

Project Report: AI-Powered Study Assistant

1. Project Overview

This project involves the development of an **AI-powered Study Assistant Chatbot** designed to help students with academic queries. The core objective was to build a system that not only answers questions but also maintains a **Persistent Contextual Memory**. Unlike basic chatbots, this assistant can recall previous interactions, providing a more personalized and human-like tutoring experience.

2. Tech Stack Used

- **Backend Framework:** FastAPI (Python)
- **LLM Model:** Llama-3.3-70b (via Groq Cloud API)
- **AI Orchestration:** LangChain
- **Database:** MongoDB Atlas (Cloud)
- **API Testing:** Swagger UI (FastAPI Docs)
- **Deployment:** Render

3. Memory Efficient (How's it works)

- The most critical feature of this project is the **Persistent Memory** handled via MongoDB Atlas.
- **Data Persistence:** Every interaction (User Query and Bot Response) is stored as a document in the MongoDB 'chats' collection, mapped to a unique user_id.
- **Contextual Retrieval:** Before processing a new request, the system queries the database to fetch the **last 5 interactions** for that specific user_id.
- **Contextual Injection:** These past messages are formatted into LangChain HumanMessage and AIMessage objects and passed to the LLM as a prefix to the current question. This ensures the AI model "remembers" the user's name and previous academic context.

4. Key Features

- **Session Persistence:** The bot retains memory across different sessions and server restarts.
- **High Performance:** Utilizing FastAPI and Groq's LPU (Language Processing Unit) ensures near-instantaneous response times.
- **Educational Guardrails:** Configured with a SystemMessage to act specifically as a "Helpful AI Study Assistant."

5. Project Links

- **GitHub Repository:** <https://github.com/knehaprajapati/chatbot/tree/main>
- **Hosted API Link:** <https://chatbot-2tog.onrender.com/>

6. API Test Screenshots

1. Screenshot 1 (Introduction):

AI Study Bot 0.1.0 OAS 3.1
/openapi.json

The screenshot displays an API testing tool interface. At the top, the title bar shows 'default' and a dropdown arrow. Below this, a breadcrumb navigation bar indicates the current path: 'GET / Home'. The main section is titled 'POST /chat Chat Endpoint' and includes 'Cancel' and 'Reset' buttons. Under the 'Parameters' tab, it states 'No parameters'. The 'Request body' section is marked as 'required' and has a dropdown menu set to 'application/json'. The 'Edit Value' tab is active, showing a JSON payload:

```
{  "user_id": "neha",  "message": "what is my name?"}
```

2. Screenshot 2 (Memory Proof):

The screenshot displays a REST client interface with the following details:

- Curl:**

```
curl -X 'POST' \
  'http://127.0.0.1:8000/chat' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
    "user_id": "neha",
    "message": "what is my name?"
  }'
```
- Request URL:** `http://127.0.0.1:8000/chat`
- Server response:**
 - Code:** 200
 - Response body:**

```
{
  "user_id": "neha",
  "bot_response": "Your name is Neha Prajapati. You told me earlier when you introduced yourself."
}
```
 - Response headers:**

```
content-length: 114
content-type: application/json
date: Sun, 22 Feb 2026 09:11:09 GMT
server: uvicorn
```
- Responses:**

| Code | Description |
|------|---------------------|
| 200 | Successful Response |

3. Screenshot 3 (MongoDB):

The screenshot shows the MongoDB Atlas interface for a cluster named 'study_bot_db' and a collection named 'chats'. The interface includes the following elements:

- Navigation:** Cluster1 > study_bot_db > chats. Buttons for 'View monitoring' and 'Visualize Your Data' are present.
- Tabs:** Documents (4), Aggregations, Schema, Indexes (1), Validation.
- Query Bar:** A search bar with the placeholder 'Type a query: { field: 'value' } or [Generate query](#)'. Buttons for 'Explain', 'Reset', 'Find', and 'Options' are available.
- Actions:** Buttons for 'ADD DATA', 'UPDATE', 'DELETE', and 'EXPORT CODE'.
- Data List:** A table showing 4 documents. The first document is expanded, showing the following fields:

```
user_id: "neha"
user_query: "hii this is neha prajapati"
bot_response: "Hello Neha Prajapati, it's nice to meet you. How can I assist you with..."
```
- Footer:** Links for 'Terms', 'Privacy', 'Atlas Blog', and 'Contact Sales'.