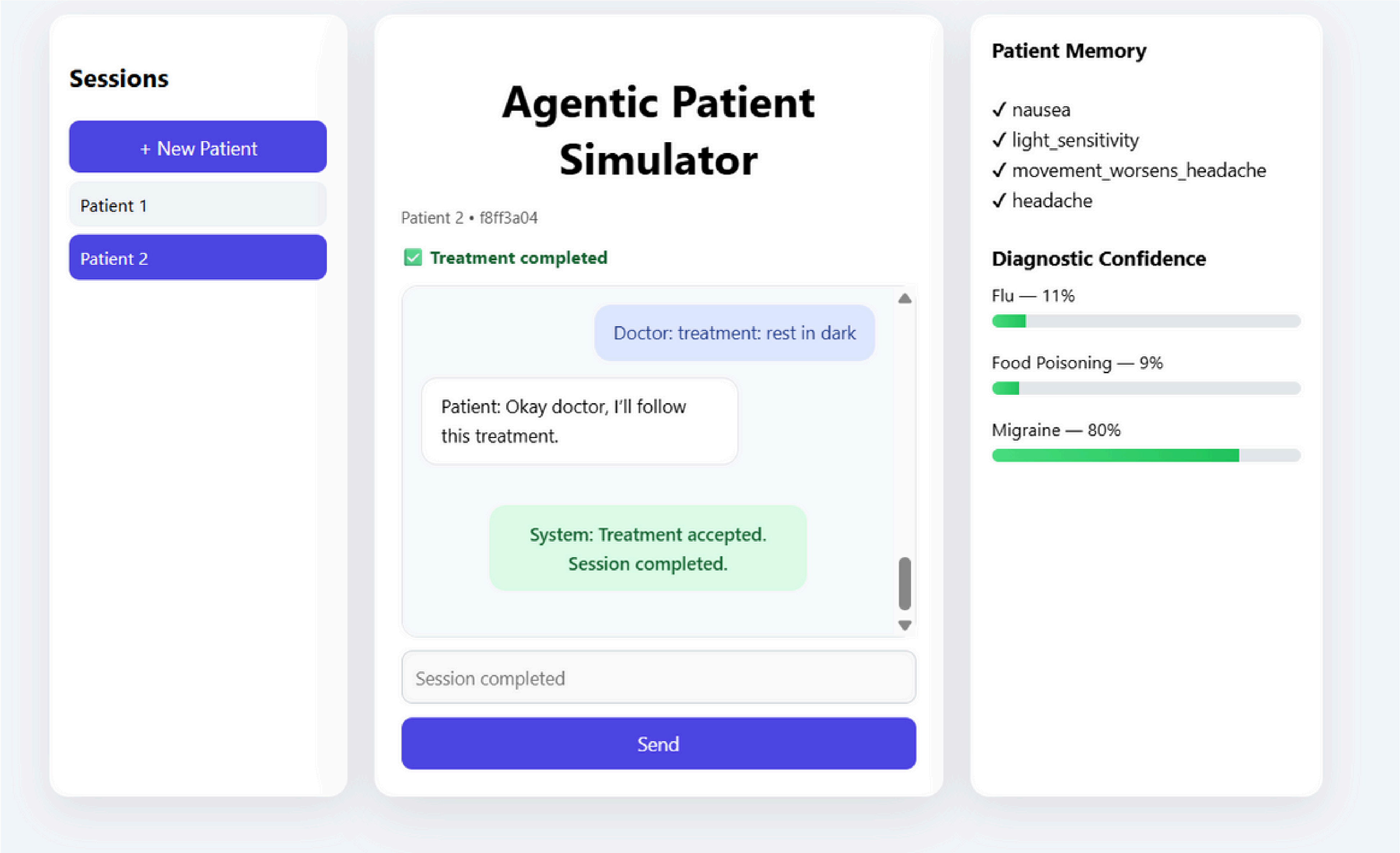
Agentic Patient Simulator

A Multi-User Agentic Medical Interview System
Built with LangGraph, LangChain, Flask, and Vanilla Web



Patient 2 • f8ff3a04

☑ Treatment completed

Doctor: treatment: rest in dark

Patient: Okay doctor, I'll follow this treatment.

System: Treatment accepted. Session completed.

Session completed

Patient Memory

- ✓ nausea
- ✓ light_sensitivity
- ✓ movement_worsens_headache
- ✓ headache

Diagnostic Confidence

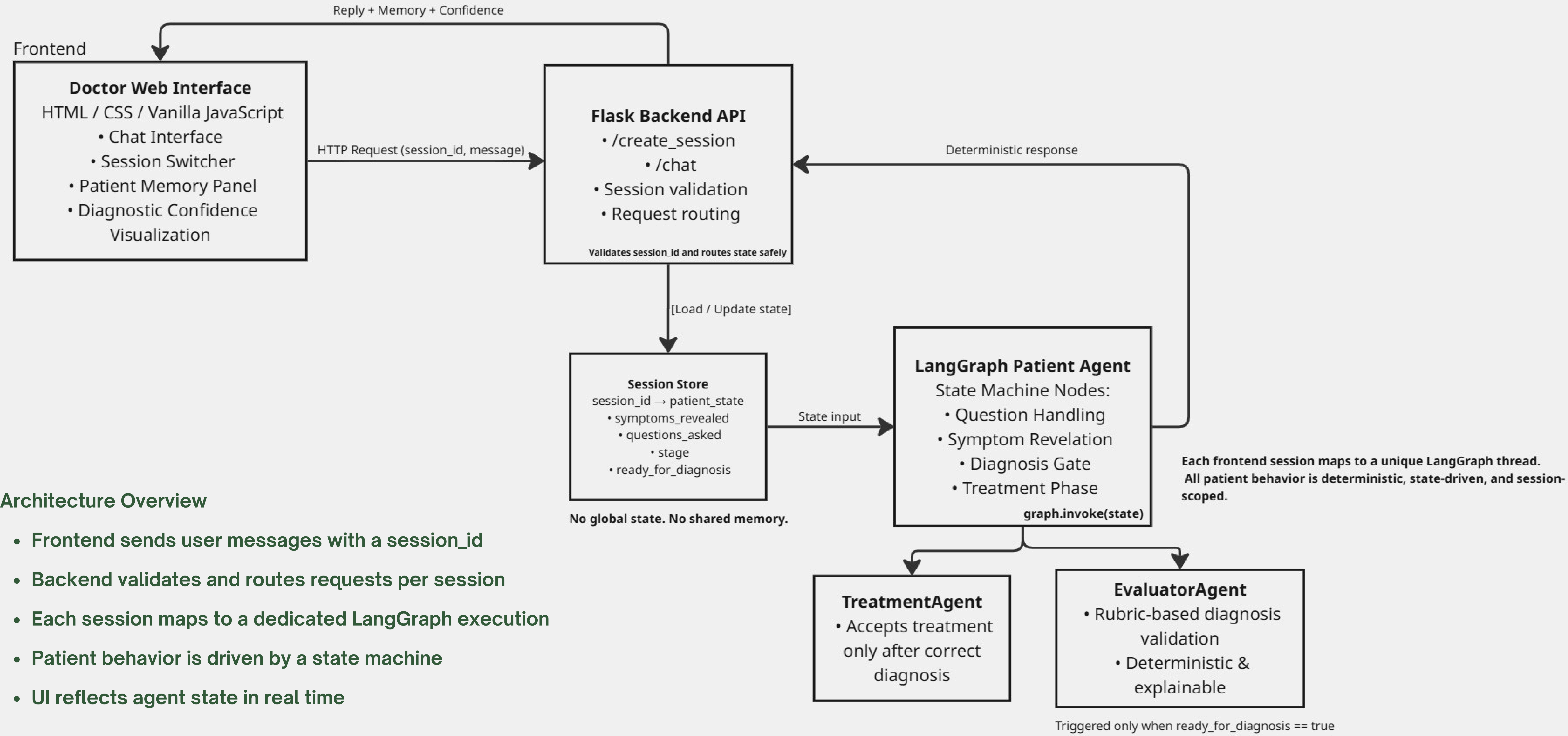
Flu — 11%

Food Poisoning — 9%

Migraine — 80%

Ashmit Verma(2024134)
IIIT Delhi

System Architecture



Reply + Memory + Confidence

Frontend

Doctor Web Interface
HTML / CSS / Vanilla JavaScript
- Chat Interface
- Session Switcher
- Patient Memory Panel
- Diagnostic Confidence Visualization

HTTP Request (session_id, message)

Flask Backend API
- /create_session
- /chat
- Session validation
- Request routing

Validates session_id and routes state safely

Deterministic response

[Load / Update state]

Session Store
session_id → patient_state
- symptoms_revealed
- questions_asked
- stage
- ready_for_diagnosis

No global state. No shared memory.

State input

LangGraph Patient Agent
State Machine Nodes:
- Question Handling
- Symptom Revelation
- Diagnosis Gate
- Treatment Phase

graph.invoke(state)

Each frontend session maps to a unique LangGraph thread. All patient behavior is deterministic, state-driven, and session-scoped.

TreatmentAgent
- Accepts treatment only after correct diagnosis

EvaluatorAgent
- Rubric-based diagnosis validation
- Deterministic & explainable

Triggered only when ready_for_diagnosis == true

Architecture Overview

- Frontend sends user messages with a session_id
- Backend validates and routes requests per session
- Each session maps to a dedicated LangGraph execution
- Patient behavior is driven by a state machine
- UI reflects agent state in real time

Technologies & Tooling

TechStack

Frontend

- HTML
- CSS
- Vanilla JavaScript

Backend

- Python
- Flask

Agent Framework

- LangGraph
- LangChain

Data / State

- In-memory session store (session-scoped)

HTML / CSS / Vanilla JavaScript

- Lightweight and framework-free UI
- Full control over session switching and real-time UI updates
- No hidden abstractions during evaluation

Flask (Backend API)

- Simple, explicit request routing
- Easy session validation and isolation
- Ideal for deterministic, state-driven APIs

LangGraph

- Models patient behavior as a state machine
- Enforces stage-gated transitions (question → diagnosis → treatment)
- Guarantees deterministic, session-scoped graph execution

LangChain

- Used for controlled LLM interactions
- Separates language understanding from decision logic
- Avoids embedding reasoning inside prompts

Session Store

- Maps session_id → patient_state
- No global memory
- Enables true multi-user isolation

Key Design Choices & Engineering Decisions

Session-Scoped State (No Global Memory)

Each user session maintains its own isolated patient state.

Why

- Prevents cross-user contamination
- Enables true multi-user support
- Makes behavior reproducible and debuggable

Separation of Intent Classification and Symptom Revelation

User questions are classified first; symptoms are revealed only if valid.

Why

- Prevents hallucinated symptoms
- Keeps patient responses medically consistent
- Allows transparent reasoning

Real-Time Confidence Visualization

Diagnostic confidence updates dynamically as symptoms are revealed.

Why

- Makes reasoning visible
- Turns the agent into a teaching tool
- Avoids black-box decision making

LangGraph for Control Flow, Not Prompts

Patient behavior is implemented as a LangGraph state machine.

Why

- Enforces strict medical flow
- Prevents premature diagnosis
- Guarantees deterministic execution

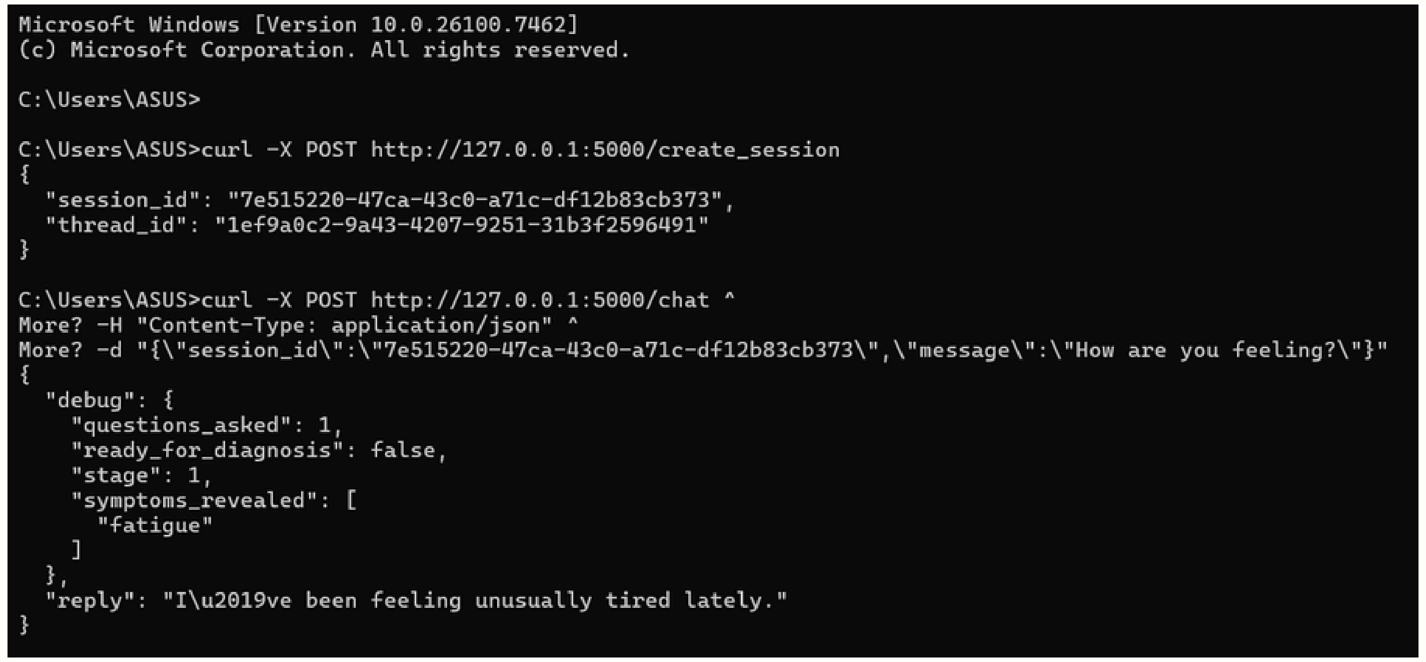
Rubric-Based Diagnosis Evaluation

Diagnosis is validated using a deterministic EvaluatorAgent.

Why

- Removes subjective LLM judgment
- Makes evaluation explainable
- Ensures fairness and consistency

Live System Demonstration & Backend Evidence



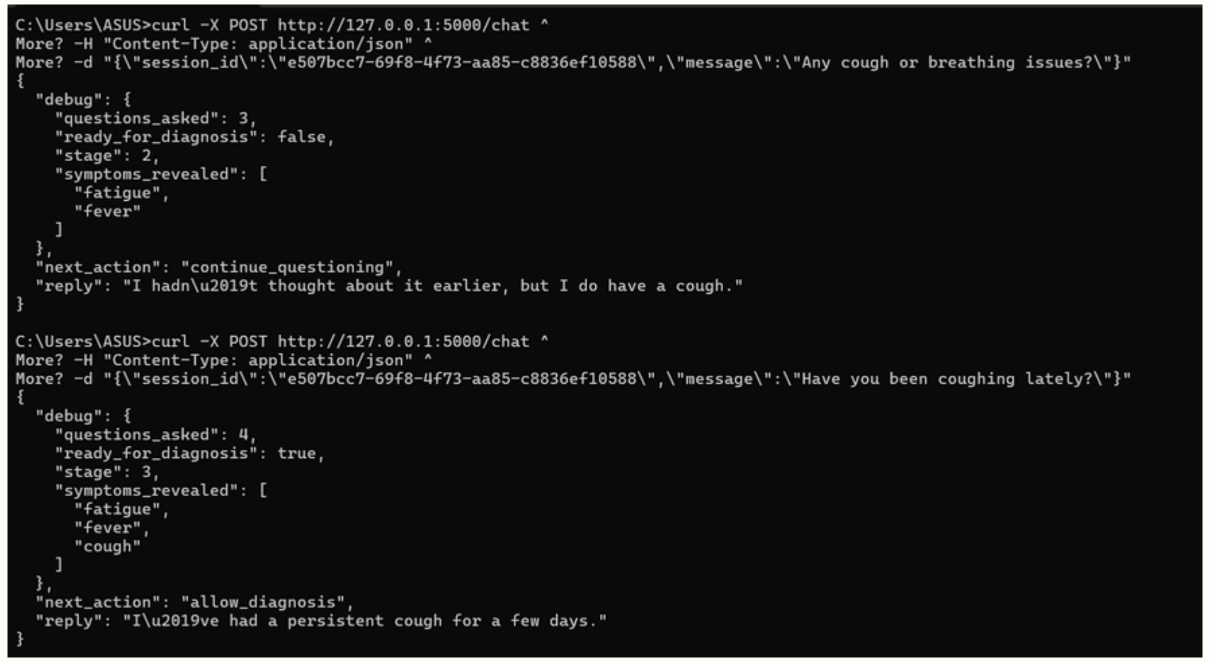
```
Microsoft Windows [Version 10.0.26100.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS>

C:\Users\ASUS>curl -X POST http://127.0.0.1:5000/create_session
{
  "session_id": "7e515220-47ca-43c0-a71c-df12b83cb373",
  "thread_id": "1ef9a0c2-9a43-4207-9251-31b3f2596491"
}

C:\Users\ASUS>curl -X POST http://127.0.0.1:5000/chat ^
More? -H "Content-Type: application/json" ^
More? -d "{\"session_id\":\"7e515220-47ca-43c0-a71c-df12b83cb373\",\"message\":\"How are you feeling?\"}"
{
  "debug": {
    "questions_asked": 1,
    "ready_for_diagnosis": false,
    "stage": 1,
    "symptoms_revealed": [
      "fatigue"
    ]
  },
  "reply": "I\u2019ve been feeling unusually tired lately."
}
```

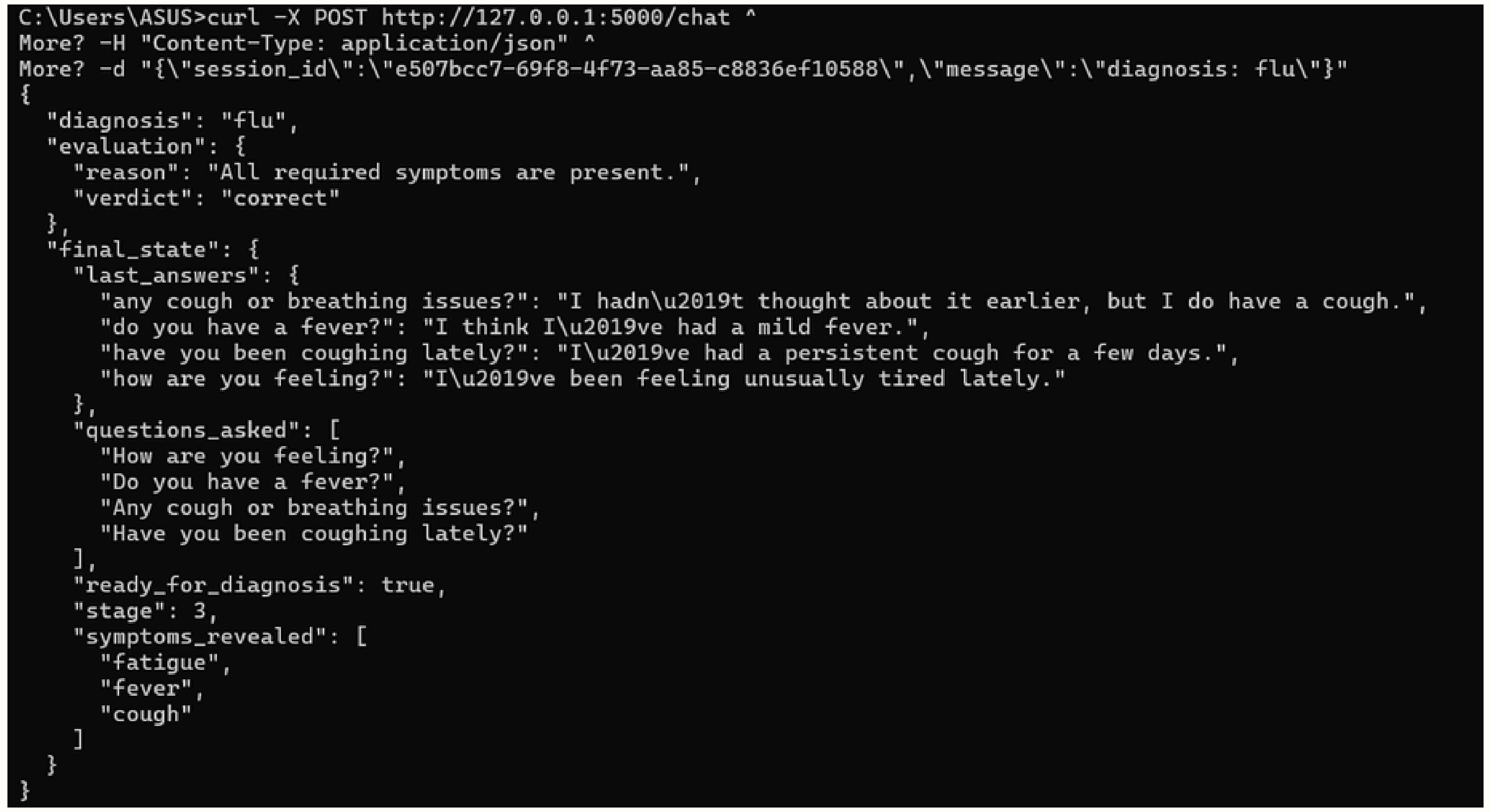
Each user initializes a unique backend session and LangGraph thread.



```
C:\Users\ASUS>curl -X POST http://127.0.0.1:5000/chat ^
More? -H "Content-Type: application/json" ^
More? -d "{\"session_id\":\"e507bcc7-69f8-4f73-aa85-c8836ef10588\",\"message\":\"Any cough or breathing issues?\"}"
{
  "debug": {
    "questions_asked": 3,
    "ready_for_diagnosis": false,
    "stage": 2,
    "symptoms_revealed": [
      "fatigue",
      "fever"
    ]
  },
  "next_action": "continue_questioning",
  "reply": "I hadn\u2019t thought about it earlier, but I do have a cough."
}

C:\Users\ASUS>curl -X POST http://127.0.0.1:5000/chat ^
More? -H "Content-Type: application/json" ^
More? -d "{\"session_id\":\"e507bcc7-69f8-4f73-aa85-c8836ef10588\",\"message\":\"Have you been coughing lately?\"}"
{
  "debug": {
    "questions_asked": 4,
    "ready_for_diagnosis": true,
    "stage": 3,
    "symptoms_revealed": [
      "fatigue",
      "fever",
      "cough"
    ]
  },
  "next_action": "allow_diagnosis",
  "reply": "I\u2019ve had a persistent cough for a few days."
}
```

Symptoms are revealed gradually and gated by medical relevance.



```
C:\Users\ASUS>curl -X POST http://127.0.0.1:5000/chat ^
More? -H "Content-Type: application/json" ^
More? -d "{\"session_id\":\"e507bcc7-69f8-4f73-aa85-c8836ef10588\",\"message\":\"diagnosis: flu\"}"
{
  "diagnosis": "flu",
  "evaluation": {
    "reason": "All required symptoms are present.",
    "verdict": "correct"
  },
  "final_state": {
    "last_answers": {
      "any cough or breathing issues?": "I hadn\u2019t thought about it earlier, but I do have a cough.",
      "do you have a fever?": "I think I\u2019ve had a mild fever.",
      "have you been coughing lately?": "I\u2019ve had a persistent cough for a few days.",
      "how are you feeling?": "I\u2019ve been feeling unusually tired lately."
    },
    "questions_asked": [
      "How are you feeling?",
      "Do you have a fever?",
      "Any cough or breathing issues?",
      "Have you been coughing lately?"
    ],
    "ready_for_diagnosis": true,
    "stage": 3,
    "symptoms_revealed": [
      "fatigue",
      "fever",
      "cough"
    ]
  }
}
```

Diagnosis is allowed only after sufficient evidence and validated deterministically.

LangGraph transition: Questioning → Diagnosis Gate (ready_for_diagnosis = true)

Key Contributions & Future Scope

Key Contributions

- Designed a multi-user, session-safe agentic medical simulator with no shared memory
- Implemented a LangGraph-driven patient state machine enforcing medical interview flow
- Introduced deterministic, rubric-based diagnosis evaluation for explainability

Future Scope

- Expand disease models and symptom graphs
- Add learning analytics for medical training evaluation
- Integrate persistent storage for longitudinal patient histories
- Support multi-agent simulations (patient + nurse + supervisor)

Designed for reproducibility, determinism, and educational evaluation