Generative AI Integration

1) Integration of Generative AI (2 pts)

We integrated **Gemini 2.0** into the backend (/analyze route).

- When a user submits a message for analysis, the backend:
    1. **Preprocesses the text** (extract sender info, detect URLs, phishing keywords).
    2. **Calls Gemini API** with a carefully crafted system prompt to return results in a **structured JSON format** with keys:
        - judgment: Safe | Phishing | Suspicious | Other
        - explanation: concise reasoning
        - tips: practical suggestions for the user
- Example response stored in DB:

  {

  "judgment": "Phishing",

  "explanation": "The message asks for personal bank details and contains a suspicious link.",

  "tips": [

  "Never share personal information via email",

  "Verify the sender domain",

  "Do not click unknown links"

  ]

  }
- This integration ensures every analysis is **consistent** and **structured**, allowing us to power the **pie chart breakdown** in the dashboard. So we fully meet the **integration** requirement.

2) Performance Improvements & Experiments

We worked on **prompt engineering and response handling** to make the AI more effective:

1. **Prompt Refinement**
    - Initial prompts returned unstructured or overly verbose text.
    - We improved the prompt to explicitly enforce JSON output, which solved parsing errors.
2. **Error Handling**
    - Added a cleaning step to remove code blocks (json ...) before parsing.
    - Implemented a fallback: if parsing fails, show plain text response instead of crashing.
3. **Hybrid Rule-based + AI**
    - Before calling Gemini, the backend runs simple checks:
        - If text contains **"password"** or **"bank login"**, mark as suspicious.
        - Extracts URLs and marks if they don't match known domains.
    - This hybrid approach improves both **speed** (AI not needed for obvious cases) and **accuracy**.
4. **Caching**
    - Results of identical inputs are cached in the DB (avoids repeated API calls).
    - This reduces latency for repeated tests.
5. **User Feedback Loop**
    - Analyses are stored per user. This allows retraining or improving future prompts by observing real phishing patterns in user data.

Even if API has limits, these improvements show **effort and methodology** for maximizing its performance.

3) Look and Feel Improvements

We worked on **user experience while using the AI chatbot**:

- **Chat-style interface** on /analyze:
    - User messages appear right-aligned in a colored bubble.

    - Bot responses appear left-aligned with structured sections (Judgment, Explanation, Tips).
- **Loading Indicator**:
    - While waiting for AI response, we show "Analyzing your message...".
    - This improves usability and prevents confusion.
- **Dashboard Integration**:
    - Analysis results are aggregated into **line, bar, and pie charts**.
    - This transforms raw AI outputs into actionable insights.

Look & feel was significantly improved beyond the raw AI integration.

Conclusion for D

- **Integration :** Gemini's API with structured JSON results.
- **Performance :** prompt engineering, error handling, hybrid rules, caching, feedback loop.
- **Look & Feel :** Chat-style interface + dashboard charts.