# MLOps Assignment I — End-to-End ML Model Development, CI/CD, and Production Deployment

## 1. Objective

Build a machine learning classifier for heart disease prediction and deploy it as a cloud-ready, monitored API using modern MLOps best practices.

## 2. Demo Video & Repository Link

- **Demo Video**: Pipeline Demo on YouTube - Complete end-to-end pipeline demonstration
- **GitHub Repository**: github.com/chaitudevi/Heart-Disease-Classification-2

## 3. Dataset (Heart Disease UCI)

- **Source**: UCI Machine Learning Repository
- **Data acquisition**:
  - Script: `src/data/download_data.py`
  - Raw data folder: `data/raw/`

Screenshot(s)

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % python src/data/download_dat
a.py
Dataset downloaded to /Users/darshan/Developer/repos/Heart-Disease-Classification-2/data/r
aw/heart_disease.csv
Column headers added successfully
```

## 4. Data Acquisition, Cleaning, and EDA (Task 1 — 5 marks)

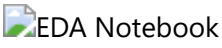
### 4.1 Data Cleaning & Preprocessing

- Missing value handling, encoding, transformations
- Implementation: `src/data/preprocess.py`

### 4.2 EDA

- Notebook: `notebooks/01_eda.ipynb`
- Visuals: class balance, correlation heatmap, distributions

Screenshot(s)


EDA Notebook

## 5. Feature Engineering & Model Development (Task 2 — 8 marks)

### 5.1 Feature Pipeline

- Implementation: `src/features/feature_pipeline.py`

### 5.2 Models

- Logistic Regression
- Random Forest

### 5.3 Evaluation

- Cross-validation metrics: accuracy, precision, recall, ROC-AUC
- Training entrypoint: `src/models/train.py`

### Screenshot(s)

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % python src/models/train.py
/Users/darshan/.pyenv/versions/3.10.18/lib/python3.10/site-packages/pydantic/_internal/_fi
elds.py:149: UserWarning: Field "model_name" has conflict with protected namespace "model_
".

You may be able to resolve this warning by setting `model_config['protected_namespaces'] =
 ()`.
  warnings.warn(
/Users/darshan/.pyenv/versions/3.10.18/lib/python3.10/site-packages/mlflow/tracking/_track
ing_service/utils.py:178: FutureWarning: The filesystem tracking backend (e.g., './mlruns'
) will be deprecated in February 2026. Consider transitioning to a database backend (e.g.,
 'sqlite:///mlflow.db') to take advantage of the latest MLflow features. See https://githu
b.com/mlflow/mlflow/issues/18534 for more details and migration guidance. For migrating ex
isting data, https://github.com/mlflow/mlflow-export-import can be used.
  return FileStore(store_uri, store_uri)
[2026-01-03 13:54:13,265] INFO | src.features.feature_pipeline | Building full feature pip
eline
[2026-01-03 13:54:13,265] INFO | src.features.feature_pipeline | Creating ColumnTransforme
r
[2026-01-03 13:54:13,265] INFO | src.features.feature_pipeline | Building numeric feature
pipeline
[2026-01-03 13:54:13,265] INFO | src.features.feature_pipeline | Building categorical feat
ure pipeline
[2026-01-03 13:54:13,268] INFO | src.features.feature_pipeline | Creating engineered featu
res
[2026-01-03 13:54:13,277] INFO | src.features.feature_pipeline | Creating engineered featu
res
/Users/darshan/.pyenv/versions/3.10.18/lib/python3.10/site-packages/sklearn/utils/extmath.
py:203: RuntimeWarning: divide by zero encountered in matmul
  ret = a @ b
/Users/darshan/.pyenv/versions/3.10.18/lib/python3.10/site-packages/sklearn/utils/extmath.
py:203: RuntimeWarning: overflow encountered in matmul
  ret = a @ b
/Users/darshan/.pyenv/versions/3.10.18/lib/python3.10/site-packages/sklearn/utils/extmath.
py:203: RuntimeWarning: invalid value encountered in matmul
  ret = a @ b
[2026-01-03 13:54:13,282] INFO | src.features.feature_pipeline | Creating engineered featu
res
/Users/darshan/.pyenv/versions/3.10.18/lib/python3.10/site-packages/sklearn/utils/extmath.
py:203: RuntimeWarning: divide by zero encountered in matmul
  ret = a @ b
/Users/darshan/.pyenv/versions/3.10.18/lib/python3.10/site-packages/sklearn/utils/extmath.
```

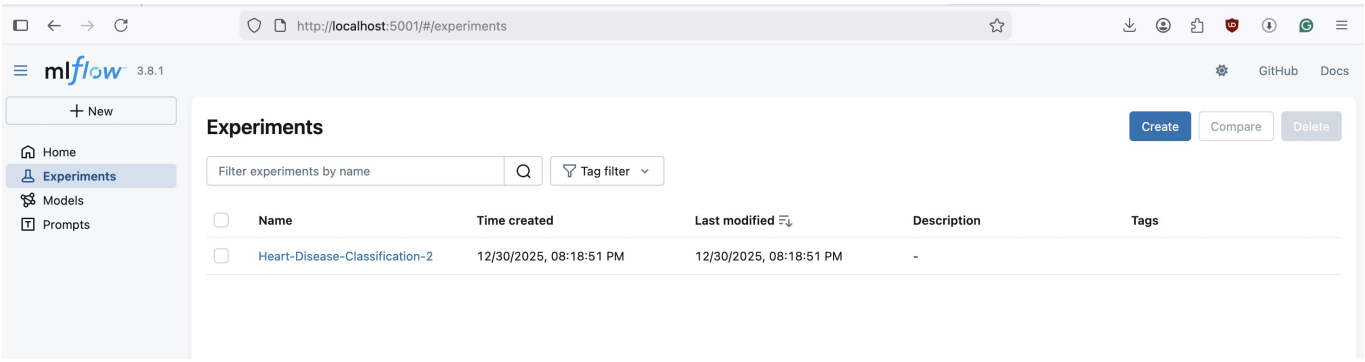
```
res
[2026-01-03 13:54:13,543] INFO | src.features.feature_pipeline | Creating engineered featu
res
[2026-01-03 13:54:13,559] INFO | src.features.feature_pipeline | Creating engineered featu
res
[2026-01-03 13:54:13,576] INFO | src.features.feature_pipeline | Creating engineered featu
res
[2026-01-03 13:54:13,662] INFO | src.features.feature_pipeline | Creating engineered featu
res
[2026-01-03 13:54:13,680] INFO | src.features.feature_pipeline | Creating engineered featu
res

Logistic Regression
accuracy: 0.8429
precision: 0.8620
recall: 0.7838
roc_auc: 0.9056

Random Forest
accuracy: 0.7933
precision: 0.8024
recall: 0.7297
roc_auc: 0.8853
[2026-01-03 13:54:13,700] INFO | src.features.feature_pipeline | Building full feature pip
eline
[2026-01-03 13:54:13,700] INFO | src.features.feature_pipeline | Creating ColumnTransforme
```

```
r
[2026-01-03 13:54:13,700] INFO | src.features.feature_pipeline | Building numeric feature
pipeline
[2026-01-03 13:54:13,700] INFO | src.features.feature_pipeline | Building categorical feat
ure pipeline
[2026-01-03 13:54:13,701] INFO | src.features.feature_pipeline | Creating engineered featu
res
Best model selected: Logistic Regression
[2026-01-03 13:54:13,707] INFO | src.features.feature_pipeline | Creating engineered featu
res
```

## 6. Experiment Tracking (Task 3 — 5 marks)

- Tool: MLflow
- Logged:
  - Parameters
  - Metrics
  - Artifacts (ROC curve, confusion matrix, reports)

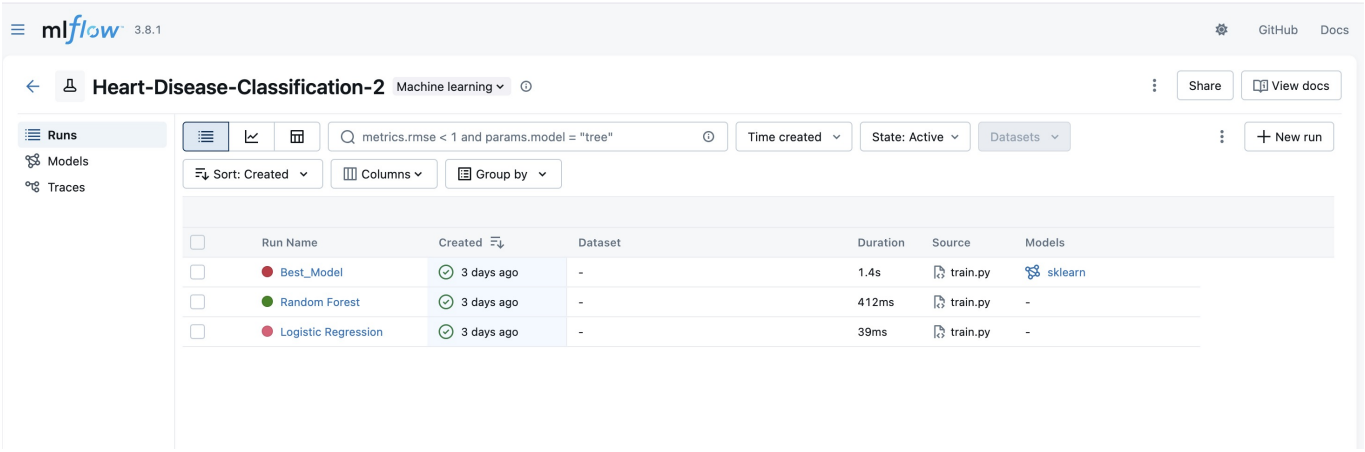
### Screenshot(s)

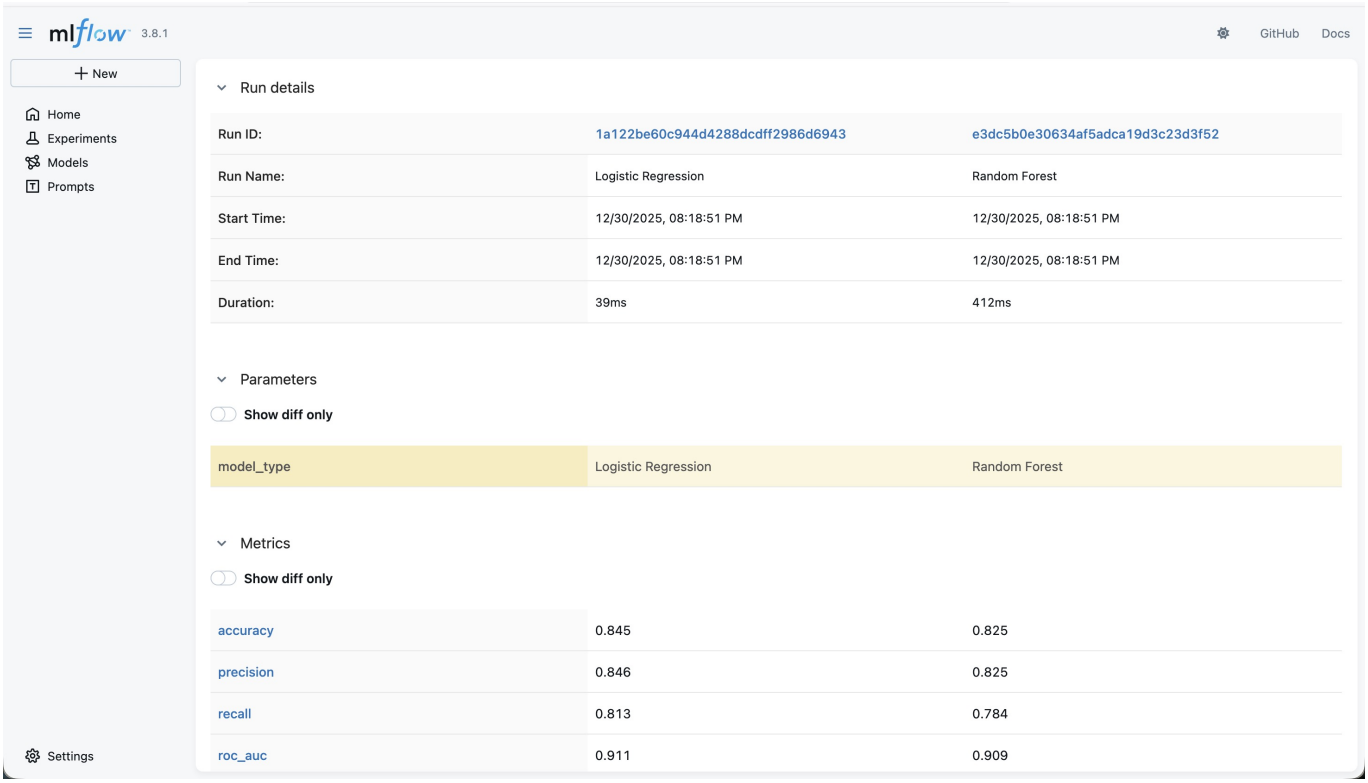


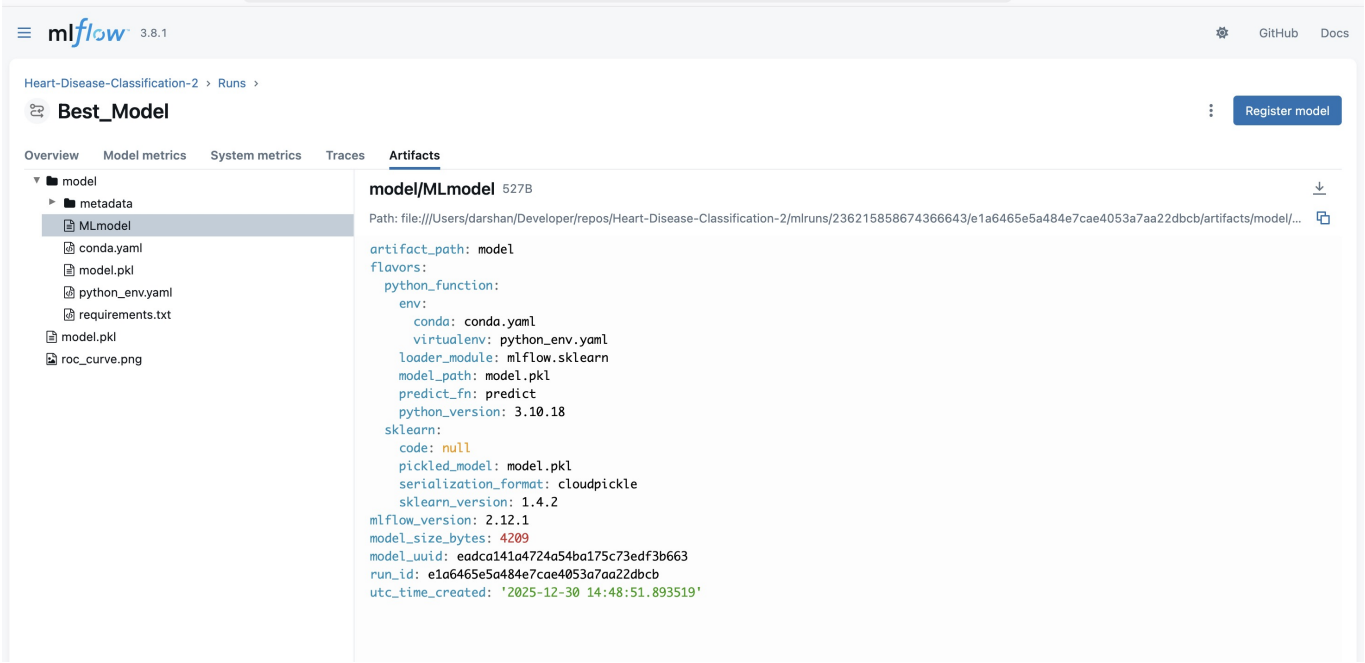

## 7. Model Packaging & Reproducibility (Task 4 — 7 marks)

- Model bundle saved at: `artifacts/model.pkl`
- Reproducible environment: `requirements.txt`
- Preprocessing/feature transformations are part of the saved sklearn `Pipeline`

### Screenshot(s)



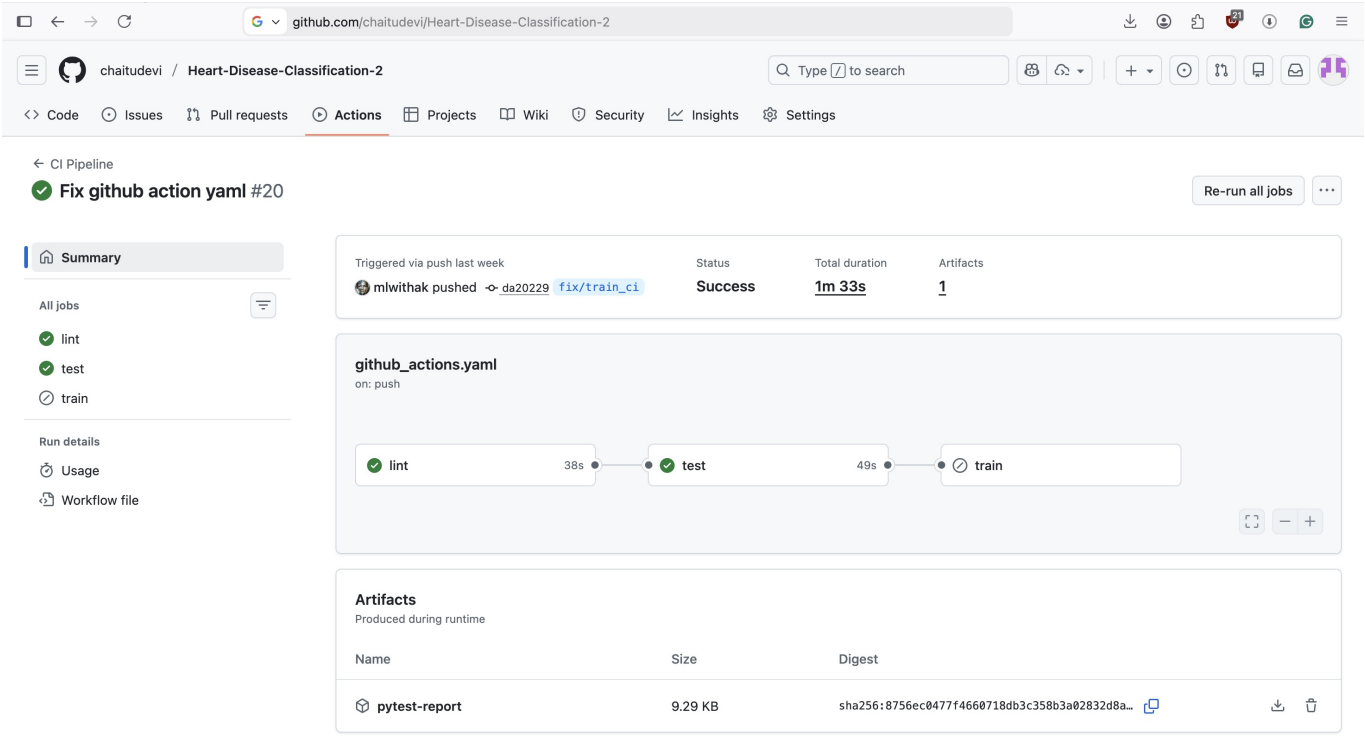



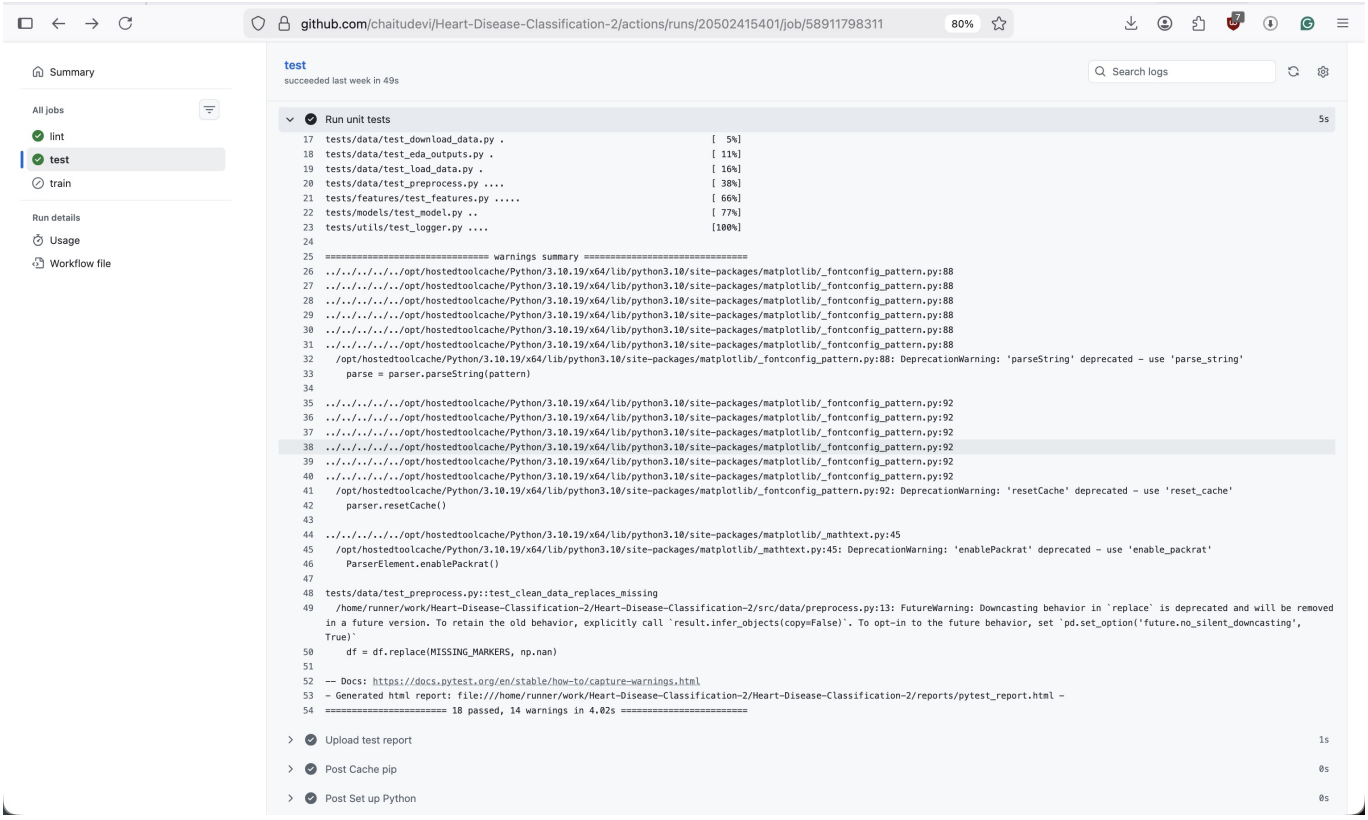



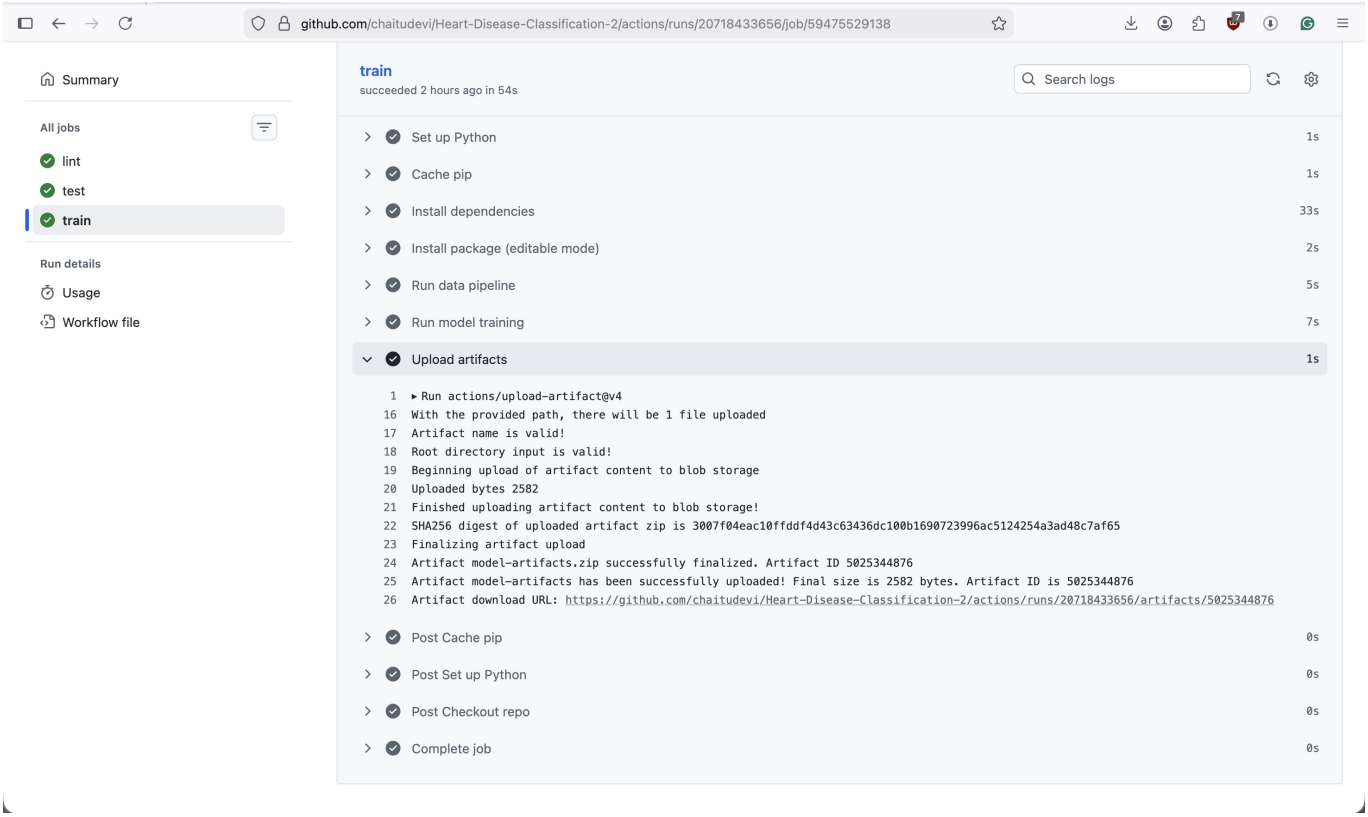

# 8. CI/CD Pipeline & Automated Testing (Task 5 — 8 marks)

- Tests under: `tests/`
- CI workflow: `.github/workflows/github_actions.yaml`
- Stages: lint, test, train

## Screenshot(s)






# 9. Model Containerization (Task 6 — 5 marks)

- Dockerfile: `Dockerfile`
- API:
  - `/predict` accepts JSON, returns prediction + confidence
  - `/metrics` exposes Prometheus metrics

Screenshot(s)

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % docker build -t heart-disease-api:latest .
[+] Building 2.5s (11/11) FINISHED                                  docker:desktop-linux
 => [internal] load build definition from Dockerfile                                0.0s
 => => transferring dockerfile: 826B                                                0.0s
 => [internal] load metadata for docker.io/library/python:3.10-slim                 2.3s
 => [internal] load .dockerignore                                                   0.0s
 => => transferring context: 207B                                                   0.0s
 => [1/6] FROM docker.io/library/python:3.10-slim@sha256:7b68a5fa7cf0d20b4cedb1dc9a  0.0s
 => => resolve docker.io/library/python:3.10-slim@sha256:7b68a5fa7cf0d20b4cedb1dc9a  0.0s
 => [internal] load build context                                                   0.0s
 => => transferring context: 1.09MB                                                 0.0s
 => CACHED [2/6] WORKDIR /app                                                       0.0s
 => CACHED [3/6] RUN apt-get update   && apt-get install -y --no-install-recommends  0.0s
 => CACHED [4/6] COPY requirements.txt ./requirements.txt                           0.0s
 => CACHED [5/6] RUN pip install --no-cache-dir -r requirements.txt                 0.0s
 => [6/6] COPY . .                                                                  0.0s
 => exporting to image                                                              0.1s
 => => exporting layers                                                             0.0s
 => => exporting manifest sha256:9e558d454705301252952463e92faeee02e025a8eed006548a  0.0s
 => => exporting config sha256:351153723d5ba02edbd88e8fca055982324e9f1b23aff0b9ca9b  0.0s
 => => exporting attestation manifest sha256:658ff0f73f69b0ed51642dbe9168e6a75f20ec  0.0s
 => => exporting manifest list sha256:6bc4d960591420c677ee391fa46bd4a36fb4193421f89  0.0s
 => => naming to docker.io/library/heart-disease-api:latest                         0.0s
 => => unpacking to docker.io/library/heart-disease-api:latest                      0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/p51azyqgbz6g5tu3rnlvp1e03
```

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % docker run --rm -p 8000:8000 -v $(pwd)/artifacts:/app/artifacts heart-d
isease-api:latest
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
[2026-01-03 08:46:01,542] INFO | src.api.app | Handled GET /health -> 200 in 0.013s
INFO:     127.0.0.1:45314 - "GET /health HTTP/1.1" 200 OK
```

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % curl -X POST http://localhost:8000/predict -H "Content-Type: applicatio
n/json" --data @sample_data/sample_request.json
{"prediction":0,"confidence":0.8095673592559162}%
```

# 10. Production Deployment (Task 7 — 7 marks)

You deployed on **Docker Desktop Kubernetes** using manifests.

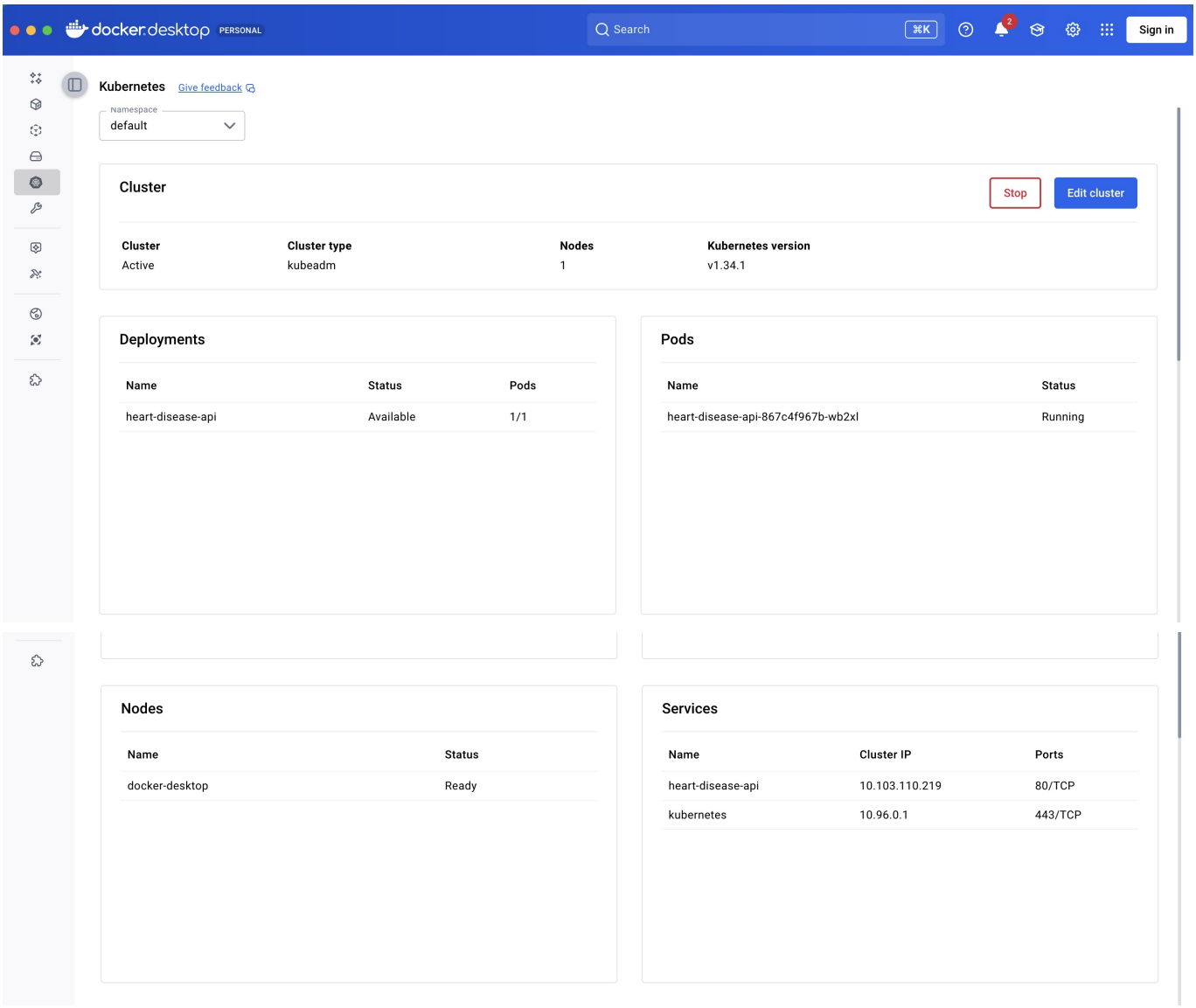
- Manifest: `k8s/deployment.yaml`
- Service exposure: LoadBalancer/NodePort

## Screenshot(s)

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % kubectl apply -f k8s/deployment.yaml
deployment.apps/heart-disease-api unchanged
service/heart-disease-api unchanged
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 %
```

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % kubectl get svc heart-disease-api
NAME                TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
heart-disease-api   LoadBalancer   10.103.110.219   localhost     80:31677/TCP   3d2h
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 %
```

```
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 % kubectl get pods
NAME                                 READY   STATUS    RESTARTS   AGE
heart-disease-api-867c4f967b-wb2xl   1/1     Running   0          3d2h
darshan@Darshans-MacBook-Air Heart-Disease-Classification-2 %
```




# 11. Monitoring & Logging (Task 8 — 3 marks)

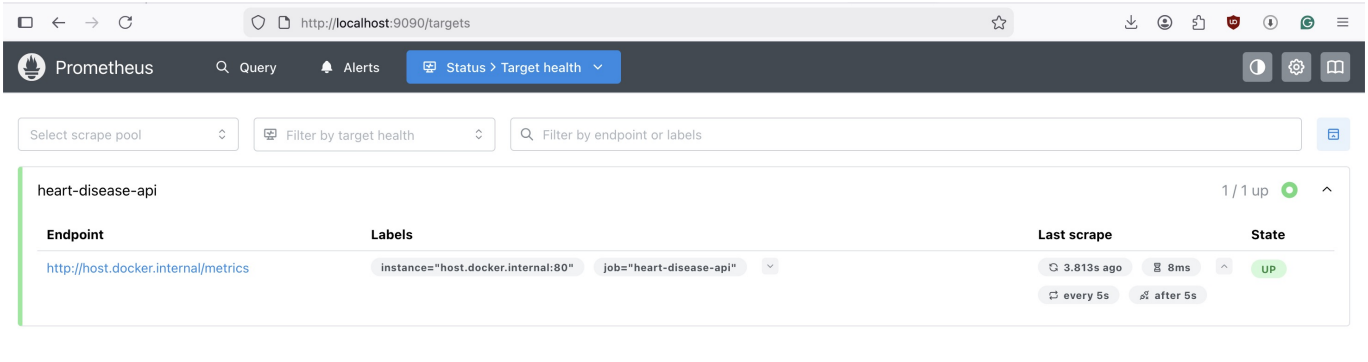
## 11.1 API Logging

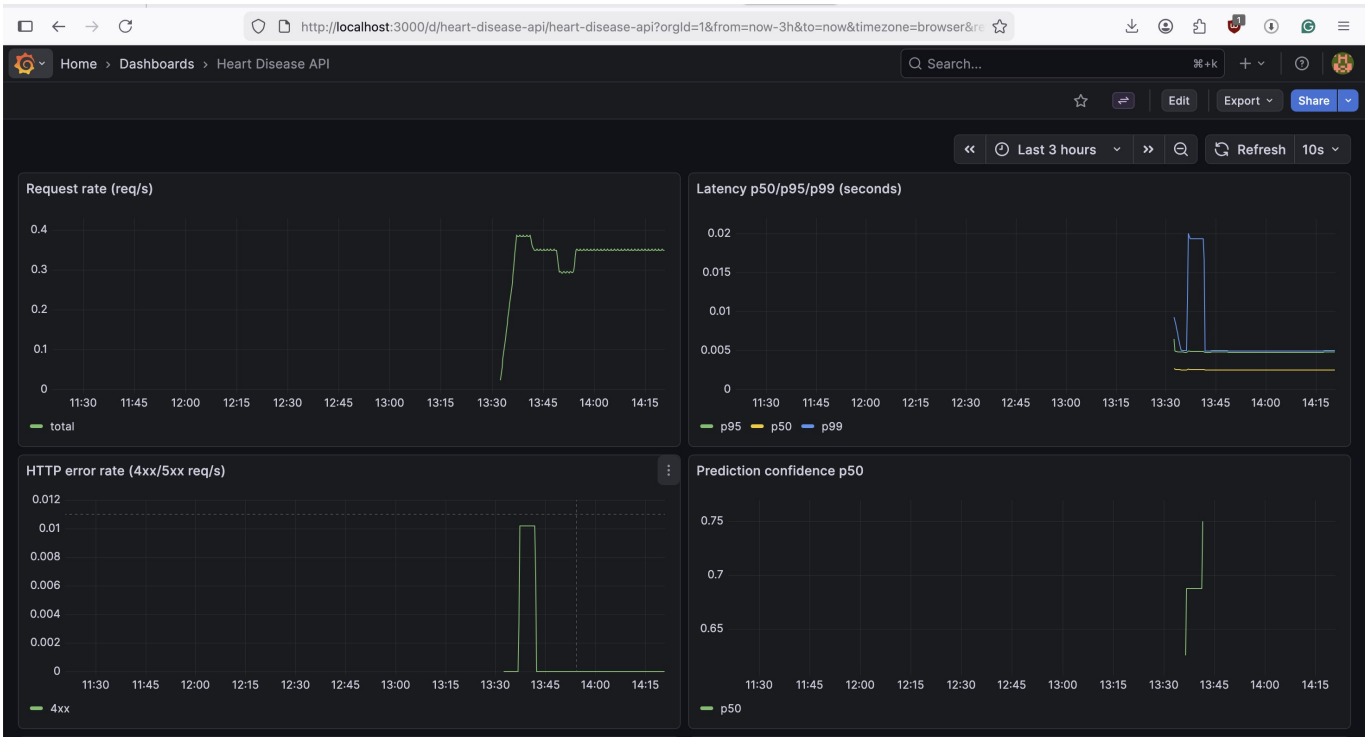
- Middleware request logging in: `src/api/app.py`

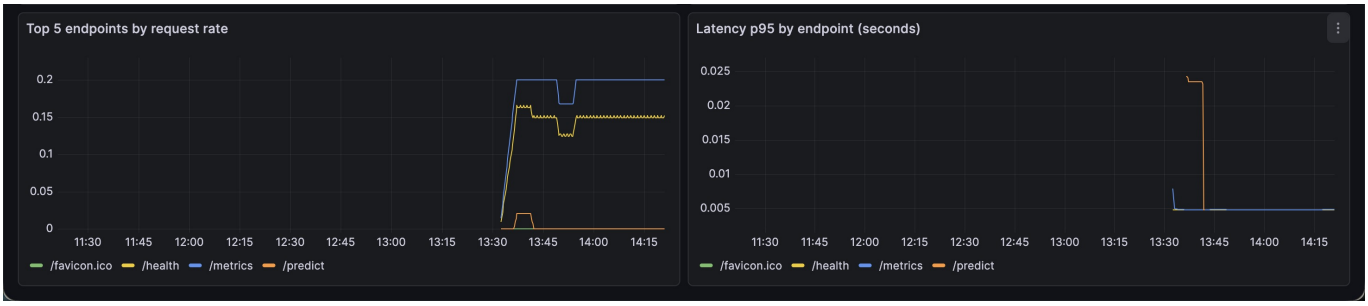
## 11.2 Metrics + Dashboards

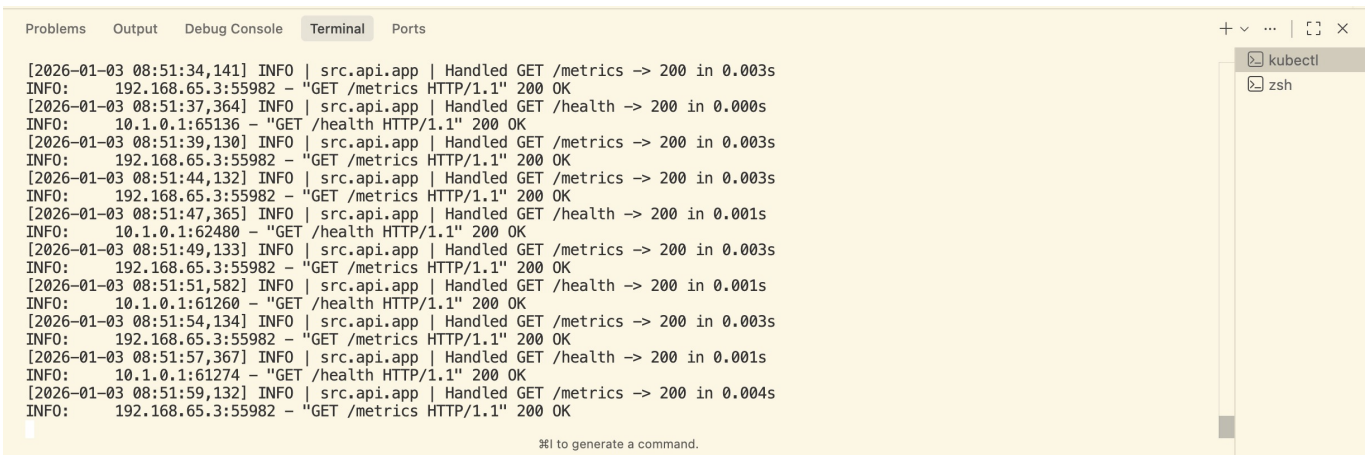
- Metrics endpoint: `/metrics`
- Prometheus config: `monitoring/prometheus.yml`
- Grafana provisioning + dashboard:
  - `monitoring/grafana/provisioning/`
  - `monitoring/grafana/dashboards/heart-disease-api-dashboard.json`

## Screenshot(s)



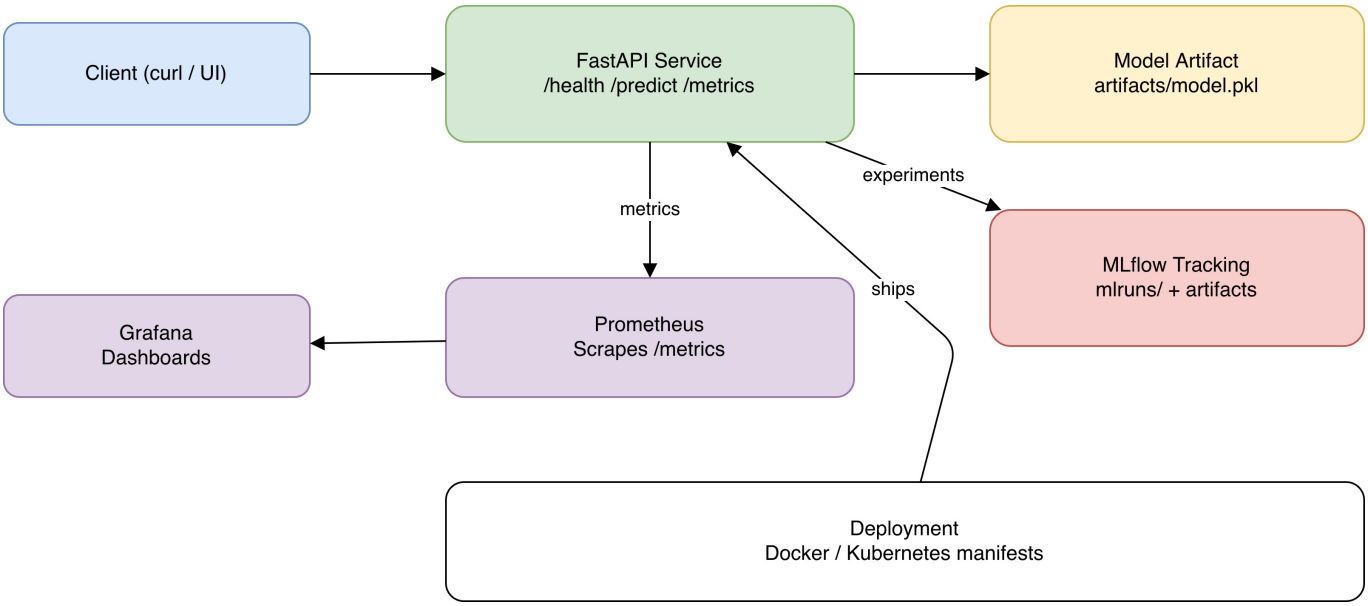

# 12. System Architecture Diagram

- Draw.io XML: `reports/system_architecture.xml`

- Architecture flow:

```
graph TD
  A[Client / UI] --> B[FastAPI Service]
  B --> C[Model Artifact joblib]
  B --> D[Prometheus Metrics]
  B --> E[Logging]
  E --> F[Stdout / Aggregator]
  D --> G[Grafana / Prometheus]
```

Screenshot(s)




# 13. How to Reproduce (Production-Readiness)

## 13.1 From clean environment

```
# 1. Clone and setup
git clone https://github.com/chaitudevi/Heart-Disease-Classification-2
cd Heart-Disease-Classification-2
python -m venv venv && source venv/bin/activate
pip install -r requirements.txt

# 2. Data pipeline
python -m src.data.download_data
python -m src.data.load_data

# 3. Train model (generates artifacts/model.pkl)
python -m src.models.train
# 4. Run API locally
uvicorn src.api.app:app --host 0.0.0.0 --port 8000

# 5. Test API
curl -X POST http://localhost:8000/predict -H "Content-Type: application/json" \
```

```
  -d @sample_data/sample_request.json

# 6. Run tests
./venv/bin/python -m pytest tests -v

# 7. Build container
docker build -t heart-disease-api .

# 8. Deploy to Kubernetes
kubectl apply -f k8s/deployment.yaml

# 9. Start monitoring stack
docker-compose -f docker-compose.monitoring.yml up -d

# 10. View MLflow experiments
mlflow ui --host 0.0.0.0 --port 5000
```

### 13.2 Key Config Files

- `monitoring/prometheus.yml`: Prometheus scrape config
- `monitoring/grafana/provisioning/`: Grafana auto-provision
- `k8s/deployment.yaml`: Kubernetes manifests
- `.github/workflows/github_actions.yaml`: CI/CD pipeline
- `src/models/train.py`: Training script with MLflow logging
- `src/api/app.py`: FastAPI app with metrics and logging

### 13.3 Ports & Endpoints

- API: http://localhost:8000 (or k8s service IP)
- Health: http://localhost:8000/health
- Metrics: http://localhost:8000/metrics
- Prometheus: http://localhost:9090
- Grafana: http://localhost:3000 (admin/admin123)
- MLflow: http://localhost:5000

## 14. Appendix: Evidence Checklist

- CI run screenshots
- Deployment screenshots
- Grafana dashboard screenshot
- MLflow runs screenshot