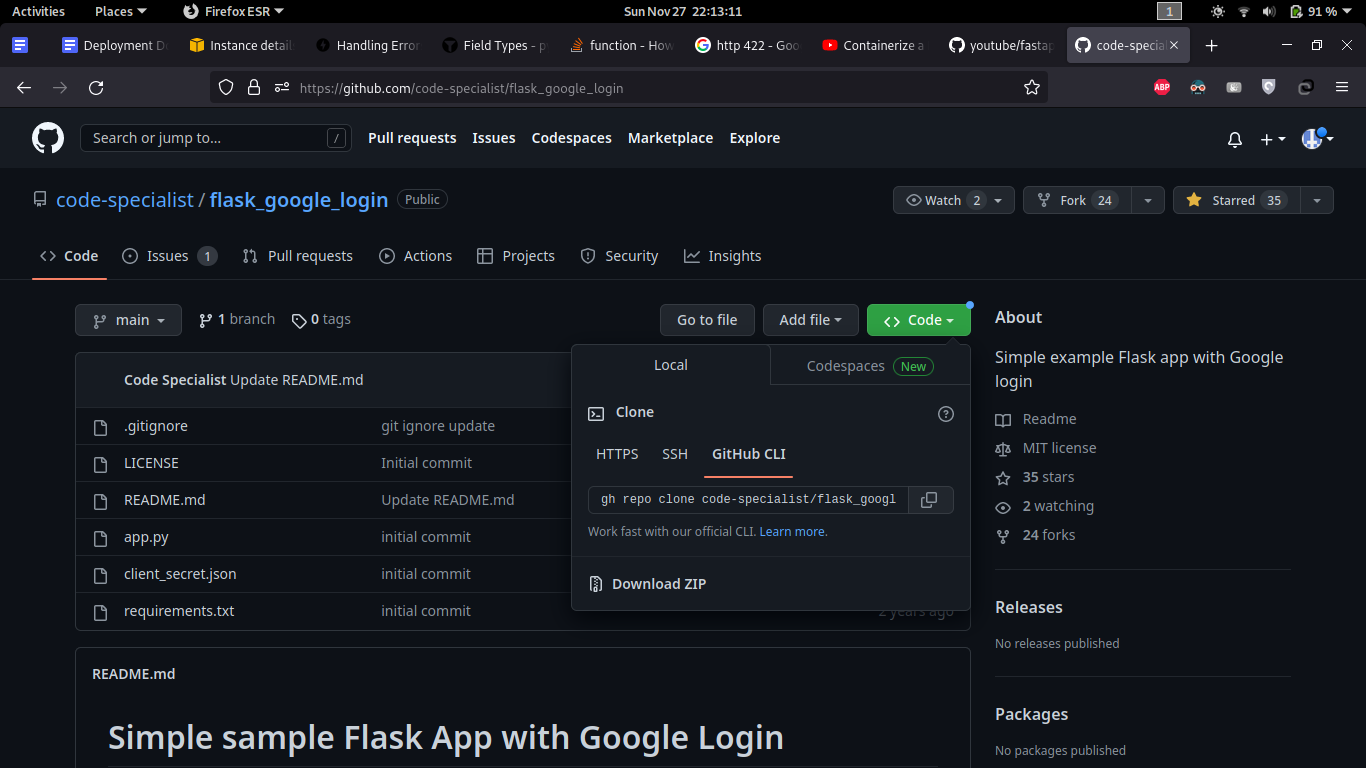
**Deployment Documentation**

1. **GitHub Repo**

Initial step - clone repository to the local machine.

**>> git clone <repo link>**

**# sample link**



To add ssh access in a new server to clone private repo

**>> ssh-keygen -t ed25519 -C "example@gmail.com"**

**>> eval "$(ssh-agent -s)"**

**>> ssh-add ~/.ssh/id\_ed25519**

**>> cat ~/.ssh/id\_ed25519.pub**

Copy this public key and paste it in SSH keys section in Github

1. **Install Dependencies**

Run the - to install dependent packages

**>> sudo apt update**

**>> sudo apt install python3-pip**

**>> sudo apt install uvicorn**

**>> pip3 install -r requirements.txt**

For NVIDIA GPU enabled servers install this additionally

**>> sudo apt install nvidia-driver-525 nvidia-dkms-525**

**>> sudo apt install nvidia-kernel-common-525**

1. **Execution**
   1. **Setting up environment variable**

Create .env file in the same directory, specify the following variables

**Hash Module**

**<API\_KEY>** - secret key generated from openai

**<AUTH\_KEY>** - secret phrase to encode and decode tokens

**Zeroshot Module**

**<AUTH\_KEY>** - secret phrase to encode and decode tokens

**Topic Module**

**<AUTH\_KEY>** - secret phrase to encode and decode tokens

**Keyword Module**

**<AUTH\_KEY>** - secret phrase to encode and decode tokens

* 1. **Config file**

Each module has its own **config\_xxx.ini** file to store parameter values. Changing the values will reflect in module execution.

* 1. **Server**

Run **reg\_login\_xxx, xxx\_server** prefix file to start the server, Once the server is up and running you can use <http://127.0.0.1:8000/><endpoint> to access registration/login endpoint and

[http://127.0.0.1:8001/](http://127.0.0.1:8000/)<endpoint> to access prediction endpoint

**>> uvicorn <reg\_login module filename\_without\_py\_extension>:app –port 8000 –reload**

**>> uvicorn <prediction filename\_without\_py\_extension>:app –port 8001 –reload**

* 1. **Client Script**

You can also access the endpoints using the client script. Run **client\_xxx** prefix file to automate the endpoint access. You can do bulk prediction by feeding the .csv file to the client script. CSV file should have only one column named ‘text’ containing the text for prediction.

**>> Python3 <filename\_with\_extension>**

1. **Dockerize**
   1. **Installation**

Dockerizing the application can help to run it in different environments irrespective of operating system, versions of packages installed. Install the docker using the following commands in linux operating system

**>> sudo apt update -qq**

**>> sudo apt install apt-transport-https ca-certificates curl software-properties-common -qq**

**>> curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -**

**>> sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"**

**>> sudo apt update -qq**

**>> sudo apt install docker-ce**

* 1. **Dockerfile**

This hold the information how the docker image has to be built and how it has to run while executing it in container

**FROM** - base image on which the application has to run

**COPY** - path to copy the files from local system into docker image

**RUN** - run requirements.txt file to install all dependent packages required to run in docker container

**EXPOSE** - port on which our application has to run

**WORDIR** - specify working directory in docker image

**CMD** - command to run the application after starting the container

* 1. **Download necessary files**

Download required files from google drive place it in the folder named - **<files>** in the same directory. You can download these and put it in your Network storage and then attach it as a volume in your docker. These will be required during run time.

Topic module - [**automl**](https://drive.google.com/drive/folders/1ye1F4JKASw2OAhYzXtl6Wo4CErLavcLI?usp=sharing)

Keyword module -[**files**](https://drive.google.com/drive/folders/15StouS_kQv7dIAr8NPBd0B2ZQQzd0lWZ?usp=sharing)

Zeroshot module - [**files**](https://drive.google.com/drive/folders/1A1DBl9zGvQrZgMocQuNfxoQ09AcVZ3uD?usp=sharing)

For redundancy, we put the same files in other location as well. If the above link is not working, please download it fromthe below location:

**Location1**

Topic module - [**automl**](https://drive.google.com/drive/folders/1OEnvC24vnPscO5zFPKgVMp7NxSN12IZT?usp=sharing)

Keyword module -[**files**](https://drive.google.com/drive/folders/1jLyfpLg6UeHPU_BaY08SahLvWQKKO4nA?usp=sharing)

Zeroshot module - [**files**](https://drive.google.com/drive/folders/10RXuDWBPULETUAg8xpqzcu-faTRJJPUg?usp=sharing)

I strongly recommend to keep a copy of it with you.

* 1. **Building docker image**

Use the below command to build the image, it will take some time to build as it have to download all dependent packages mentioned in requirements file

**>> sudo docker build -t <image\_name>:<version\_tag> <source\_directory>**

**for egs: sudo docker build -t zero:1.0 .**

We have to create two containers for each module - register and prediction module. Run both containers in different ports while running client script.

* 1. **Running docker container**

Now we have to load the built docker image into a container to run

**>> sudo docker run -p <port>:<port> <img\_name>:<version\_tag>**

**for egs:**

**sudo docker run -p 8000:8000 zero:1.0**

**sudo docker run -p 8001:8001 registry:1.0**

gpu enabled container:

**>> sudo docker run --rm -p 8000:8000 --gpus all -d zeroshot:1.0**

**>> sudo docker run --rm -p 9000:8000 --gpus all -d zeroshot:1.0**

1. **Deploying in AWS**

Choose the apt configuration available in EC2 instances and transfer the file from local machine via SCP

In my case I transferred model files and built the docker image in EC2 instance. We need to open a custom TCP port 8000 in network configuration to allow access through that channel. Then type following command to enable to traffic

**>> sudo ufw allow 8000/tcp**

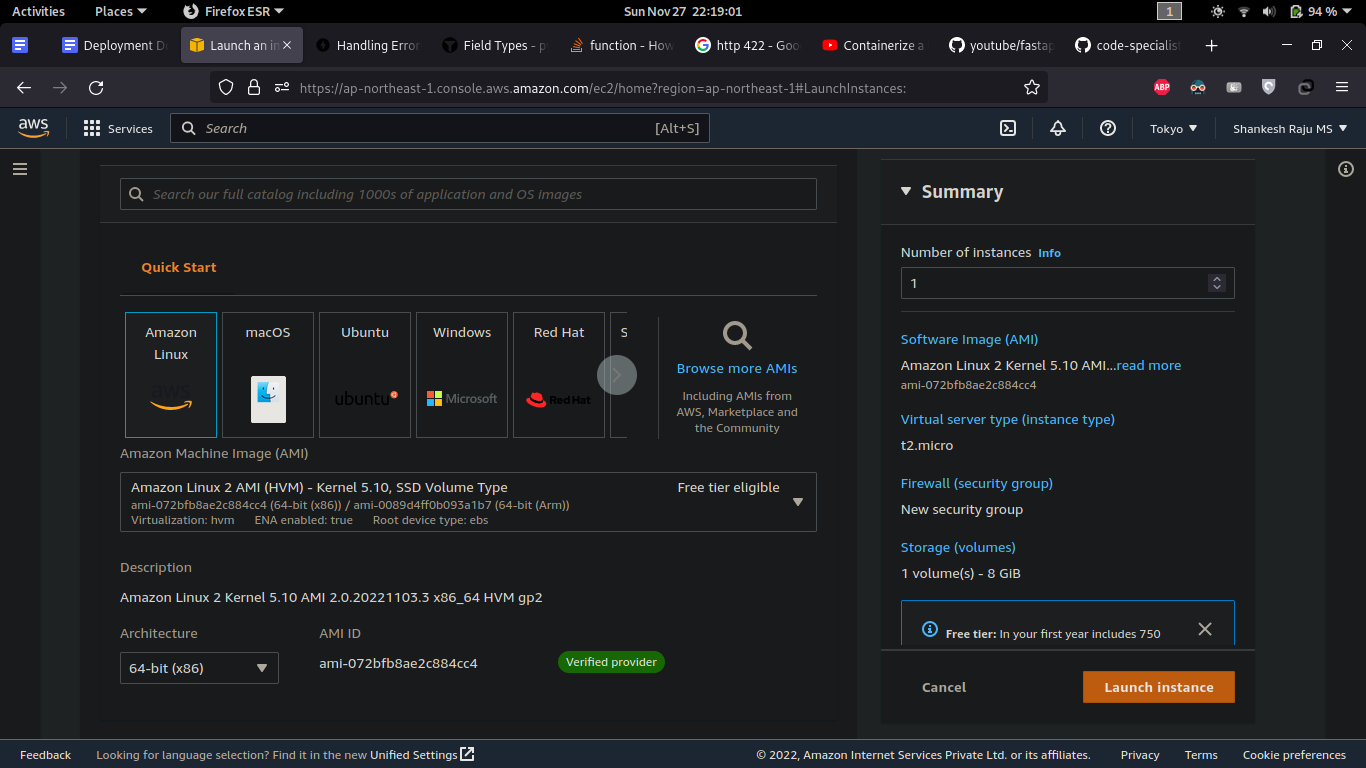
**>> sudo ufw allow 8001/tcp**

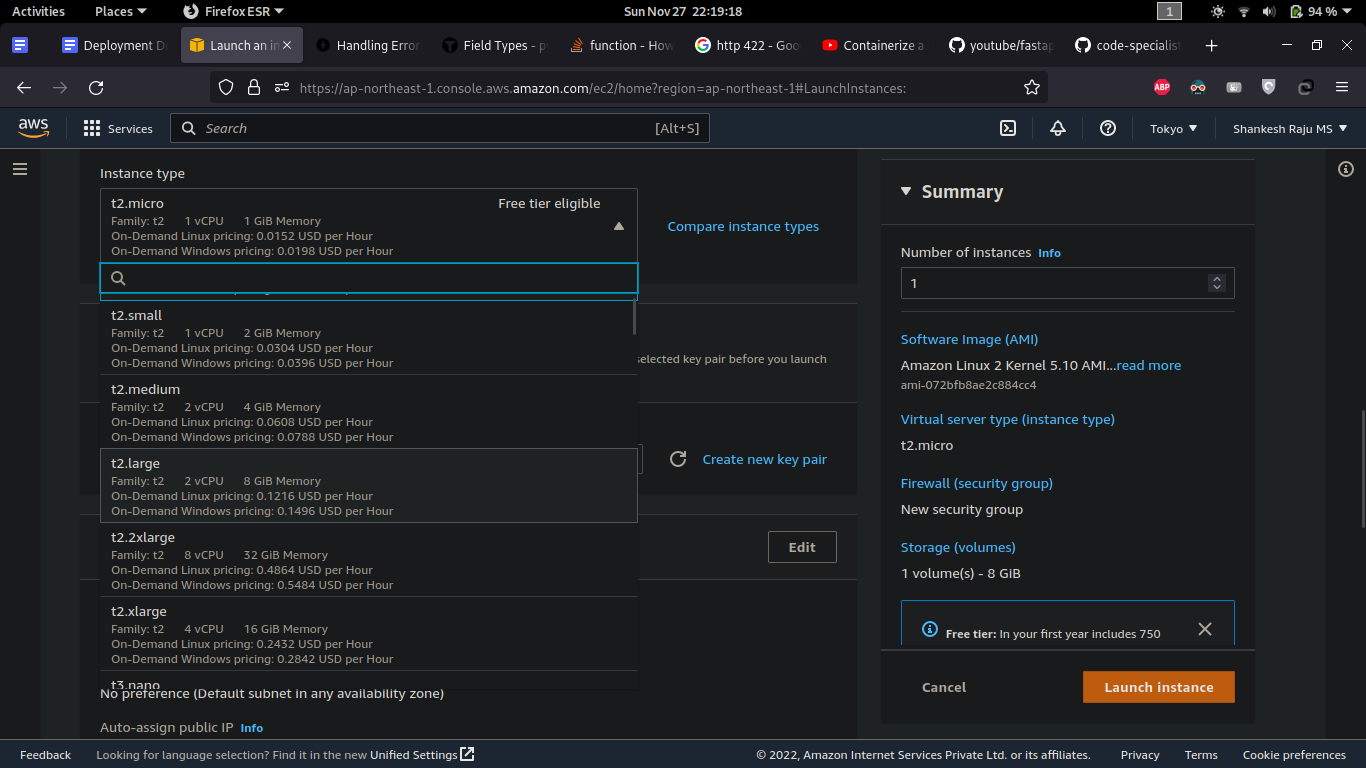
**To transfer files from local system to AWS instance**

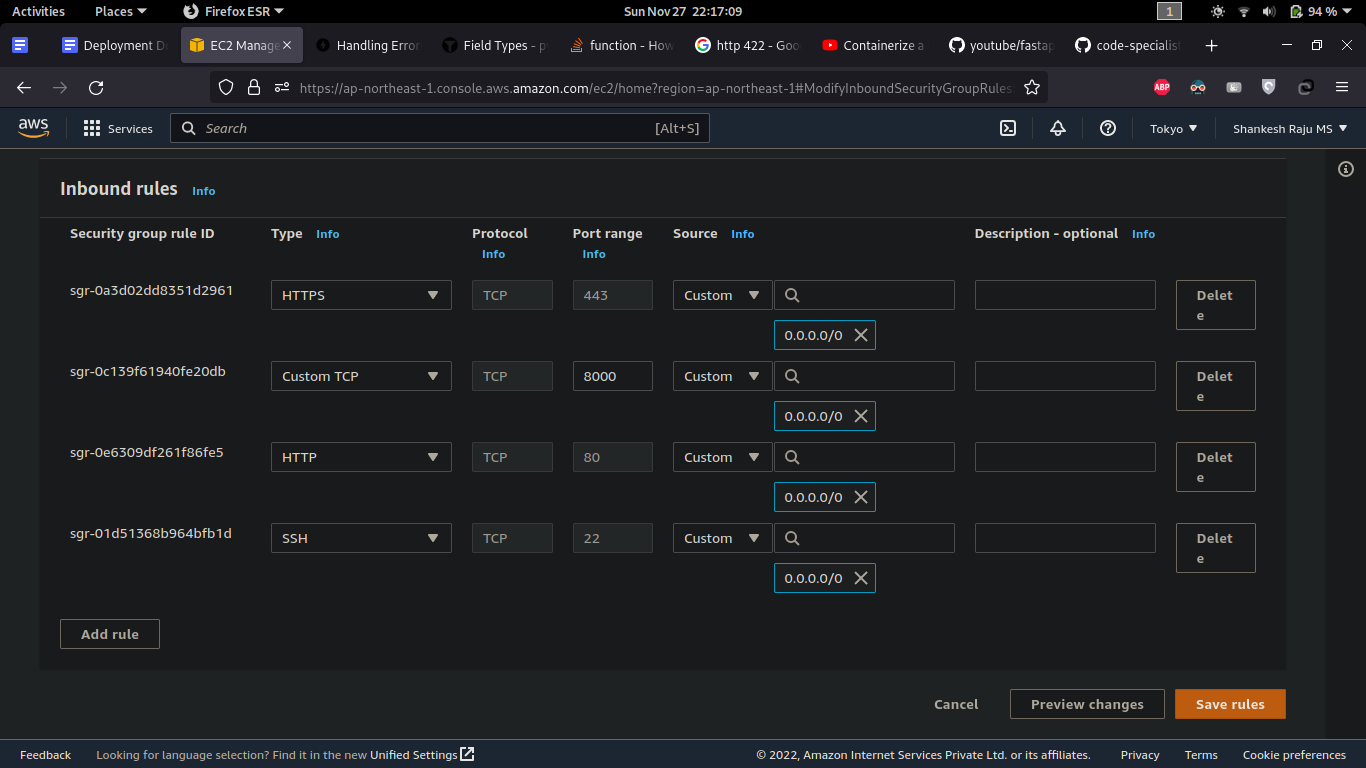
**scp -i <private key.pem> <localpath\_tocopy> username@ec2address:<destination\_path>**

Egs:

**scp -i <token.pem> <source file location>** [**ubuntu@ec2-xx-xx-xx-xx.ap-south-1.compute.amazonaws.com**](mailto:ubuntu@ec2-xx-xx-xx-xx.ap-south-1.compute.amazonaws.com)**:<destination path>**







Once enabling the port run the docker image in a container and access it through the link

**http://<EC2\_instance\_public\_ip>:8000/<endpoints> - prediction module**

**http://<EC2\_instance\_public\_ip>:8001/<endpoints> - register, login module**

For the zeroshot module use port 8000 for prediction and 8001 for register, because load balancer is configured in port 8000 enabling port 7000,9000 to run three more instances of the prediction module with shared resources.