Interaction Logs and chat history :

These are the AI interaction logs from the development of my Campus Companion AI Agent project. I used different LLMs at various stages- for planning, coding assistance, UI inspiration, and system design. Below are the detailed logs, prompts, and links.Here is the description of what AI tool I used for what specific tasks

- **1. Core Idea & Planning –** Gemini Pro
- **2. Important Updates Section –** ChatGPT, Gemini Pro
- **3. Roadmap Feature –** ChatGPT, Gemini Pro
- **4.Debugging Smaller bugs-** ChatGPT , Perplexity Pro
- **5. Debugging Complex Bugs –** Vercel V0
- **6. UI Theme & Landing Page –** Gemini Nano
- **7. Visual tasks and Low-Level System Design –** Claude

1.) Used **Gemini pro** for framing the core idea basically thinking part like making the roadmap for execution, suggestions for tech stack and how to execute the features , probable directory structure and outline for the system design document etc

https://g.co/gemini/share/bcc935e3079a

2.) For the important updates section the most part was for basic setup code generation and taking small suggestions so found **ChatGPT** was a good go

https://chatgpt.com/share/68ca5430-20c4-800c-8427-b2f12ff8c147

3.) Again for the **Roadmap section**, I have deal with generating the roadmap downloading the roadmap and also achieving the feature to be able to edit the agent's output and then display them , faced a bug too "Attempting to parse an unsupported color function "lab" which was due to heavy css theme I was using so our parser was unable to download it

https://chatgpt.com/share/68ca573f-df90-800c-8fca-02fe022bae03

4.) There are some of the time where I faced some serious bugs which basic LLMs were not able to figure it out and also they are uncommon so for that I used **Vercel V0 model** to fix the code and ambiguity in it
Example log where I faced parsing out the agent's output of roadmap section into our wanted json format , it was wrapping it in string and adding unwanted escape characters:

https://v0.app/chat/roadmap-generation-and-editing-v6ufbpPMO94

5.) For **app theme and the landing page cartoon**: While I was searching what could be the best possible UI theme which resembles our app idea and also might looks cool so found this kind of doodle style cartoon, so I took inspiration from them Credits and Source

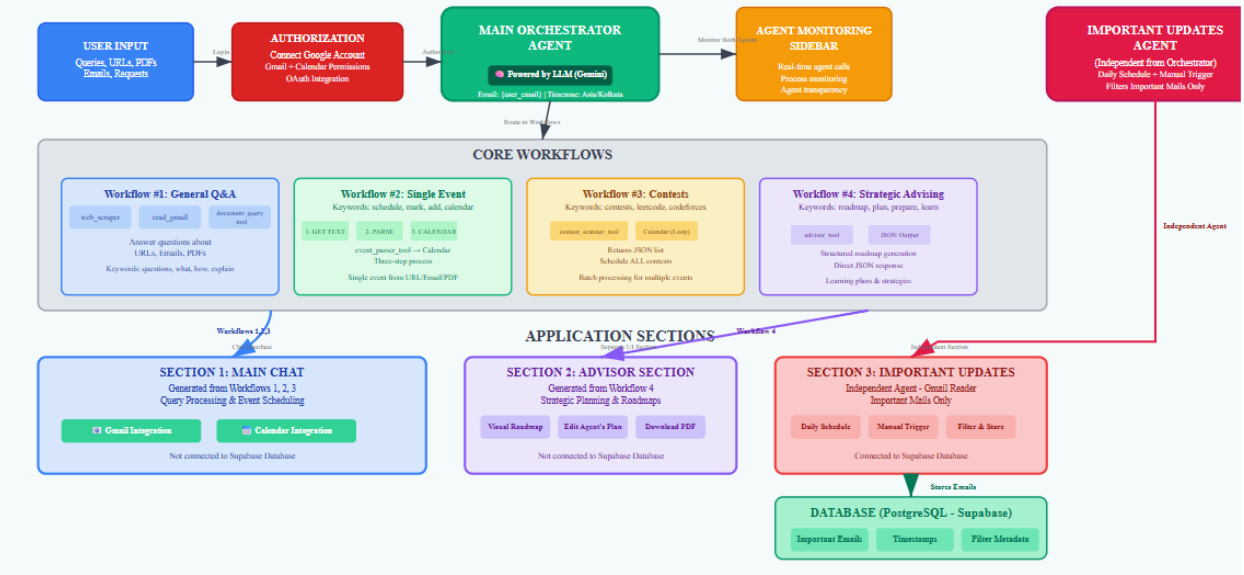
Used **Gemini Nano Banana** model to generate the image on our landing page as its excellent in image generation and quite fast and accurate then some of the recent image generation model

Prompt:(With this image)

“Take inspiration from a playful doodle illustration filled with pets, like dogs, cats, and bones, where everything is hand-drawn in a minimal black outline style with orange accents. Now, instead of pets, recreate this doodle theme for a student productivity and AI assistant concept. Keep the same quirky, abstract doodle look, with objects scattered around in a fun, casual way. Show a cheerful student with a backpack working on a laptop, and a small friendly robot assistant beside them. Replace the pet doodle icons with elements like a calendar with dates marked, a clock for time management, a stack of books with an apple on top, a graduation cap, a roadmap scroll, a PDF file icon, and an email envelope with a notification bell. Add doodles of bar graphs, a download arrow, a chain link symbol, laptops, and a web dashboard window with charts. Fill the background with small accents like stars, hearts, arrows, and playful abstract shapes. Keep the exact same doodle art vibe as the pet illustration, simple, cute, and engaging, with black outlines and orange highlights”

So used this and updated the landing page with the cartoon
https://g.co/gemini/share/9e32fdaa53c7

6.) For the low level system design document generation I have used **Claude AI model** which is quite good in styling and creating visuals for the application



Prompt:

“Create a structured system architecture workflow diagram in a clean and organized style, using rectangular nodes, color-coded sections, and directional arrows to show the flow of processes. The diagram should represent an AI Orchestrator Application for students that integrates multiple agents and tools.

At the top, show a User Input box (queries, URLs, PDFs, emails, requests). Connect it to an Authorization box (Google Account, Gmail + Calendar permissions via OAuth). From there, link to the central Main Orchestrator Agent node.

From the Orchestrator, branch into Core Workflows:

1. General Q&A – connects to web_scraper, read_gmail, and document_query_tool.
2. Single Event Scheduling – follows the strict 3-step process (Get Text → Parse Event → Calendar).
3. Contests – fetch contests from Codeforces/LeetCode using contest_scanner_tool, then schedule via Calendar.

4. Strategic Advising – advisor_tool generates JSON roadmap.

On the right, include an Agent Monitoring Sidebar showing real-time process monitoring and tool call transparency.

Below, add Application Sections:

- Section 1: Main Chat (generated from Workflows 1, 2, 3, connected to Gmail + Calendar).
- Section 2: Advisor Section (from Workflow 4, shows roadmap → edit → download).
- Section 3: Important Updates (independent Gmail Reader agent for important emails, with daily schedule + manual trigger, connected to database).

At the bottom, include a Database (PostgreSQL - Supabase) storing important emails, timestamps, and metadata.

Also show Google Services Integration (Gmail API, Calendar API, OAuth 2.0, Clist API, Admin SDK, Workspace). Finally, at the bottom row, include Data Sources & Inputs: Websites/URLs, PDFs, Gmail, Hackathons, Internships, Opportunities, Codeforces, LeetCode, Contacts, Learning, User Queries, Roadmaps.

Use arrows to clearly show how user input flows into the Orchestrator, through workflows, into app sections, and down to database/services. Keep the diagram neat, with well-labeled boxes, arrows, and grouped colored sections.”