

Snello SDE Internship – Take-Home Assignment

Welcome Builder!

At Snello we create AI agents that think, remember, and act. This assignment lets you show how you would design a small agentic system. Don't panic if you haven't built any AI systems. This assignment is simple and you are free to refer/use any platform and coding tools.

Task

Build a chatbot that uses the Gemini API (Google AI Studio) and follows an agentic architecture.

It must:

1. Hold a basic conversation, remembering the user's name and past messages.
2. Store a personal to-do list for the user.
3. Read from and write to that to-do list **through LLM tool calls** (e.g., `add_todo`, `list_todos`, `remove_todo`).
4. Persist both conversation history and the to-do list in memory (you may keep them in RAM, a file, or a lightweight store such as Supabase).

Use **LangGraph** or **LangChain** to organise the agent's steps, memory, and tool invocation. Plan the architecture however you like, but make sure your code shows clear agentic components (prompt, planner, memory, tools, executor, etc.).

Tech Guidelines

Layer	Requirements	Options
LLM	Gemini via Google AI Studio (free tier)	You can use any other LLM if you have subscription. Google AI Studio API is free.
Orchestration	LangGraph or LangChain	Custom additions welcome

Layer	Requirements	Options
Backend	Python	—
Storage	Any simple method that shows persistence of memory (RAM, file, SQLite, Supabase)	—
Frontend	Optional. CLI, or a small web UI. If you prefer a web UI you may use Node.js with React or anything similar	—

What to Submit

1. GitHub repo

- Create a GitHub repository (public)
- Include all your project source code (Python/JS/Node/etc.).
- Clear commit history: Commit every 1 hour (minimum) to reflect progress.

2. README that includes:

- Chosen architecture diagram or flow description.
 - How memory is stored and retrieved.
 - How each tool call is defined and registered with the LLM.
 - Set up and run instructions with example prompts.
 - Any limitations or future improvements.
-

Deadline

Submit within **5 days** of receiving the assignment.

Evaluation Focus

- Sound agentic design and use of LangGraph or LangChain.
- Clean, well-documented Python code and clear tool definitions.
- Thoughtful README and architecture explanation.

- Bonus points for a smooth user experience or creative extras.

Use any modern tool that boosts your productivity—Gemini, ChatGPT, Cursor, Windsurf, or others. We value smart builders who can leverage the best tech available.

Build boldly, document clearly, and have fun. We look forward to seeing what you create! 🚀

Sample Conversation

User: Hey there!

Agent: Hey! What can I help you with today?

User: Add "Finish LangChain tutorial" to my to-do list.

Agent: Got it! I've added "Finish LangChain tutorial" to your to-do list.

User: Add "Email professor" to my list.

Agent: Added "Email professor" as well.

User: What's on my to-do list?

Agent: Here's your current to-do list:

1. Finish LangChain tutorial
2. Email professor
3. Buy groceries
4. Complete assignment

User: Remove "Complete assignment" from my to-do list. I've already completed it.

Agent: Great! I've removed "Complete assignment" from your list.

User: Show my current to-dos.

Agent: You currently have:

1. Finish LangChain tutorial
2. Email professor
3. Buy groceries