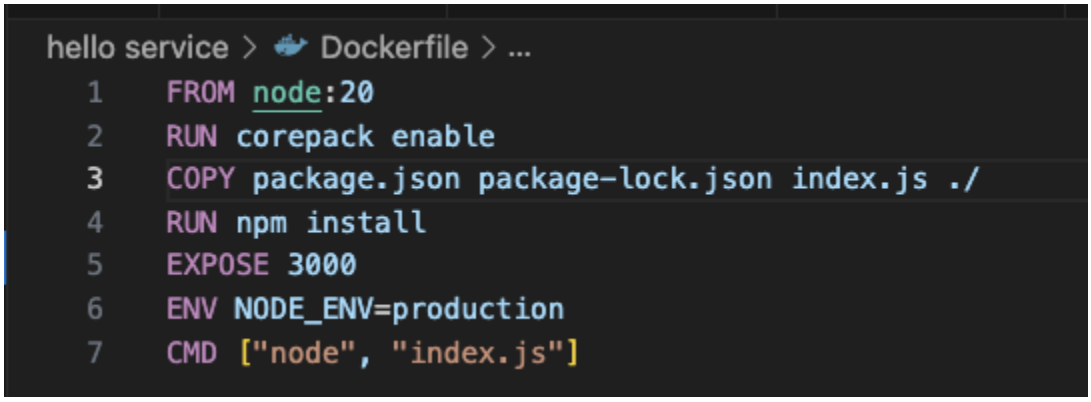
```
hello service > JS index.js > ...
1   const express = require('express');
2
3   const app = express();
4   const port = 3000;
5
6   app.get('/hello', (req, res) => {
7       console.log('hello called');
8       res.send('Hello');
9   });
10
11  app.listen(port, () => {
12      console.log(`hello service listening at http://localhost:${port}`)
13  })
14
```

I tested the endpoints with postman and got the expected return values.



Step 3

I created a dockerfile for each service. The dockerfile specifies the node version and commands necessary to run the service in a docker container. These dockerfiles are located in the project folder of each service.



I first tested the docker locally. I then pushed the images to dockerrhub. The images can be found here: https://hub.docker.com/repositories/bmltera

Step 4

I chose to use Minikube for this project. I defined a deployment YAML file for each service. I also created service YAML files using LoadBalancer so that the local machine can access the kubernetes services.

The deployment files pull the docker images that I pushed to dockerhub. This simplifies the setup process, so that other testers do not need to create the images locally.

After the kubernetes files are deployed, I run the command “minikube tunnel”. This runs the services and opens up the specified ports on the local machine (5000 and 5001) to interact with the kubernetes services.

I then packaged all of the commands into a shell script “setup.sh” located in the main directory of the repository.

Here is the kubernetes deployment script:

```
$ setup.sh
1   #!/bin/bash
2   minikube delete
3   minikube start
4
5   # Apply Kubernetes manifest files
6   kubectl apply -f deployment-hello.yml
7   kubectl apply -f deployment-world.yml
8   kubectl apply -f service-hello.yml
9   kubectl apply -f service-world.yml
10
11  # start minikube tunnel
12  (minikube tunnel &) &> /dev/null
13
```

Step 5

I tested the hello and world kubernetes services and successfully retrieved the return values. I wrote a shell script to test this by calling the endpoints and returning the combined values “Hello World”. The script file is “app.sh”, located in the main directory of the repository.

Step 6

I documented the process in this file. I also created a readme for setup and testing instructions. Here are the contents of the readme:

Project tested on Mac with M2 chip

- Make sure ports 5000 and 5001 on localhost are not occupied as these are the ports the services listen on.
- Dockerfiles are located in the respective microservice folders.
- Documentation file is documentation.pdf.

Steps to run application:

1. in the main project directory, run the script `bash script.sh` from terminal. This script deletes any existing minikubes in docker and sets up a new minikube. It will then deploy the hello service and world service, and the kubernetes service for both.
2. wait 1-5 minutes for kubernetes deployments to be ready
3. Test the microservices by running the script `bash app.sh`. This will call both services and return their concatenation, resulting in a console output of `Hello World`.

github: https://github.com/bmltera/helloworld-microservices/tree/main

dockerhub: https://hub.docker.com/repositories/bmltera