Gemini Pro Financial Decoder - Streamlit Interface

This module (app.py) provides the main user interface for the Gemini Pro Financial Decoder application. Built using Streamlit, it allows users to upload financial documents, initialize the LLM backend, and generate AI-powered summaries and visualizations.

What This Module Does

- Sets up the Streamlit UI with a page title, layout, and custom header.
- Initializes the LLM used for financial document summarization.
- Provides a sidebar file uploader for:
- Balance Sheet
- Profit & Loss Statement
- Cash Flow Statement
- On button click, processes the uploaded files:
- Loads each file
- Summarizes it using the LLM
- Displays the summary
- Generates and shows visualizations

Dependencies

This module depends on:

- streamlit - for building the UI
- file_uploader.load_file() - for loading uploaded documents into a DataFrame
- llm_handler.initialize_llm() - for setting up the large language model
- llm_handler.generate_summary() - for generating text summaries

- visualizer.create_enhanced_visuals() - for visual analysis of financial data

UI Layout Overview

- Header: Centered title and description

- Sidebar: File uploaders for three financial statements

- Main Panel: When "Generate Comprehensive Financial Analysis" is clicked:

- Each uploaded document is:

- Parsed

- Summarized by LLM

- Visualized using charts/graphs

LLM Fallback Handling

If the LLM fails to initialize, the app:

- Displays an error

- Stops further execution to avoid crashes

Sample Usage (UI Flow)

1. Launch the app: streamlit run app.py

2. Upload your financial documents via the sidebar.

3. Click "Generate Comprehensive Financial Analysis".

4. View AI-generated summaries and visual reports directly on the screen.

Note

This file does not handle:

- Backend LLM configuration

- File parsing logic

- Visualization details

Those are delegated to separate modules to maintain modularity and separation of concerns.

Functionality

- Sets up the **Streamlit** page layout and interface settings.
- Loads **custom CSS styling** using a helper from styles.py.
- Displays a centered **header** with the application title and tagline.
- **Initializes the LLM** and handles any failure with a user-friendly error.
- Provides a **sidebar** for uploading:
    - Balance Sheet
    - Profit & Loss Statement
    - Cash Flow Statement
- On pressing the **Generate Analysis** button:
    - Reads uploaded files.
    - Passes them to the LLM for generating summaries.
    - Displays both text-based insights and visual charts.
    - Shows a final success confirmation.

Contribution

This app.py module has been developed independently to handle the full interactive workflow of the application—from file upload to AI-enhanced output. It acts as the main control layer, coordinating between user inputs and backend logic, and ensuring that financial insights are presented in a streamlined, professional format.