Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	**31 January 2026**
Team ID	**LTVIP2026TMIDS61777**
Project Name	**Gemini Pro Financial Decoder**
Maximum Marks	**4 Marks**

Technical Architecture

The technical architecture of the Gemini Pro Financial Decoder system describes how different components interact to process financial data and generate AI-based financial insights using Generative AI.

In this application:

1. The user interacts with the Streamlit web interface.
2. The user uploads a financial file in CSV or XLSX format.
3. The application logic reads and processes the uploaded file using Pandas.
4. The financial data is formatted into a structured prompt.
5. The request is sent to the Google Gemini 2.5 Flash API.
6. The AI model generates structured financial insights.
7. The application calculates dashboard metrics and creates visualizations.
8. The generated summary, metrics, and charts are displayed to the user.

Table 1: Components and Technologies

S.No	Component	Description	Technology
1	User Interface	Web interface where user uploads financial files and views output	Streamlit (Python Web UI)
2	Application Logic 1	File validation and data preprocessing	Python, Pandas
3	Application Logic 2	Integration with Generative AI model	Google Generative AI SDK
4	Application Logic 3	Financial metric calculation and prompt formatting	Python
5	Database	Not required as data is processed dynamically	Not Applicable
6	Cloud Database	Not used in this project	Not Applicable
7	File Storage	Temporary handling of uploaded files	Local Filesystem
8	External API 1	Generates AI-based financial summaries	Google Gemini 2.5 Flash API
9	External API 2	Not used in this project	Not Applicable
10	Machine Learning Model	Generates structured financial insights	Gemini 2.5 Flash
11	Infrastructure Server or Cloud	Application deployment environment	Local System or Streamlit Clou

Table 2 Application Characteristics

S.No	Characteristics	Description	Technology Used
1	Open Source Frameworks	Frameworks used to build the application	Python, Streamlit, Pandas
2	Security Implementations	API key stored securely using environment variables	.env configuration
3	Scalable Architecture	Can be extended to multi-year analysis, ratios, PDF export, advanced tools	Python, Gemini API
4	Availability	Accessible whenever Streamlit server is running	Streamlit Deployment
5	Performance	Fast response due to optimized API calls and lightweight UI	Gemini API, Streamlit