CIS 325 Final Exam Part-1
Submitted by: Anurag Yadav

GitHub
[Click to Access Repo]

Bank Telemarketing Classification Using Artificial Neural Network

Introduction

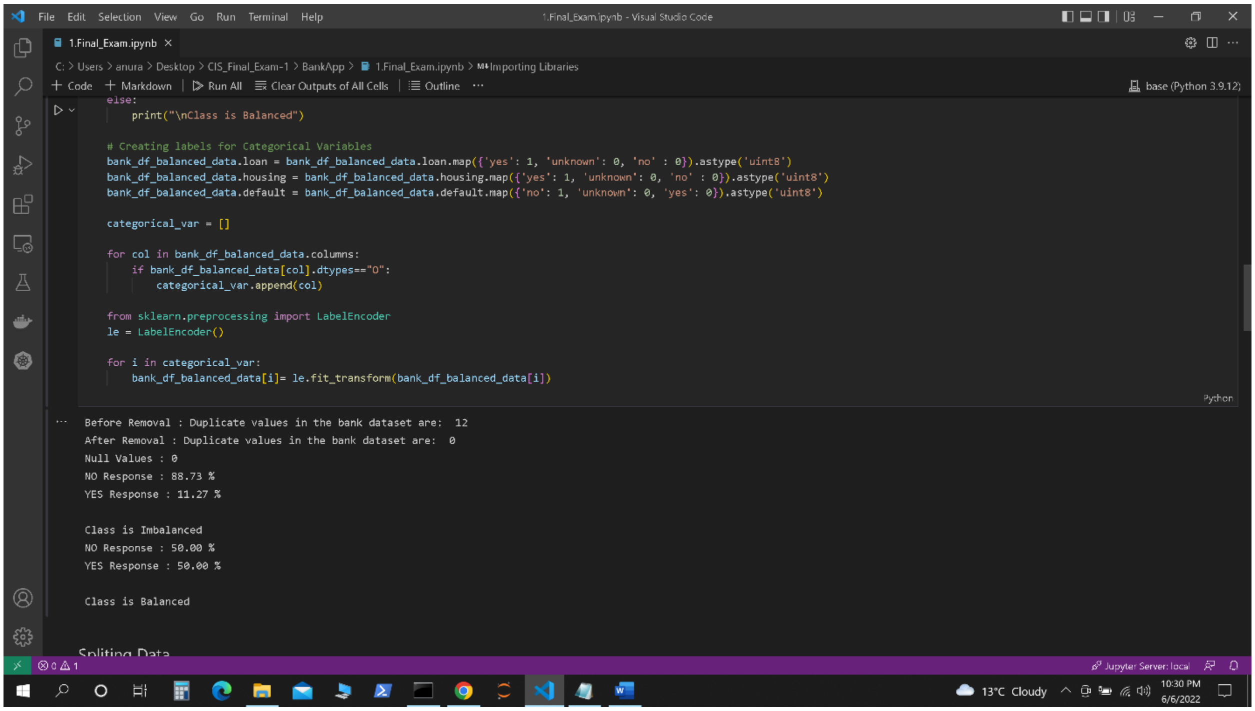
The data is related with direct marketing campaigns of a banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be ('yes') or not ('no') subscribed.

Bank-Telemarketing.csv with all examples (41188) and 20 inputs, ordered by date (from May 2008 to November 2010). The classification goal is to predict if the client will subscribe (yes/no) a term deposit (variable y)

Download Link: https://archive.ics.uci.edu/ml/datasets/bank+marketing

Pre-Processing

The data that I used for this project was highly imbalanced and was dominated by NO responses, while only 11.27% responses were YES. This means that the data need to be processed first. Using **sklearn.utils.resample**, the data have been balanced.


File Edit Selection View Go Run Terminal Help
1.Final_Exam.ipynb - Visual Studio Code
1.Final_Exam.ipynb
C: > Users > anura > Desktop > CIS_Final_Exam-1 > BankApp > 1.Final_Exam.ipynb > M↓Importing Libraries
+ Code + Markdown | Run All Clear Outputs of All Cells | Outline ···
base (Python 3.9.12)
Python
Spliting Data
⊗ 0 ⚠ 1
Jupyter Server: local
13°C Cloudy
10:30 PM
6/6/2022

```
else:
    print("\nClass is Balanced")

# Creating labels for Categorical Variables
bank_df_balanced_data.loan = bank_df_balanced_data.loan.map({'yes': 1, 'unknown': 0, 'no' : 0}).astype('uint8')
bank_df_balanced_data.housing = bank_df_balanced_data.housing.map({'yes': 1, 'unknown': 0, 'no' : 0}).astype('uint8')
bank_df_balanced_data.default = bank_df_balanced_data.default.map({'no': 1, 'unknown': 0, 'yes': 0}).astype('uint8')

categorical_var = []

for col in bank_df_balanced_data.columns:
    if bank_df_balanced_data[col].dtypes=="O":
        categorical_var.append(col)

from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

for i in categorical_var:
    bank_df_balanced_data[i]= le.fit_transform(bank_df_balanced_data[i])
```

```
Before Removal : Duplicate values in the bank dataset are:  12
After Removal : Duplicate values in the bank dataset are:  0
Null Values : 0
NO Response : 88.73 %
YES Response : 11.27 %

Class is Imbalanced
NO Response : 50.00 %
YES Response : 50.00 %

Class is Balanced
```

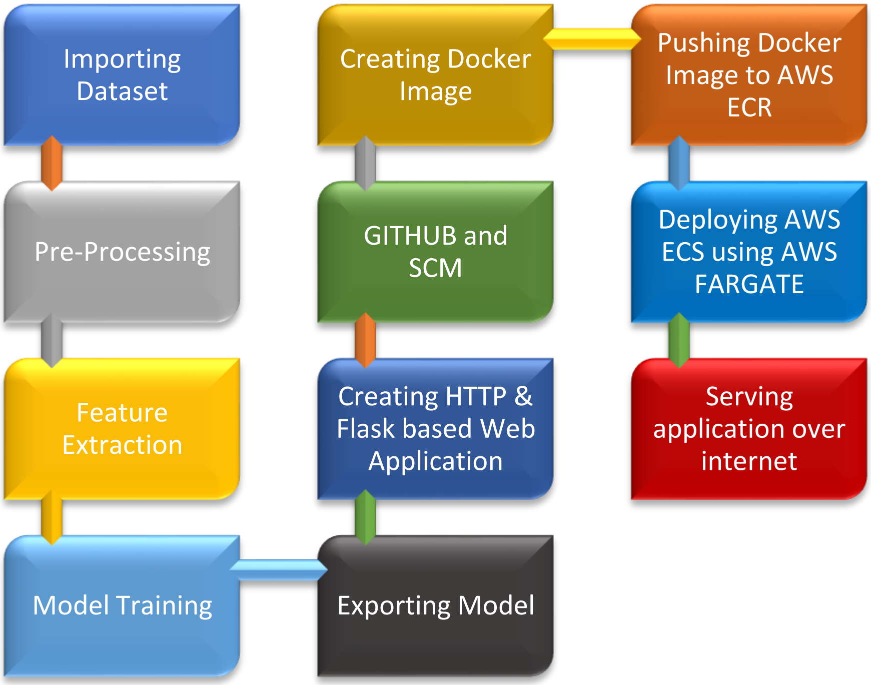
CIS 325 Final Exam Part-1

Submitted by: Anurag Yadav

GitHub

[Click to Access Repo]

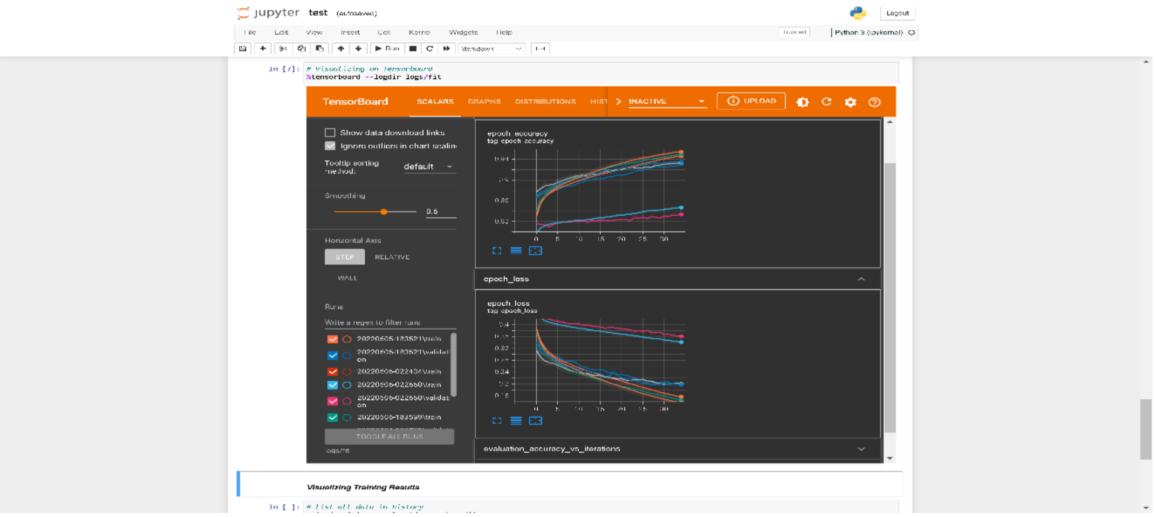
Project Workflow


Importing Dataset
Pre-Processing
Feature Extraction
Model Training
Exporting Model
Creating HTTP & Flask based Web Application
GITHUB and SCM
Creating Docker Image
Pushing Docker Image to AWS ECR
Deploying AWS ECS using AWS FARGATE
Serving application over internet

TRAINING MODEL

The Python programming language was used to create the model. In addition, the packages Matplotlib, Keras, and NumPy were utilized for system implementation. Keras provides built-in functions such as activation functions, optimizers, layers, etc. TensorFlow was also used as the system's back-end.

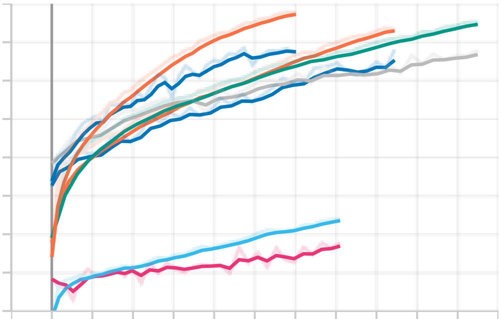
Furthermore, Tensor board was used for model tracking.

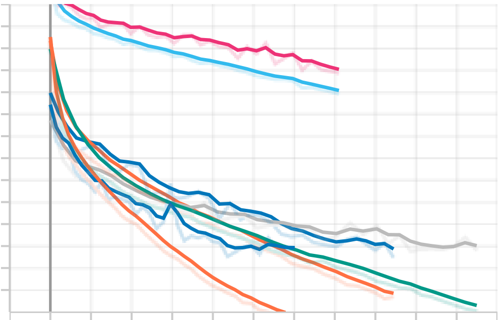


```
In [7]: # Visualizing on Tensorboard
        %tensorboard --logdir logs/fit
```

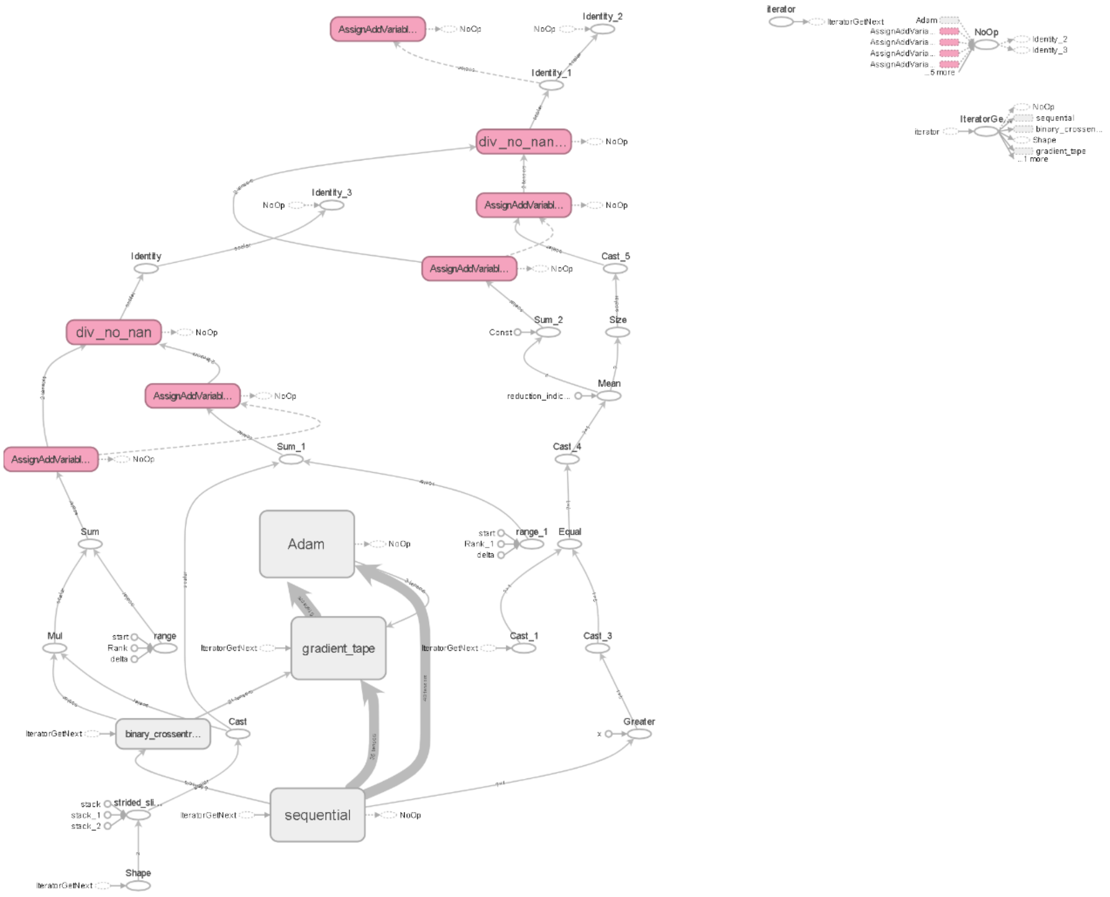
Visualizing Training Results

```
In [ ]: # List all data in history
```



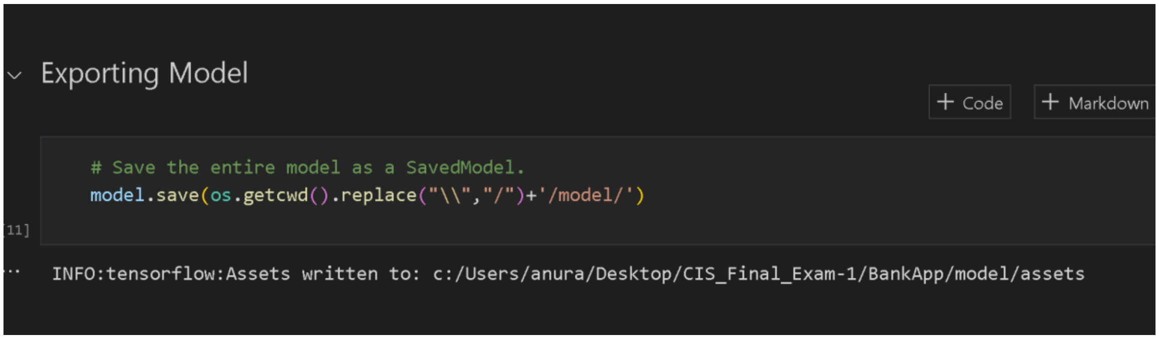


Conceptual Graph Generated from Tensor Board



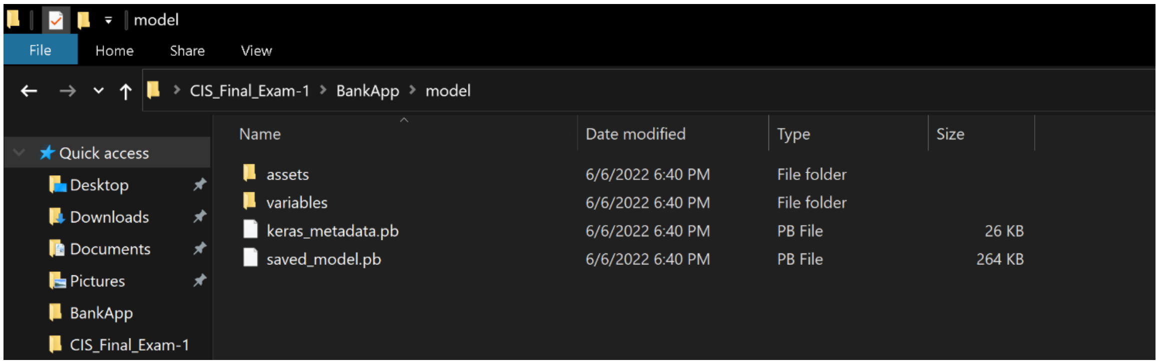
EXPORTING MODEL

You must first export your trained models in TensorFlow Saved_Model format before deploying them to AI Platform Prediction and using them to deliver predictions. TensorFlow's recommended format for exporting models is a Saved_Model, and it is required for deploying trained TensorFlow models on AI Platform Prediction. When you export your trained model as a Saved_Model, you save your training graph, including its assets, variables, and metadata, in a format that AI Platform Prediction can ingest and restore for predictions.



```
# Save the entire model as a SavedModel.
model.save(os.getcwd().replace("\\","/")+'/model/')
```

```
INFO:tensorflow:Assets written to: c:/Users/anura/Desktop/CIS_Final_Exam-1/BankApp/model/assets
```



Name	Date modified	Type	Size
assets	6/6/2022 6:40 PM	File folder	
variables	6/6/2022 6:40 PM	File folder	
keras_metadata.pb	6/6/2022 6:40 PM	PB File	26 KB
saved_model.pb	6/6/2022 6:40 PM	PB File	264 KB

CREATING HTTP & FLASK BASED WEB APPLICATION

In order to create a Gender Detection web application, we have created two different files:

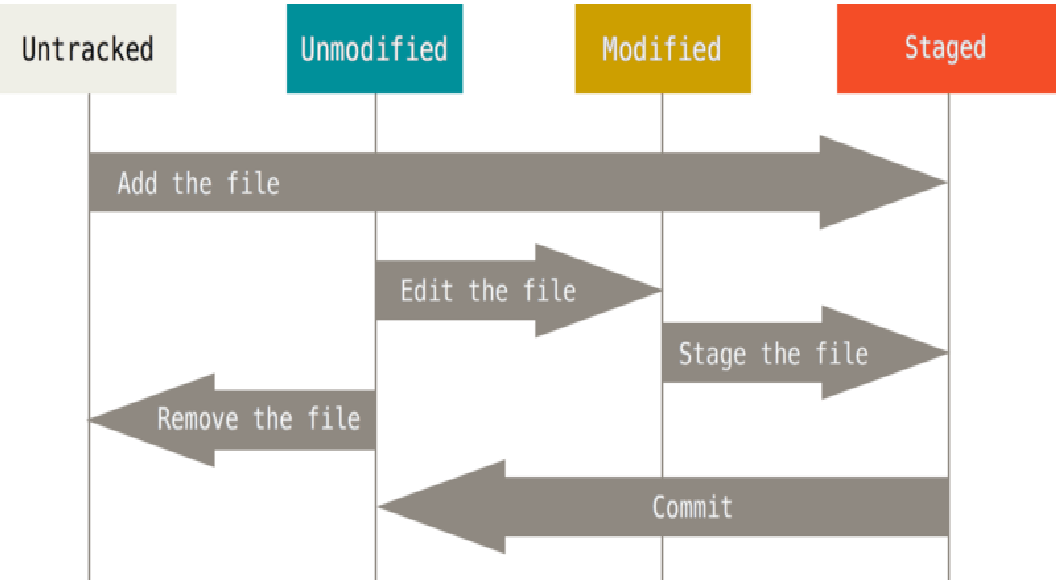
- Flask for the back-end engine - app.py
- HTML for the front-end - /templates/index.html & /output.html

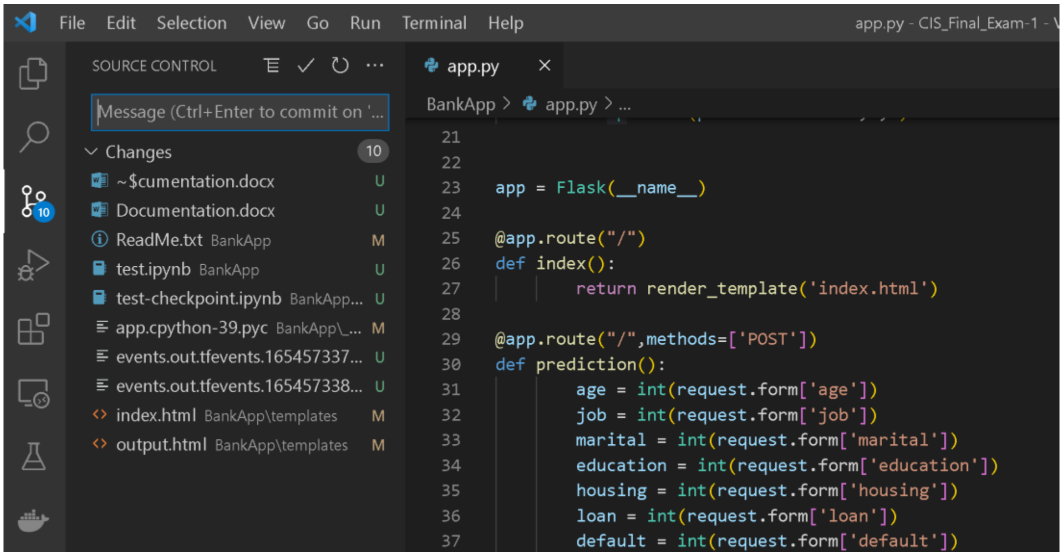
The front-end HTML acts as a medium to interact with and accept input from the user, who then receives predictions from the model. First, the POST request is received from the HTML. Then, when a request is received, the ML model is loaded, and the input file is pre-processed according to the steps described in the Flask back-end engine. Finally, the model generates the prediction, which is subsequently returned to the user via the *render_template(output.html)* function.

GITHUB & SCM

Source code management (SCM) is a technique for tracking changes to a source code repository. SCM maintains a running history of modifications to a code base and aids in dispute resolution when integrating updates from various contributors.

Visual Studio Code can be a valuable tool if you're working with a remote repository. You only need to add your remote repository to VS to be able to control the changes. Another option is to execute git commands from the terminal. Both of them can be seen in the snapshot below.





Message (Ctrl+Enter to commit on '...

Changes 10

- ~$cumentation.docx U
- Documentation.docx U
- ReadMe.txt BankApp M
- test.ipynb BankApp U
- test-checkpoint.ipynb BankApp... U
- app.cpython-39.pyc BankApp_... M
- events.out.tfevents.165457337... U
- events.out.tfevents.165457338... U
- index.html BankApp\templates M
- output.html BankApp\templates M

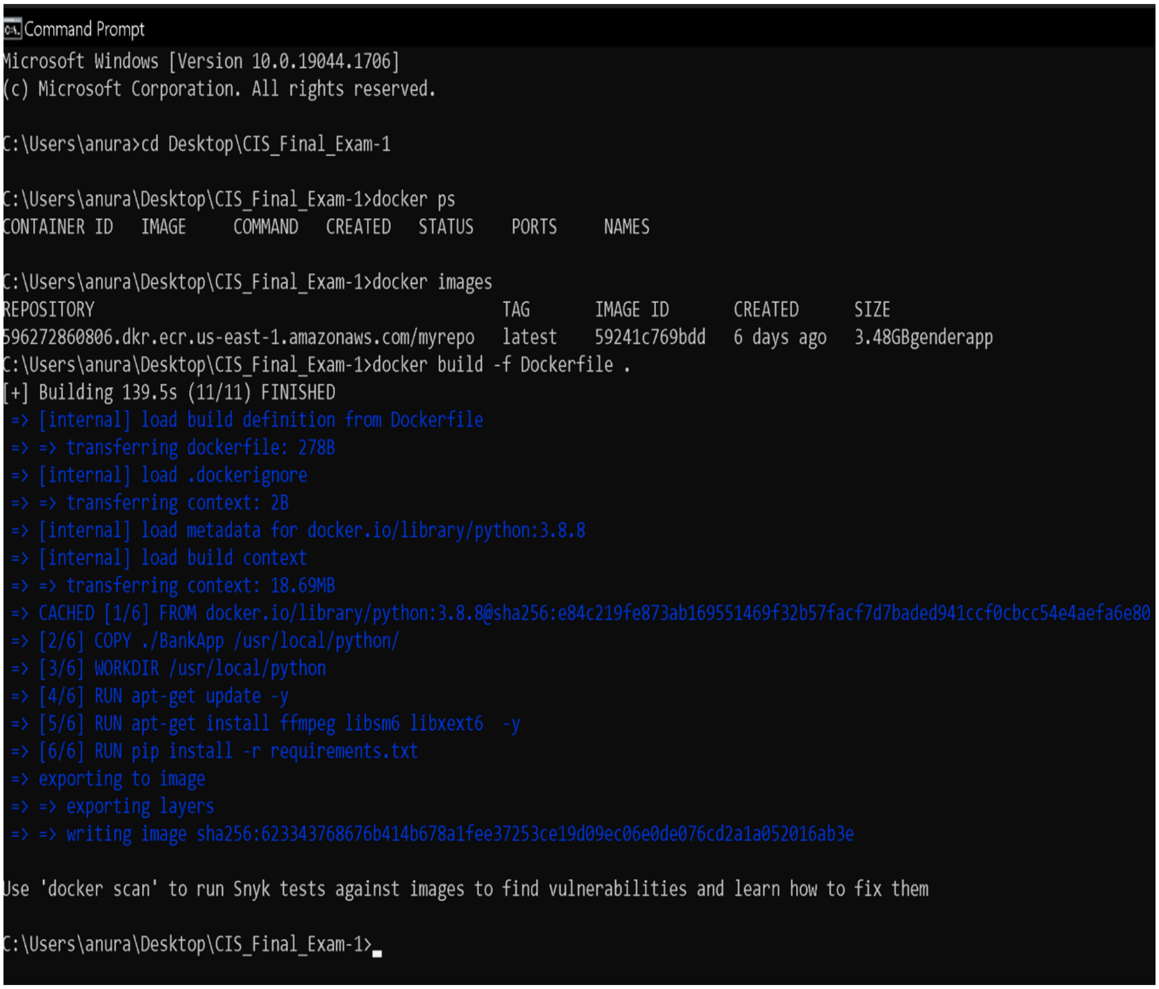
```


app = Flask(__name__)

@app.route("/")
def index():
        return render_template('index.html')

@app.route("/",methods=['POST'])
def prediction():
        age = int(request.form['age'])
        job = int(request.form['job'])
        marital = int(request.form['marital'])
        education = int(request.form['education'])
        housing = int(request.form['housing'])
        loan = int(request.form['loan'])
        default = int(request.form['default'])
```

CREATING A CUSTOM IMAGE FROM DOCKER

When we want to automate and operate a custom application, the easiest option to deploy your containerized application is to create your own docker image with the desired configuration. This custom configuration may be passed using the Dockerfile, and a container can then be created using this image.

Once the image is ready, it can be used to deploy containers in Docker. In order to test our application on Docker, we have created a container using *docker run -d -p 5000:5000 --name BankApplication <IMAGE_NAME: VERSION>* command.

The application can be seen running at *localhost:5000* in the last image.

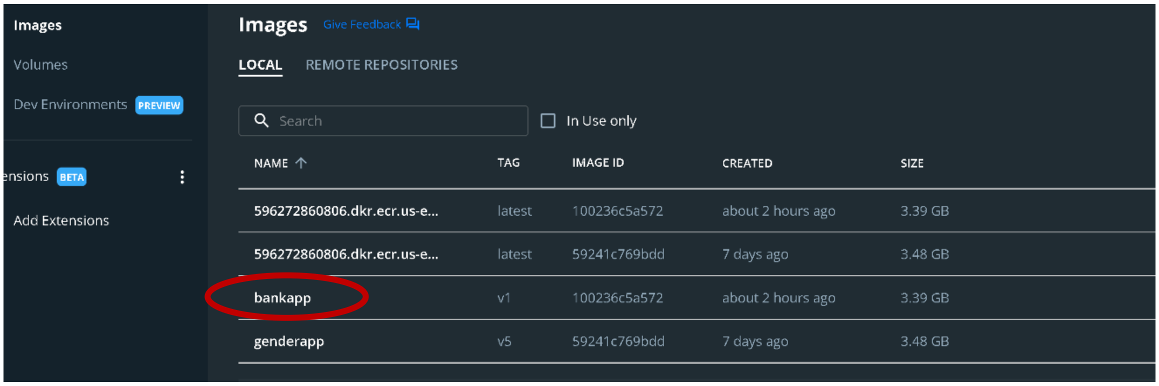


```
Command Prompt
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\anura>cd Desktop\CIS_Final_Exam-1

C:\Users\anura\Desktop\CIS_Final_Exam-1>docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

C:\Users\anura\Desktop\CIS_Final_Exam-1>docker images
REPOSITORY                                      TAG       IMAGE ID       CREATED      SIZE
596272860806.dkr.ecr.us-east-1.amazonaws.com/myrepo   latest    59241c769bdd   6 days ago   3.48Gbgenderapp
C:\Users\anura\Desktop\CIS_Final_Exam-1>docker build -f Dockerfile .
[+] Building 139.5s (11/11) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 278B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/python:3.8.8
 => [internal] load build context
 => => transferring context: 18.69MB
 => CACHED [1/6] FROM docker.io/library/python:3.8.8@sha256:e84c219fe873ab169551469f32b57facf7d7baded941ccf0cbcc54e4aefa6e80
 => [2/6] COPY ./BankApp /usr/local/python/
 => [3/6] WORKDIR /usr/local/python
 => [4/6] RUN apt-get update -y
 => [5/6] RUN apt-get install ffmpeg libsm6 libxext6  -y
 => [6/6] RUN pip install -r requirements.txt
 => exporting to image
 => => exporting layers
 => => writing image sha256:623343768676b414b678a1fee37253ce19d09ec06e0de076cd2a1a052016ab3e

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

C:\Users\anura\Desktop\CIS_Final_Exam-1>
```

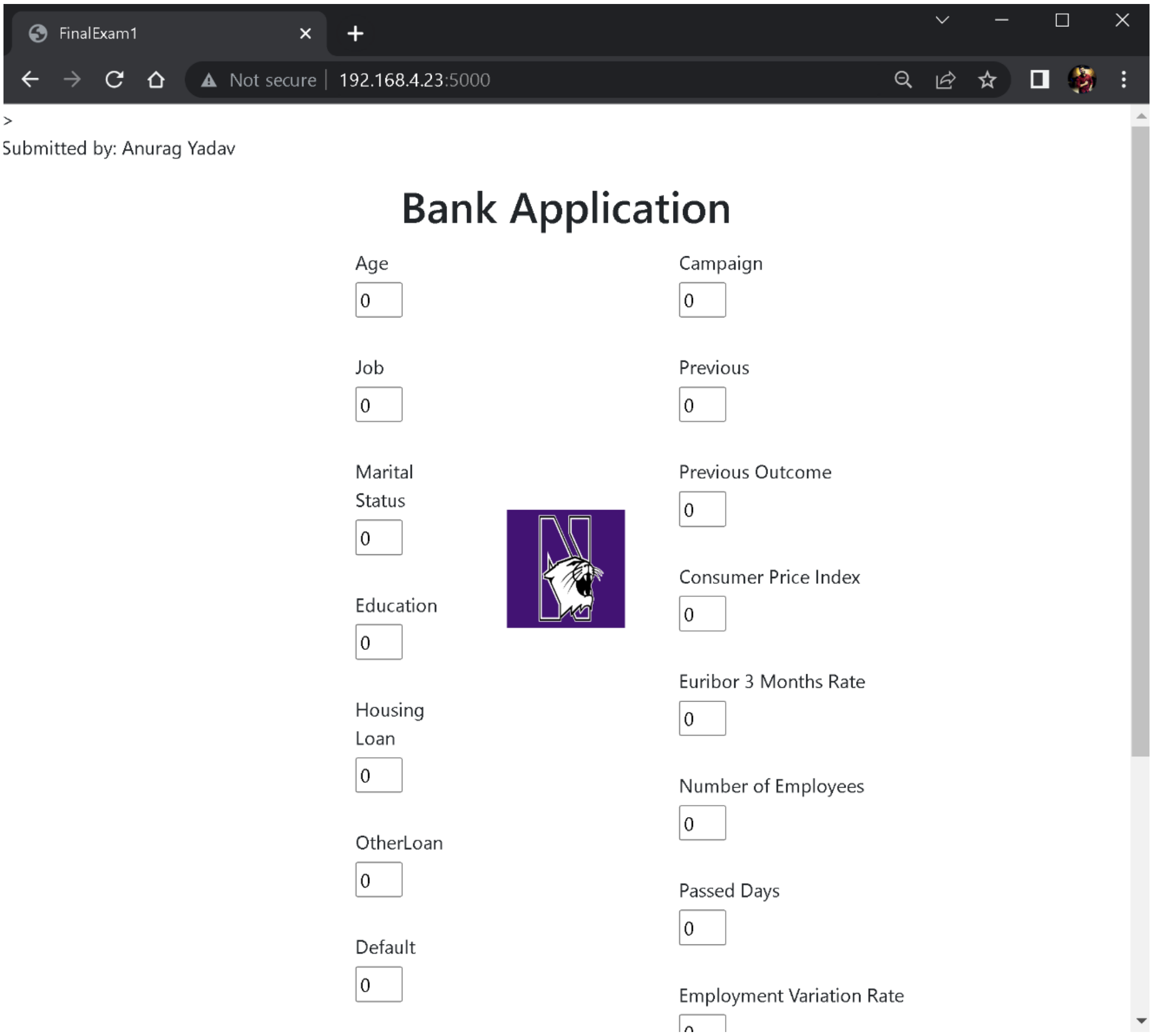
CIS 325 Final Exam Part-1

Submitted by: Anurag Yadav

GitHub
[Click to Access Repo]



NAME	TAG	IMAGE ID	CREATED	SIZE
596272860806.dkr.ecr.us-e...	latest	100236c5a572	about 2 hours ago	3.39 GB
596272860806.dkr.ecr.us-e...	latest	59241c769bdd	7 days ago	3.48 GB
bankapp	v1	100236c5a572	about 2 hours ago	3.39 GB
genderapp	v5	59241c769bdd	7 days ago	3.48 GB



>
Submitted by: Anurag Yadav

Bank Application

Age: 0

Job: 0

Marital Status: 0

Education: 0

Housing Loan: 0

OtherLoan: 0

Default: 0

Campaign: 0

Previous: 0

Previous Outcome: 0

Consumer Price Index: 0

Euribor 3 Months Rate: 0

Number of Employees: 0

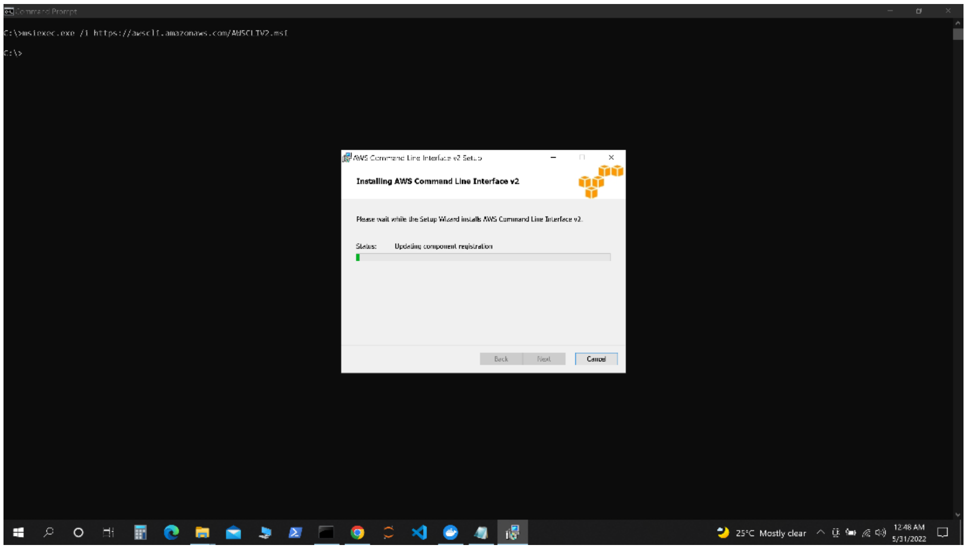
Passed Days: 0

Employment Variation Rate: 0

Access this Application over Internet: http://44.203.104.134:5000/

PUSHING LOCAL DOCKER IMAGE TO AWS ELASTIC CONTAINER REGISTRY ECR

To push a local docker image to AWS, we must first configure AWS CLI for the first time. This can be performed by using the instructions listed in the ReadMe file. Once the CLI is ready, we must use the terminal to create a repository and push the local docker image to AWS ECR.

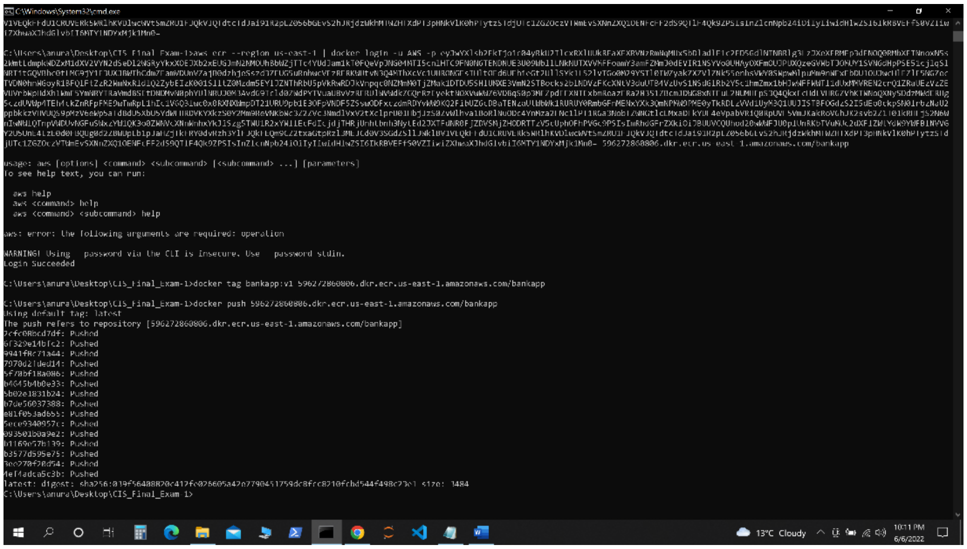


```
C:\>msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi

C:\>
```

Installing AWS Command Line Interface v2

Please wait while the Setup Wizard installs AWS Command Line Interface v2.

Status: Updating component registration



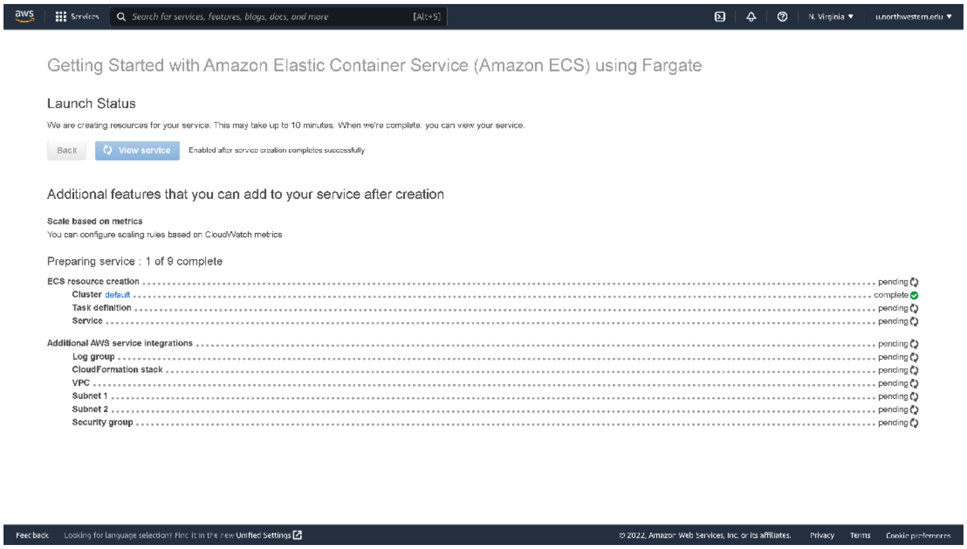
```
V1VEQkFFdU1CRUVERk5WRlhKVDlwcWVtSmZRU1FJQkVJQldtcldJai91R2pLZ056bGEvS2hJRjdzWkhMTNZHTXdPT3pHNkVlK0hPTytzSTdjUTc1ZGZOczVTNmEvSXNnZXQ1OENFcFF2dS9QTlF4Qk9ZPSIsInZlcnNpb24iOiIyIiwidHlwZSI6IkRBVEFfS0VZIiw
iZXhwaXJhdGlvbiI6MTY1NDYxMjk1Mn0-

C:\Users\anura\Desktop\CIS Final Exam-1>aws ecr --region us-east-1 | docker login -u AWS -p eyJwYXlsb2FkTjo1c04yRkU2TlcxRXlHUkRFaXFXRVNzRmNqMUx5bDladlF1c2FDSGdlNTNBRlg3LzJXeXFRMFp3dFNOQ0RMbXFTNnoxN5s
2WmtLdmpkNDZxM1dXV2VVYN2dSeDl2NGRyYkxXOEJXb2xEUGJmN2NMOUhBbWZjTTc4YUdJam1kT0FQeVpJNG04MTI5cnlHTC9FN0NGTENDNUE3U09Wbl1LNkNUTXVVMFFoamY3amFZMmJ0dEVIR1NSYVo0UHAyOXFmOUJPUXQzeGVHbTJOMUY1SVNGdHpPSE51cjlqS1
NRT1tGQVBhc0tlMG9jY1FBUXJBWThCdm7FamVDUnVZajR0dzhjcSszd3FUG5uRnhwcVFzRFRKWUtvN3Q4MThXcVc1UHR0NGFsTHltdFFd6UFhfcGt2UllSYk1LS2lvTGo0M29YSTl0TW7yak7X7VlZNk55enhsVWY0SNpwMlpuMm9nNFxFbDU1OUJwcHlFZ1lFSNG7oc
TVDN0hnWGoyR1BFQlFtZzR2WmNxR1dlQ2ZybEIzK001S1ltZ0Mzdm5EY1JZNThRbU5pVkRwRDJkVnpqc0NZMmM0TjZMak1DTDU5SH1UMXE3VmN2STBocks2b1NDVzFKcXNtV3duUTB4VzUvS1NSdG1Rb2Y5c1hmZmx1bHJwNFFWWT11dUxMMVREN2crQ1ZRaUEzVzZE
VUVrbWpHdXh1WmF5YmNRYTRaVmdBSFtDNDMvN0phYU1NRUJ0M3AvdG9Tcld0ZNdPYTVuaUBVVzRFRUlWVWdkZGQrRzEyektNOXVwWWZ6VDBqS0p3MFZpdFFXNTExbmRoazFRa2H3S1ZBcmJDWG8xNTFuL2NUMHFpS3Q4QkxFcHdLVHRGZVhKTWNoQXNySDdzMWdERUg
5czdUVWp4TEh4ckZnRFpFMEOwTmRpL1hIc1VGQ3lwc0x0RXNXWmpDT21URU9pb1E3OFpVNDF5ZSswODFxczdmRDYvWN0KQ2FibUZGcDBaTENzaUlWbWk1RURUY0RmbGFnMENNxYXk3QmNPNW0PME0yTkRDLzVVd1UyM3Q1UUJISTBFOGdzS2I5dEo0ckpSN01rbzNaU2
ppbkkzVlVUQS9pMzV6eWp5aTdBdU5XbU5YdWIRDVKYXkzS0Y2Mm9ReVNKbWc3Z2ZVcJNmdlVxV2tXclprU01HbjJzS0ZvWlhva1BoRlNuODc4YnMza2INc1lPT1RGa3NobIZWNGtlcLMxaDIkYUI4eVpabVR1QURpUVI5VmJKakRoVGhJK2svb2Z1T01kRHIjS2N6W
nIwNWlQTnpVWDUvNGFuSWxzYW1QK3o0ZWNVcXNnWnhxYkJiSzg5TWU1R2xYW1lEcFdIcjdjTHRjUnhLbnh3NyLEd2JXTFdNR0FjZDVSMjZHODRTTZV5cUphOFhPVGc9PSIsImRhdGFrZXkiOiJBUUVCQUhod20wWNFJU0plUnRKbTVuMUc2dXFIZWtYdW9YWFB1NVVG
Y2U5UnE4LzE0d0FBQUg0d2ZBWUpLb1pJaHZjTkFRY0dvRzh3YlFJQkFEQm9CZ2txaGtpRzl3MEJCd0V3SGdZSllJWklBV1VEQkFFdU1CRUVERk5WRlhKVDlwcWVtSmZRU1FJQkVJQldtcldJai91R2pLZ056bGEvS2hJRjdzWkhMTNZHTXdPT3pHNkVlK0hPTytzSTd
jUTc1ZGZOczVTNmEvSXNnZXQ1OENFcFF2dS9QTlF4Qk9ZPSIsInZlcnNpb24iOiIyIiwidHlwZSI6IkRBVEFfS0VZIiwiZXhwaXJhdGlvbiI6MTY1NDYxMjk1Mn0- 596272860806.dkr.ecr.us-east-1.amazonaws.com/bankapp

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help

aws: error: the following arguments are required: operation

WARNING! Using   password via the CLI is insecure. Use   password stdin.
Login Succeeded

C:\Users\anura\Desktop\CIS_Final_Exam-1>docker tag bankapp:v1 596272860806.dkr.ecr.us-east-1.amazonaws.com/bankapp

C:\Users\anura\Desktop\CIS_Final_Exam-1>docker push 596272860806.dkr.ecr.us-east-1.amazonaws.com/bankapp
Using default tag: latest
The push refers to repository [596272860806.dkr.ecr.us-east-1.amazonaws.com/bankapp]
2cfc08bcd7df: Pushed
6f329e14bfc2: Pushed
9941f8c71a44: Pushed
7970d21ded14: Pushed
5f70bf18a086: Pushed
b4645b4b0e33: Pushed
5b02e1831b24: Pushed
b7de56037388: Pushed
e81f053ad655: Pushed
5ece9340957c: Pushed
093501b0a9e2: Pushed
b1169e57b139: Pushed
b3577d595e75: Pushed
3ee270f20d54: Pushed
4ef4adca5c3b: Pushed
latest: digest: sha256:019f56408820c412fe026605a42e7790451759dc8fcc8210fcbd544f498c23e1 size: 3484
C:\Users\anura\Desktop\CIS_Final_Exam-1>
```

DEPLOYING ELASTIC CONTAINER SERVICE ECS USING AWS FARGATE

AWS FARGATE is a technology that you can use with Amazon ECS to run containers without having to manage servers or clusters of Amazon EC2 instances. With AWS FARGATE, you no longer have to provision, configure, or scale clusters of virtual machines to run containers.

This removes the need to choose server types, decide when to scale your clusters or optimize cluster packing.

When running your tasks and services with the FARGATE launch type, you package your application in containers, specify the CPU and memory requirements, define networking and IAM policies, and launch the application. Each FARGATE task has its own isolation boundary and does not share the underlying kernel, CPU resources, memory resources, or elastic network interface with another task.



Getting Started with Amazon Elastic Container Service (Amazon ECS) using Fargate

Launch Status

We are creating resources for your service. This may take up to 10 minutes. When we're complete, you can view your service.

Back | View service | Enabled after service creation completes successfully

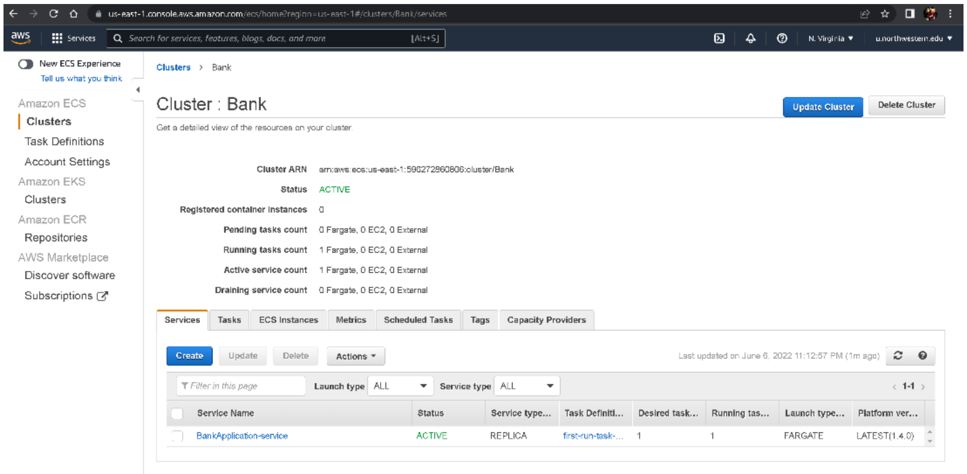
Additional features that you can add to your service after creation

Scale based on metrics
You can configure scaling rules based on CloudWatch metrics

Preparing service : 1 of 9 complete

```
ECS resource creation ........................ pending
    Cluster default .......................... complete
    Task definition .......................... pending
    Service .................................. pending
Additional AWS service integrations .......... pending
    Log group ................................ pending
    CloudFormation stack ..................... pending
    VPC ...................................... pending
    Subnet 1 ................................. pending
    Subnet 2 ................................. pending
    Security group ........................... pending
```

Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
u.northwestern.edu
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/clusters/Bank/services

Clusters > Bank

Cluster : Bank

Get a detailed view of the resources on your cluster

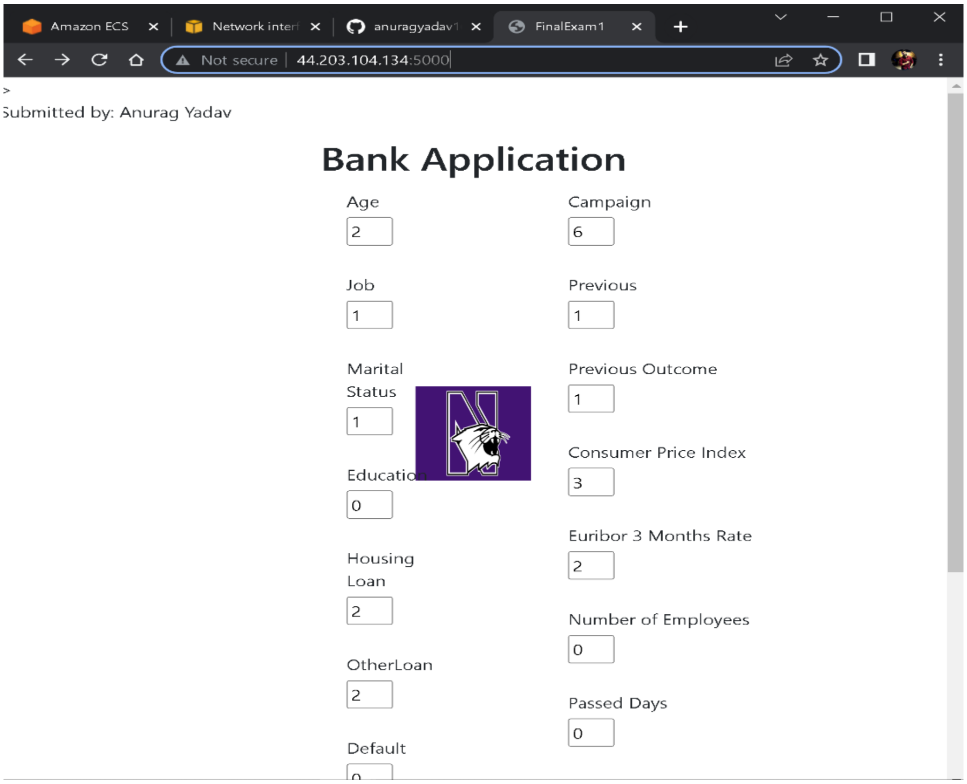
Cluster ARN	arn:aws:ecs:us-east-1:598272960806:cluster/Bank
Status	ACTIVE
Registered container instances	0
Pending tasks count	0 Fargate, 0 EC2, 0 External
Running tasks count	1 Fargate, 0 EC2, 0 External
Active service count	1 Fargate, 0 EC2, 0 External
Draining service count	0 Fargate, 0 EC2, 0 External

Last updated on June 6, 2022 11:12:57 PM (1m ago)

Launch type: ALL; Service type: ALL

Service Name	Status	Service type...	Task Definiti...	Desired task...	Running tas...	Launch type...	Platform ver...
BankApplication-service	ACTIVE	REPLICA	first-run-task-...	1	1	FARGATE	LATEST(1.4.0)

RESULTS:



>
Submitted by: Anurag Yadav

Bank Application

Age
2

Job
1

Marital Status
1

Education
0

Housing Loan
2

OtherLoan
2

Default
0

Campaign
6

Previous
1

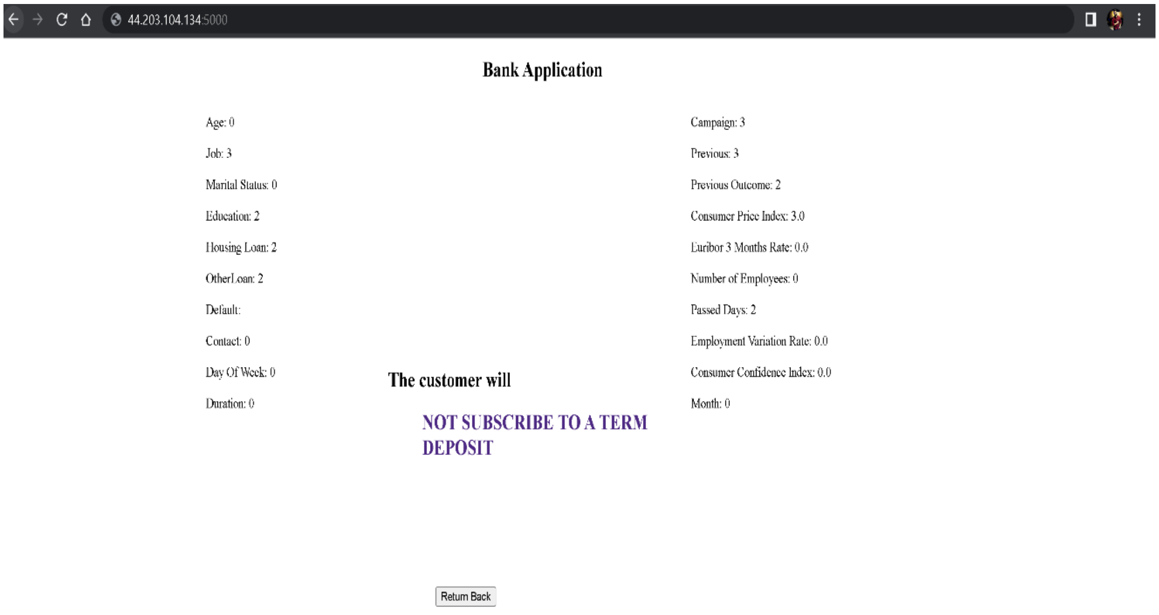
Previous Outcome
1

Consumer Price Index
3

Euribor 3 Months Rate
2

Number of Employees
0

Passed Days
0



Bank Application

Age: 0

Job: 3

Marital Status: 0

Education: 2

Housing Loan: 2

OtherLoan: 2

Default:

Contact: 0

Day Of Week: 0

Duration: 0

Campaign: 3

Previous: 3

Previous Outcome: 2

Consumer Price Index: 3.0

Euribor 3 Months Rate: 0.0

Number of Employees: 0

Passed Days: 2

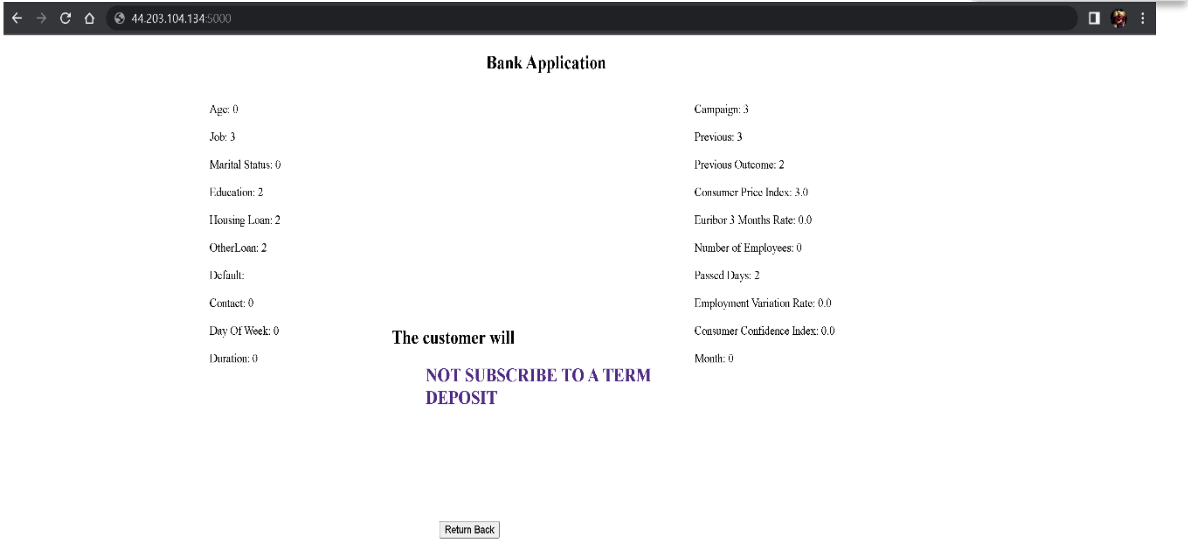
Employment Variation Rate: 0.0

Consumer Confidence Index: 0.0

Month: 0

The customer will

NOT SUBSCRIBE TO A TERM DEPOSIT



44.203.104.134:5000

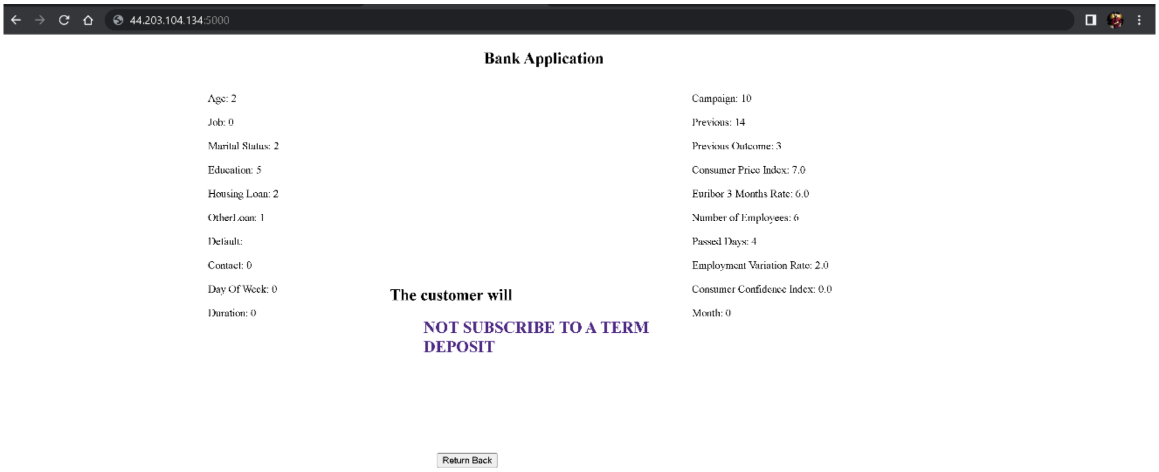
Bank Application

Age: 0
Job: 3
Marital Status: 0
Education: 2
Housing Loan: 2
OtherLoan: 2
Default:
Contact: 0
Day Of Week: 0
Duration: 0

The customer will

NOT SUBSCRIBE TO A TERM DEPOSIT

Campaign: 3
Previous: 3
Previous Outcome: 2
Consumer Price Index: 3.0
Euribor 3 Months Rate: 0.0
Number of Employees: 0
Passed Days: 2
Employment Variation Rate: 0.0
Consumer Confidence Index: 0.0
Month: 0



44.203.104.134:5000

Bank Application

Age: 2
Job: 0
Marital Status: 2
Education: 5
Housing Loan: 2
OtherLoan: 1
Default:
Contact: 0
Day Of Week: 0
Duration: 0

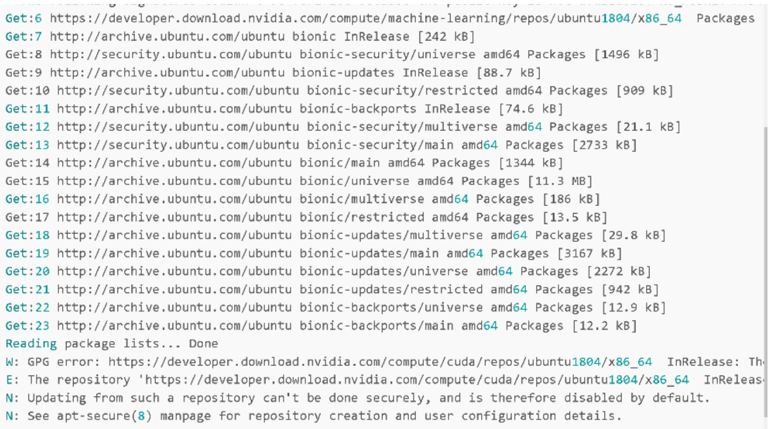
The customer will

NOT SUBSCRIBE TO A TERM DEPOSIT

Campaign: 10
Previous: 14
Previous Outcome: 3
Consumer Price Index: 7.0
Euribor 3 Months Rate: 6.0
Number of Employees: 6
Passed Days: 4
Employment Variation Rate: 2.0
Consumer Confidence Index: 0.0
Month: 0

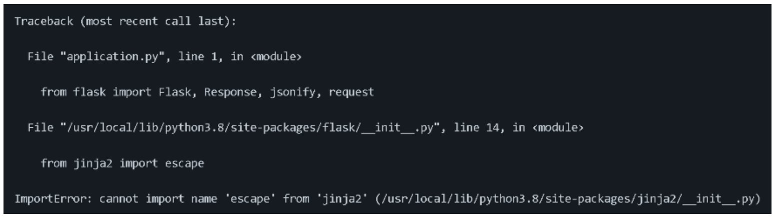
CHALLENGES / ERRORS ENCOUNTERED:

1. Invalid public key for CUDA apt repository



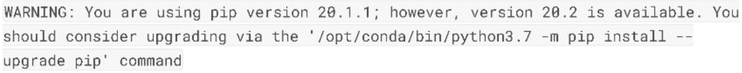
```
Get:6 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64  Packages
Get:7 http://archive.ubuntu.com/ubuntu bionic InRelease [242 kB]
Get:8 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [1496 kB]
Get:9 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:10 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64 Packages [909 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:12 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64 Packages [21.1 kB]
Get:13 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [2733 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic/main amd64 Packages [1344 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic/universe amd64 Packages [11.3 MB]
Get:16 http://archive.ubuntu.com/ubuntu bionic/multiverse amd64 Packages [186 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic/restricted amd64 Packages [13.5 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages [29.8 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [3167 kB]
Get:20 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [2272 kB]
Get:21 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages [942 kB]
Get:22 http://archive.ubuntu.com/ubuntu bionic-backports/universe amd64 Packages [12.9 kB]
Get:23 http://archive.ubuntu.com/ubuntu bionic-backports/main amd64 Packages [12.2 kB]
Reading package lists... Done
W: GPG error: https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64  InRelease: Th
E: The repository 'https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64  InReleas
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
```

2. Flask failing to startup due to Jinja2 breaking change



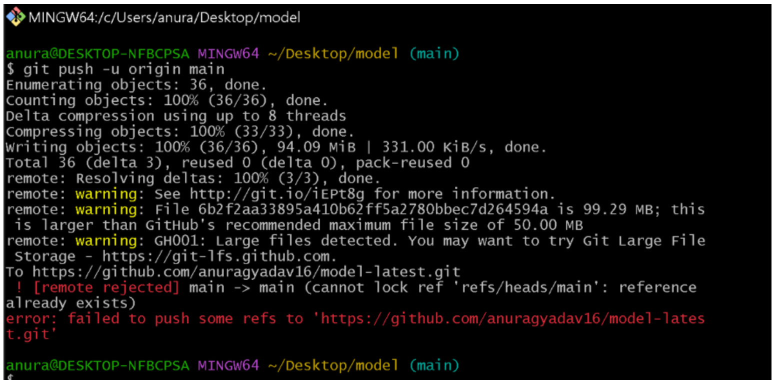
```
Traceback (most recent call last):

  File "application.py", line 1, in <module>

    from flask import Flask, Response, jsonify, request

  File "/usr/local/lib/python3.8/site-packages/flask/__init__.py", line 14, in <module>

    from jinja2 import escape

ImportError: cannot import name 'escape' from 'jinja2' (/usr/local/lib/python3.8/site-packages/jinja2/__init__.py)
```

3. Pip version error



```
WARNING: You are using pip version 20.1.1; however, version 20.2 is available. You
should consider upgrading via the '/opt/conda/bin/python3.7 -m pip install --
upgrade pip' command
```

4. Git size limit error



```
MINGW64:/c/Users/anura/Desktop/model

anura@DESKTOP-NFBCPSA MINGW64 ~/Desktop/model (main)
$ git push -u origin main
Enumerating objects: 36, done.
Counting objects: 100% (36/36), done.
Delta compression using up to 8 threads
Compressing objects: 100% (33/33), done.
Writing objects: 100% (36/36), 94.09 MiB | 331.00 KiB/s, done.
Total 36 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
remote: warning: See http://git.io/iEPt8g for more information.
remote: warning: File 6b2f2aa33895a410b62ff5a2780bbec7d264594a is 99.29 MB; this
 is larger than GitHub's recommended maximum file size of 50.00 MB
remote: warning: GH001: Large files detected. You may want to try Git Large File
 Storage - https://git-lfs.github.com.
To https://github.com/anuragyadav16/model-latest.git
 ! [remote rejected] main -> main (cannot lock ref 'refs/heads/main': reference
already exists)
error: failed to push some refs to 'https://github.com/anuragyadav16/model-lates
t.git'

anura@DESKTOP-NFBCPSA MINGW64 ~/Desktop/model (main)
$
```