Memory in LangChain

Understanding how LangChain enables LLMs to remember past interactions and maintain conversational context for intelligent, context-aware applications.

What is Memory in LangChain?

Definition

Memory in LangChain refers to the **ability of a system to store and recall past interactions** during a conversation or workflow.

🧠 In simple terms: It allows the chatbot or application to "remember what was said earlier" instead of treating every input as new.

Why It's Essential

- To make LLMs context-aware
- To maintain natural conversation flow
- To deliver personalised, continuous responses
- To build meaningful user experiences

Why Do We Need Memory?

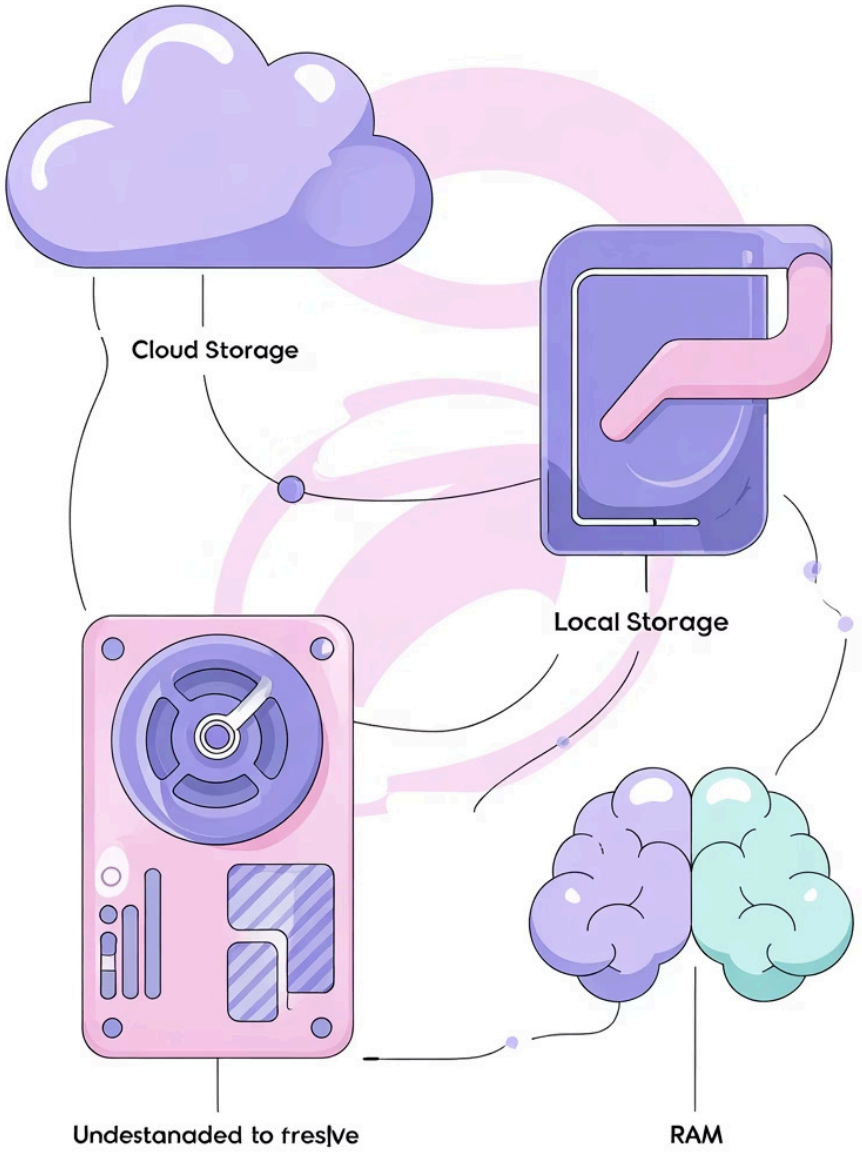
Problem Without Memory

- Model forgets past messages
- Repeats information unnecessarily
- No personalisation possible
- Disconnected conversations

Solution With Memory

- Retains complete chat history
- Refers to earlier responses
- Builds long-term context
- Enables smooth dialogue flow

💡 **Example:** User: "Who is Elon Musk?" Bot: "CEO of Tesla and SpaceX." User: "Where was he born?" → **Without memory:** Bot doesn't know "he" = Elon Musk → **With memory:** Bot understands context perfectly.



Types of Memory in LangChain

LangChain provides different types of memory depending on your specific use case and requirements:

01

ConversationBufferMemory

Stores all messages in memory, maintaining complete chat history

02

ConversationSummaryMemory

Keeps a summarised version of previous messages for efficiency

03

VectorStoreRetrieverMemory

Uses embeddings to store and recall contextually similar information

ConversationBufferMemory

1️⃣ Complete History Storage

What: Stores the **complete conversation history** in a buffer, preserving every message exchanged.

Why: Essential when full chat history is needed to maintain accurate context throughout the conversation.

When to Use

- Small, short conversations
- Customer service interactions
- Interactive FAQ systems
- When complete accuracy is required

How It Works

Keeps every message as text in memory, creating a chronological record of the entire conversation.

Example Use

Chatbot that answers follow-up questions based on the entire past conversation context.

ConversationSummaryMemory

What It Does

Instead of storing full text, it **summarises previous messages** using an LLM, keeping only the essence of the conversation.

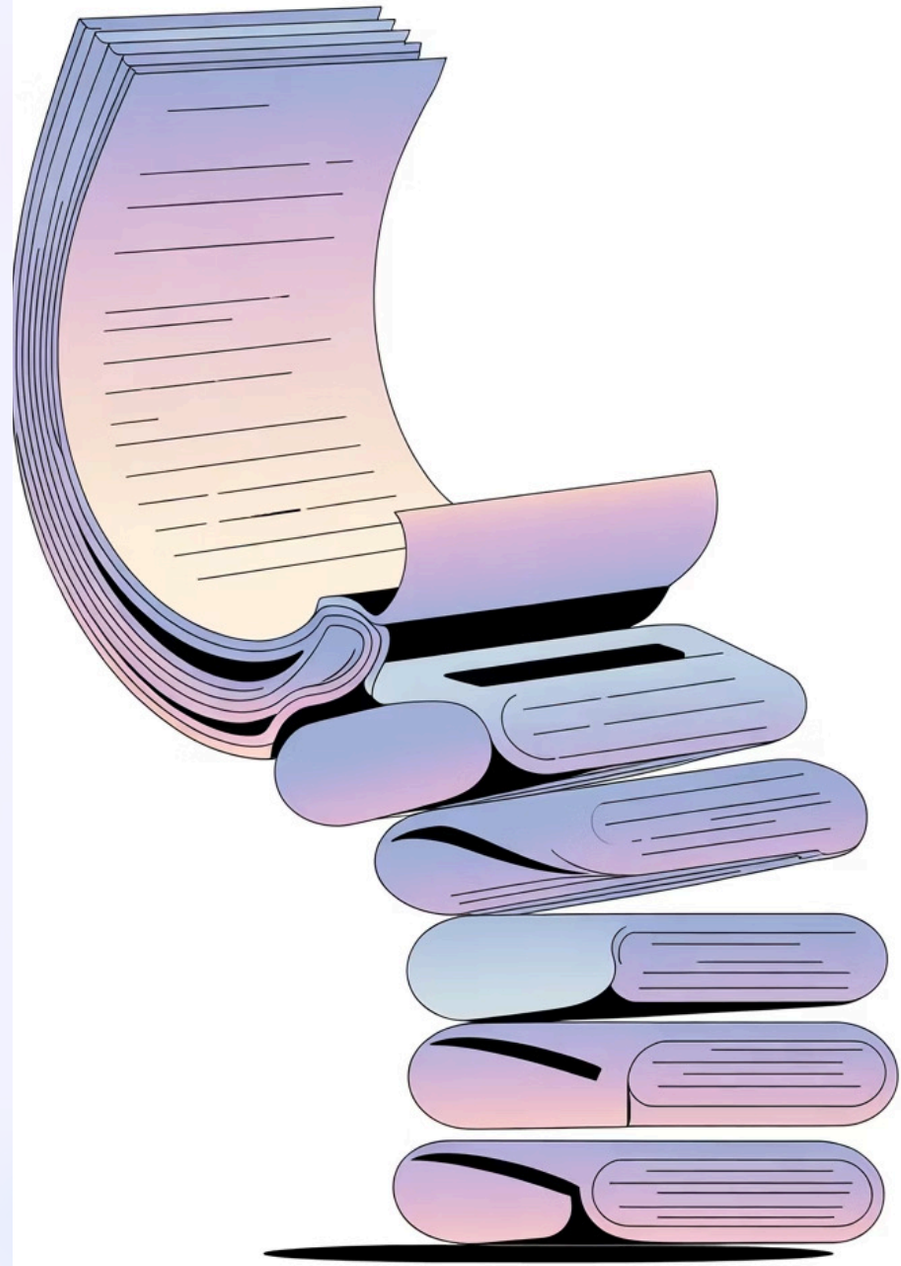
Why Use It

Saves memory and improves efficiency for long conversations without losing critical context.

Best For

Lengthy conversations, lightweight context storage, and scalable chat systems.

> **Example:** Used in chat assistants that need to remember context across 100+ messages whilst maintaining optimal performance.



VectorStoreRetrieverMemory

3️⃣ Semantic Memory

What: Stores information as **embeddings (numerical vectors)** in a database like FAISS or Chroma.

Why: Allows semantic recall — retrieves contextually relevant information, not just exact matches.

When to Use

- Knowledge-based Q&A bots
- Connecting chatbots to documents
- Database-driven applications
- Complex information retrieval

How It Works

Uses similarity search to retrieve previous context or relevant facts based on semantic meaning rather than exact text matching.

Example: Used in document-based assistants like "Chat with PDF" applications where contextual understanding is crucial.

Use Case: Context-Aware Chatbots

User Input

"Summarise the document."

Bot Response

Provides comprehensive summary

Follow-up

"Now explain point 3 in more detail."

Memory Recall

Bot knows "point 3" refers to earlier summary

Virtual Assistants

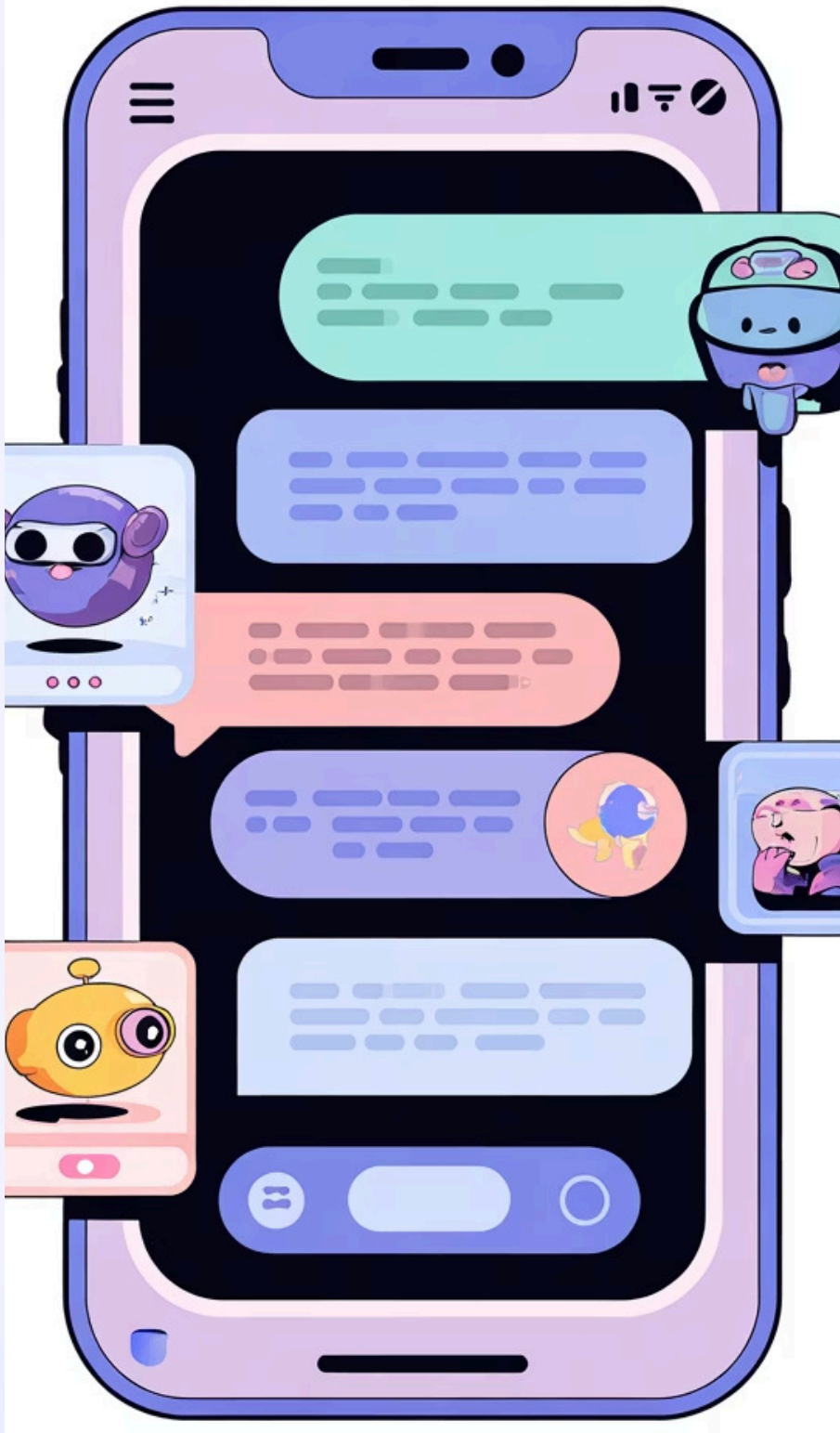
Personal AI helpers that remember preferences and history

Customer Service Bots

Support systems that maintain ticket context

Personalised Tutoring

Educational apps that adapt to learner progress



Summary: Memory in LangChain

Memory Type	Description	Best Used For	Key Benefit
ConversationBufferMemory	Stores full chat history	Short conversations	Complete recall
ConversationSummaryMemory	Keeps summarised context	Long chats	Saves space
VectorStoreRetrieverMemory	Stores semantic embeddings	Knowledge bots	Contextual recall

Key Takeaway

Memory is the "short-term and long-term brain" of LangChain apps — enabling true conversational intelligence and context-aware interactions.