■ Technical Approach – AI Medical Scheduling Agent

1. Architecture Overview

The AI Medical Scheduling Agent was designed and implemented as a modular, intelligent scheduling system using Streamlit. It follows an agent-inspired workflow, integrating automated patient intake, schedule management, and report generation. The architecture separates the user interface, logic, and data handling into modular service layers for easier debugging and scalability.

Core Components

- PatientDB – Manages patient data stored in CSV.
- ScheduleDB – Manages doctor availability and booked appointments in Excel.
- Exporter – Generates Admin Review Reports (Excel) after every booking.
- EmailClient / SMSClient – Simulates confirmations and reminders for demo purposes.

Workflow Summary

Patient enters details → System checks existing records (CSV) → Applies scheduling rules (60/30-minute logic) → Finds available slots → Confirms booking → Updates Excel calendar → Exports Admin Review Report → Queues reminder notification.

2. Framework Choice

LangGraph and LangChain were chosen for workflow orchestration, enabling flexible, node-based scheduling logic. Streamlit serves as the front-end interface, connected to LangGraph nodes for patient lookup, slot validation, and reminders.

3. Integration Strategy

To make the system realistic yet lightweight, several integrations were implemented:
- Patients: 50 synthetic records stored in CSV.
- Doctor Schedules: Excel workbooks with availability and booking sheets.
- Appointment Form: Generates PDF post-booking.
- Reminders: Stubbed Email/SMS notifications.
- Reports: Admin Excel file generated per appointment.

4. Challenges and Solutions

- Excel file locking on Windows → Solved using context managers and atomic writes.
- Streamlit UI reruns resetting state → Fixed using st.session_state caching.
- Double booking issues → Prevented with dynamic slot filtering.
- Reminder testing without APIs → Built stub clients for local simulation.

5. Design Decisions

- Separation between UI and logic using service classes.
- CSV and Excel used for transparency and easy demo setup.
- Modular architecture for scalability and future AI integration.

6. Future Enhancements

- Integrate Google/Outlook Calendar APIs.
- Add LLM-based conversational interface.
- Implement HIPAA-compliant security.
- Extend Streamlit UI for mobile responsiveness.

7. Outcome

The project demonstrates a complete AI-driven scheduling system capable of automating patient appointments end-to-end. It showcases expertise in workflow automation, data handling, and integration of AI frameworks into practical healthcare solutions.