Project Report: Conversational IVR Modernization Framework

Topic: In-Patient Service Request & Facility Dispatch IVR

Milestone: 1 (Legacy System Analysis and Requirements Gathering)

1. Introduction

The modern hospital environment relies heavily on efficient communication between patients and care teams. However, for decades, this communication has relied on a binary, hardware-defined mechanism: the "Nurse Call Button." In the current "Legacy Reality," this button sends a signal that lacks context—a nurse does not know if a patient is having a heart attack or simply needs a glass of water without physically entering the room.

This project, the "In-Patient Service Request & Facility Dispatch IVR," aims to modernize this workflow. By implementing a Conversational AI layer—utilizing either Cloud Telephony (Twilio) or a multimodal Stand-Alone Web Simulator acting as a smart bedside tablet—alongside Python-based logic, the system will decouple non-clinical requests from clinical workflows. The proposed solution is a Voice-Activated Triage System that intelligently routes requests to the appropriate department (Housekeeping, Maintenance, or Nursing), thereby optimizing hospital operations, providing real-time visual feedback to the patient, and reducing "Nurse Fatigue."

2. Problem Statement

The current manual dispatch system presents three critical operational failures:

- **Undifferentiated Signaling:** The legacy buzzer system is binary (On/Off). It treats a request for a "pillow" with the same urgency as a request for "pain medication," forcing nurses to triage every signal manually.
- **Nurse as a Router:** Highly trained nurses spend approximately 30-40% of their shift managing logistical tasks. They act as "human switchboards," answering intercoms and manually calling support departments (e.g., Environmental Services) to report spills or broken equipment.
- **Operational Latency:** The manual relay of information creates a "Service Latency Gap." A patient request typically goes from *Patient -> Nurse -> Unit Secretary -> Facility Team*. This multi-step process is prone to delays and errors, negatively impacting HCAHPS scores regarding hospital environment responsiveness.

3. Objective

The primary objective is to design and prototype a Conversational IVR Framework that acts as an intelligent intermediary for in-patient requests.

- Primary Goal: To automate the dispatch of non-clinical service requests (Housekeeping, Maintenance, Dietary) directly to the relevant department, bypassing the nursing station.
- Technical Goal: To build a middleware layer using Python (Flask) that integrates legacy VXML concepts with modern Natural Language Understanding (NLU), deployed via Cloud Telephony (Twilio) or a custom Stand-Alone Web Simulator utilizing the Web Speech API.
- Operational Goal: To achieve a "Zero-Touch" dispatch workflow where a spoken request triggers a database entry or SMS alert without human intervention, while also providing real-time visual feedback and ticket status updates to the patient through the web simulator interface.

4. Existing vs. Proposed Systems

Feature	Existing System (Legacy Nurse Call)	Proposed System (Conversational IVR)
Input Interface	Single Hardware Button (Binary Signal).	Natural Voice Interface (Speech-to-Text).
Context	**Zero Context.** Nurse does not know the nature of the request until they answer via intercom.	**Full Context.** System captures specific intent (e.g., "Spill in Room 302") immediately.
Routing	**Manual/Inefficient.** All calls route to the Nurse Station first.	**Intelligent/Automated.** Calls are routed to Housekeeping, Maintenance, or Nurse based on keywords.
Prioritization	First-Come-First-Serve (FIFO).	Severity-Based. "Pain" overrides "Water" immediately.
Scalability	Limited by physical line cards and hardware wiring.	Infinite scalability using Cloud Telephony (Twilio)or Stand Alone Web Simulator.

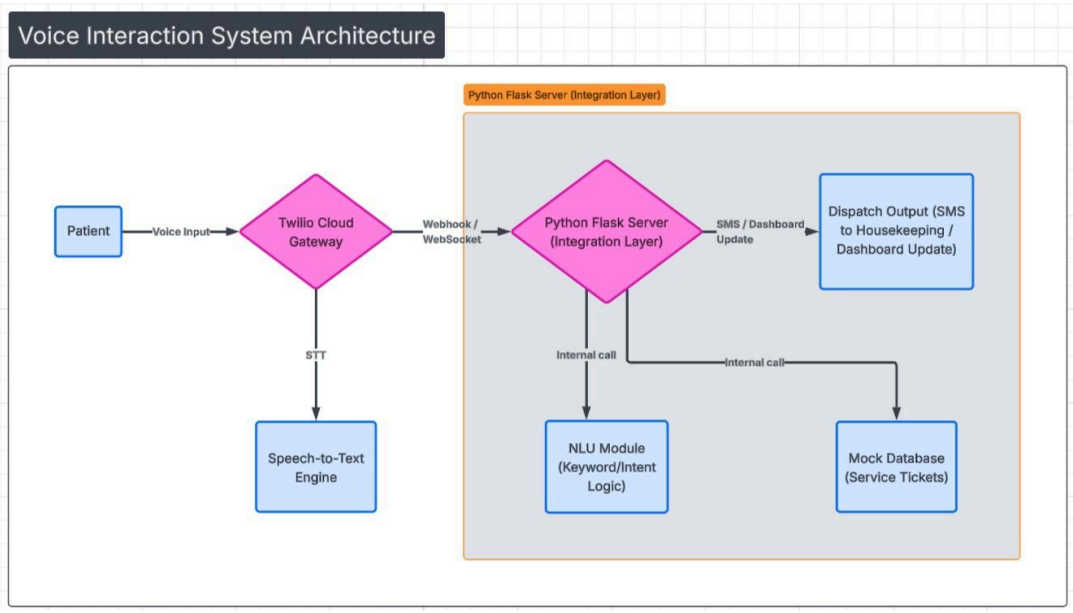
5. Architecture & Component Diagram

There are two approaches which can be used to implement this project:
1) Using Twilio : The system follows an event-driven microservices architecture, integrating cloud telephony with a local logic engine.
2) Using Stand-Alone Web Simulator : The system transitions from a traditional telephony IVR to a **Web-Based Conversational Interface**, architected as a decoupled Client-Server model. This stand-alone approach eliminates third-party telecom dependencies, allowing for rapid iteration and testing.

5.1 System Flow Description using Twilio:

1. **User (Patient):** Initiates the interaction via voice (simulating a bedside phone).
2. **Gateway Layer (Twilio):** Handles the SIP signaling and converts voice audio into a data stream.
3. **Integration Layer (Python/Flask):** The core middleware running on the server. It receives the voice input, maintains the call state, and executes the logic.
4. **Intelligence Layer (NLU - Natural Language Understanding):** Analyzes the text to identify the **Intent** (e.g., CLEANING_REQUEST) and **Entities** (e.g., Room 304).
5. **Dispatch Layer:** Based on the identified intent, the system triggers an external action (updating a Mock Database or sending an SMS to the facility team).

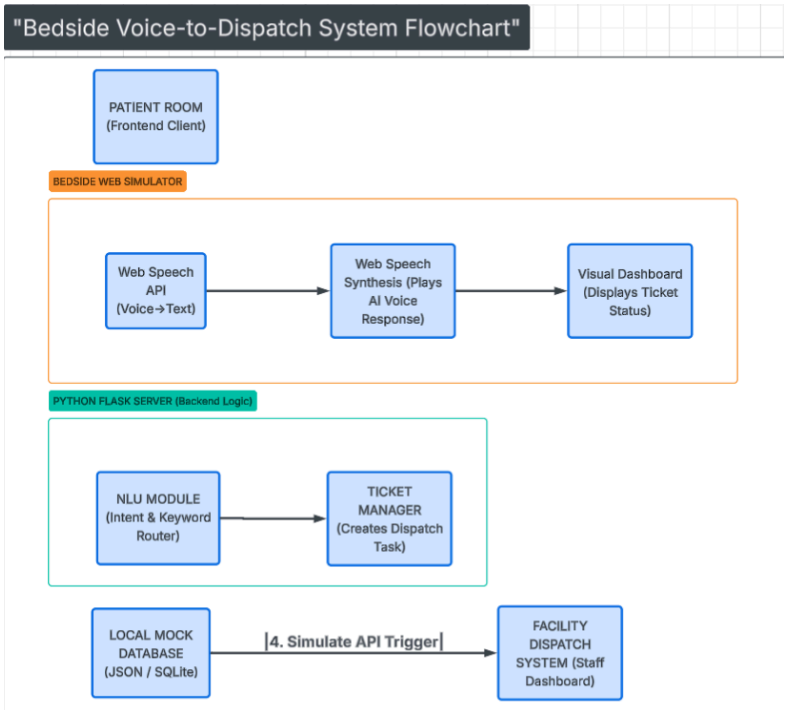
Component Diagram :



5.2 System Flow Description using Stand-Alone Web Simulator:

- **Frontend Interface (The Bedside Simulator):** A web application (HTML/CSS/JavaScript) acting as the patient's smart bedside tablet.
- **Speech-to-Text Layer:** The UI utilizes the browser's native Web Speech API to capture the patient's voice and convert it into a text string locally.
- **Data Transmission:** The frontend sends the transcribed text as a JSON payload via a RESTful HTTP POST request to the backend.
- **Integration & Logic Layer (Python/Flask):** The core middleware. It receives the text, processes it through the Natural Language Understanding (NLU) module to extract the **Intent** (e.g., MAINTENANCE) and **Keywords** (e.g., AC, broken).
- **Dispatch & Feedback Loop:** * *Backend:* The Python server logs the request into a mock database (JSON/SQLite).
    - *Frontend:* The server sends a JSON response back to the web UI, which uses a Text-to-Speech (TTS) synthesizer to speak back to the patient (e.g., *"Maintenance has been notified"*) while updating the visual ticket status on the screen.

Component Diagram :



6. Summary

The "Conversational IVR Modernization Framework" represents a shift from hardware-defined, reactive systems to software-defined, proactive intelligence. By automating the "Patient-to-Facility" loop, this project directly addresses the critical issue of nurse burnout by removing non-clinical interruptions.