User Manual For EmotionLens

EmotionLens is an AI application that detects emotions from facial images. It categorizes emotions into seven categories: anger, disgust, fear, happiness, sadness, surprise, and neutral.

GitHub Link for this Project: https://github.com/HSF007/EmotionLens

Features

- Data versioning with DVC
- Automated ML pipeline for data preprocessing, model training, and evaluation
- Experiment tracking with MLflow
- Model deployment via FastAPI
- Monitoring with Prometheus and Grafana

App Structure

```
EmotionLens/
├── data/              # Data directory (tracked by DVC)
├── src/               # Source code
├── app/               # API application
├── monitoring/        # Monitoring configuration
├── models/            # Model storage
├── dvc.yaml           # DVC pipeline configuration
├── params.yaml        # Parameters for the pipeline
├── requirements.txt   # Python dependencies
└── Dockerfile
```

Setup

1. Clone the repository:

```
git clone https://github.com/HSF007/EmotionLens.git
cd EmotionLens
```

1. Create a virtual environment:

```
python -m venv venv
source venv/bin/activate
# On Windows: venv\Scripts\activate
```

1. Install dependencies:

```
pip install -r requirements.txt
```

1. Initialize DVC:

```
dvc init
```

Data Structure

Place your emotion dataset in the `data` directory with the following structure:

```
data/
├── raw/
    ├── train/
    │   ├── anger/
    │   ├── disgust/
    │   ├── fear/
    │   ├── happiness/
    │   ├── sadness/
    │   ├── surprise/
    │   └── neutral/
    ├── test/
    └── validation/
├── processed/
```

Each emotion folder should contain facial images displaying that emotion.

- For training model we have used data from: https://www.kaggle.com/datasets/geolek/grayscale-face-images

User Manual for Developers

Running the Pipeline

Execute the entire ML pipeline:

```
dvc repro
```

This will run:

1. Data preprocessing
2. Model training
3. Model evaluation
4. Deployment of the best model

Viewing Experiments

Start the MLflow UI:

```
mlflow ui
```

Visit http://localhost:5000 to view experiment results.

Running the API

Start the FastAPI application:

```
cd app
uvicorn main:app --reload
```

The API will be available at http://localhost:8000

Monitoring

1. Start Prometheus:

```
docker run -d --name prometheus -p 9090:9090 -v $(pwd)/monitoring/pro
metheus/prometheus.yml:/etc/prometheus/prometheus.yml prom/prometh
eus
```

1. Start Grafana:

```
docker run -d --name grafana -p 3000:3000 grafana/grafana
```

1. Import the dashboard from `monitoring/grafana/dashboard.json`

Visit http://localhost:3000 to access the Grafana dashboard.

Using Docker

The entire application is containerized using Docker Compose, which sets up:

- The EmotionLens application
- Prometheus for metrics collection
- Grafana for monitoring dashboards

Option 1: Quick Start with Docker Compose

1. Run the complete pipeline:

```
chmod +x run-pipeline.sh
./run-pipeline.sh
```

This script will:

- Build the Docker image
- Run data preprocessing, model training, and evaluation
- Start all services

1. Access the services:

- EmotionLens API: http://localhost:8000/statci/index.html
- Prometheus: http://localhost:9090
- Grafana: http://localhost:3000 (login with admin/admin)

Option 2: Manual Docker Setup

1. Build and start the services:

```
docker-compose build
docker-compose up -d
```

1. Run the ML pipeline:

```
docker-compose exec app python src/data_preprocessing.py
docker-compose exec app python src/train.py
docker-compose exec app python src/evaluate.py
```

Stopping the Application

```
docker-compose down
```

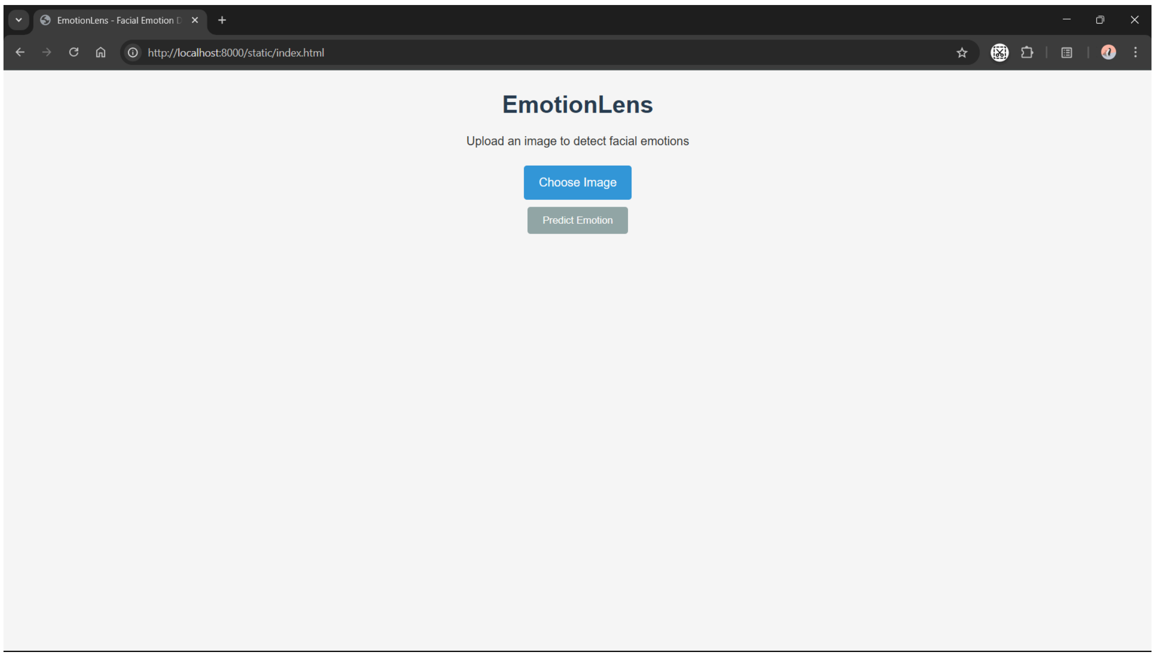
Persisted Data

The following data is persisted through Docker volumes:

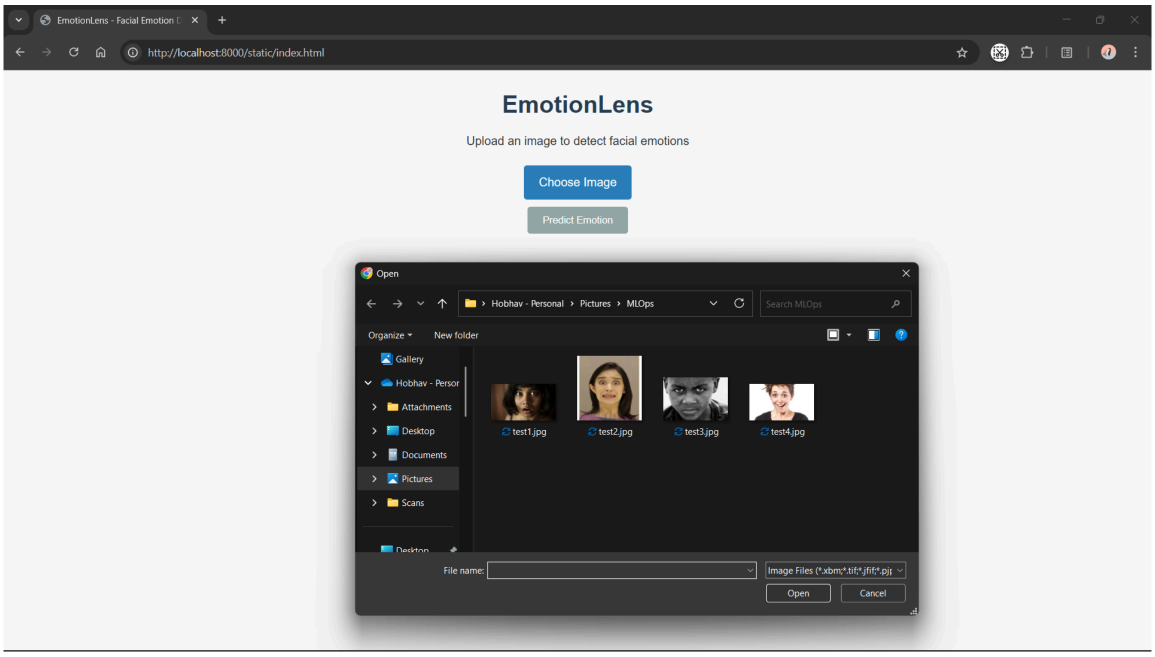
- ML models: `./models` directory
- Training data: `./data` directory
- MLflow tracking: `./mlruns` directory
- Prometheus data: Docker volume `prometheus-data`
- Grafana data: Docker volume `grafana-data`

Application User Manual for Application Users

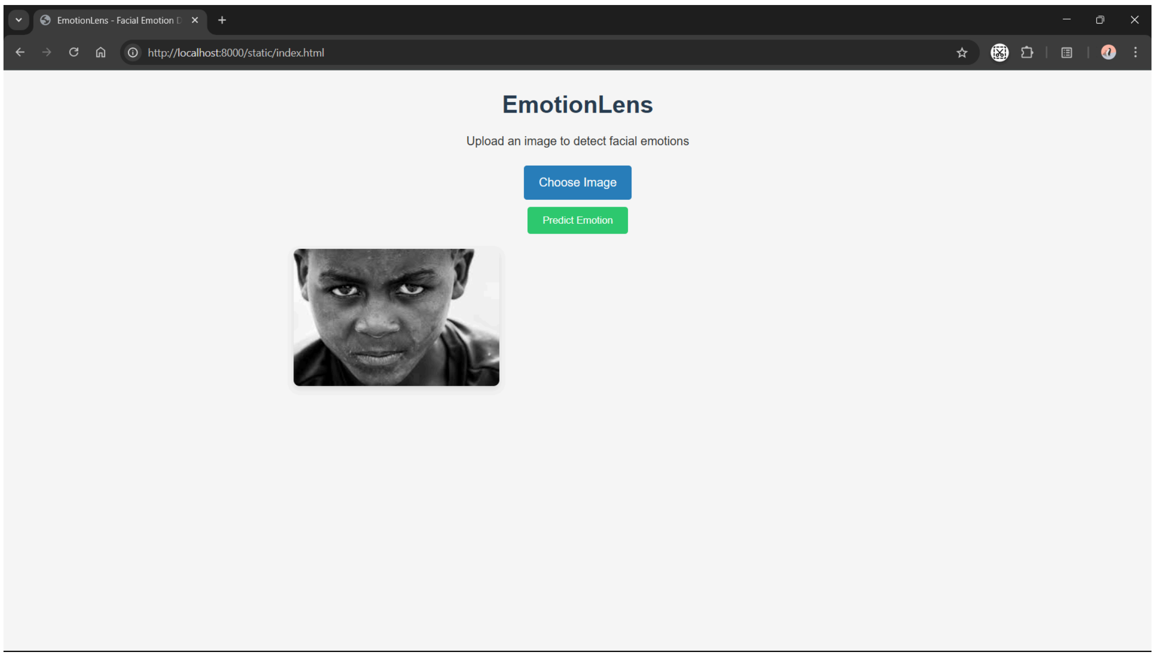
- After running Application in any way given above
  - Mostly by `docker-compose build` and `docker-compose up -d`
- Then you can access the application and it will be as shown below


EmotionLens - Facial Emotion D
http://localhost:8000/static/index.html
EmotionLens
Upload an image to detect facial emotions
Choose Image
Predict Emotion

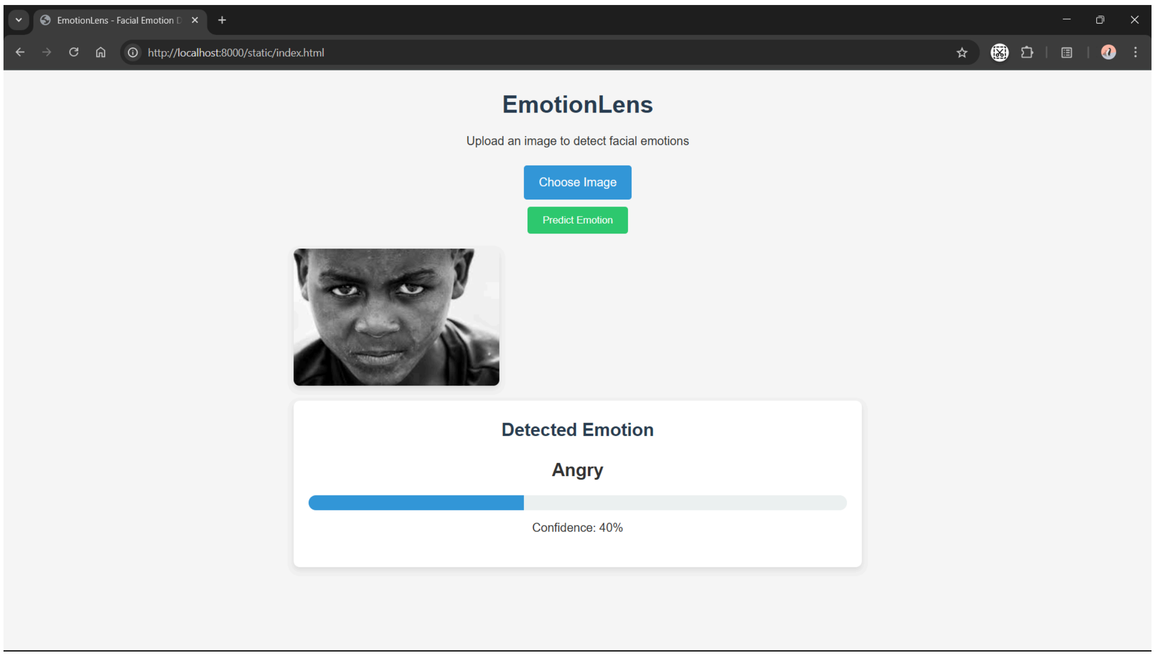
- Then click on `Choose Image` which will open your file manager shown below



- Then just select a picture of your liking the image should be selected


EmotionLens - Facial Emotion D
http://localhost:8000/static/index.html
EmotionLens
Upload an image to detect facial emotions
Choose Image
Predict Emotion

- Then click on `predict Emotion` to get prediction with confidence value.


EmotionLens - Facial Emotion D
http://localhost:8000/static/index.html
EmotionLens
Upload an image to detect facial emotions
Choose Image
Predict Emotion

Detected Emotion

Angry

Confidence: 40%

- To close application just do

  `docker compose down` in your terminal in the same folder where you have done `docker compose up`.