**Amazon Web Service Понятия**

**EC2** — это облачный сервис, предоставляющий виртуальные сервера

**Docker** - Программа, позволяющая операционной системе запускать процессы в изолированном окружении на базе специально созданных образов.

Ход работы

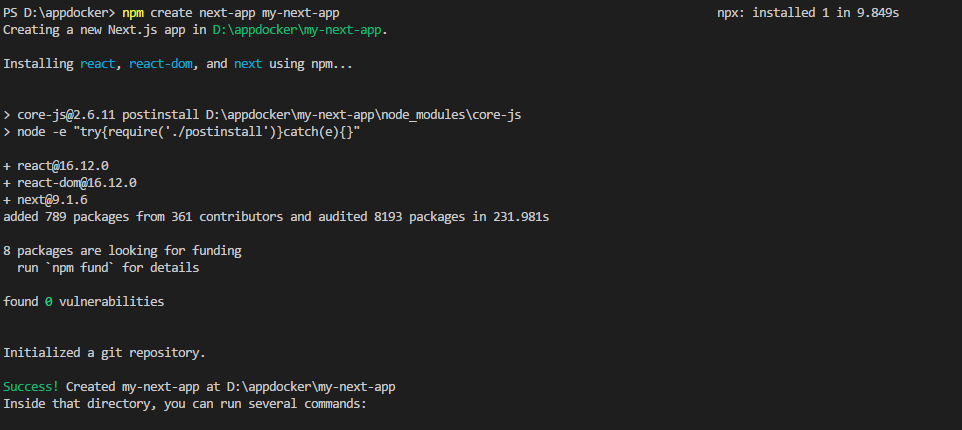
1. Создание простого NodeJs приложения.
2. Создание docker образа с нашим приложением.
3. Создание виртуальной машины Linux на AWS EC2.
4. Установка на виртуальную машину Docker, и запуск нашего приложения.
5. **Создание простого WebApi приложения на базе NodeJs.**

Открываем редактор, например VSCode. Открываем в нем консоль и создаем проект:

npm install

npm create next-app my-next-app (Windows)

npx create-next-app (Linux)



1. **Создание docker образа с нашим приложением.**

Создаем в папке проекта файл «Dockerfile» без расширения.

Вставляем в него содержимое:

FROM node:12

# Set working directory. Paths will be relative this WORKDIR.

WORKDIR /usr/src/app

# Install dependencies

COPY package\*.json ./

RUN npm install

# Copy source files from host computer to the container

COPY . .

# Build the app

RUN npm run build

# Specify port app runs on

EXPOSE 3000

# Run the app

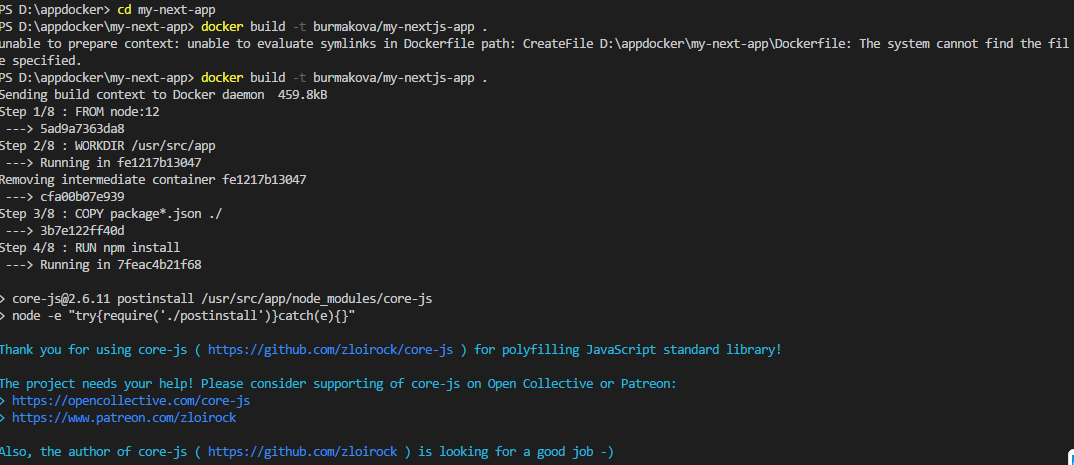
CMD [ "npm", "start" ]

Далее перейдем в папку нашего проекта и создаем образ контейнера,

cd my-next-app

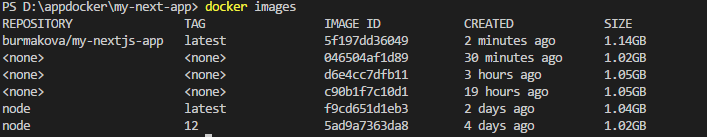
docker build -t burmakova/my-nextjs-app .

где burmakova - это логин на Docker Hub, my-next-app - название нашего контейнера. (Аналогично на GitHub-е хранятся ваши репозитории, имя\_аккаунта/имя\_репозитория).



Чтобы убедится в создании контейнера, можно запустить команду:

docker images



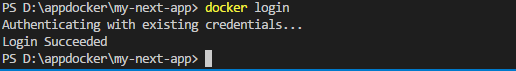
Попробуем запустить наш контейнер, команда docker run.

docker run -it -p 3000:80 burmakova/ my-nextjs-app

Проверяем работу приложение в браузере localhost:3000.

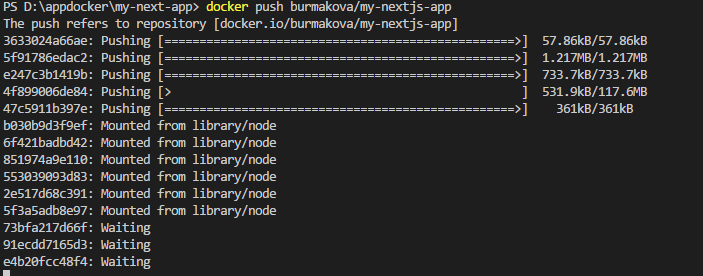
Далее нужно доставить наш контейнер на Docker Hub, для этого авторизуемся в консоли.

docker login



И пушим контейнер в Docker Hub.

docker push burmakova/my-nextjs-app



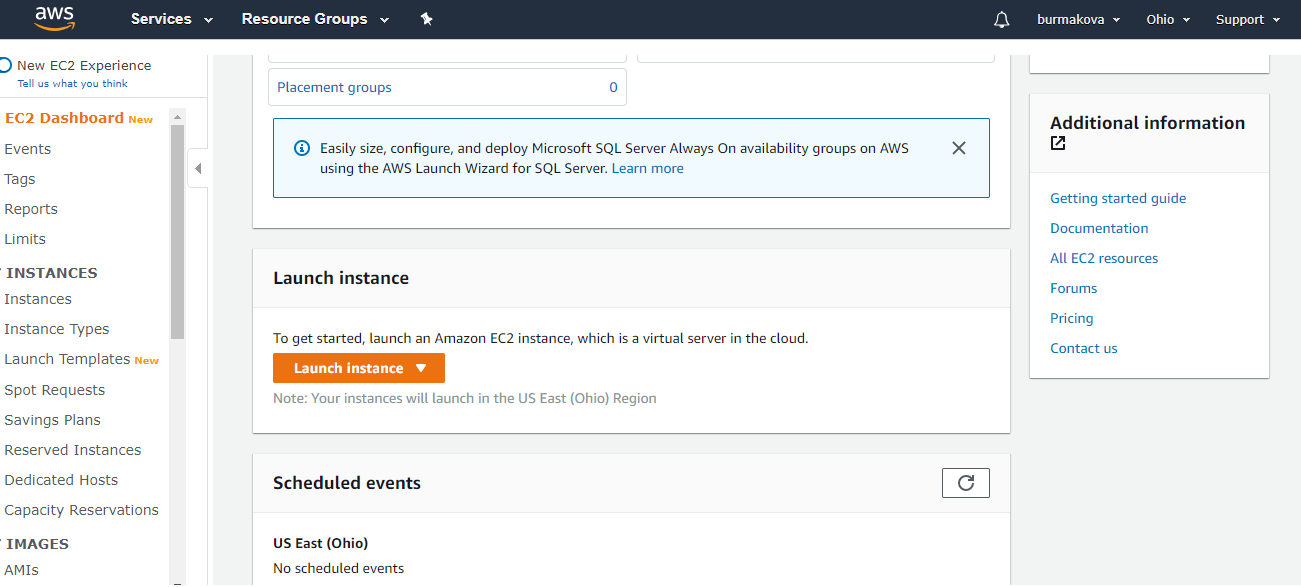
(Далее чтобы скачать данный контейнер на машину с Docker Hub, нужно ввести команду):

docker pull

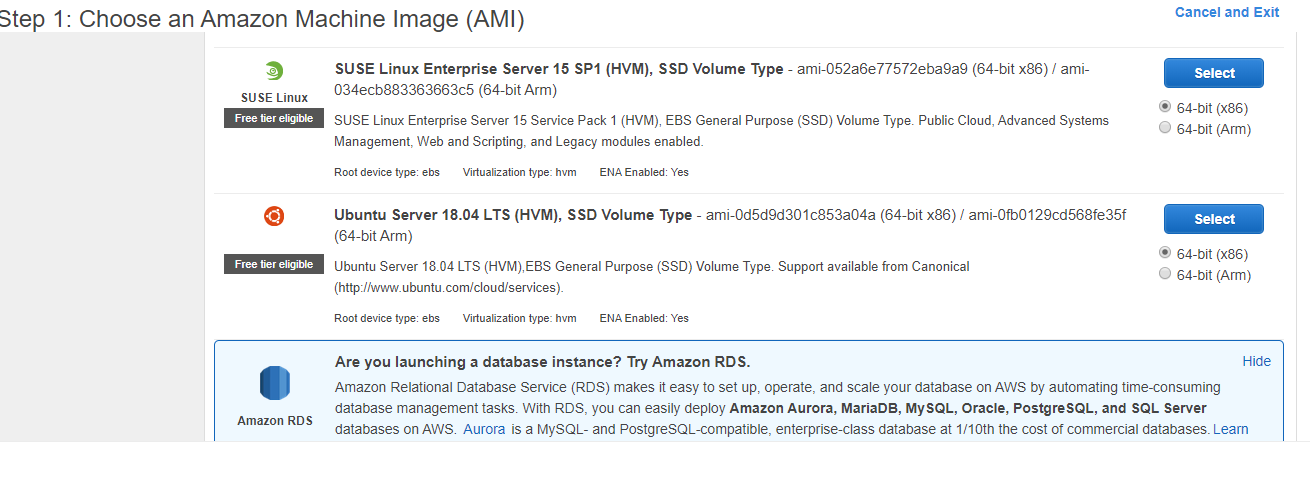
Пример: docker pull burmakova/my-nextjs-app

**3. Создание виртуальной машины Linux на AWS EC2.**

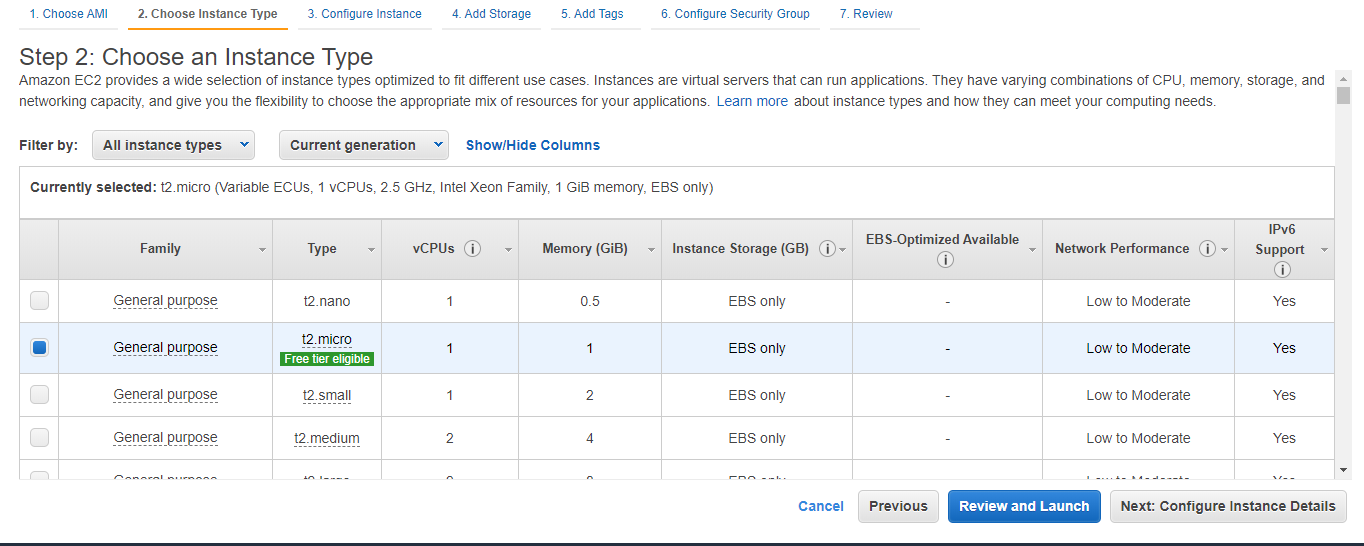
Регистрируемся на AWS и переходим на главную страницу и кликаем на **EC2** и далее выбираем кнопку **Launch instance.**



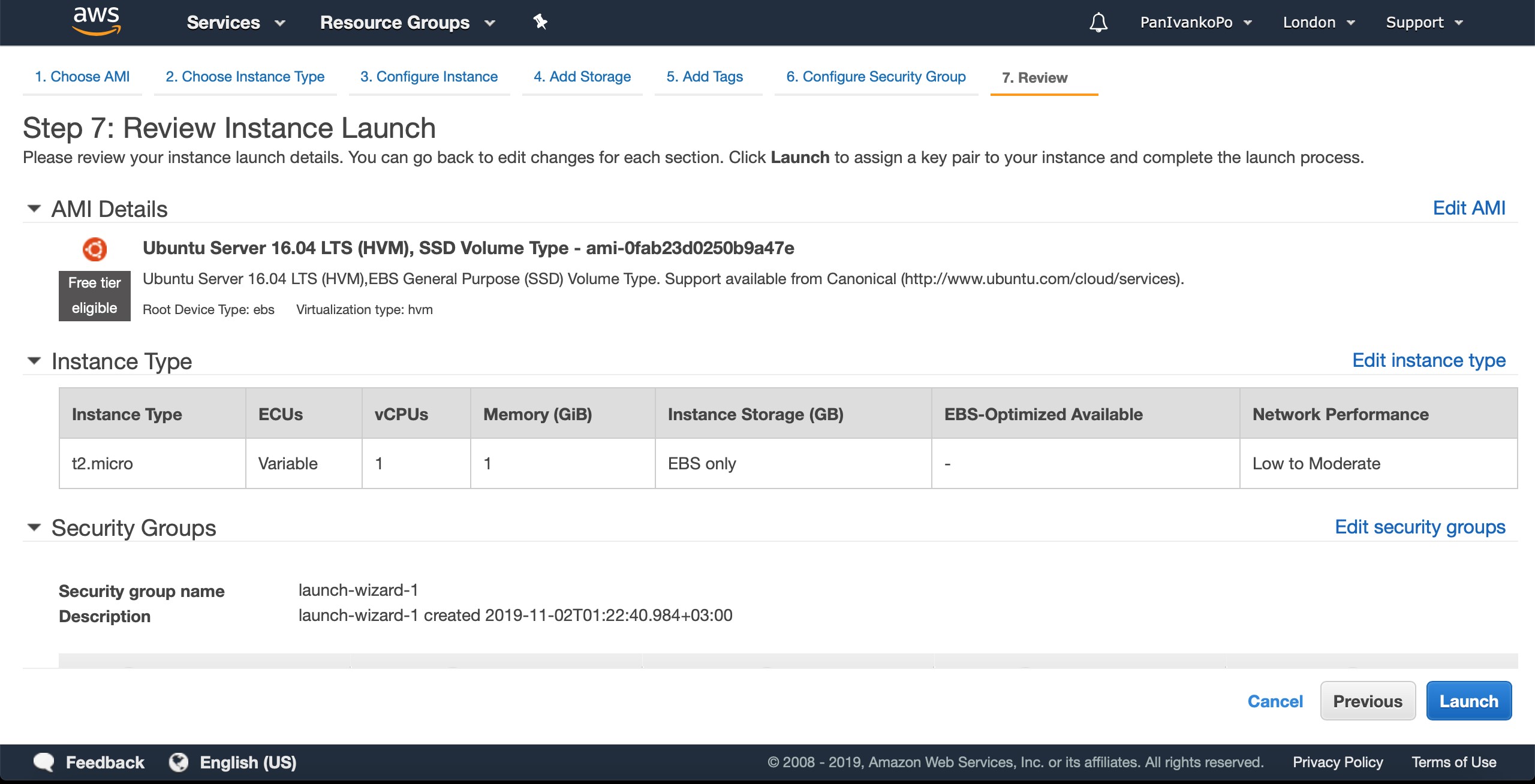
На следующем шаге **в**ыбираем Ubuntu Server 18.04 LTS Нажимаем **Review and Launch.**



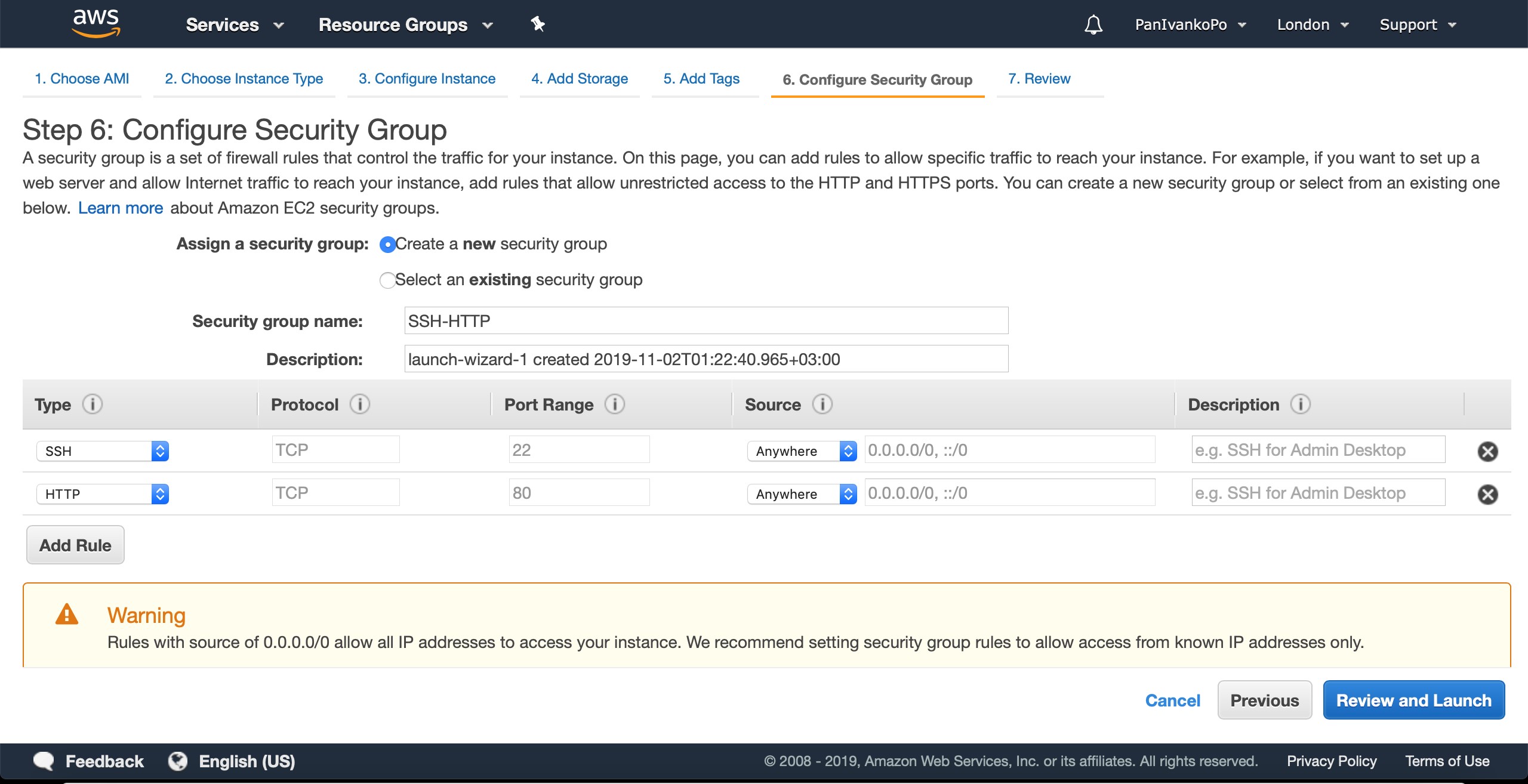
Выбираем Ubuntu Server 16.04 LTS Тут выбираем базовый тип инстанса t2.micro, он доступен бесплатно. Нажимаем Review and Launch



Нужно подредактировать **Security Groups**



Выбираем Edit Security groups



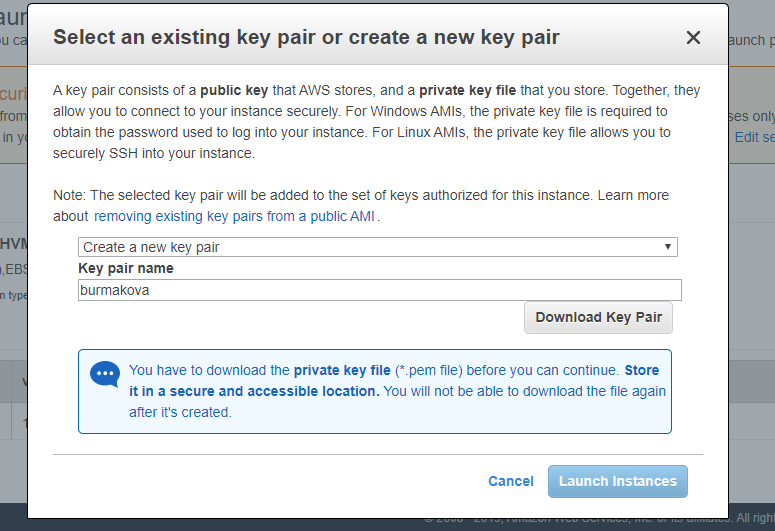
Добавили два правила

Для **http** доступ с любого ip адресса, 80 порт открыть.

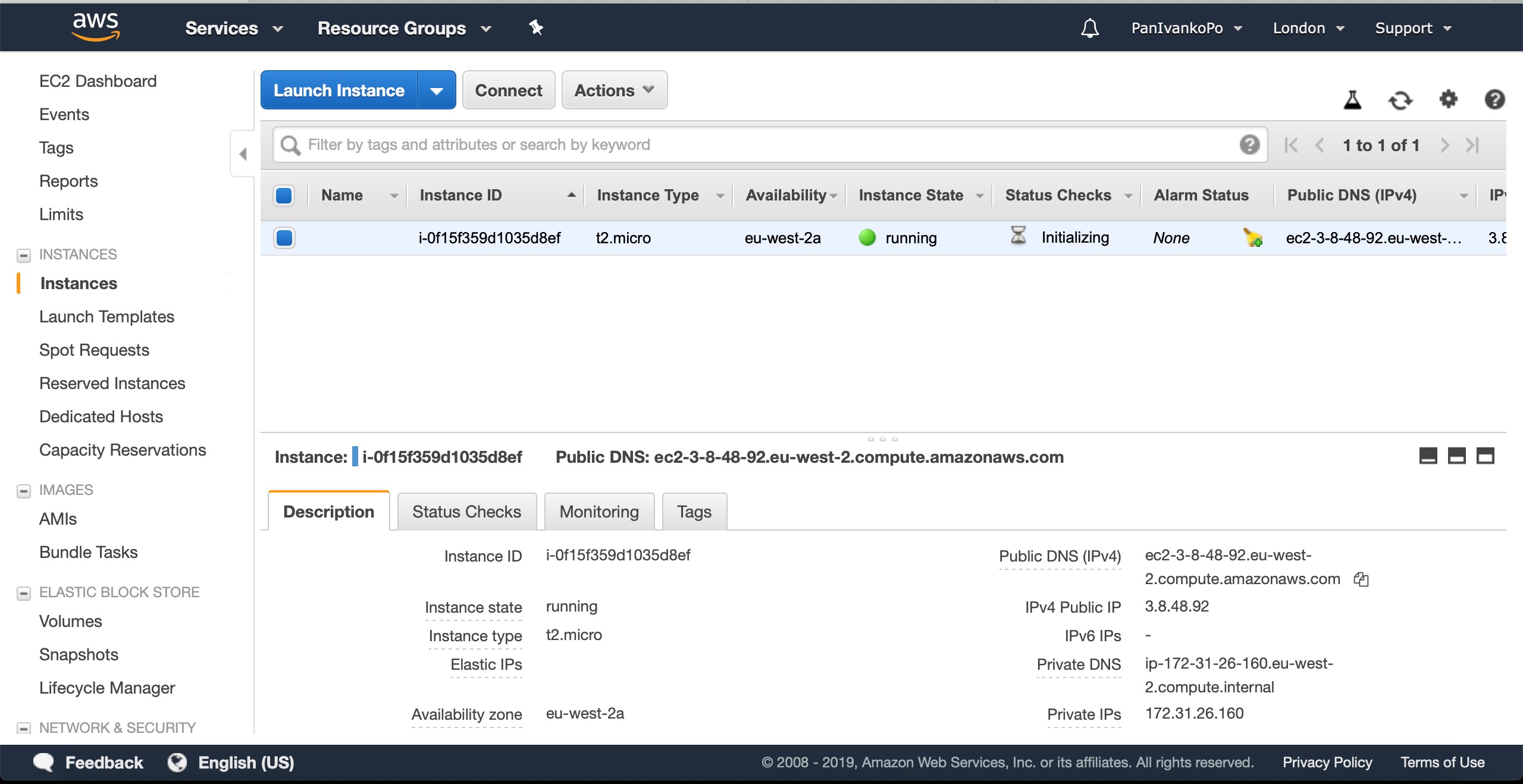
Для **ssh** доступ с любого ip адресса, 22 порт открыть,

Далее Review and Launch

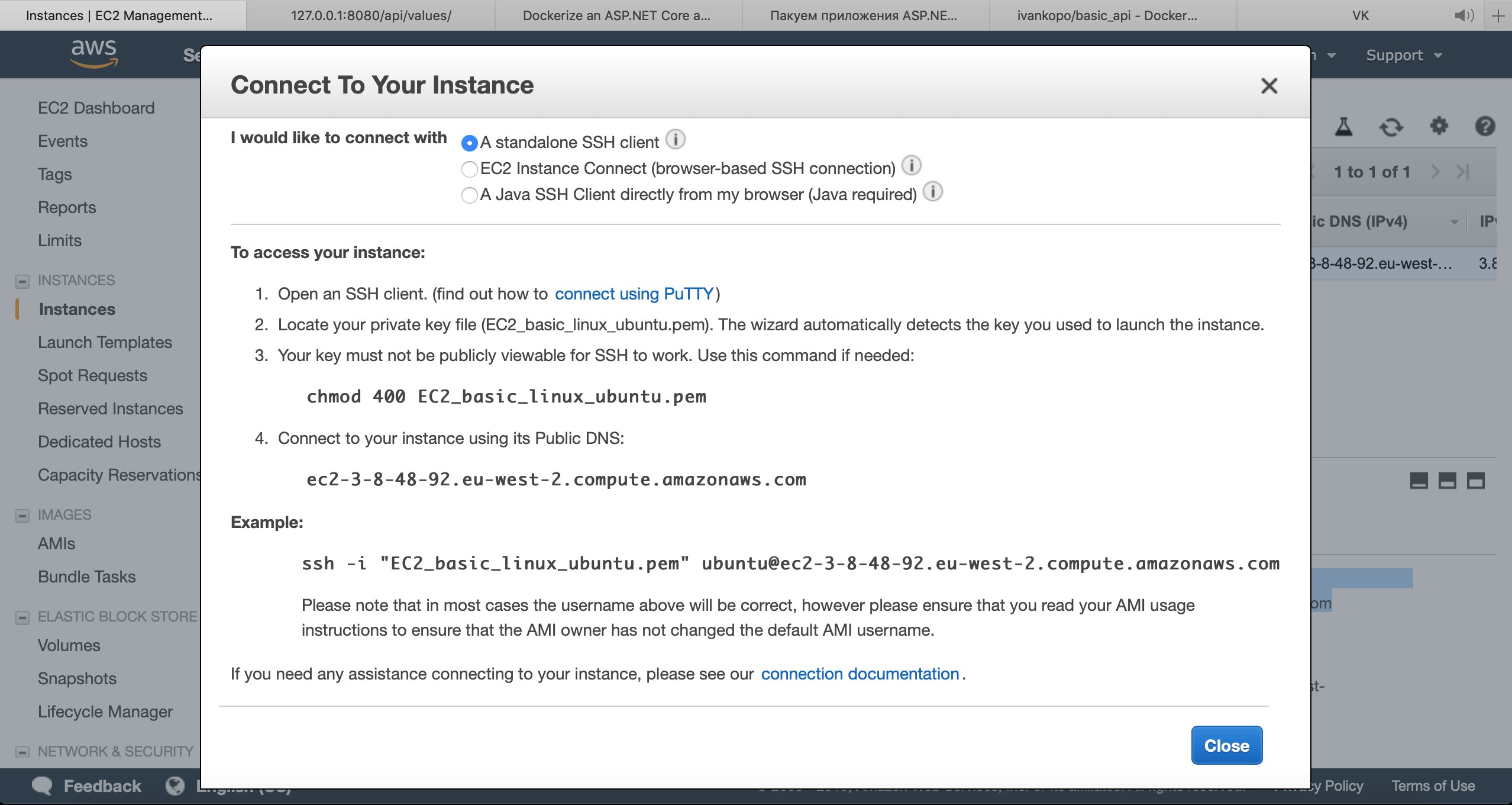
Создаем и скачиваем ключ для подключения по **ssh**



Готово, теперь у нас есть одна виртуальная машина. Детали машины(ip, dns адреса) можно посмотреть во вкладке Description.



Нажмем на кнопку **Connect,** посмотрим, как подключится



Для подключение вводим в консоли

sudo ssh -i "EC2\_basic\_linux\_ubuntu.pem" [ubuntu@ec2-3-8-48-92.euwest-2.compute.amazonaws.com](mailto:ubuntu@ec2-3-8-48-92.euwest-2.compute.amazonaws.com)

Или если вы на Windows воспользуйтесь программой PuTTY

Нас приветствует Linux Ubuntu

Для начала работы, обновим пакеты

sudo apt-get update

Установим docker на ОС

sudo apt install [docker.io](http://docker.io/)

Скачаем наш образ

docker pull burmakova/my-nextjs-app

И запустим его

docker run -it -p 80:3000 burmakova/my-nextjs-app

Проверим работу приложения, например в Postman.

Переходим по адресу

<http://ec2-3-8-48-92.eu-west-2.compute.amazonaws.com/api/values>

(Этот адрес вы увидите в последнем окне амазона «Сonnect to your instance» под пунктом «Сonnect to your instance using its Public DNS»)

