Lab 6 - MLOps CI Pipeline/Workflow with GitHub Actions

Create a Github Workflow for Streamlit

To understand how GitHub workflow works, lets create a simple pipeline for Streamlit app which will

- Trigger when code inside `streamlit_app/` path changes or optionally be triggered manually.
- Build a container image by reading `streamlit_app/Dockerfile`
- Login and publish the image to DockerHub Container Registry

```
cd house-price-predictor
mkdir -p .github/workflows
```

File : `.github/workflows/streamlit-ci.yaml`

```
name: Streamlit CI

on:
  push:
    paths:
      - 'streamlit_app/**'
  workflow_dispatch:

jobs:
  build-and-push:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout code
        uses: actions/checkout@v4

      - name: Set up Docker Buildx
        uses: docker/setup-buildx-action@v3

      - name: Log in to DockerHub Container Registry
```

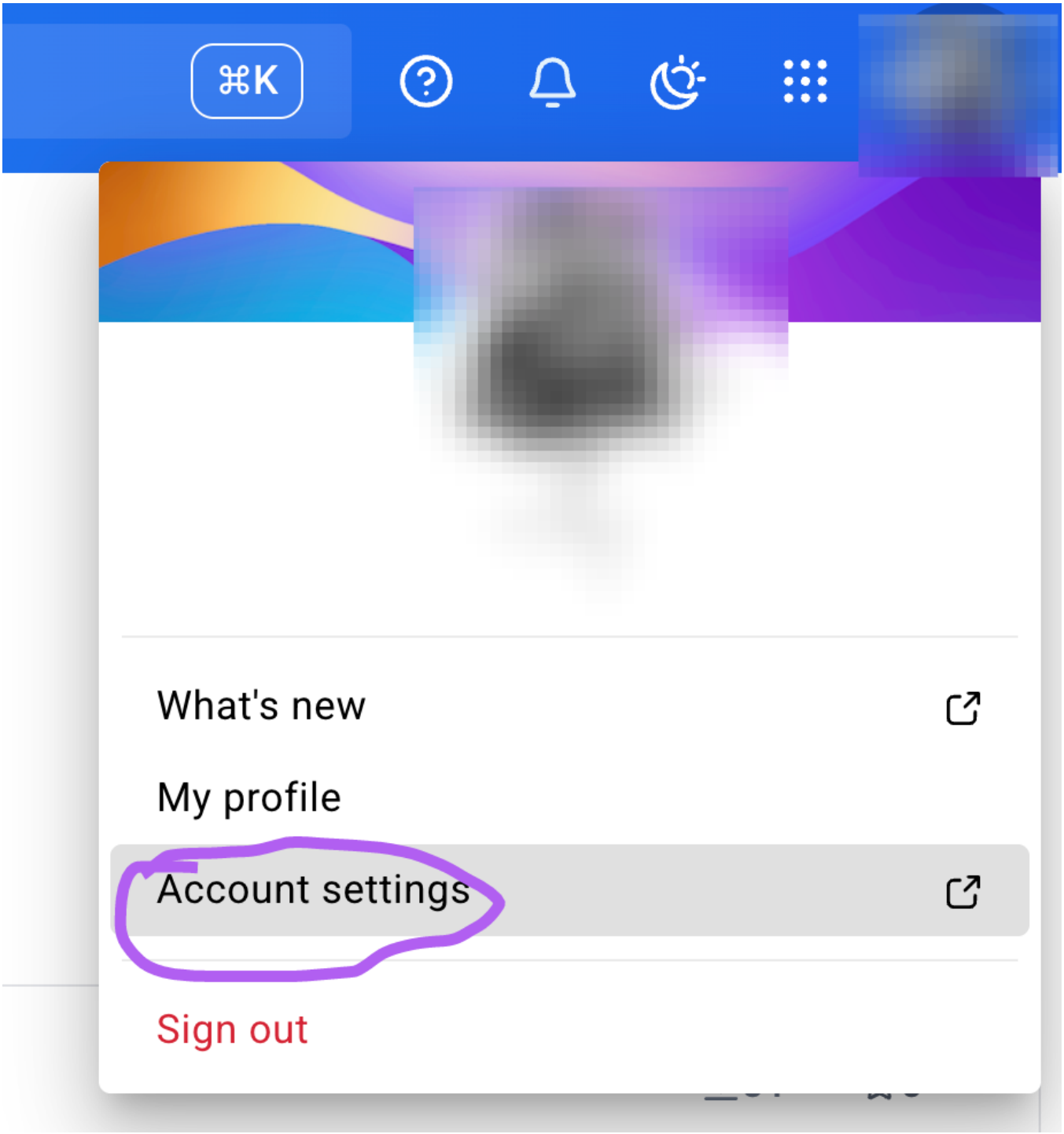
```
        uses: docker/login-action@v3
        with:
          registry: docker.io
          username: ${{ vars.DOCKERHUB_USERNAME }}
          password: ${{ secrets.DOCKERHUB_TOKEN }}

      - name: Build and push Docker image
        uses: docker/build-push-action@v5
        with:
          context: ./streamlit_app
          push: true
          tags: docker.io/${{ vars.DOCKERHUB_USERNAME }}/streamlit:latest
```

```
git add .github/workflows/streamlit-ci.yaml
git commit -am "adding CI workflow for streamlit"
git push origin main
```

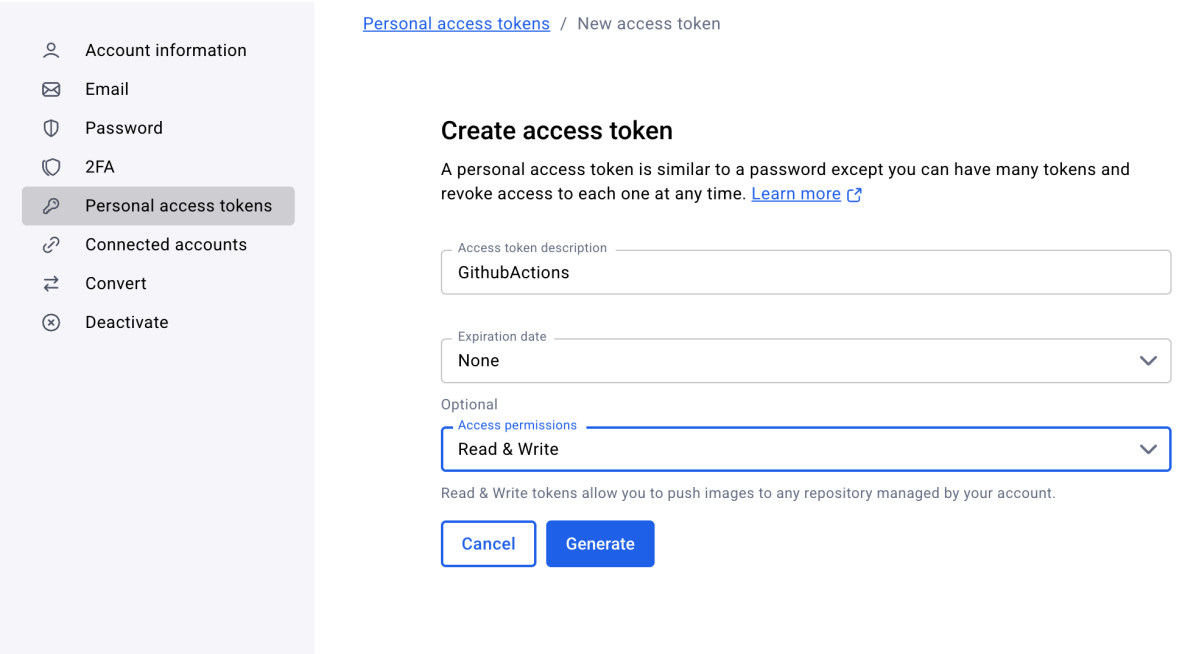
Once committed and pushed you are ready to trigger the workflow/pipeline from GitHub ➜ Actions. However, you still need to add the authentication required to connect to DockerHub to pubish the image with. Lets add that first.

Setting up Registry Credentials for Pipeline Builds

Login to Dockerhub and brose to Account Settings ➜ Personal Access Tokens ➜ Generate Token. You could also visit the page directly at https://app.docker.com/settings/personal-access-tokens/create

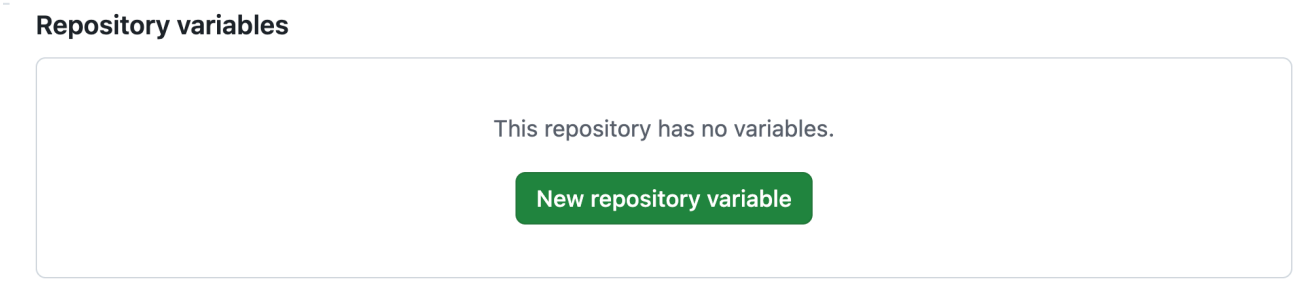


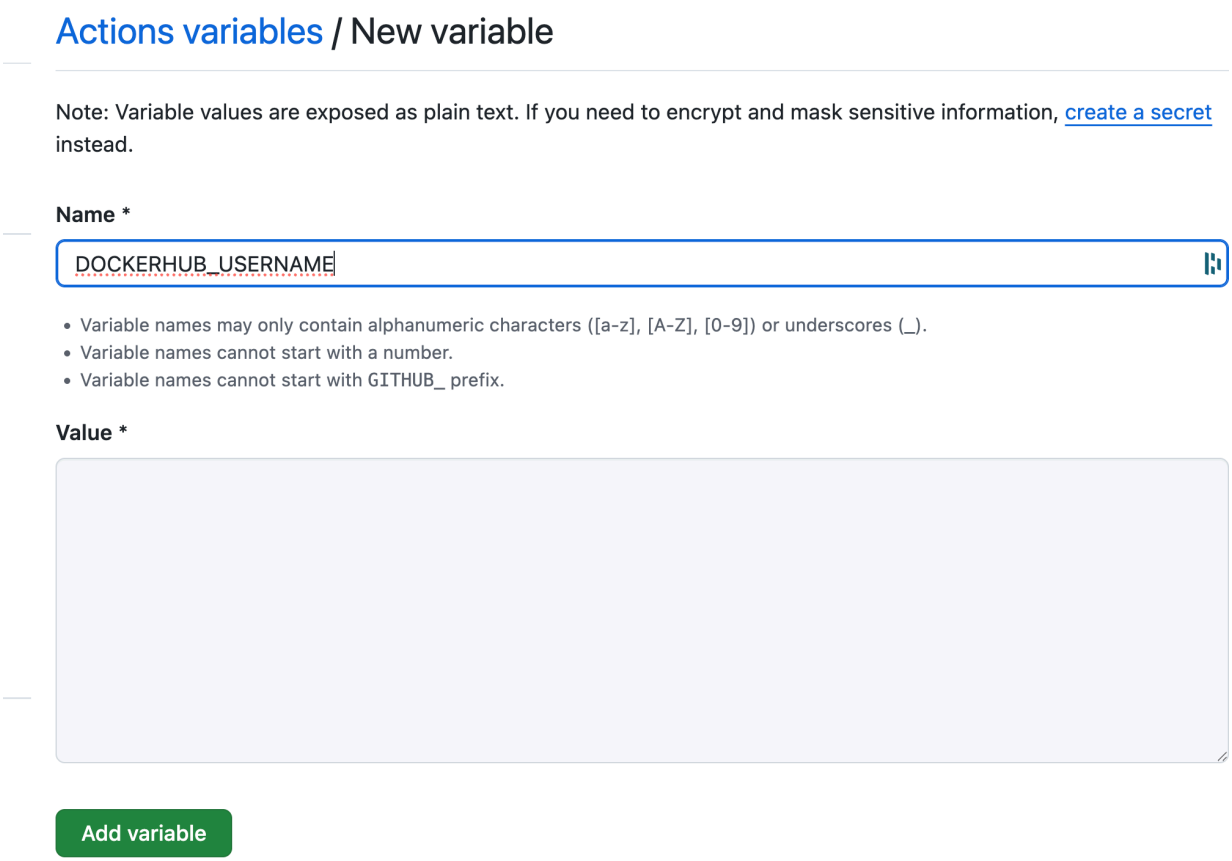
Generate the token with `Read & Write` access, copy it over.



Now from GitHub Repository Settings, select `Secrets and Variables` ➜ Actions ➜ New Repository Variable

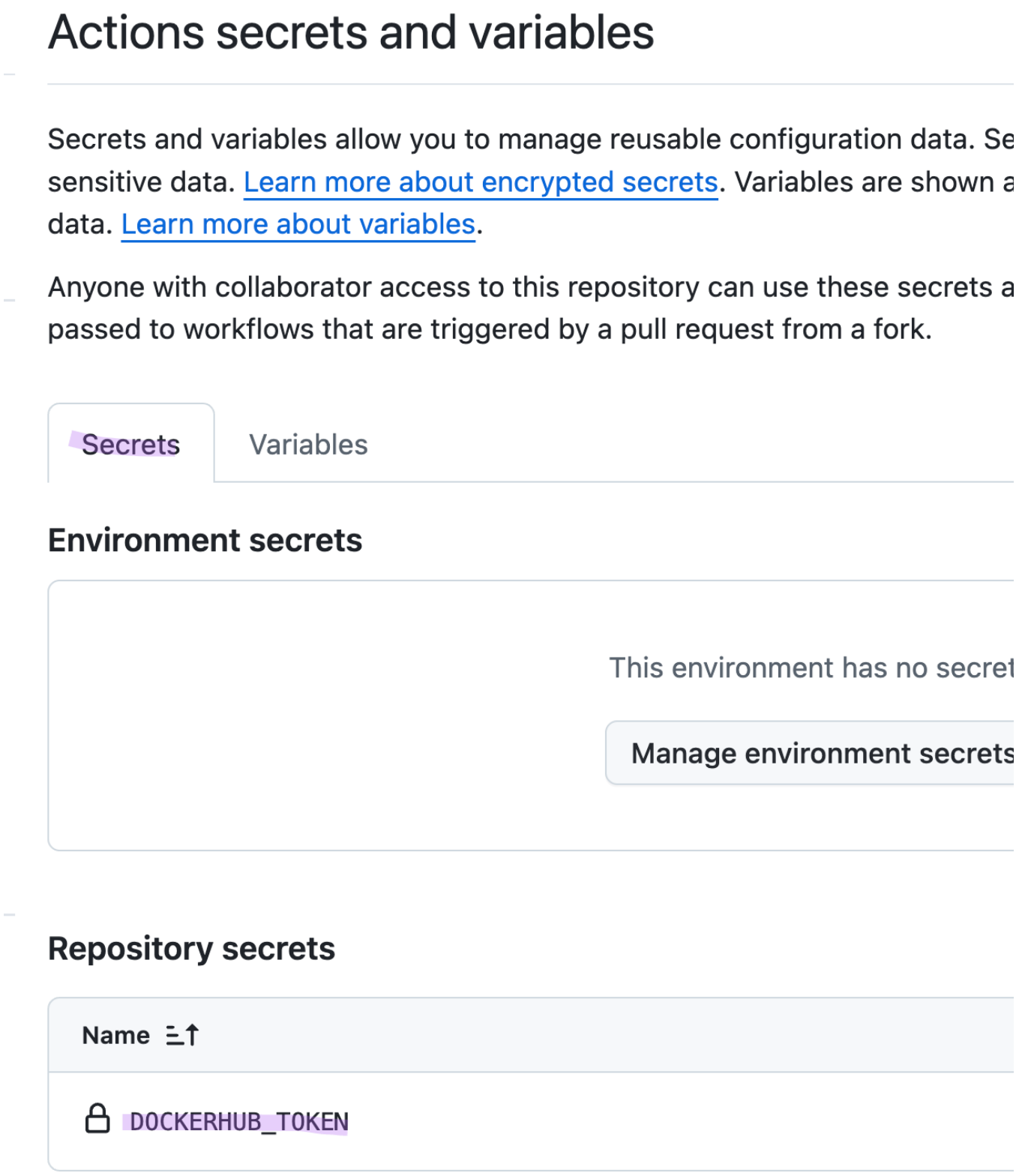
and add the variable `DOCKERHUB_USERNAME` with your actual docker id.





Note : Make sure its **Repository Variable**.

Also add the repository secret (token copied from DockerHub) which should be set as `DOCKERHUB_TOKEN`



Once you are done with this, enable and trigger the Streamlit CI pipeline from GitHub ➜ Actions.

Setup MLOps CI Pipeline with GitHub Actions

Create the MLOPs workflow at `.github/workflows/mlops-pipeline.yml`

```
# .github/workflows/mlops-pipeline.yml
name: MLOps Pipeline

on:
  push:
    branches: [ main ]
    tags: [ 'v*.*.*' ]
  pull_request:
    branches: [ main ]

jobs:
  data-processing:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v2

    - name: Set up Python
      uses: actions/setup-python@v2
      with:
        python-version: '3.11.9'

    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        pip install -r requirements.txt

    - name: Process data
      run: |
        python src/data/run_processing.py --input data/raw/house_data.csv --
output data/processed/cleaned_house_data.csv

    - name: Engineer features
      run: |
        python src/features/engineer.py --input data/processed/
cleaned_house_data.csv --output data/processed/featured_house_data.csv --
preprocessor models/trained/preprocessor.pkl

    - name: Upload processed data
```

```
      uses: actions/upload-artifact@v4
      with:
        name: processed-data
        path: data/processed/featured_house_data.csv

    - name: Upload preprocessor
      uses: actions/upload-artifact@v4
      with:
        name: preprocessor
        path: models/trained/preprocessor.pkl

  model-training:
    needs: data-processing
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v2

    - name: Set up Python
      uses: actions/setup-python@v2
      with:
        python-version: '3.11.9'

    - name: Install dependencies
      run: |
        python -m pip install --upgrade pip
        pip install -r requirements.txt

    - name: Download processed data
      uses: actions/download-artifact@v4
      with:
        name: processed-data
        path: data/processed/

    - name: Set up MLflow
      run: |
        docker pull ghcr.io/mlflow/mlflow:latest
        docker run -d -p 5000:5000 --name mlflow-server ghcr.io/mlflow/
mlflow:latest mlflow server --host 0.0.0.0 --backend-store-uri sqlite:///
mlflow.db

    - name: Wait for MLflow to start
```

```
      run: |
        for i in {1..10}; do
          curl -f http://localhost:5000/health || sleep 5;
        done

    - name: Train model
      run: |
        mkdir -p models
        python src/models/train_model.py --config configs/model_config.yaml --
data data/processed/featured_house_data.csv --models-dir models --mlflow-
tracking-uri http://localhost:5000

    - name: Upload trained model
      uses: actions/upload-artifact@v4
      with:
        name: trained-model
        path: models/

    - name: Clean up MLflow
      run: |
        docker stop mlflow-server || true
        docker rm mlflow-server || true

  build-and-publish:
    needs: model-training
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v2

    - name: Download trained model
      uses: actions/download-artifact@v4
      with:
        name: trained-model
        path: models/

    - name: Download preprocessor
      uses: actions/download-artifact@v4
      with:
        name: preprocessor
        path: models/trained/
```

```
    - name: Set up Docker Buildx
      uses: docker/setup-buildx-action@v3


    - name: Log in to DockerHub Container Registry
      uses: docker/login-action@v3
      with:
        registry: docker.io
        username: ${{ vars.DOCKERHUB_USERNAME }}
        password: ${{ secrets.DOCKERHUB_TOKEN }}


    - name: Build and push Docker image
      uses: docker/build-push-action@v5
      with:
          context: .
          file: ./Dockerfile
          push: true
          tags: docker.io/${{ vars.DOCKERHUB_USERNAME }}/house-price-
model:latest
```

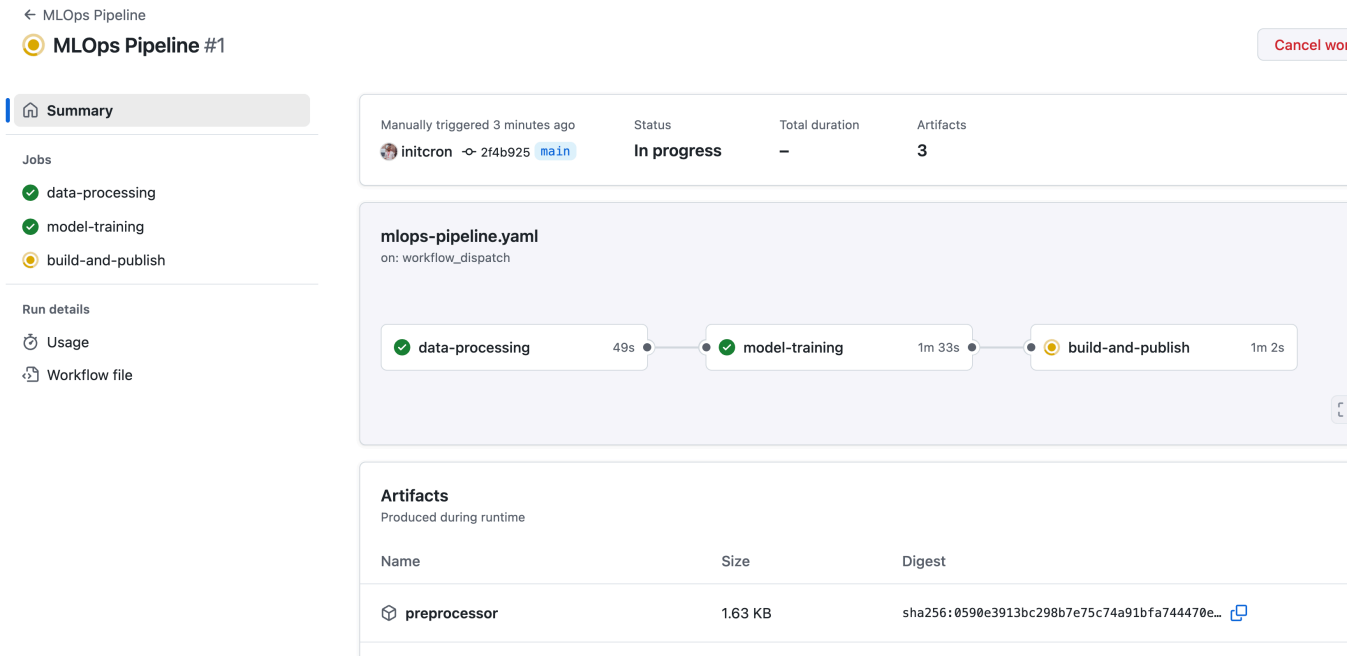
Source : mlops-pipeline

Commit the changes

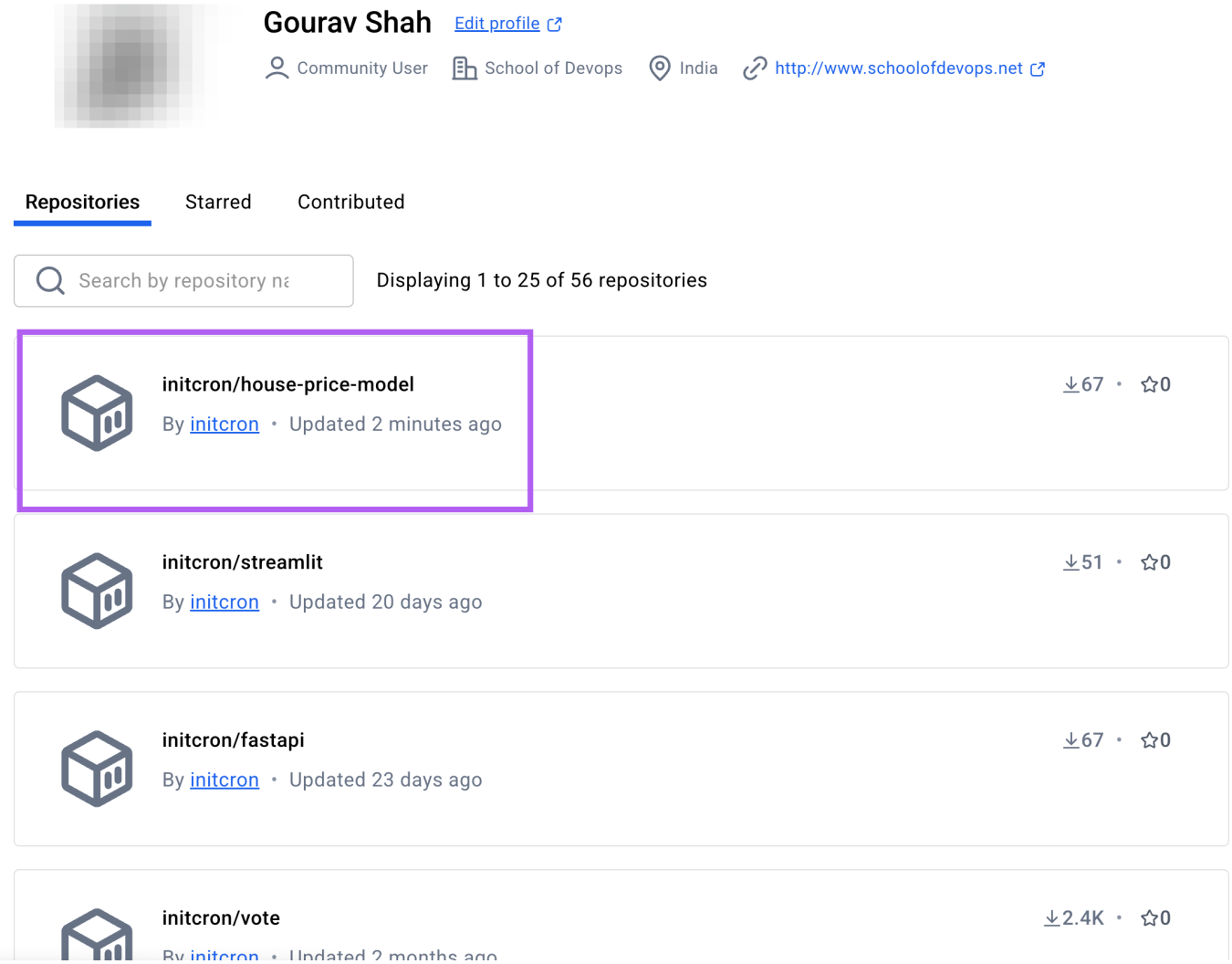
```
git add .github/workflows/mlops-pipeline.yml
git commit -am "adding github mlops pipeline workflow"
git push origin main
```

Once you push the changes, go to GitHub Repo ➜ Actions.

You will see the complete Pipeline from Data Processing ➜ Model Training ➜ Build and Publish execute, just like any other CI Pipeline, just with a differnt type of application i.e. a Machine Learning Model wrapped into FastAPI.



You will also see a new image published for the packaged model on the registry.



Congratulations for setting up a complete MLOps CI Workflow.

#courses/mlops