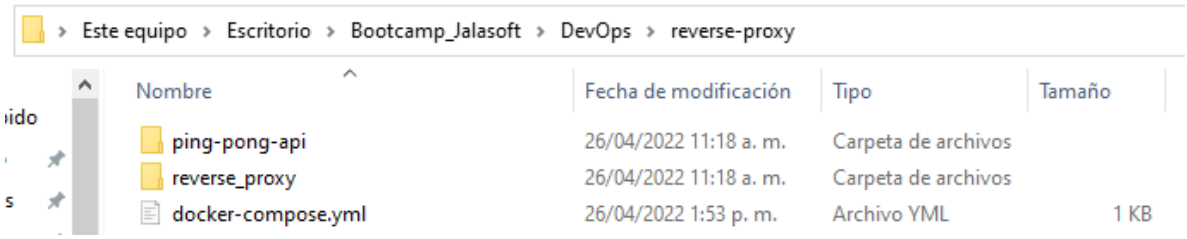
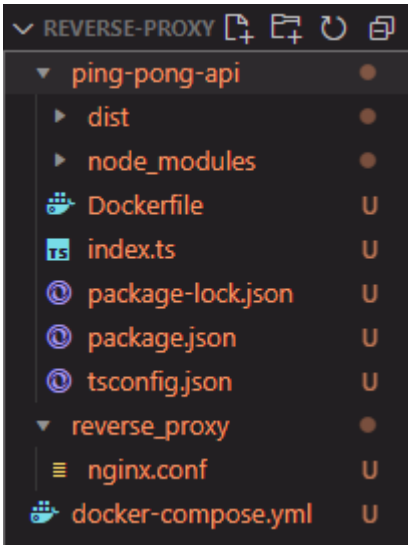
Setting up a reverse proxy with Nginx for an Express web app

Build the web app using Express and TypeScript

Create a directory to hold the project. In this case, the directory is called reverse-proxy



The directory structure should be like the following



Make sure to have TypeScript, Express and type declaration files for Express installed

```
npm i express

npm i -D @types/express
```

Create a simple API with two get requests: /ping and /pong. When /ping is called it should respond with pong; when /pong is called it should respond with ping.

ping-pong-api > index.ts > ...

```typescript
import express from 'express';

const app = express();

// Ping => pong
app.get('/ping', (req, res) => {
    res.send('pong');
});

// Pong => ping
app.get('/pong', (req, res) => {
    res.send('ping');
});

const port = process.env.PORT || 3000;

app.listen(port, () => console.log(`App listening on PORT ${port}`));
```

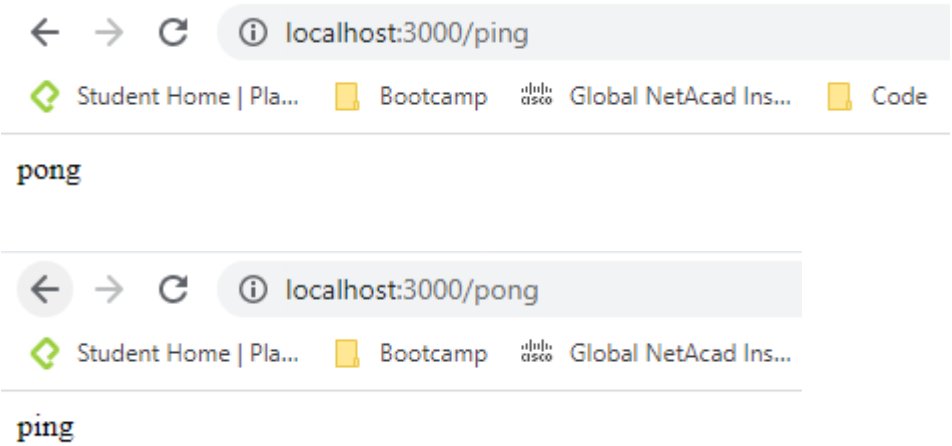
Make sure to add the following scripts on the package.json file:

```
ping-pong-api > package.json > ...
{
  "name": "ping-pong-api",
  "version": "1.0.0",
  "description": "Simple API that returns ping or pong",
  "main": "dist/index.js",
  ▷ Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node dist/index.js",
    "dev": "tsc -w & nodemon dist/index.js"
  },
  "author": "J. Sebastián Beltrán S.",
  "license": "ISC",
  "devDependencies": {
    "@types/express": "^4.17.13",
    "typescript": "^4.6.3"
  },
  "dependencies": {
    "express": "^4.18.0"
  }
}
```

Run the app by typing in the console `npm run start`

```
> ping-pong-api@1.0.0 start
> node dist/index.js

App listening on PORT 3000
```

Test it by accessing http://localhost:3000/ping or http://localhost:3000/pong



Use Docker Compose to set up the app and the reverse proxy

Create a Docker image based on Node to run the app on a container. This Dockerfile copies the package.json and package-lock.json files to the working directory /api inside the container, then runs `npm install`, then copies the whole content of the project into the directory, exposes port 3000 and runs `npm run start` command

```
ping-pong-api > Dockerfile > ...
1  FROM node:latest
2  WORKDIR /api
3  COPY package*.json ./
4  RUN npm install
5  COPY ./ .
6  EXPOSE 3000
7  CMD ["npm", "run", "start"]
```

Create a docker-compose file to integrate nginx and the web application using the previously set image

docker-compose.yml

```yaml
services:

  # Reverse proxy using nginx
  reverse-proxy:
    image: nginx:latest
    container_name: reverse-proxy
    depends_on:
      - ping-pong
    volumes:
      - ./reverse_proxy/nginx.conf:/etc/nginx/nginx.conf
    ports:
      - 80:80


  # Main API
  ping-pong:
    image: ping-pong-api
    container_name: ping-pong-api
    build:
      context: ./ping-pong-api
    ports:
      - 3000:3000
    restart: on-failure
```

Create a `nginx.conf` file inside the folder reverse_proxy with the following information. This configuration tells the proxy to listen in port 80 in localhost, capture all traffic in that port, and redirect to the network of the container running the web app (ping-pong) if the entry point is localhost:80/ping or localhost:80/pong

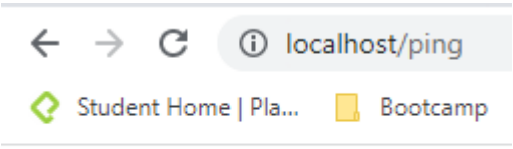
reverse_proxy > nginx.conf

```
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;

events {
    worker_connections 1024;
}

http {

    # Ping Pong API reverse proxy
    server {
        listen 80;
        server_name localhost 127.0.0.1;

        location /ping {
            proxy_pass          http://ping-pong:3000/ping;
            proxy_set_header    X-Forwarded-For $remote_addr;
        }

        location /pong {
            proxy_pass          http://ping-pong:3000/pong;
            proxy_set_header    X-Forwarded-For $remote_addr;
        }
    }
}
```

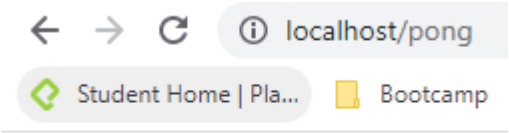
Stand up containers using docker-compose up

```
Starting ping-pong-api ... done
Starting reverse-proxy ... done
Attaching to ping-pong-api, reverse-proxy
ping-pong-api    |
ping-pong-api    | > ping-pong-api@1.0.0 start
ping-pong-api    | > node dist/index.js
ping-pong-api    |
ping-pong-api    | App listening on PORT 3000
reverse-proxy    | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
reverse-proxy    | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
reverse-proxy    | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
reverse-proxy    | 10-listen-on-ipv6-by-default.sh: info: IPv6 listen already enabled
reverse-proxy    | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
reverse-proxy    | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
reverse-proxy    | /docker-entrypoint.sh: Configuration complete; ready for start up
```

Test the reverse proxy is running by entering http://localhost/ping or http://localhost/pong


localhost/ping
Student Home | Pla...
Bootcamp

pong


localhost/pong
Student Home | Pla...
Bootcamp

ping