

Przygotowanie platformy

Instalacja terraform

wget https://releases.hashicorp.com/terraform/1.8.1/terraform_1.8.1_linux_386.zip

unzip terraform_1.8.1_linux_386.zip

echo "export PATH=\$PATH:~/local/bin" >> ~/.bashrc

cp terraform ~/local/bin

```
eee_W_311562@runweb121348:~/local/bin$ cd ..
eee_W_311562@runweb121348:~/local$ cd ..
eee_W_311562@runweb121348:~$ terraform
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init      Prepare your working directory for other commands
  validate  Check whether the configuration is valid
  plan      Show changes required by the current configuration
  apply     Create or update infrastructure
  destroy   Destroy previously-created infrastructure

All other commands:
  console   Try Terraform expressions at an interactive command prompt
  fmt       Reformat your configuration in the standard style
  force-unlock Release a stuck lock on the current workspace
  get       Install or upgrade remote Terraform modules
  graph     Generate a Graphviz graph of the steps in an operation
  import    Associate existing infrastructure with a Terraform resource
  login     Obtain and save credentials for a remote host
  logout    Remove locally-stored credentials for a remote host
  metadata  Metadata related commands
  output    Show output values from your root module
  providers Show the providers required for this configuration
  refresh   Update the state to match remote systems
  show      Show the current state or a saved plan
  state     Advanced state management
  taint     Mark a resource instance as not fully functional
  test     Execute integration tests for Terraform modules
  untaint   Remove the 'tainted' state from a resource instance
  version   Show the current Terraform version
  workspace Workspace management

Global options (use these before the subcommand, if any):
  -chdir=DIR Switch to a different working directory before executing the
             given subcommand.
  -help      Show this help output, or the help for a specified subcommand.
  -version   An alias for the "version" subcommand.
eee_W_311562@runweb121348:~$
```

Uruchomienie Linuxa przez terraform

ssh-keygen -t ed25519 -f ~/.ssh/id_ed25519

git clone <https://github.com/phajder/ask-node-react-docker>

terraform init

terraform apply -var "public_key=\$(cat ~/.ssh/id_ed25519.pub)"

```
+ revoke_rules_on_delete = false
+ tags_all               = {
  + "source" = "tf-ask-docker"
}
+ vpc_id           = (known after apply)
}

Plan: 3 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ ask_docker_private_ips = [
  + (known after apply),
]
+ ask_docker_public_ips  = [
  + (known after apply),
]

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_key_pair.main: Creating...
aws_security_group.ask_docker_sg: Creating...
aws_key_pair.main: Creation complete after 0s [id=ask-vm-key]
aws_security_group.ask_docker_sg: Creation complete after 3s [id=sg-0407f49ffa7c4ec8]
aws_instance.ask_docker_ec2[0]: Creating...
aws_instance.ask_docker_ec2[0]: Still creating... [10s elapsed]
aws_instance.ask_docker_ec2[0]: Still creating... [20s elapsed]
aws_instance.ask_docker_ec2[0]: Still creating... [30s elapsed]
aws_instance.ask_docker_ec2[0]: Still creating... [40s elapsed]
aws_instance.ask_docker_ec2[0]: Still creating... [50s elapsed]
aws_instance.ask_docker_ec2[0]: Creation complete after 53s [id=i-0c3b64f0064c39e44]

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:

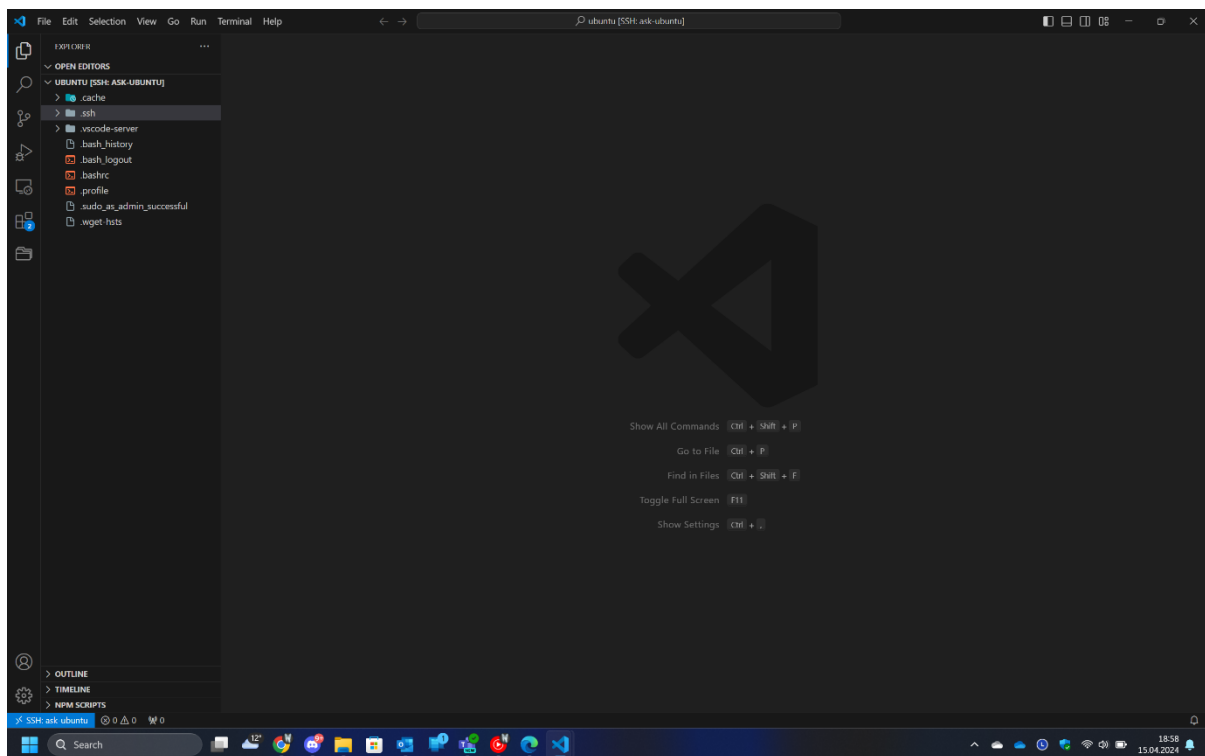
ask_docker_private_ips = [
  "172.31.50.234",
]
ask_docker_public_ips  = [
  "54.242.192.138",
]
eee_h_3115626@runweb121348:~/lab1/ask-node-react-docker/infra$
```

Skonfigurowanie Visual Studio Code

Pobranie wtyczki Remote-SSH

Ustawienie konfiguracji połączenia

```
Host ask-ubuntu
  HostName <adres ip>
  User ubuntu
  Port 22
  IdentityFile <sciezka do klucza publicznego>
```



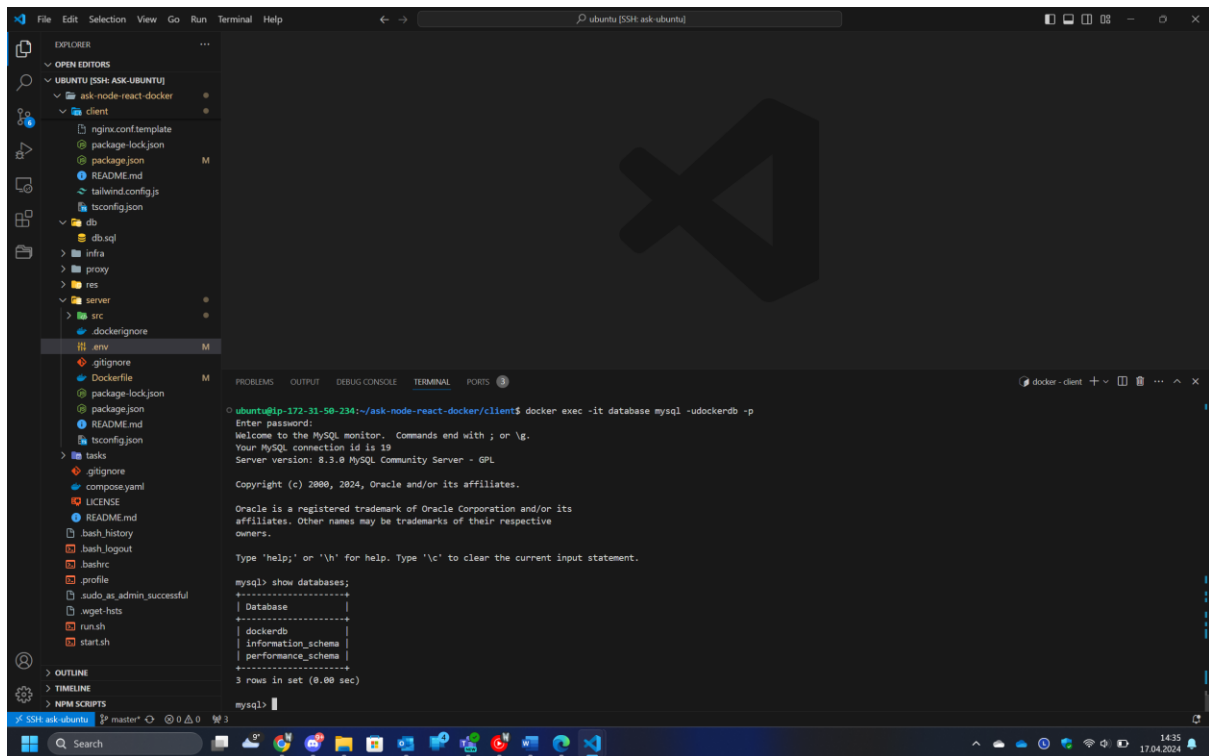
Skonfigurowanie bazy danych

docker pull mysql

docker run --name database -e MYSQL_ROOT_PASSWORD=<haslo root> -d mysql:latest

docker exec -i database sh -c 'exec mysql -uroot -p"\$MYSQL_ROOT_PASSWORD"' < ~/ask-node-react-docker/db/db.sql

docker exec -it database mysql -udockerdb -p



Skonfigurowanie aplikacji backendowej

docker pull node

Plik Dockerfile

```
FROM node:latest

WORKDIR /app

COPY package*.json ./

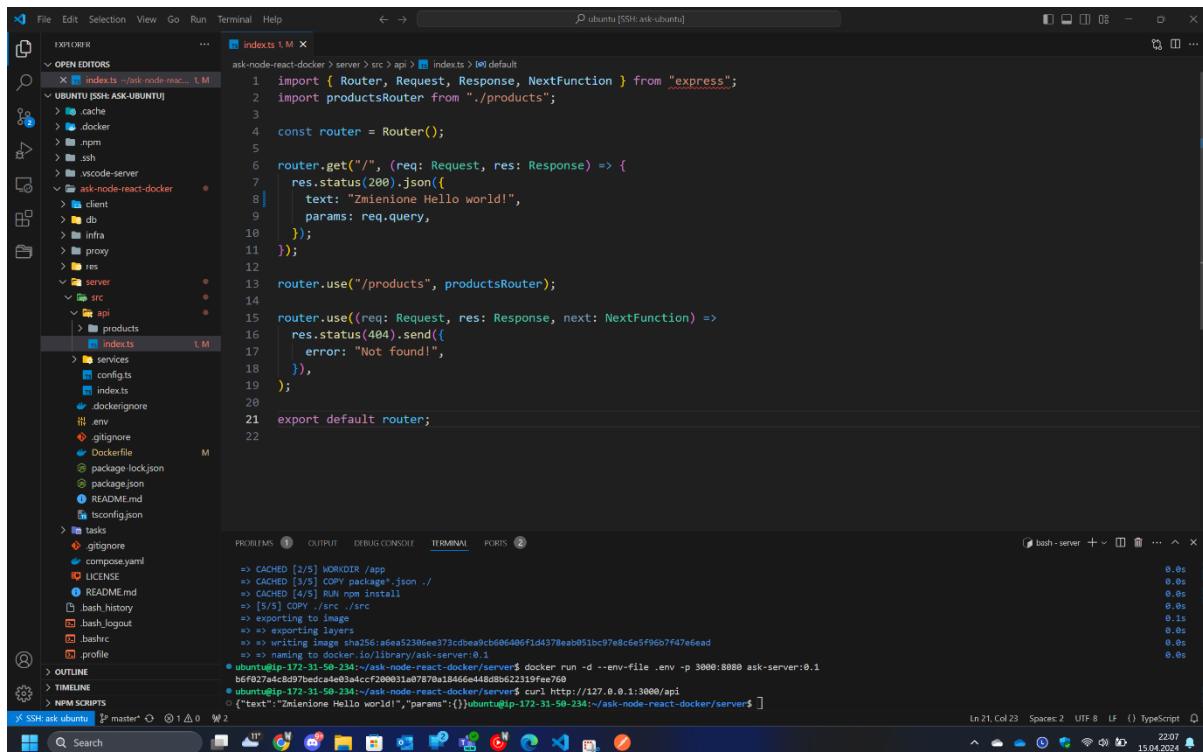
RUN npm install

EXPOSE ${PORT}

CMD ["npm", "run", "dev"]
```

docker build -t ask-server:0.1 ~/ask-node-react-docker/server/

docker run -d --name backend --env-file ~/ask-node-react-docker/server/.env -p 8000:8080 -v
~/ask-node-react-docker/server/src:/app/src ask-server:0.1



Skonfigurowanie aplikacji frontendowej

Plik Dockerfile

```
FROM node:latest

WORKDIR /app

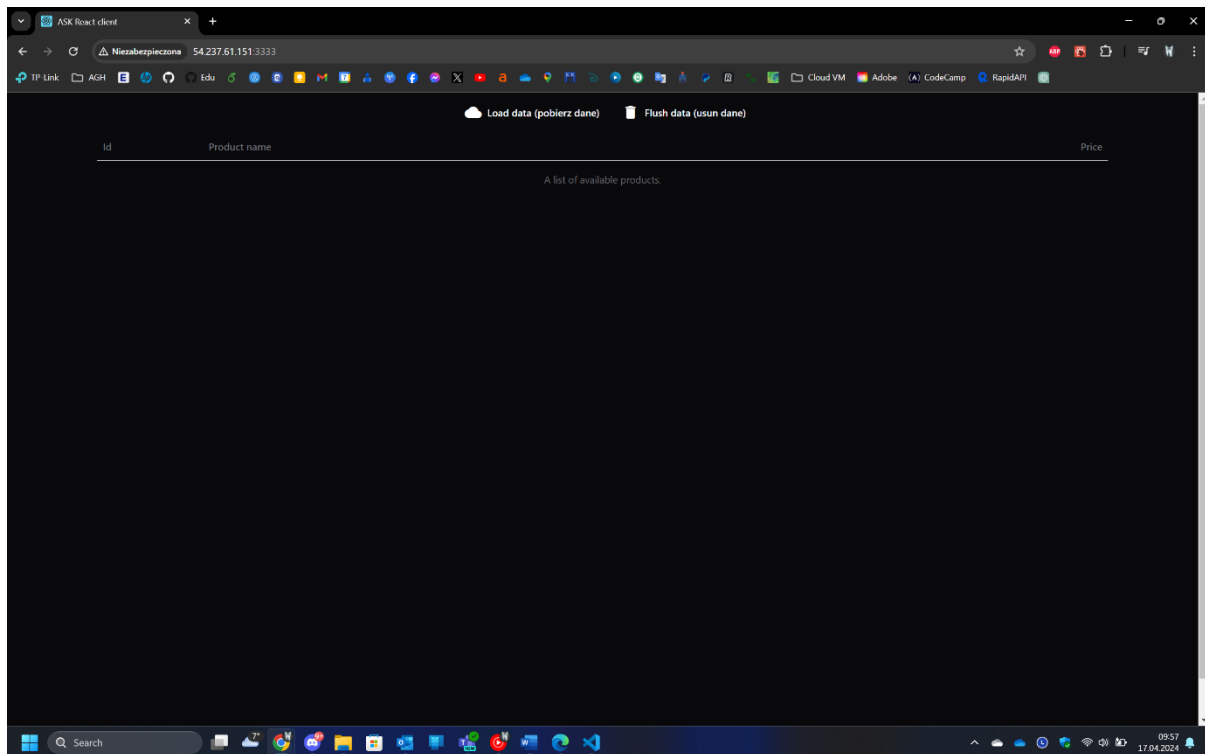
COPY package*.json ./
COPY *.config.js ./
COPY *.json ./

RUN npm install

CMD ["npm", "start"]
```

`docker build -t ask-front:0.1 ~/ask-node-react-docker/client/`

`docker run -d --name frontend -e CI=true -p 3333:3000 -v ~/ask-node-react-docker/client/src/app/src -v ~/ask-node-react-docker/client/public:/app/public ask-front:0.1`



Połączenie wszystkich trzech aplikacji

Tworzymy własną sieć

`docker network create --subnet=192.168.46.0/24 custom_net`

Ustawiamy odpowiednie adresy IP na kontenerach oraz w plikach konfiguracyjnych aplikacji

Plik `.env`

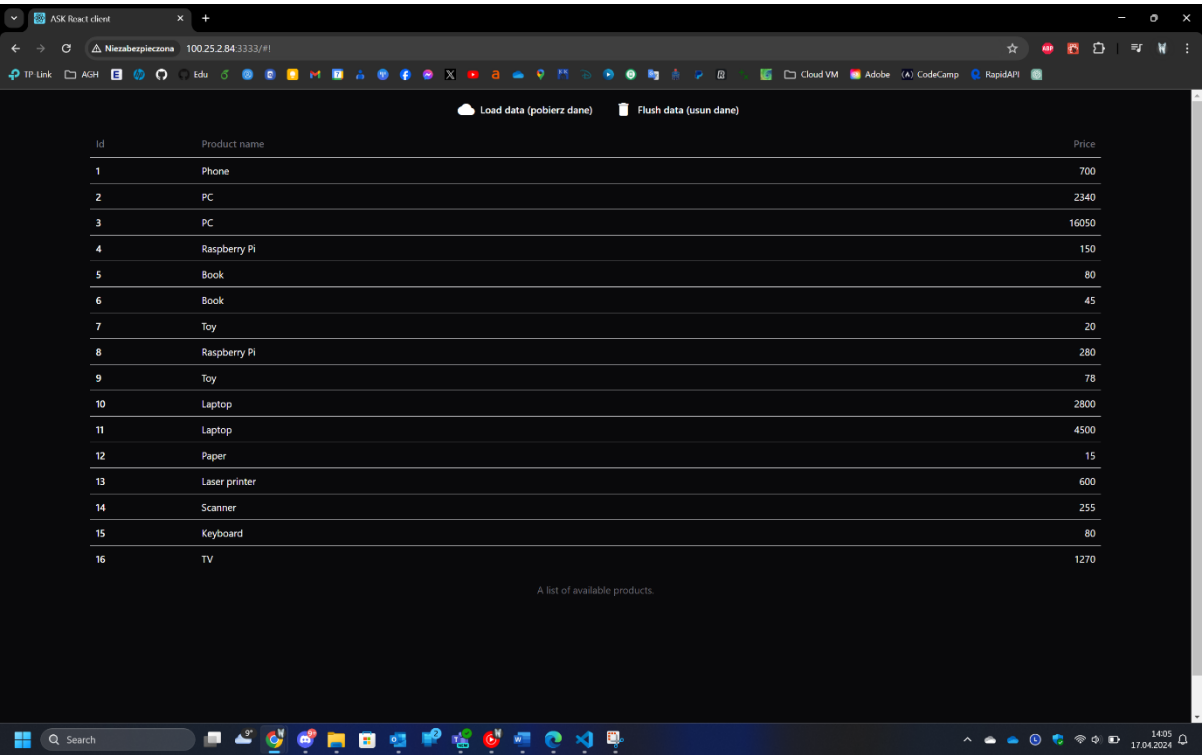
```
# Database
DB_HOST=192.168.46.100
DB_PORT=3306
DB_NAME=dockerdb
DB_USER=dockerdb
DB_PSWD=Zaq12wsx
```

Plik `package.json`

```
"proxy": "http://192.168.46.101:8080",
```

Uruchamiamy kontenery z poprawnymi adresami IP. Do komendy `docker run` dodajemy odpowiednie parametry:

1. **Baza danych:** `--net custom_net --ip 192.168.46.100`
2. **Backend:** `--net custom_net --ip 192.168.46.101`
3. **Frontend:** `--net custom_net --ip 192.168.46.102`



Aplikacja działa poprawnie.