AI Agent Architecture Document

1. Overview

The **AI Personal Assistant** automates **email triage and calendar management** for IIT Delhi students and faculty. It integrates directly with IIT Delhi's webmail (IMAP protocol) and **Google Calendar**, intelligently classifies emails, and creates calendar events for timed academic/professional activities.

The system was designed around the principle of **high recall for critical events** (to avoid missed quizzes, exams, or deadlines) while maintaining reasonable precision (avoiding excessive false alarms).

---

2. Core Components

A. Data Layer

- **IMAP Fetcher (`imap_tool.py`)**:
  - Retrieves the last 30 days of emails (up to 50 at a time).
  - Extracts metadata: subject, sender, body, and timestamp.
  - Implements error handling for login failures, malformed MIME, and connection drops.
- **Google Calendar Tool (`calendar_tool.py`)**:
  - Uses OAuth2 authentication with the Google API.
  - Reads existing events (to avoid duplicates).
  - Creates new events based on AI extraction.
- **Credential Management**:
  - `.env` file stores the IITD webmail credentials.

- `token.json` stores Google API tokens (refreshable).
- No hardcoded passwords → secure setup.

B. AI Models

1. **Triage Model (Priority Classification)**
   - Base: **FLAN-T5-Small** (80M params).
   - Fine-tuned with **LoRA adapters** on labeled IITD emails.
   - Task: Output label **HIGH/MEDIUM/LOW**.
   - Optimized for **high recall on HIGH class** to minimize missed deadlines.
2. **Event Extraction Model (Structured Parsing)**
   - Base: **FLAN-T5-Small** (separate LoRA adapter).
   - Task: Convert free-form email into structured JSON (event_name, date, time, location).
   - Handles noisy, unstructured formats (e.g., “Quiz tomorrow at 10 AM in LH-101”).
   - Default rules applied when partial info is missing (e.g., no time → 23:59).

C. Processing Pipeline

1. **Email Retrieval**: Fetches raw emails via IMAP.
2. **Classification**: The Triage model assigns HIGH/MEDIUM/LOW.
3. **Extraction**: If HIGH → event extraction model parses details.
4. **Calendar Update**: The Calendar tool creates or updates events.

5. **Analytics Logging**: Stores statistics for reporting (distribution of priorities, % of events captured).
6. **Dynamic Re-run**: Each execution reclassifies emails (no stale labels).

D. Web Interface (`app.py`)

- **Flask-based dashboard**.
- Features:
    - Inbox view with AI-labeled emails.
    - Real-time run of the agent.
    - Integrated Google Calendar view.
    - Analytics charts (e.g., distribution of HIGH/MEDIUM/LOW).
- Designed for **non-technical users** (faculty/students).

---

3. Interaction Flow

[IMAP Fetcher]

↓

[Triage Model (LoRA)] → Assigns HIGH/MEDIUM/LOW

↓

(if HIGH)

↓

[Event Extraction Model (LoRA)] → JSON Event Object

↓

[Google Calendar Tool] → Create/Update Event

↓

[Web UI] → Display Emails + Calendar + Analytics

---

4. Models Used & Rationale

- **FLAN-T5-Small**
    - Small footprint → fast inference, deployable on student laptops.
    - Instruction-tuned → works well with minimal prompt engineering.
- **LoRA Fine-tuning**
    - Parameter-efficient → trainable with limited compute (Local computer, 16GB RAM, no GPU).
    - Adapters allow **switching tasks** (classification vs extraction) without retraining the base model.
- **Separation of Concerns**
    - Two specialized models instead of one multitask model.
    - Improves interpretability: easier to debug classification vs extraction errors.