

MLOps Heart Disease Prediction

Group – 53

SOUMYA RANJAN MOHANTY : 2024AA05498

DHUPKAR SAURABH SHARACHCHANDRA : 2024AA05496

SRIDHAR R : 2024AA05357

TANMAY RASTOGI : 2024AA05742

SUMANGALA B S : 2023AC05941

1. Setup & Installation Instructions :

a) Ensure the following are installed on the system:

- Mini conda (or any other virtual environment system)
- Python **3.10+** (**Installed within environment cmd : conda create -n flops python -y , conda activate mlops**)
- Git
- Docker Desktop (with Kubernetes enabled)
- kubectl

b) pip install -r requirements.txt within the environment “mlops”

2. Exploratory Data Analysis (EDA) & Modelling Choices :

a) Exploratory Data Analysis (EDA):

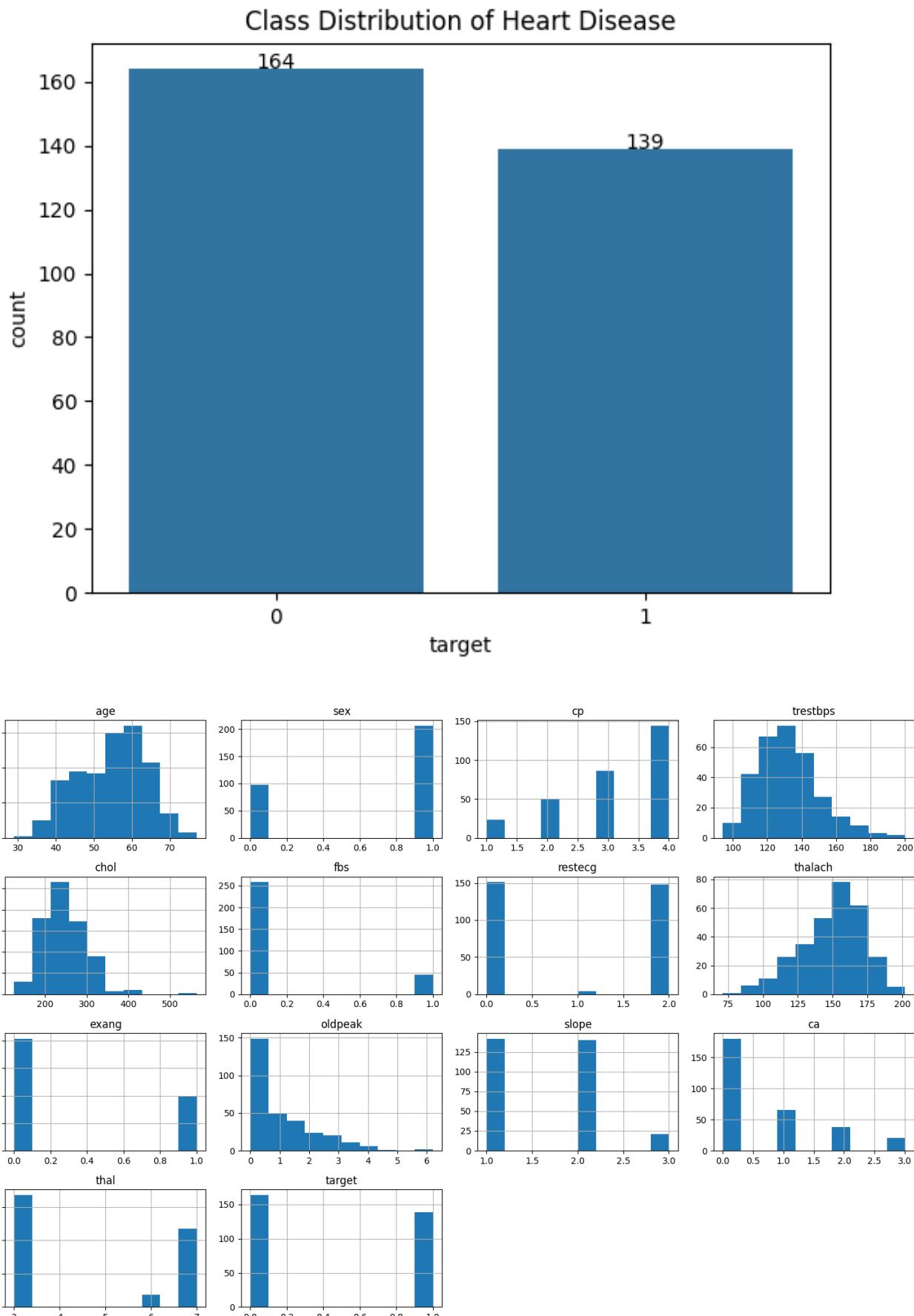
The initial exploratory data analysis was performed to understand the structure, distribution, and quality of the dataset.

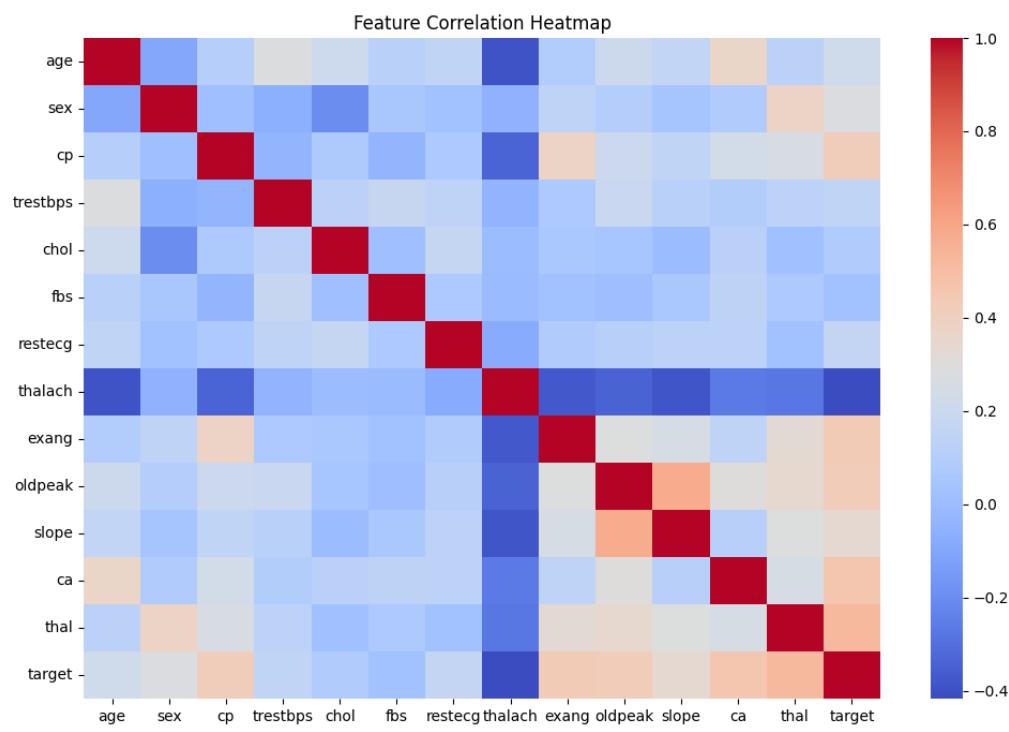
Key EDA observations:

- The dataset contains clinical and demographic attributes related to heart disease.
- The target variable (target) is binary, representing the presence or absence of heart disease.
- Numerical features such as age, chol, trestbps, thalach, and oldpeak exhibited varying scales and distributions.
- Categorical and ordinal features (e.g., cp, restecg, slope, thal) were already encoded numerically but represented discrete clinical categories.
- No missing values were present in the processed dataset, ensuring suitability for direct model training.

EDA conclusions:

- Feature scaling was required for continuous variables.
- Categorical and ordinal features could be used directly without one-hot encoding.





b) Modelling Choices

Two classification models were selected to balance interpretability and predictive performance:

1. Logistic Regression

- Chosen as a **baseline model** due to its simplicity and interpretability.
- Commonly used in medical and clinical prediction tasks.
- Provides a linear decision boundary and probabilistic outputs.
- Helps establish a benchmark for more complex models.

2. Random Forest Classifier

- Selected as a **non-linear ensemble model** capable of capturing complex feature interactions.
- Robust to feature scaling and noise.
- Provides improved predictive performance compared to linear models.

3. Feature Engineering & Preprocessing

- Preprocessing was implemented using a scikit-learn Pipeline combined with a ColumnTransformer.
- Continuous features were standardized using StandardScaler.
- Binary, ordinal, and discrete numerical features were passed through without transformation.
- This ensured consistent preprocessing during both training and inference.

4. Model Selection Strategy

- Models were evaluated using **cross-validation**.
- Multiple metrics were considered:
 - Accuracy
 - Precision
 - Recall
 - ROC-AUC
- A **multi-metric selection strategy** was applied, prioritizing ROC-AUC recall to account for the clinical importance of minimizing false negatives.

3. Experiment Tracking Using MLflow

MLflow was used to manage and track machine learning experiments throughout the model development process. This enabled systematic comparison of different models and ensured reproducibility of results.

a) Tracked Experiments :

Two independent experiments were conducted:

- 1. Logistic Regression**
- 2. Random Forest Classifier**

Each model training run was logged as a **separate MLflow run** under a common experiment.

b) Parameters Logged :

For each run, the following parameters were logged:

- Model type (e.g., Logistic Regression, Random Forest)
- Model-specific hyperparameters (e.g., max_iter, n_estimators)

c) Metrics Logged :

```

2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade bda7b8c39065 -> cbc13b556ace, add V3 trace schema columns
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade cbc13b556ace -> 770bee3ae1dd, add assessments table
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade 770bee3ae1dd -> a1b2c3d4e5f6, add spans table
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade a1b2c3d4e5f6 -> de4033877273, create entity_associations table
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade de4033877273 -> 1a0cddfcaa16, Add webhooks and webhook_events tables
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade 1a0cddfcaa16 -> 534353b11cbc, add scorer tables
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade 534353b11cbc -> 71994744cf8e, add evaluation datasets
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade 71994744cf8e -> 3da73c924c2f, add outputs to dataset record
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade 3da73c924c2f -> bf29a5ff90ea, add jobs table
2026/01/06 00:18:44 INFO alembic.runtime.migration: Running upgrade bf29a5ff90ea -> 1bd49d398cd23, add secrets tables
2026/01/06 00:18:44 INFO alembic.runtime.migration: Context impl SQLiteImpl.
2026/01/06 00:18:44 INFO alembic.runtime.migration: Will assume non-transactional DDL.
2026/01/06 00:18:44 INFO mlflow.store.db.utils: Creating initial MLflow database tables...
2026/01/06 00:18:44 INFO mlflow.store.db.utils: Updating database tables
2026/01/06 00:18:44 INFO alembic.runtime.migration: Context impl SQLiteImpl.
2026/01/06 00:18:44 INFO alembic.runtime.migration: Will assume non-transactional DDL.
[MLflow] Security middleware enabled with default settings (localhost-only). To allow connections from other hosts, use --host 0.0.0.0 and configure --allowed-hosts and --cors-allowed-origins.
INFO: Uvicorn running on http://127.0.0.1:5000 (Press CTRL+C to quit)
INFO: Started parent process [60348]
INFO: Started server process [60354]
INFO: Waiting for application startup.
INFO: Started server process [60351]
INFO: Waiting for application startup.
INFO: Started server process [60353]
INFO: Waiting for application startup.
INFO: Application startup complete.

```

Each experiment recorded the following evaluation metrics using cross-validation:

- Accuracy
- Precision
- Recall
- ROC-AUC

These metrics allowed objective comparison between models based on both predictive performance and clinical relevance.

d) Artifacts Logged

MLflow automatically stored:

- The trained model pipeline (including preprocessing steps)
- Environment metadata required to reload the model

This ensured that each experiment could be reproduced exactly at a later time.

e) Model Comparison & Selection

- MLflow's UI was used to compare runs visually.
- A multi-metric evaluation strategy was applied to select the best-performing model.
- The final selected model was serialized separately as a production-ready artifact (final_model.pkl).

1. Prefer higher ROC-AUC
2. If close (± 0.02), choose higher Recall
3. If still close, choose higher Precision

f) Benefits of Using MLflow

- Centralized experiment tracking
- Clear separation of training runs
- Reproducibility through logged artifacts
- Transparent comparison of model performance

g) Reproducibility

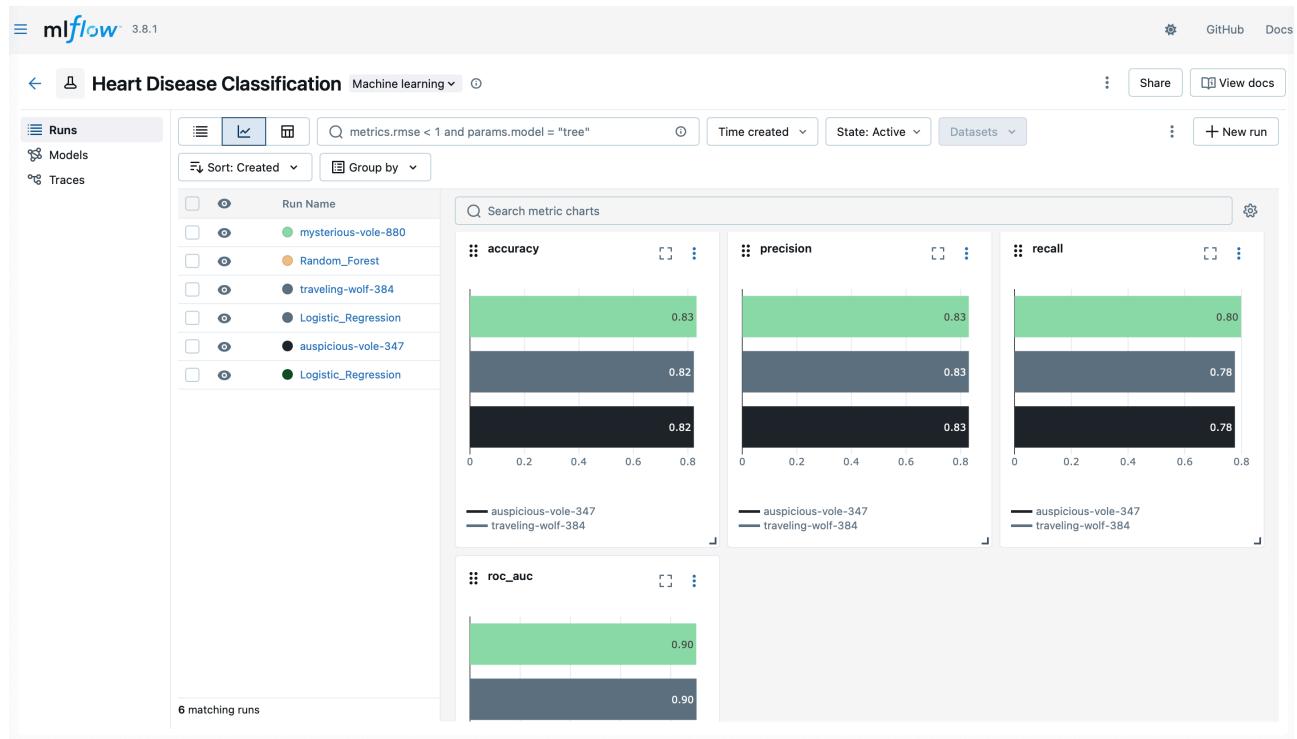
Because preprocessing, training, and evaluation were encapsulated in a single pipeline and logged using MLflow, the experiments can be reliably reproduced across environments.

3. Architecture Diagram:

The architecture follows a modular MLOps design. Raw data is first processed using a reproducible preprocessing pipeline implemented with scikit-learn.

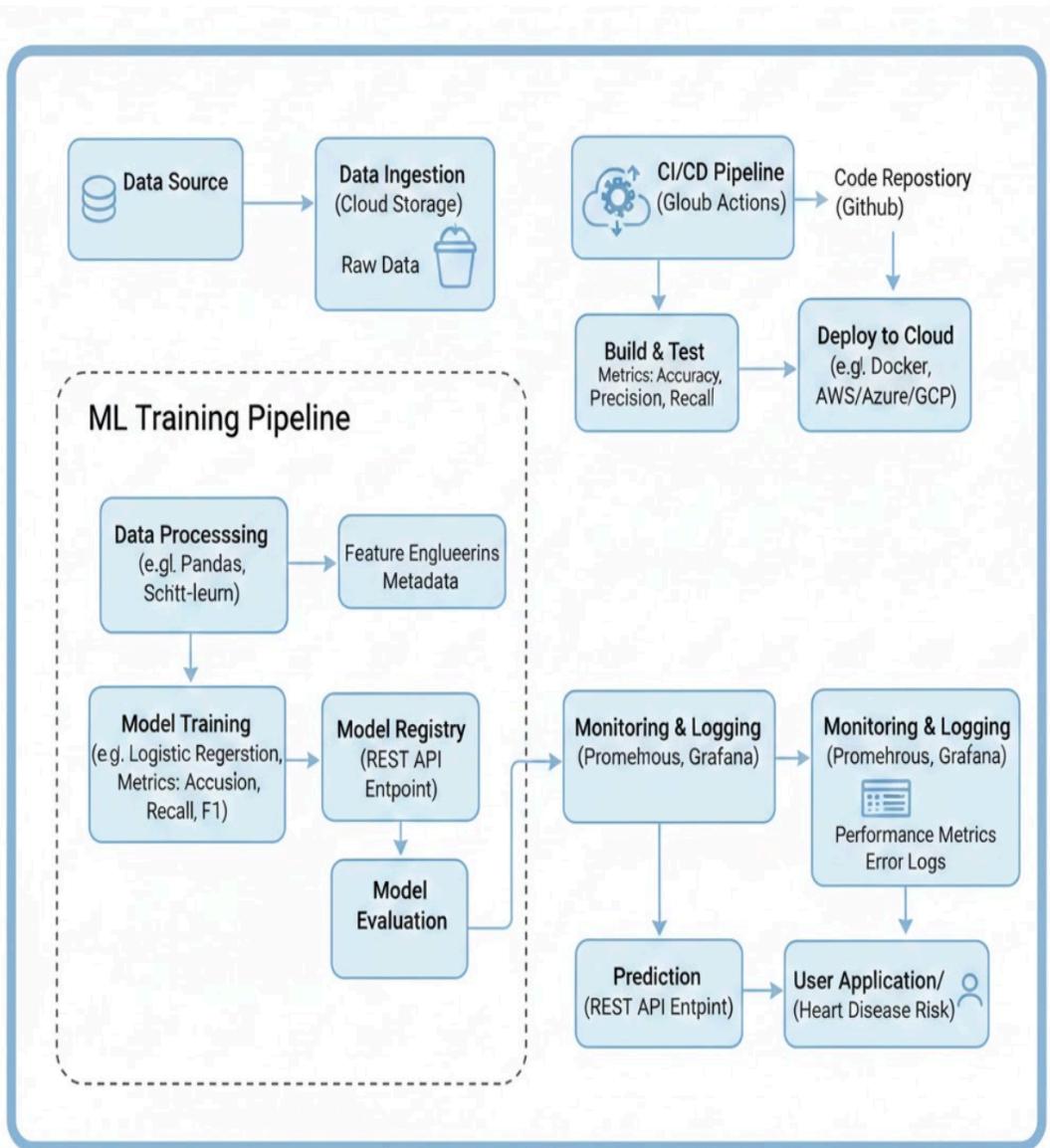
Multiple machine learning models are trained and tracked using MLflow, which logs parameters, metrics, and artifacts. A multi-metric evaluation strategy selects the best performing model, which is serialized as a standalone artifact.

A CI/CD pipeline implemented using GitHub Actions automates testing, training, and artifact generation.



The final model is served using a FastAPI application, containerized with Docker, and deployed to a Kubernetes cluster.

Monitoring is enabled through application logs, a metrics endpoint, and optional Prometheus integration, providing observability into runtime behavior.



5. CI/CD and deployment workflow screenshots.

The screenshot shows the GitHub Actions interface for the 'mlops-heart-disease' repository. On the left, there's a sidebar with navigation links like 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The main area is titled 'All workflows' and shows '15 workflow runs'. The runs are listed in descending order of age. Each run includes information such as the commit message, branch (main), event time, status (green for success, red for failure), and duration. The runs include tasks like updating the README, adding screenshots, updating workspace-related changes, updating mlflow scripts, changing os path, adding logging and metrics page, and performing K8s deployment.

```
(mlops) soumya@Soumyas-MacBook-Air mlops-heart-disease % docker build -t heart-disease-api .
[+] Building 56.4s (12/12) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 409B
=> [internal] load metadata for docker.io/library/python:3.10-slim
=> [auth] library/python:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/6] FROM docker.io/library/python:3.10-slim@sha256:7b68a5fa7cf0d20b4cedb1dc9a134fdd394fe27edbc4c2519756c91d21df2
=> => resolve docker.io/library/python:3.10-slim@sha256:7b68a5fa7cf0d20b4cedb1dc9a134fdd394fe27edbc4c2519756c91d21df2
=> => sha256:120d040cf64151012b1a116b1e2cb2a30615d5ac476546c37b74ef396e770a66 249B / 249B
=> => sha256:b432ca87a2f064b7ee413f636db60e8cfaf6ebc9bb007cd8fdf6e2bbbc2 13.78MB / 13.78MB
=> => sha256:01116e55a56dc7760137538145b313cc63db9515e7161784122173b961226ba 1.27MB / 1.27MB
=> => sha256:2ae15a20160209c6fd6cff4886e4ba2e666fa5bedd7b54a2c0097ea6646f0273 30.14MB / 30.14MB
=> => extracting sha256:2ae15a20160209c6fd6cff4886e4ba2e666fa5bedd7b54a2c0097ea6646f0273
=> => extracting sha256:01116e55a56cd7760137538145b313cc63db9515e7161784122173b961226ba
=> => extracting sha256:b432ca87a2f064b7ee413f636db60e8cfaf6ebc9bb007cd8fdf6e2bbbc2
=> => extracting sha256:120d040cf64151012b1a116b1e2cb2a30615d5ac476546c37b74ef396e770a66
=> [internal] load build context
=> => transferring context: 1.71MB
=> [2/6] WORKDIR /app
=> [3/6] COPY requirements.txt .
=> [4/6] RUN pip install --no-cache-dir -r requirements.txt
=> [5/6] COPY api/ api/
=> [6/6] COPY artifacts/ artifacts/
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:519506951864852eed190596f17ccb5f21a5f22d85971aa1f369366617b17924
=> => exporting config sha256:96bb2640183d22997398aedc4f38c2069b0b5c8f4d0eeaee1e8485ab13ec827c
=> => exporting attestation manifest sha256:401478a956de8e6c65309125d3e93e5878525d8513975a79a4f1bff572ccfec
=> => exporting manifest list sha256:bf7a9b6d0b0f17b6904722f6ed9e5cdac2c7cf0b5cb1bab47ebdf01657f7744d
=> => naming to docker.io/library/heart-disease-api:latest
=> => unpacking to docker.io/library/heart-disease-api:latest
(mlops) soumya@Soumyas-MacBook-Air mlops-heart-disease %
(mlops) soumya@Soumyas-MacBook-Air mlops-heart-disease %
(mlops) soumya@Soumyas-MacBook-Air mlops-heart-disease %
(mlops) soumya@Soumyas-MacBook-Air mlops-heart-disease %
```

```

● (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease % kubectl config current-context
  docker-desktop
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
● (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease % kubectl get services
  NAME          TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
  heart-disease-service   NodePort    10.111.72.156   <none>        80:30007/TCP   3m49s
  kubernetes       ClusterIP    10.96.0.1       <none>        443/TCP      16m
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %

  deployment.apps/heart-disease-api created
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
● (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease % kubectl get deployments
  NAME          READY  UP-TO-DATE  AVAILABLE  AGE
  heart-disease-api  1/1     1         1        18s
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
● (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease % kubectl get pods
  NAME          READY  STATUS  RESTARTS  AGE
  heart-disease-api-67c45ddd78-qndtv  1/1     Running   0        26s
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %

```

```

● (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease % kubectl get pods
  NAME          READY  STATUS  RESTARTS  AGE
  heart-disease-api-67c45ddd78-qndtv  1/1     Running   0        12m
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %

```

```

● (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease % kubectl get nodes -o wide
  NAME          STATUS  ROLES      AGE   VERSION      INTERNAL-IP      EXTERNAL-IP      OS-IMAGE      KERNEL-VERSION      CO
  NTAINER-RUNTIME
  docker-desktop  Ready   control-plane  13m  v1.34.1    192.168.65.3   <none>        Docker Desktop  6.12.54-linuxkit  do
  cker://29.1.3
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
● (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease % curl -X POST "http://localhost:30007/predict" \
-H "Content-Type: application/json" \
-d '{
  "age": 55,
  "sex": 1,
  "cp": 2,
  "trestbps": 130,
  "chol": 250,
  "fbs": 0,
  "restecg": 1,
  "thalach": 150,
  "exang": 0,
  "oldpeak": 1.5,
  "slope": 2,
  "ca": 0,
  "thal": 3
}'
{"prediction":0,"probability":0.15}
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %
○ (mllops) soumya@Soumyas-MacBook-Air mllops-heart-disease %

```

6. Link to code repository :

<https://github.com/soumyaranmohanty/mllops-heart-disease>

7. Video Link :

