Smarter Reconciliation Anomaly Detection System - Runbook

🔧 Project Overview

This project provides a FastAPI-powered real-time and batch anomaly detection system for financial reconciliation data. It integrates ML models, LLMs for categorization/resolution, WebSockets for live updates, and email notifications.

🗂️ Project Structure

econ\_anomaly\_detection/

├── api/

│ ├── \_\_init\_\_.py

│ ├── anomalydetection.py

│ ├── models.py

│ └── websocket\_manager.py

├── config/

│ └── config.yaml

├── data/

│ └── generated\_reconciliation\_data.csv

├── models/

│ ├── isolation\_forest\_model.pkl

│ ├── kmeans\_model.pkl

│ └── label\_encoder.pkl

├── modules/

│ ├── data\_ingestion.py

│ ├── data\_preparation.py

│ ├── model\_layer.py

│ ├── llm\_integration.py

│ ├── data\_persistence.py

│ ├── data\_validation.py

│ ├── agentic\_ai.py

│ └── utils.py

├── scripts/

│ └── model\_training.py

└── ui/

└── ui\_integration.py

└── reatime\_streaming.py

└── batch\_update.py

🚀 How to Run

1. Setup Environment

Go to src folder which is checkout from github

https://github.com/ewfx/sradg-ai-innovators/tree/main/code/src

Go to VS code terminal / any ide terminal,

Execute the below commands to install the libs if already not installed libs/face module not found error.

1. pip install fastapi uvicorn[standard] pyyaml pandas scikit-learn openai python-multipart aiofiles email-validator
2. pip install aiohttp websockets
3. Create a file named requirements.txt and add below packages

fastapi

uvicorn[standard]

pandas

pyyaml

scikit-learn

openai

email-validator

python-multipart

aiofiles

aiohttp

websockets



1. pip install -r requirements.txt
2. pip install jupyterlab ipython black isort mypy

sample requirments.txt attached for the reference.

2. Configure the App

Edit config/config.yaml with proper email credentials, paths, and thresholds.

Example:

paths:

data\_file: data/generated\_reconciliation\_data.csv

anomaly\_output: output/detected\_anomalies\_{timestamp}.csv

log\_file\_name: logs/recon\_anomaly\_detection\_{timestamp}.log

data\_validation:

quantity\_threshold: 10

email:

sender: your\_email@gmail.com

password: your\_password

recipient: recipient@example.com

api\_keys:

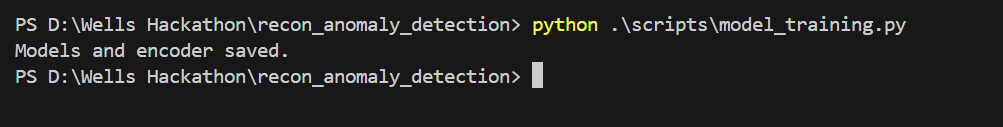
openai: YOUR\_OPENAI\_API\_KEY

3. Model Training

Run the following command in terminal:

python .\scripts\model\_training.py

output

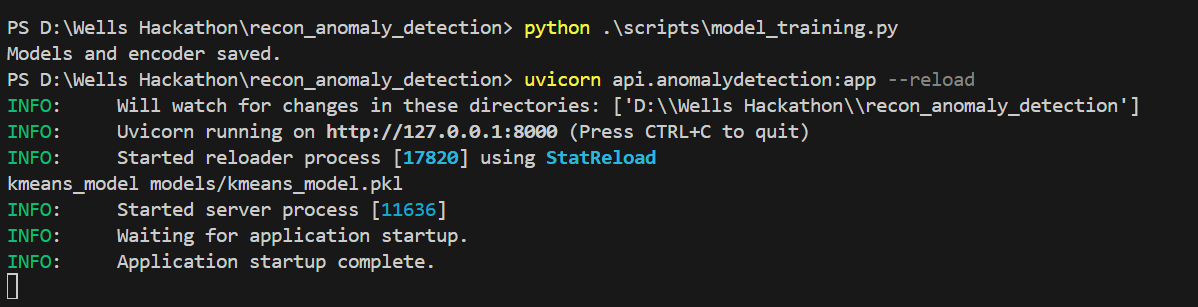


4. Run the API in terminal

uvicorn api.anomalydetection:app --reload

# Or if renamed

uvicorn api.anomalydetection:app --reload



4. Test API Endpoints open the new terminal and run the below commands

- POST /realtime\_anomaly/ – Detect anomaly in a single record

- POST /batch\_anomaly/ – Detect and categorize batch data

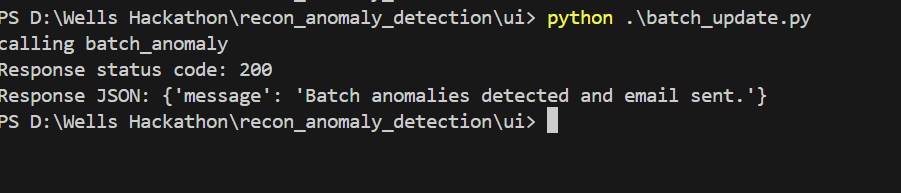
- GET /health – Check app status

- WS /ws – Real-time WebSocket anomaly updates

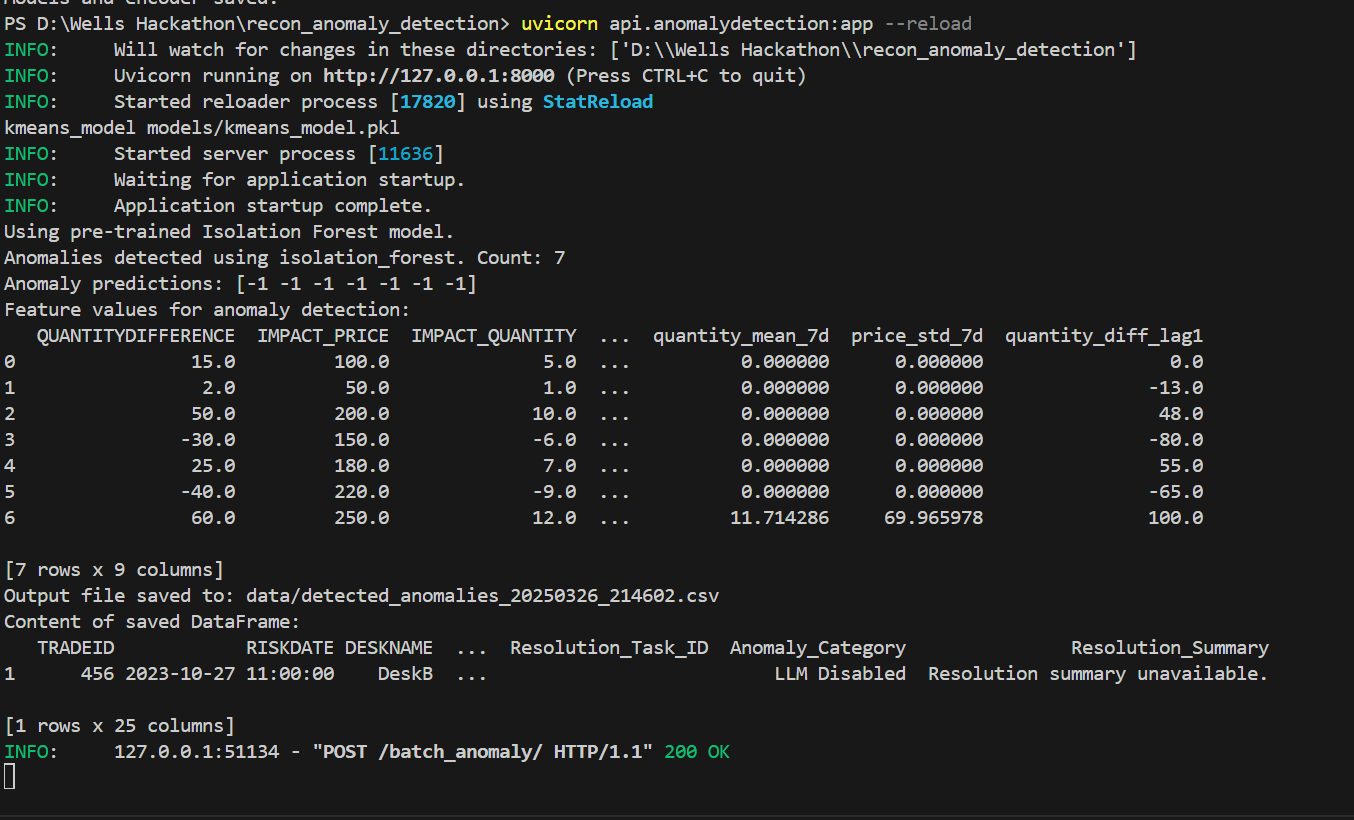
Go to src/ui folder then run the following comments to test the real time reconciliation anomaly detection.

python .\batch\_update.py

**Output:**

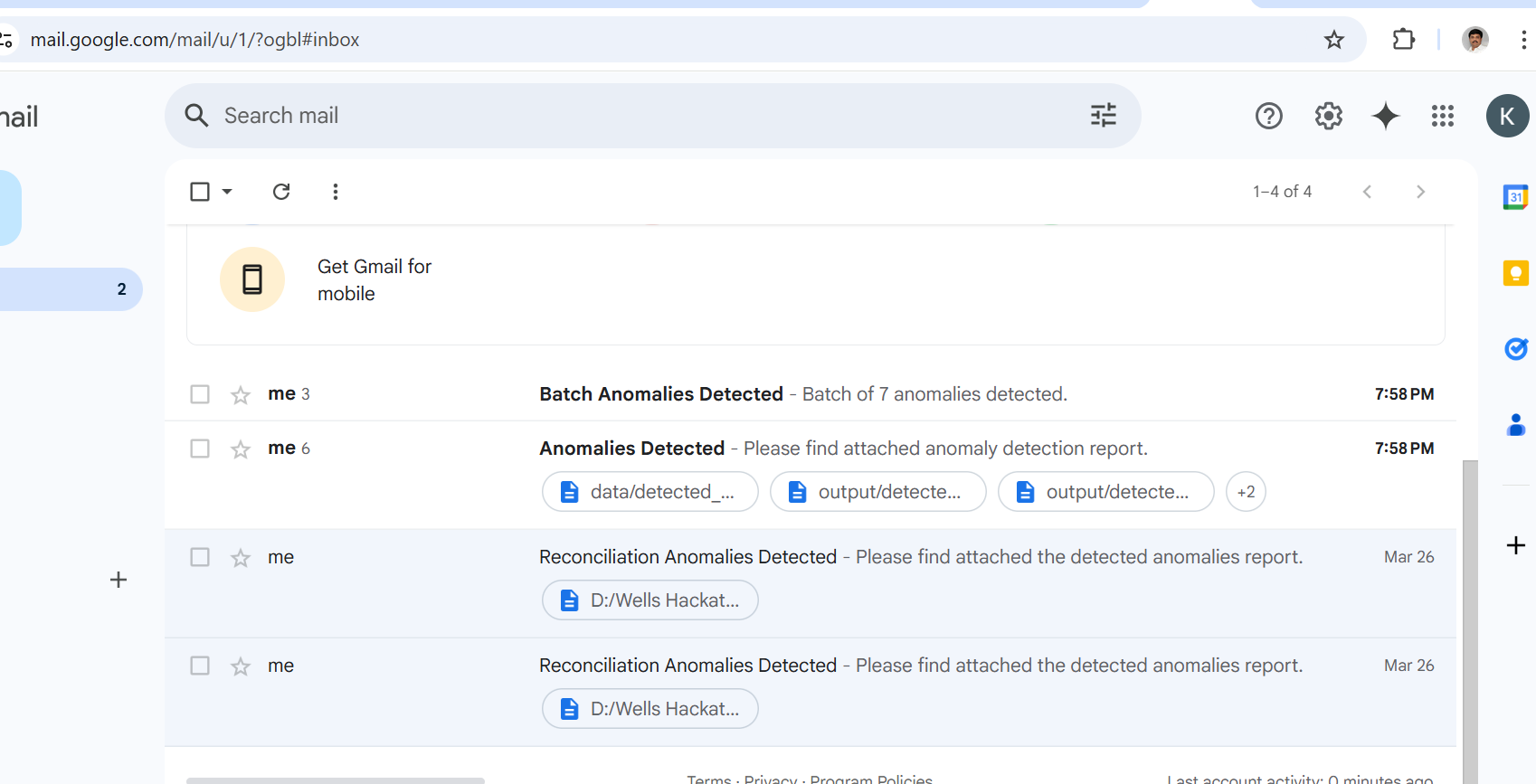
****

**Server side out put**



Output file available: src/data/detected\_anomalies\_20250326\_214602.csv

**Also above output file will send through email to configured users**

****

**Logfile available : src/logs/ recon\_anomaly\_detection.log**

**Explore other anomalies as well**

**Src/ui> python reatime\_streaming.py**

**Src/ui> ui\_integration.py**

If needed please change the test data in the above file and verify it.

💼 Usage Examples

Realtime Example:

POST /realtime\_anomaly/

{

        "TRADEID": 123,

        "RISKDATE": "2023-10-26T10:00:00",

        "DESKNAME": "DeskA",

        "QUANTITYDIFFERENCE": 15.0,

        "IMPACT\_PRICE": 100.0,

        "IMPACT\_QUANTITY": 5.0,

        "COMMENT": "Data entry error"

    },

Batch Example:

POST /batch\_anomaly/

[

{

        "TRADEID": 123,

        "RISKDATE": "2023-10-26T10:00:00",

        "DESKNAME": "DeskA",

        "QUANTITYDIFFERENCE": 15.0,

        "IMPACT\_PRICE": 100.0,

        "IMPACT\_QUANTITY": 5.0,

        "COMMENT": "Data entry error"

    },

    {

        "TRADEID": 456,

        "RISKDATE": "2023-10-27T11:00:00",

        "DESKNAME": "DeskB",

        "QUANTITYDIFFERENCE": 2.0,

        "IMPACT\_PRICE": 50.0,

        "IMPACT\_QUANTITY": 1.0,

        "COMMENT": "Rounding error"

    },

      {

    "TRADEID": 789,

    "RISKDATE": "2024-03-20T14:30:00",

    "DESKNAME": "DeskC",

    "QUANTITYDIFFERENCE": 50.0,

    "IMPACT\_PRICE": 200.0,

    "IMPACT\_QUANTITY": 10.0,

    "COMMENT": "Large quantity difference, potential manual error"

  },

  {

    "TRADEID": 101,

    "RISKDATE": "2024-03-21T09:15:00",

    "DESKNAME": "DeskA",

    "QUANTITYDIFFERENCE": -30.0,

    "IMPACT\_PRICE": 150.0,

    "IMPACT\_QUANTITY": -6.0,

    "COMMENT": "Negative impact, review required"

  },

    {

    "TRADEID": 202,

    "RISKDATE": "2024-03-22T11:45:00",

    "DESKNAME": "DeskB",

    "QUANTITYDIFFERENCE": 25.0,

    "IMPACT\_PRICE": 180.0,

    "IMPACT\_QUANTITY": 7.0,

    "COMMENT": "Unusual quantity difference, needs investigation"

  },

    {

    "TRADEID": 303,

    "RISKDATE": "2024-03-23T13:20:00",

    "DESKNAME": "DeskC",

    "QUANTITYDIFFERENCE": -40.0,

    "IMPACT\_PRICE": 220.0,

    "IMPACT\_QUANTITY": -9.0,

    "COMMENT": "Significant negative impact, immediate review"

  },

  {

    "TRADEID": 404,

    "RISKDATE": "2024-03-24T10:00:00",

    "DESKNAME": "DeskA",

    "QUANTITYDIFFERENCE": 60.0,

    "IMPACT\_PRICE": 250.0,

    "IMPACT\_QUANTITY": 12.0,

    "COMMENT": "Extremely high quantity difference, urgent review"

  }

]

🧠 LLM & Agentic AI

- llm\_integration.py uses OpenAI/Gemini to categorize anomalies & generate summaries.

- agentic\_ai.py can apply feedback or auto-resolution logic.

📬 Notifications

Sends an email with the CSV report when anomalies are detected.

🧹 Cleanup / Cache Issues

If FastAPI app behaves unexpectedly:

find . -type d -name "\_\_pycache\_\_" -exec rm -r {} +

find . -name "\*.pyc" -delete

PowerShell:

Get-ChildItem -Recurse -Include "\_\_pycache\_\_" | Remove-Item -Recurse -Force

Get-ChildItem -Recurse -Include "\*.pyc" | Remove-Item -Force

🛠️ Model Training

python scripts/model\_training.py

📈 Future Enhancements

We developed all features which are requested in the hackathon problem statement, don’t have more time to develop more features, will develop it, if provide the time.

-Kafka & Redis Integration

- Database integration (PostgreSQL/MongoDB)

- UI dashboard for monitoring

- Power BI Reports Generation

- Docker containerization

- CI/CD pipeline

- Feedback learning loop API