Data Ingestion API System

A robust RESTful API system for handling asynchronous data ingestion requests with priority-based processing, rate limiting, and real-time status tracking.

# Repository Link

GitHub: https://github.com/yourusername/data\_ingestion\_api

# Contents

* - Features
* - Project Structure
* - Prerequisites
* - Installation
* - Running the Application
* - API Endpoints
* - Running Tests
* - Configuration
* - Priority Processing
* - Rate Limiting
* - Error Handling
* - Contributing
* - License
* - Author
* - Acknowledgments
* - Comments

# Features

* - Priority-based Processing: HIGH, MEDIUM, LOW
* - Rate Limiting: Configurable delay between batches
* - Batch Processing: Automatically splits large requests
* - Status Tracking: Real-time batch & request status
* - Asynchronous Processing: Non-blocking logic
* - Input Validation: Strong schema checks with Pydantic

# Project Structure

data\_ingestion\_api/  
│  
├── main.py # FastAPI application and core logic  
├── test\_main.py # Test suite  
├── requirements.txt # Dependencies  
└── README.md # Project documentation

# Prerequisites

- Python 3.8 or higher  
- pip (Python package manager)

# Running the Application

Start the FastAPI server:  
 uvicorn main:app --reload  
  
The API will be available at:  
 http://localhost:8000  
  
Interactive documentation:  
- Swagger UI: http://localhost:8000/docs  
- ReDoc: http://localhost:8000/redoc

# API Endpoints

## 1. POST /ingest

Submit data for ingestion.

Request Body:

{  
 "ids": [1, 2, 3, 4, 5],  
 "priority": "HIGH"  
}

Response:

{  
 "ingestion\_id": "uuid-string"  
}

## 2. GET /status/{ingestion\_id}

Check ingestion and batch processing status.

Response:

{  
 "ingestion\_id": "uuid-string",  
 "status": "yet\_to\_start|triggered|completed",  
 "batches": [  
 {  
 "batch\_id": "uuid-string",  
 "ids": [1, 2, 3],  
 "status": "yet\_to\_start|triggered|completed"  
 }  
 ]  
}

# Running Tests

Run the test suite:

pytest test\_main.py -v

# Configuration

- BATCH\_SIZE: Maximum number of IDs per batch (default: 3)  
- RATE\_LIMIT\_SECONDS: Minimum time between batch processing (default: 5 seconds)

# Priority Processing

The system processes ingestion requests by priority:  
1. HIGH priority first  
2. MEDIUM priority next  
3. LOW priority last  
  
FIFO order is maintained within each priority.

# Rate Limiting

- One batch is processed every RATE\_LIMIT\_SECONDS (default: 5 seconds)  
- Applies across all priority levels  
- Maintains a pending queue

# Error Handling

Handled cases:  
- IDs not in range (1 to 10^9 + 7)  
- Invalid priority values  
- Unknown ingestion ID  
- Invalid JSON payloads

# Contributing

1. Fork the repository  
2. Create a feature branch  
3. Push changes  
4. Open a Pull Request

# License

MIT License - see LICENSE file for details.

# Acknowledgments

- FastAPI  
- Pydantic  
- Uvicorn

# Comments

- Make sure Python and pip are properly installed.  
- Use the interactive Swagger docs for easy testing.  
- Adjust BATCH\_SIZE and RATE\_LIMIT\_SECONDS as per your use case.  
- The UUIDs returned can be used to track status via GET /status/{id}.

