CI/CD AND ENVIRONMENT CONFIGURATION

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
INFORMATION TECHNOLOGY

Submitted by:

Ishan Mangal | Muheet Alam | Paras Chandra | Keshu Shukla | Ujjawal Tomar
(24/ISY/05) | (24/ISY/09) | (24/ISY/13) | (24/ISY/15) | (24/ISY/25)

Under the supervision of

DR. ANKIT DESWAL

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

We, Ishan Mangal (24/ISY/05), Muheet Alam (24/ISY/09), Paras Chandra (24/ISY/13), Keshu Shukla (24/ISY/15), Ujjawal Tomar (24/ISY/25) of M.Tech (Information Technology), hereby declare that the project titled "CI/CD and Environment Configuration" which is submitted by us to the Department of Information Technology, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi
Date:

(ISHAN MANGAL)
(MUHEET ALAM)
(PARAS CHANDRA)
(KESHU SHUKLA)
(UJJAWAL TOMAR)

DEPARTMENT OF INFORMATION TECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

We hereby certify that the Project titled “CI/CD and Environment Configuration” which is submitted by Ishan Mangal (24/ISY/05), Muheet Alam (24/ISY/09), Paras Chandra (24/ISY/13), Keshu Shukla (24/ISY/15), Ujjawal Tomar (24/ISY/25), Department of Information Technology, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the students under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi
Date:

(DR. ANKIT DESWAL)
SUPERVISOR
Assistant Professor, IT Department,
Delhi Technological University
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

We would like to express my sincere gratitude to my supervisor **Dr. Ankit Deswal** (Assistant Professor, Information Technology Department) for their invaluable guidance, constructive feedback, and unwavering support throughout this research. Their mentorship has been instrumental in shaping the direction and outcome of this work.

We extend my heartfelt thanks to my institution, department, and **Prof. Dinesh Kumar Vishwakarma** (Head of the Information Technology Department) for providing the resources, infrastructure, and encouragement necessary to carry out this study. We are also grateful to our peers and colleagues for their insightful discussions and collaboration, which greatly enriched my understanding and contributed to the progress of this research.

We would like to acknowledge the contributions of the authors of the research works referenced in this project, whose foundational studies have been instrumental in shaping the direction of this work. Lastly, express our deepest gratitude to my family and friends for their unwavering support and motivation, without which this endeavour would not have been possible.

Muheet Alam (24/ISY/09), Ishan Mangal (24/ISY/05), Paras Chandra (24/ISY/13), Keshu Shukla (24/ISY/15), Ujjawal Tomar (24/ISY/25)

M.Tech(Information Technology)

Department of Information Technology

Delhi Technological University

CONTENTS

OBJECTIVE

The primary goal of this project is to automate the setup and deployment of a Machine Learning (ML) environment using **Docker** and **Ansible**, ensuring reproducibility, ease of collaboration, and consistency across systems. Specifically, this project implements a pipeline that:

- Uses **Ansible** to automate environment setup both **locally** and **remotely**.
- Uses **Docker** to containerize the ML training workflow.
- Publishes a **working Docker image** to **DockerHub**: `alamsaim/fakenews-model:latest`.

Deliverables:

- `playbook-local.yml` and `playbook-remote.yml` for local and portable environment setup respectively.
- `Dockerfile` and associated Docker image containing all ML training logic and dependencies.
- Trained model artifacts (`model.pkl` and `vectorizer.pkl`) saved to `/opt/ml/output` after container execution.

INTRODUCTION

In the modern era of software and machine learning (ML) development, the need for automation, consistency, and reliability in deployment has become critical. This project focuses on simplifying and automating the deployment of a machine learning pipeline using DevOps tools like **Docker**, **Ansible**, and **DockerHub**. It encapsulates environment provisioning and application containerization to ensure that the ML workflow can be deployed and executed consistently, regardless of the host system. The core of this implementation lies in creating a seamless **Continuous Integration and Continuous Deployment (CI/CD)** workflow and automating the environment setup for scalable, reproducible deployment.

Abstract of Implementation:

The ML training script (`train.py`) was designed to detect fake news using TF-IDF vectorization and a PassiveAggressiveClassifier. This script, along with its dependencies (`pandas`, `scikit-learn`, `nltk`), is run inside a Docker container that is automatically built and executed using Ansible playbooks.

Importance of Automation:

Automation plays a critical role in the success of ML workflows:

- It removes manual errors during environment setup.
- It ensures every team member can replicate the training environment effortlessly.
- It aligns with real-world MLOps practices.

Tools Used in This Project:

- **Docker**: To containerize the ML pipeline.
- **Ansible**: To automate system provisioning.
- **DockerHub**: To store and share the working Docker image.

TECHNOLOGY STACK

1. Python 3.9

- Python is a high-level, interpreted programming language widely used in scripting, data science, machine learning, and automation.
- Used to write the `train.py` script that loads `train.csv` and `test.csv`, vectorizes the text data, trains a fake news classifier, and saves output models.

2. Docker

- Docker is a platform that allows applications to be packaged and run in containers — lightweight, standalone, and executable environments that include everything needed to run the software. Containers ensure that applications run the same regardless of where they are deployed.
- The `Dockerfile` in `/app` builds an image with Python and ML libraries, copies source code, installs dependencies, and runs training on container start. Final image: `alamsaim/fakenews-model:latest`.

3. Ansible

- Ansible is an open-source IT automation tool that automates configuration management, application deployment, and task execution across servers. Using YAML-based playbooks, Ansible can install software, configure systems, and manage infrastructure, making deployments repeatable and error-free.
- Two playbooks were created:
  i. `playbook-local.yml`: Sets up environment locally.
  ii. `playbook-remote.yml`: Sets up environment remotely or in cloud VMs.

4. DockerHub

- DockerHub is a cloud-based repository where Docker images can be stored, shared, and pulled for use. It acts like GitHub for Docker containers, allowing developers to automate the build and deployment process directly through CI/CD pipelines.
- The trained and tested image was pushed to DockerHub for public reuse. The image runs the model training and saves artifacts to `/opt/ml/output`.

SYSTEM REQUIREMENTS

1. OS Compatibility

- Ubuntu 20.04+ / WSL2 (for Windows)
- macOS

2. Required Software

Tool	Version	Purpose
Python	3.8+	Core scripting language
pip	21+	Package manager
Docker	20.10+	Containerisation
Ansible	2.9+	Automation & environment provisioning
DockerHub	-	Required to push and pull Docker images remotely.

3. Python Packages

Specified in `requirements.txt`:

- `pandas`
- `scikit-learn`
- `numpy`
- `nltk`

These are auto-installed during Docker image build or via Ansible.

FOLDER STRUCTURE OVERVIEW

```
project-2-mlops/
├── app/
│   ├── train.py
│   ├── Dockerfile
│   ├── requirements.txt
│   ├── train.csv (local only)
│   └── test.csv (local only)
└──── ansible/
    ├── playbook-local.yml
    └── playbook-remote.yml
```

Description of each file:

- `train.py`: Trains ML model, outputs `model.pkl` and `vectorizer.pkl`.
- `Dockerfile`: Builds an image to execute training automatically.
- `playbook-local.yml`: Sets up local system with Docker and required packages.
- `playbook-remote.yml`: Does the same on a remote host or VM.

KEY FUNCTIONALITIES

End-to-End Workflow:

1. Ansible playbook installs Python, pip, and Docker.
2. Ansible copies the `app/` folder to `/opt/ml`.
3. Docker image is built using `Dockerfile`.
4. Docker container is executed, running `train.py`.
5. Trained model artifacts are saved to `/opt/ml/output/`.

Sample Output:

- Accuracy: `93.97%`
- Confusion Matrix:
```
[[583  42]
 [ 34 601]]
```
- Model saved to `/opt/ml/output/model.pkl`
- Vectorizer saved to `/opt/ml/output/vectorizer.pkl`

DOCKERHUB INTEGRATION

- **Docker Image Name:**

```
docker pull alamsaim/fakenews-model:latest
```

- **Push Steps:**

```
docker tag fakenews-model alamsaim/fakenews-model:latest
docker push alamsaim/fakenews-model:latest
```

- **Usage:**

```
docker run --rm -v $(pwd)/output:/opt/ml/output
alamsaim/fakenews-model:latest
```

- **Output:**

Files saved to `./output` directory on host machine.

EXECUTION INSTRUCTIONS

- **Local Execution:**

```
ansible-playbook ansible/playbook-local.yml
```

Performs setup, builds image, and runs container locally.

- **Remote Execution:**

```
ansible-playbook ansible/playbook-remote.yml
```

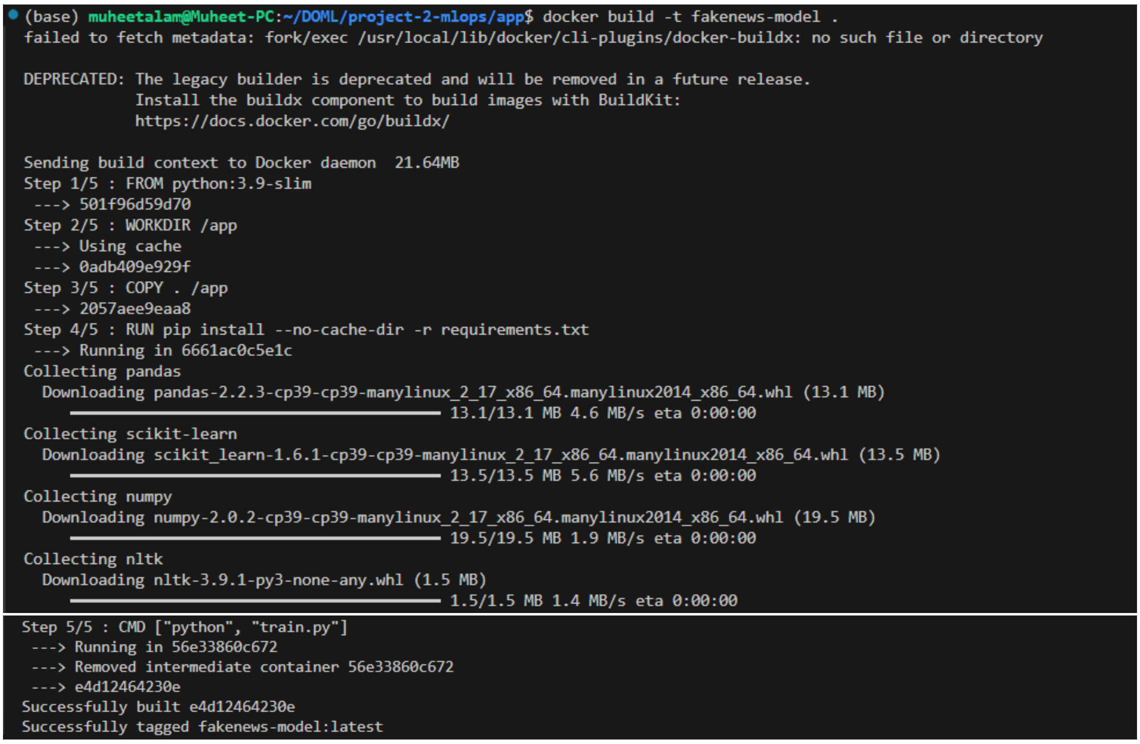
Runs same setup on a remote system with Ansible installed.

- **Docker Only:**

```
cd app
docker build -t fakenews-model .
docker run --rm -v $(pwd)/output:/opt/ml/output fakenews-model
```

SCREENSHOTS

1. Docker Build:



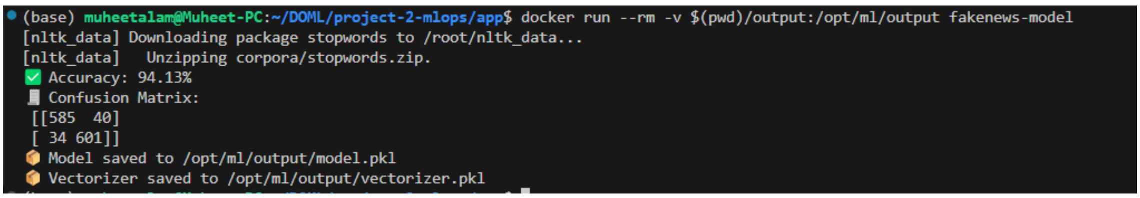
```
(base) muheetalam@Muheet-PC:~/DOML/project-2-mlops/app$ docker build -t fakenews-model .
failed to fetch metadata: fork/exec /usr/local/lib/docker/cli-plugins/docker-buildx: no such file or directory

DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  21.64MB
Step 1/5 : FROM python:3.9-slim
 ---> 501f96d59d70
Step 2/5 : WORKDIR /app
 ---> Using cache
 ---> 0adb409e929f
Step 3/5 : COPY . /app
 ---> 2057aee9eaa8
Step 4/5 : RUN pip install --no-cache-dir -r requirements.txt
 ---> Running in 6661ac0c5e1c
Collecting pandas
  Downloading pandas-2.2.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.1/13.1 MB 4.6 MB/s eta 0:00:00
Collecting scikit-learn
  Downloading scikit_learn-1.6.1-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.5/13.5 MB 5.6 MB/s eta 0:00:00
Collecting numpy
  Downloading numpy-2.0.2-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (19.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19.5/19.5 MB 1.9 MB/s eta 0:00:00
Collecting nltk
  Downloading nltk-3.9.1-py3-none-any.whl (1.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 1.4 MB/s eta 0:00:00
Step 5/5 : CMD ["python", "train.py"]
 ---> Running in 56e33860c672
 ---> Removed intermediate container 56e33860c672
 ---> e4d12464230e
Successfully built e4d12464230e
Successfully tagged fakenews-model:latest
```

Fig 1. Successful image creation

2. Docker Run:



```
(base) muheetalam@Muheet-PC:~/DOML/project-2-mlops/app$ docker run --rm -v $(pwd)/output:/opt/ml/output fakenews-model
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
✅ Accuracy: 94.13%
📊 Confusion Matrix:
 [[585  40]
 [ 34 601]]
📦 Model saved to /opt/ml/output/model.pkl
📦 Vectorizer saved to /opt/ml/output/vectorizer.pkl
```

Fig 2. Output in terminal with accuracy & confusion matrix

3. Output Files:

```
(base) muheetalam@Muheet-PC:~/DOML/project-2-mlops/app$ ls output
model.pkl  vectorizer.pkl
```

Fig 3(a). Screenshot of `model.pkl` and `vectorizer.pkl` listed inside `/opt/ml/output`

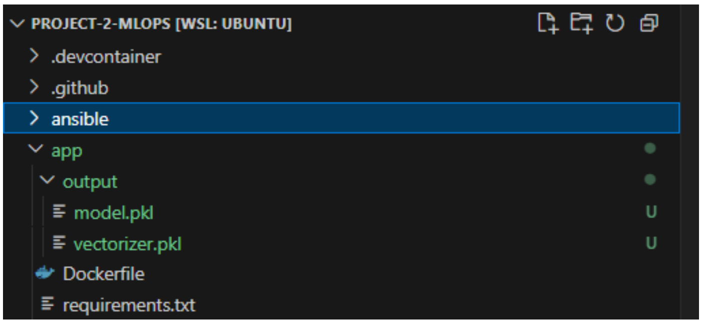


Fig 3(b). Screenshot of model.pkl and vectorizer.pkl inside /opt/ml/output

4. Ansible Local Playbook Execution:

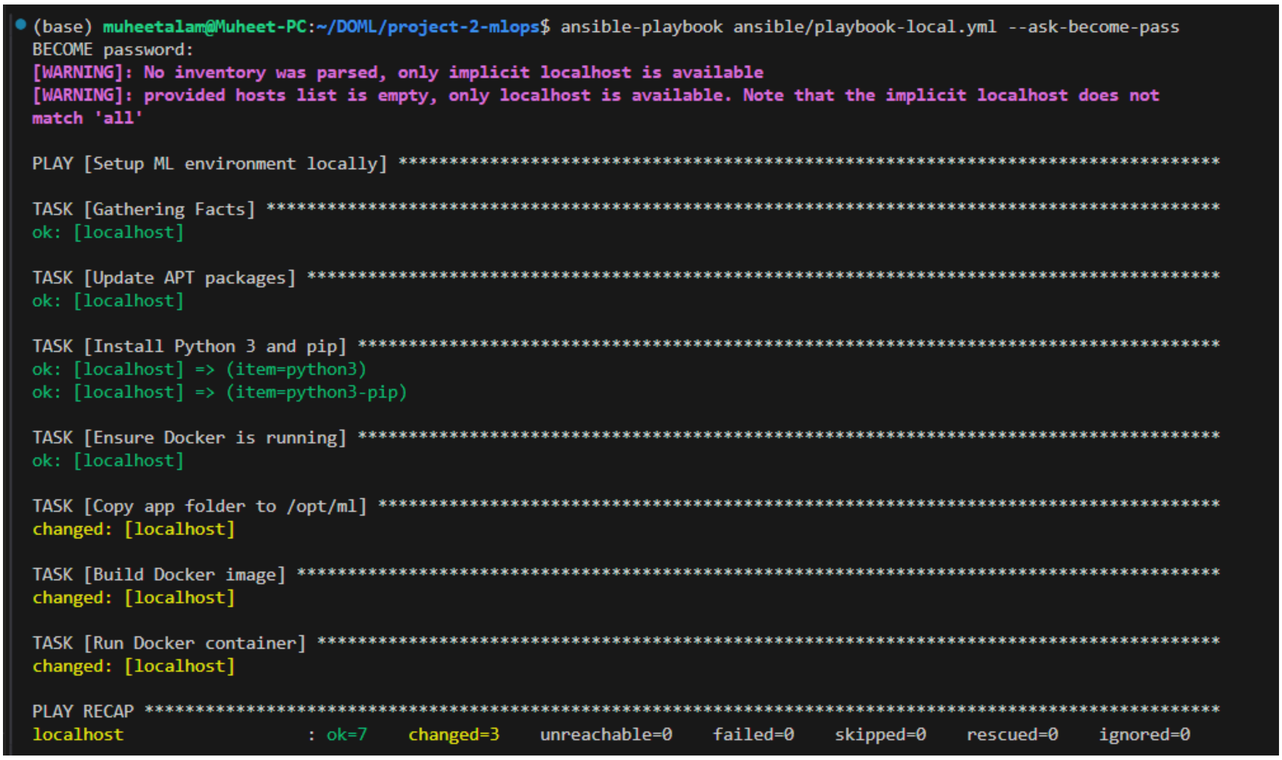


Fig 4. Ansible playbook execution locally.

5. Ansible Remote Playbook Execution:

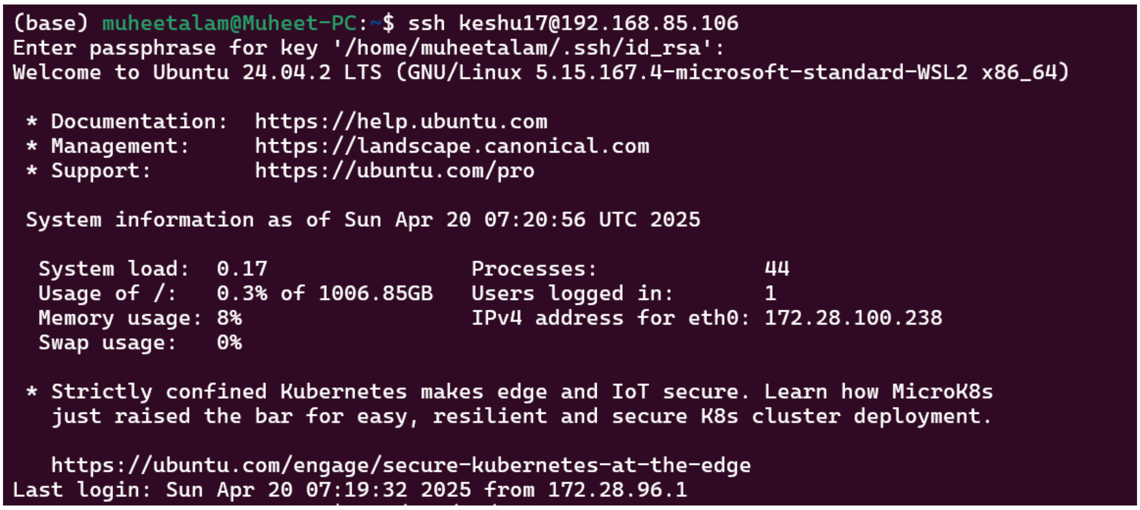


Fig 5. Establishing SSH connection.

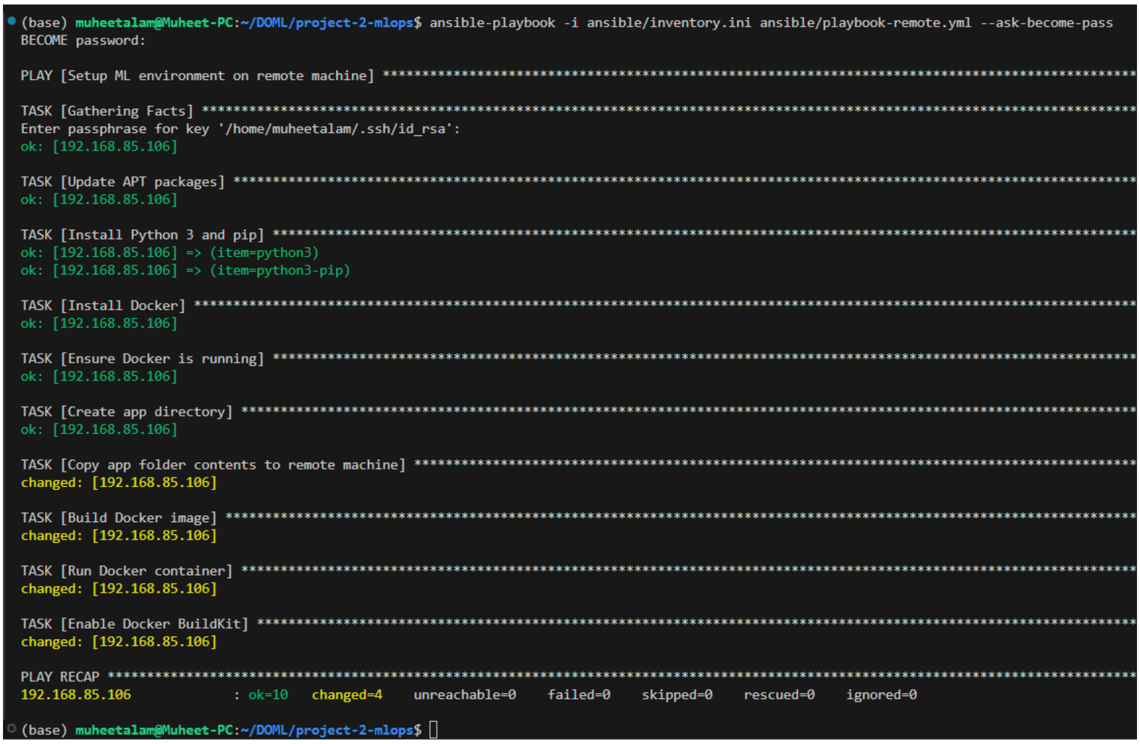


Fig 6. Remote ML environment successfully set up using Ansible playbook with automated Docker deployment.

```
keshu17@DESKTOP-6VNP240:/opt/ml/output$ docker run -it --rm -v /opt/ml/output:/opt/ml/output 88c9246548f7 /bin/bash
root@9e9ced611f4b:/app# pip install -r requirements.txt
python train.py
Collecting pandas
  Downloading pandas-2.2.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.1/13.1 MB 585.5 kB/s eta 0:00:00
Collecting scikit-learn
  Downloading scikit_learn-1.6.1-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.5/13.5 MB 517.1 kB/s eta 0:00:00
Collecting numpy
  Downloading numpy-2.0.2-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (19.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19.5/19.5 MB 245.1 kB/s eta 0:00:00
Collecting nltk
  Downloading nltk-3.9.1-py3-none-any.whl (1.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.5/1.5 MB 333.1 kB/s eta 0:00:00
Collecting pytz>=2020.1
  Downloading pytz-2025.2-py2.py3-none-any.whl (509 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 509.2/509.2 kB 365.3 kB/s eta 0:00:00
Collecting python-dateutil>=2.8.2
  Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 229.9/229.9 kB 340.4 kB/s eta 0:00:00
Collecting tzdata>=2022.7
  Downloading tzdata-2025.2-py2.py3-none-any.whl (347 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 347.8/347.8 kB 423.5 kB/s eta 0:00:00
Collecting scipy>=1.6.0
  Downloading scipy-1.13.1-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (38.6 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 38.6/38.6 MB 737.2 kB/s eta 0:00:00
Collecting threadpoolctl>=3.1.0
  Downloading threadpoolctl-3.6.0-py3-none-any.whl (18 kB)
Collecting joblib>=1.2.0
  Downloading joblib-1.4.2-py3-none-any.whl (301 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 301.8/301.8 kB 950.9 kB/s eta 0:00:00
Collecting tqdm
  Downloading tqdm-4.67.1-py3-none-any.whl (78 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 78.5/78.5 kB 559.3 kB/s eta 0:00:00
Collecting click
  Downloading click-8.1.8-py3-none-any.whl (98 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 98.2/98.2 kB 1.3 MB/s eta 0:00:00
Collecting regex>=2021.8.3
  Downloading regex-2024.11.6-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (780 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 780.9/780.9 kB 1.1 MB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.17.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: pytz, tzdata, tqdm, threadpoolctl, six, regex, numpy, joblib, click, scipy, python-dateutil, nltk, scikit-learn, pandas
Successfully installed click-8.1.8 joblib-1.4.2 nltk-3.9.1 numpy-2.0.2 pandas-2.2.3 python-dateutil-2.9.0.post0 pytz-2025.2 regex-2024.11.6 scikit-learn-1.6.1
```

Fig 7. Installing ML project dependencies inside Docker container using requirements.txt.

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
✅ Accuracy: 93.81%
📋 Confusion Matrix:
 [[583  42]
 [ 36 599]]
📦 Model saved to /opt/ml/output/model.pkl
📦 Vectorizer saved to /opt/ml/output/vectorizer.pkl
root@9e9ced611f4b:/app#
```

Fig 8. Model trained with 93.81% accuracy; saved model.pkl and vectorizer.pkl for deployment.

```
keshu17@DESKTOP-6VNP240:/mnt/c/WINDOWS/system32$ ls -lh /opt/ml/output/
total 1.9M
-rw-r--r-- 1 root root 493K Apr 20 07:32 model.pkl
-rw-r--r-- 1 root root 1.4M Apr 20 07:32 vectorizer.pkl
```

Fig 9. Successfully accessed remote machine and verified presence of saved model and vectorizer files.

6. **Dockerhub**:

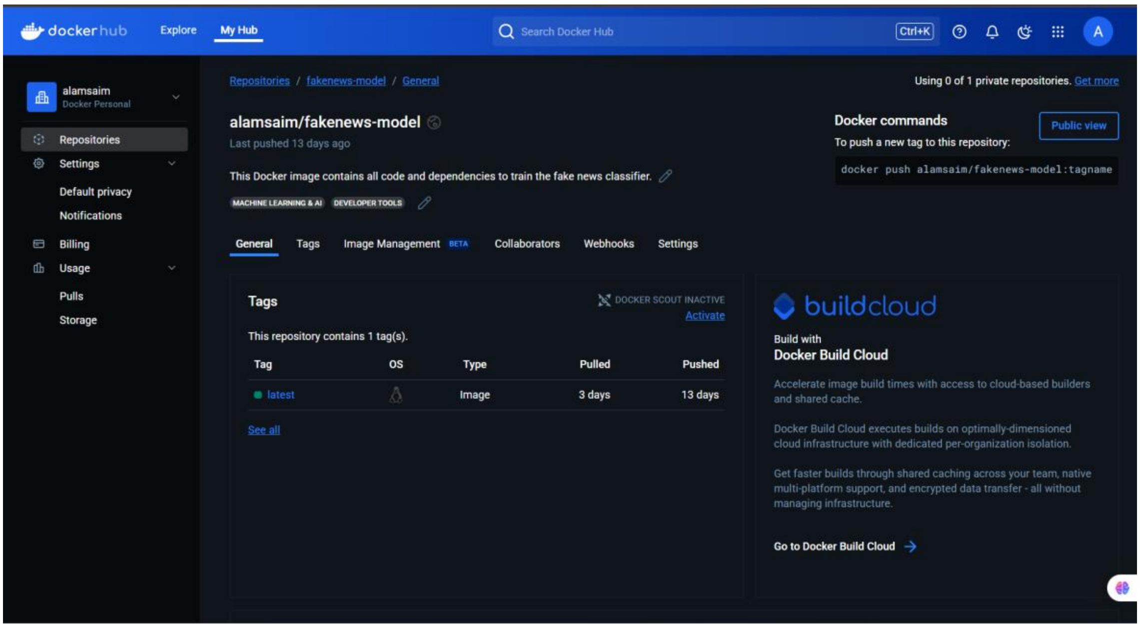


Fig 10. Dockerhub demonstration containing Docker Image

CONCLUSION

This project successfully automated an end-to-end ML workflow using Ansible and Docker. The containerized solution ensures reproducibility across environments, and the Ansible playbooks minimize manual setup. By pushing the Docker image to DockerHub, the model training process becomes easily shareable and executable by others without any configuration. This aligns with real-world MLOps practices and builds a strong foundation for more advanced automation and deployment workflows in machine learning systems.

REFERENCES

1. https://docs.ansible.com/ansible/latest/index.html#
2. https://hub.docker.com/
3. https://www.docker.com/
4. https://github.com/