# Blood Test Report Analyzer

A FastAPI-based backend service that allows users to upload PDF blood test reports and get AI-generated summaries using a queue-based task processing system. Built for the VWO AI Internship Debug Assignment.

## 🗂️ Project Structure

├── main.py # FastAPI app  
├── worker.py # Dramatiq worker logic  
├── tools.py # PDF reading helper  
├── requirements.txt  
└── README.md # This file

## 📦 Setup Instructions

### 1. Clone the Repo

cd blood-test-analyser-debug

### 2. Install Dependencies

pip install -r requirements.txt

Make sure Redis server is running locally on port 6379.

### 3. Run FastAPI App

uvicorn main:app --reload

### 4. Start Dramatiq Worker

python -m dramatiq worker

## 🧪 How It Works

### 📤 Submit a PDF for Analysis

POST /analyze

* Form fields:
  + file = blood test report PDF
  + query = your question (optional)

Returns:

{  
 "status": "submitted",  
 "task\_id": "<uuid>",  
 "file\_processed": "your\_file.pdf"  
}

### 📥 Get Result

GET /result/{task\_id}

* Returns summary and parsed content if completed.

## 📘 API Documentation

### 📤 POST /analyze

Uploads a blood test report PDF and starts the background summarization task.

#### ✅ Form Data

| Field | Type | Required | Description |
| --- | --- | --- | --- |
| file | file | ✅ Yes | PDF file of the blood test report |
| query | string | ❌ No | Optional query (default: “Summarise my Blood Test Report”) |

#### 🔄 Response

{  
 "status": "submitted",  
 "task\_id": "b8c5e1a4-31d0-4ec5-8ab9-25b6d3201bb6",  
 "file\_processed": "report.pdf"  
}

### 📥 GET /result/{task\_id}

Fetches the result of the background task using the task\_id.

#### 🔄 Path Parameter

| Parameter | Type | Description |
| --- | --- | --- |
| task\_id | string | The task ID returned by /analyze |

#### 🔄 Possible Responses

✅ **Success** (if analysis is complete):

{  
 "status": "completed",  
 "summary": "Processed file: ...",  
 "query": "Summarise my Blood Test Report",  
 "content\_excerpt": "This report contains..."  
}

⏳ **Pending** (if still processing):

{  
 "status": "PENDING"  
}

❌ **Expired or Not Found**:

{  
 "status": "not\_found\_or\_expired"  
}

### ✅ Example Usage (with curl)

curl -X POST "http://127.0.0.1:8000/analyze" \  
 -F "file=@blood\_report.pdf" \  
 -F "query=Summarise my Blood Test Report"

curl http://127.0.0.1:8000/result/<task\_id>

### ⚠️ Database Integration (Attempted but Not Included)

This project originally included an attempt to integrate persistent data storage using SQLite and SQLAlchemy. The goal was to store analysis results permanently instead of caching them in Redis.

## 🤝 Author

Om Dodia — for VWO AI Internship Assignment

## ✅ Submission Status

Queue system fully functional. PDF uploads, processing, and result retrieval work end-to-end. No database used — Redis used for temporary storage.”

Thank you for reviewing this project! 🙏