BFSI- OCR OF BANK STATEMENTS

Milestone 3 Report

PREPARED BY:
Jasneet Arora

Table of Contents

1. Introduction

The BFSI (Banking, Financial Services, and Insurance) OCR (Optical Character Recognition) project aims to develop a comprehensive system for the automated analysis of financial documents using advanced text extraction technologies. Milestone 3 focuses on visualizing data extracted by the meta-llama/Llama-3.2-11B-Vision-Instruct-Turbo model, and incorporates a querying feature, allowing users to ask specific questions about the document. This approach enhances the user experience by providing interactive visualizations and enabling deeper insights into the financial data.

2. Objectives

Milestone 3 focuses on visualizing financial data extracted by the meta-llama/Llama-3.2-11B-Vision-Instruct-Turbo model and implementing a document querying feature. The objectives include creating interactive visualizations with Plotly for easy analysis of financial data and enabling users to query the document for deeper insights. This milestone aims to enhance the efficiency and user experience of financial document analysis.

3. Methodology

The methodology involves users selecting the financial document type (e.g., bank statements, cheques, P&L statements, salary slips, transaction history) and uploading images. The meta-llama/Llama-3.2-11B-Vision-Instruct-Turbo model extracts the relevant data, which is then converted into a parameter table for visualization. Users can choose the graph type (bar/pie) to display the data. Additionally, a chatbot is integrated to allow users to query the document, enabling efficient interaction and deeper insights from the uploaded financial documents.

4. Technical Implementation

4.1 Document Upload and Processing

Users upload financial documents (images or PDFs) through Streamlit's file uploader. PDFs are converted to images using PyMuPDF (fitz), while image files are directly encoded into base64 format. This encoding allows seamless integration with the Together API, which interfaces with the meta-llama/Llama-3.2-11B-Vision-Instruct-Turbo model for text extraction.

4.2 Data Extraction

The meta-llama model extracts key financial parameters from documents, using prompts tailored to document types (e.g., bank statements, salary slips). Extracted data is processed and organized into a Pandas DataFrame, which is cleaned using

regular expressions (re) to remove currency symbols, commas, and other unnecessary characters.

4.3 Data Visualization

For data visualization, Plotly is used to create dynamic, interactive charts. Plotly Express handles basic visualizations, like bar and pie charts, while plotly.graph_objs enables more customized charts, such as grouped bar charts for multiple documents. These charts allow users to visually analyze financial data and interact with the visualizations in real-time through Streamlit. Users can select graph types, filter parameters, and download visualized data as CSV files.

4.4 Querying and Insights

The Together API enables users to query documents interactively. After submitting a query, the meta-llama model analyzes the document, providing context-specific answers, which are displayed through Streamlit for user interaction.

4.5 Temporary File Management

Temporary image files are managed using Python's tempfile and os libraries, ensuring efficient file handling and cleanup after processing, with Streamlit notifications alerting users in case of errors.

5. Results and Deliverables

5.1 Results

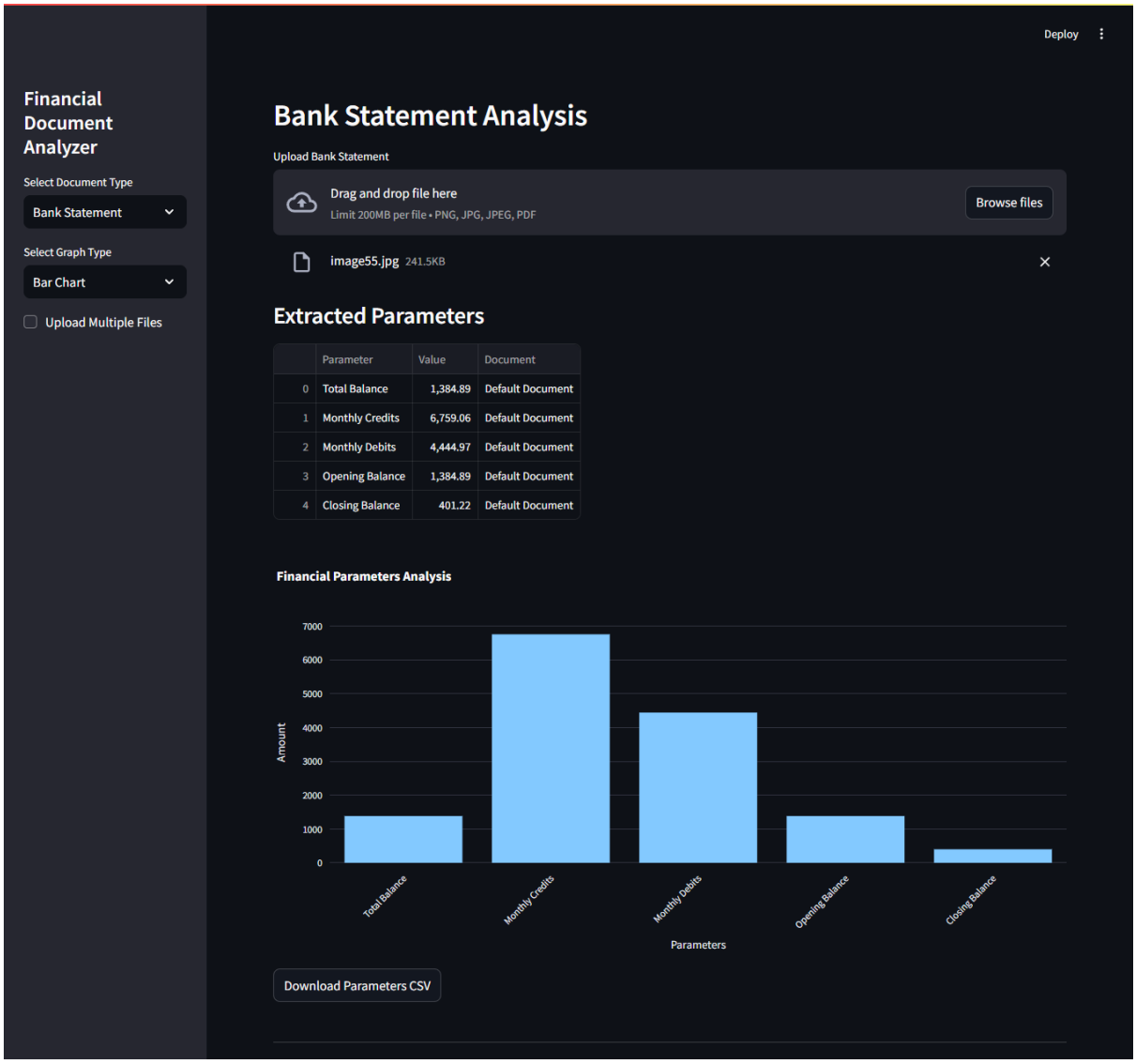
The Financial Document Analyzer successfully processes multiple document types, such as bank statements, cheques, and profit and loss statements, by extracting key financial parameters using the meta-llama model. Extracted data is organized into a Pandas DataFrame for easy analysis. Plotly is used to generate interactive Bar and Pie charts, offering insightful visualizations of financial data. Additionally, the system enables querying of the document content, providing users with a conversational interface to gain further insights.

5.2 Deliverables

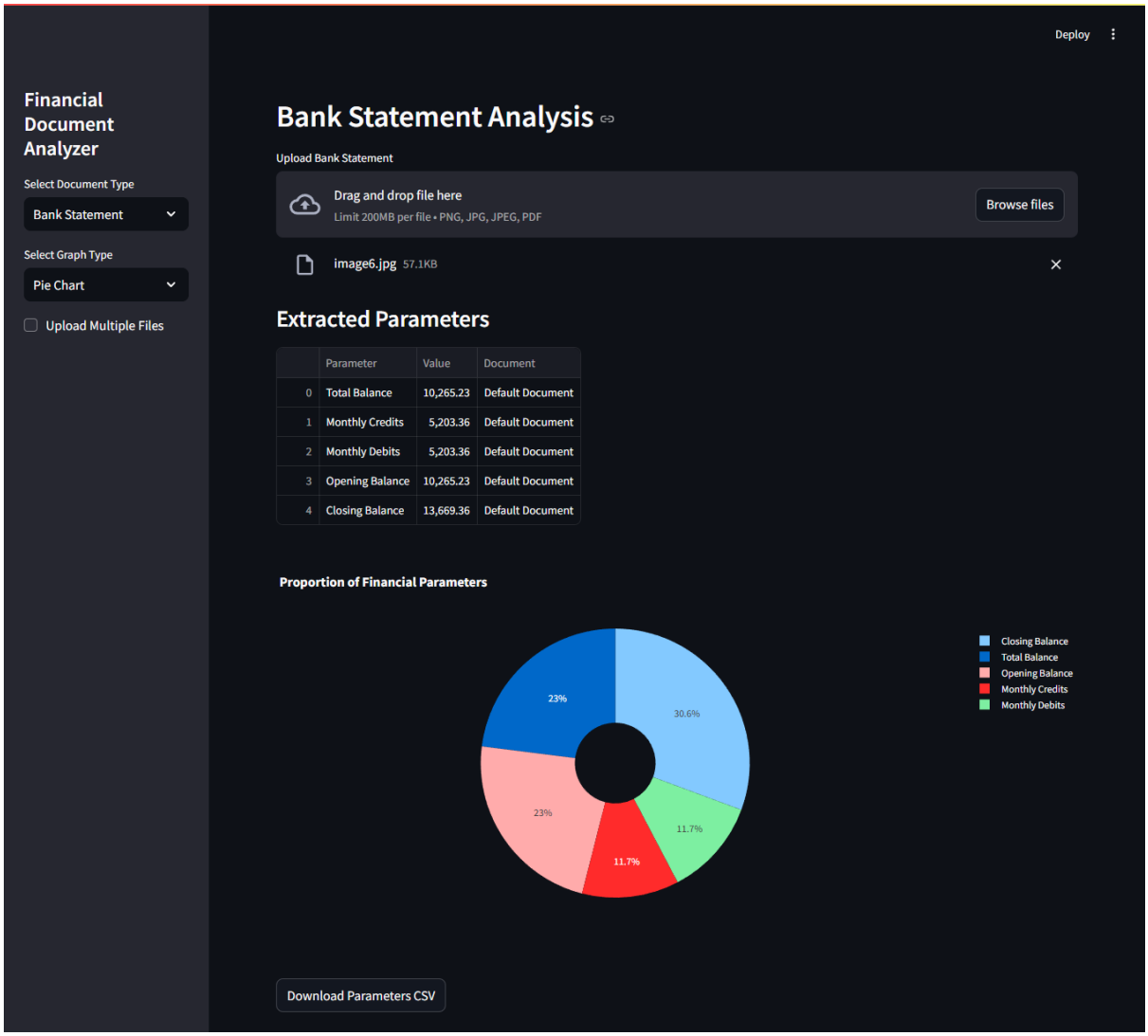
- **Document Parameter Extraction**: Automated extraction of key financial parameters (e.g., balance, credits, debits) from various document types using meta-llama and Pandas.
- **Data Visualization**: Interactive Bar and Pie charts generated using Plotly, allowing users to visualize financial data trends.
- **Querying Interface**: AI-driven querying feature that allows users to ask specific questions about the document's content using Together API and meta-llama.
- **Data Export**: Option to download the extracted financial data as a CSV file for further analysis and record-keeping.

6. Dashboard Snapshots

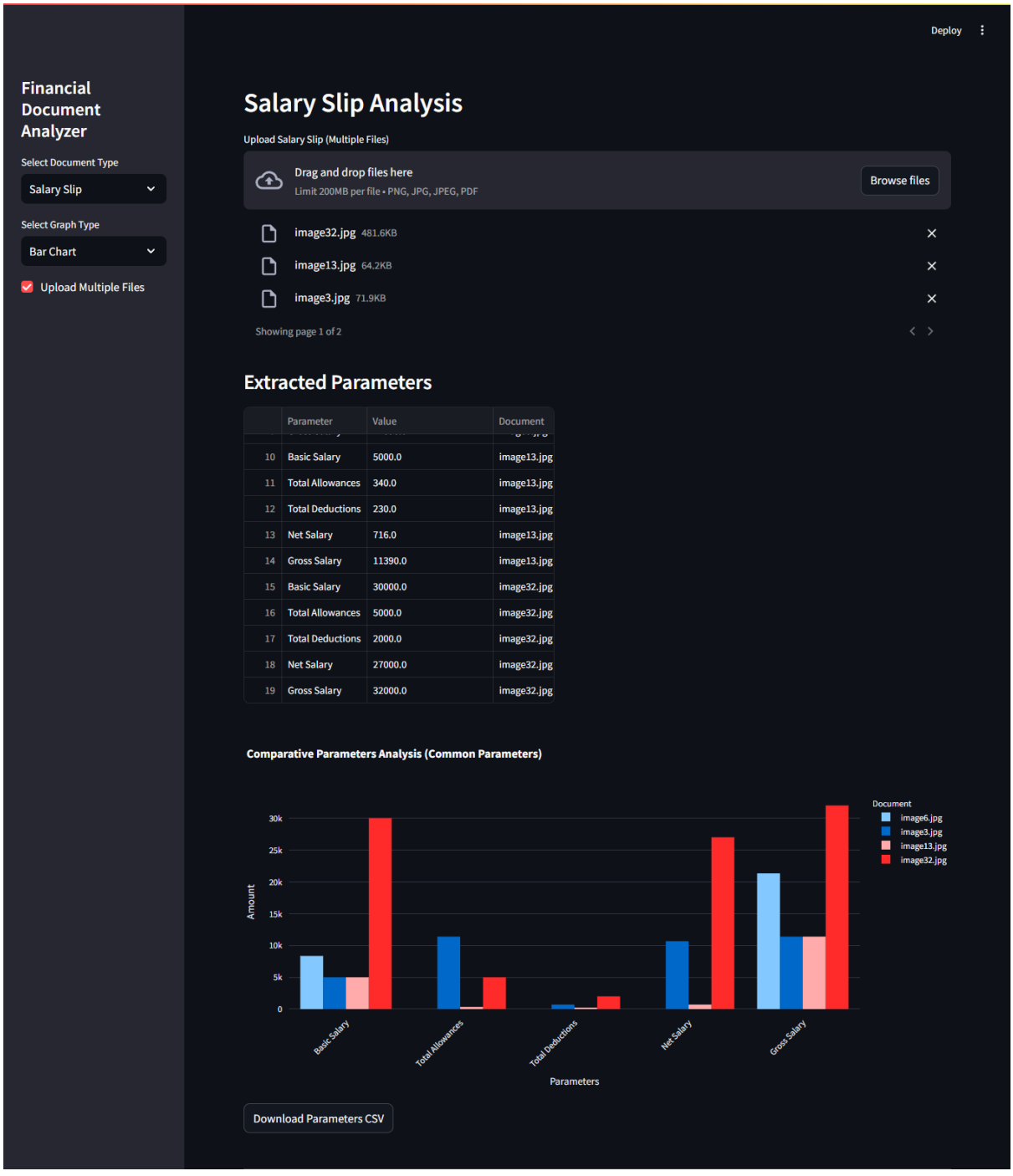
6.1 Bank Statement Analysis and Bar Chart for a Single Image



6.2 Bank Statement Analysis and Pie Chart for a Single Image

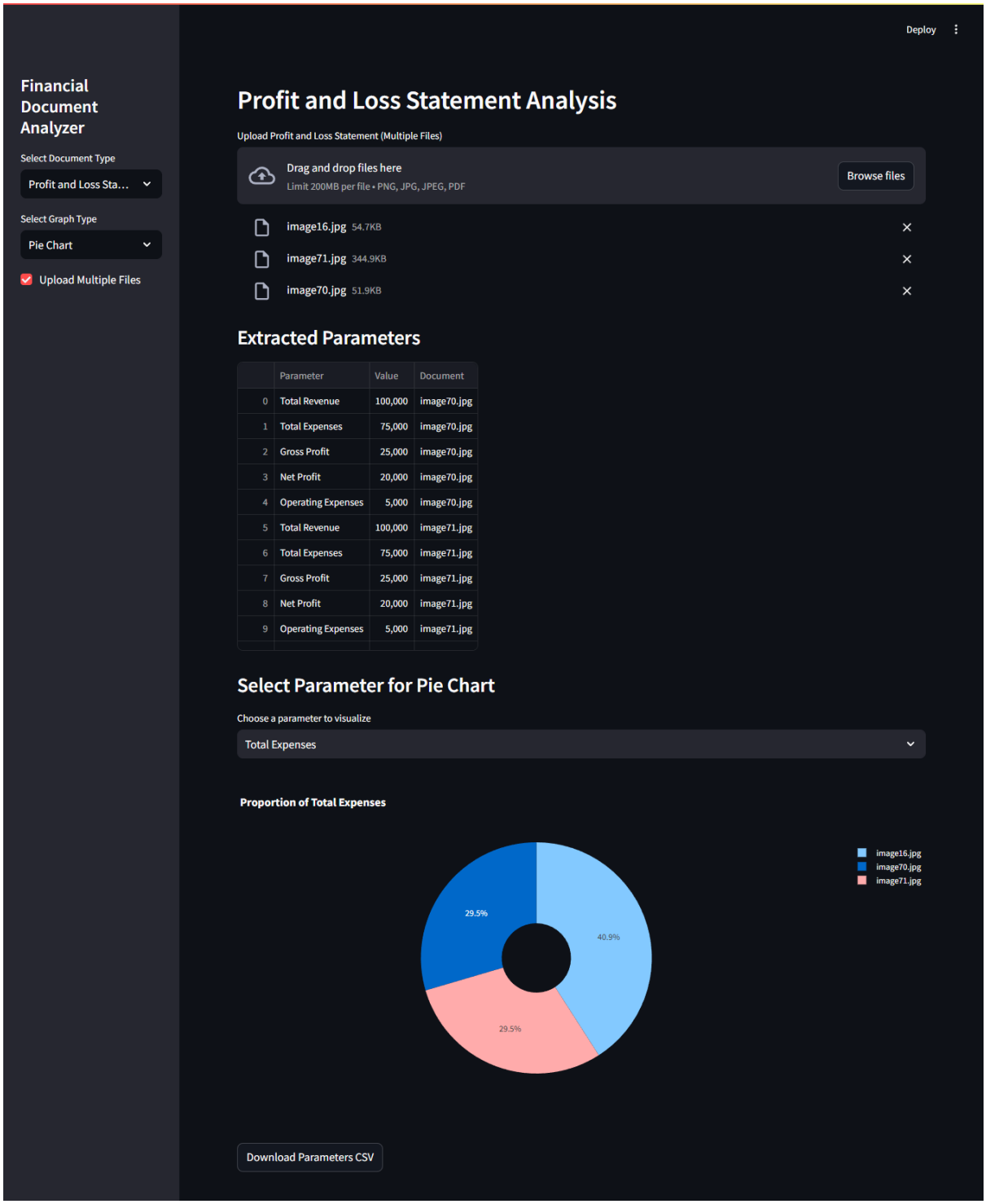


6.3 Salary Slip Analysis and Bar Chart for Multiple Images

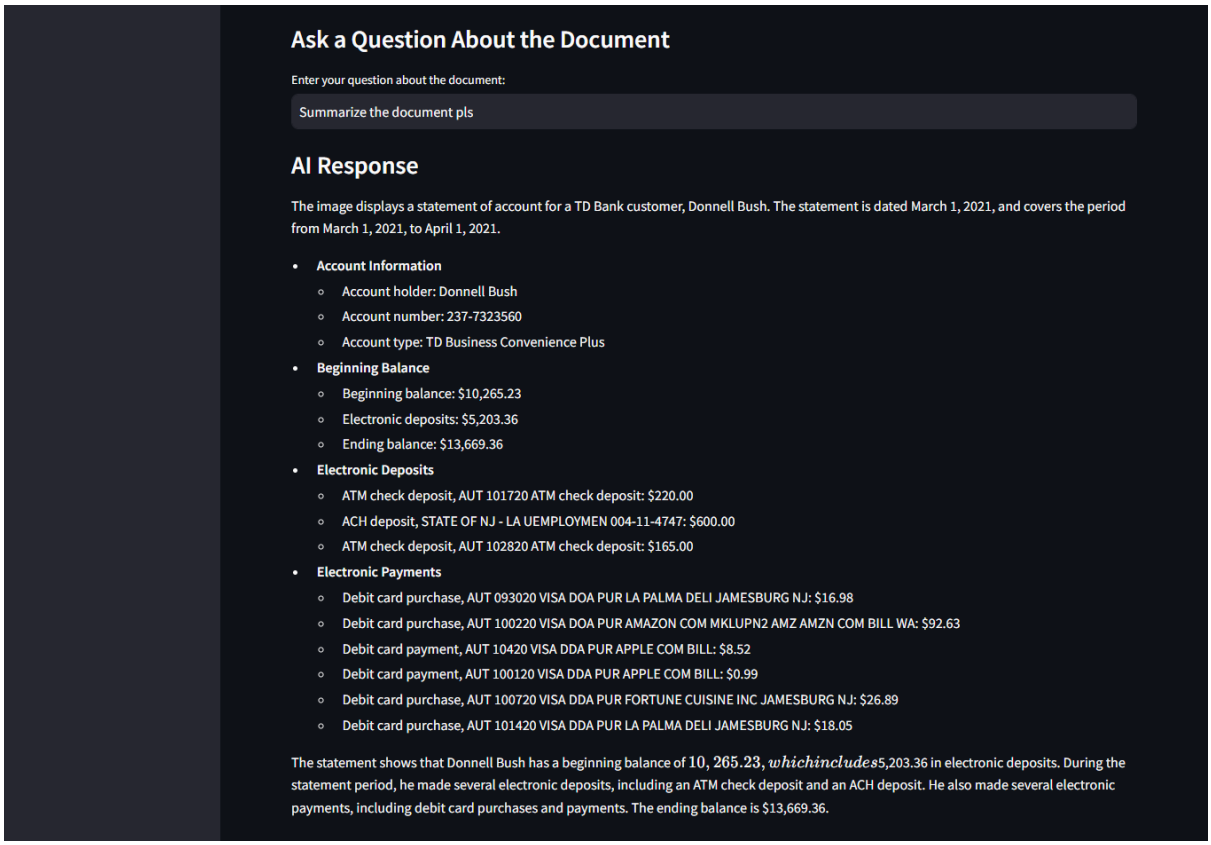


`

6.4 Profit and Loss Statement Analysis and Pie Chart for Multiple Images



6.5 Ask a Question About the Document



7. Challenges and Solutions

7.1 Document Variability

- **Challenge:** Different document types (e.g., bank statements, cheques) vary in format, making it difficult to extract consistent data.
- **Solution:** Custom prompts were provided to the model for each document type, guiding it to extract the necessary parameters in a structured and consistent format.

7.2 Accuracy in Parameter Extraction

- **Challenge:** Extracted data could be inconsistent due to variations in formatting within documents.
- **Solution:** The model was prompted to return data in a specific format, which was then organized into a parameter table, ensuring accuracy and consistency in the extracted financial values.

8. Next Steps

The next phase of the project involves deploying the OCR and data analysis system for real-time analysis and integrating it with financial platforms to streamline document processing. The aim is to ensure that the system can handle various bank statement formats, enabling efficient extraction and visualization of financial data. We will focus on creating a seamless user experience that enhances the overall process of financial document analysis.

9. Appendices

Appendix A: Technical Architecture

- **Frontend**: Streamlit for UI, allowing user interactions.
- **PDF/Image Processing**: PyMuPDF (fitz) for PDF extraction and Pillow for image handling.
- **Data Processing**: Pandas for organizing and manipulating extracted data.
- **Visualization**: Plotly for dynamic bar and pie charts.
- **AI Integration**: Together API for document analysis and parameter extraction.

Appendix B: Technical Specifications

- **Streamlit**: Framework for building interactive web apps.
- **PyMuPDF**: Used for extracting content from PDFs.
- **Pandas**: Data manipulation and parameter extraction.
- **Plotly**: Creates interactive visualizations (bar, pie charts).
- **Pillow**: Processes and converts image formats.
- **Together API**: AI-powered extraction and analysis.

Appendix C: Hardware and Software Requirements

- **Hardware**:
    - Minimum RAM: 8GB
    - CPU: 2.5 GHz or higher
    - Storage: 50GB available space for handling files and processing
- **Software**:
    - Operating System: Windows, macOS, or Linux
    - Python 3.8+
    - Libraries: Streamlit, PyMuPDF, Pillow, Pandas, Plotly, Together API
    - Web Browser for Streamlit interface (Chrome/Firefox recommended)