1. System Design

🔹 What Are System Specifications?

Component	Specification
Platform	Streamlit (web-based)
Programming Language	Python 3.10+
Voice Input	Whisper AI (OpenAI)
Voice Output	Edge TTS (Microsoft)
Face Recognition	DeepFace (with OpenCV)
AI Model	Google Gemini 1.5 Flash
Memory Storage	JSON-based per user
Reminder System	Windows Task Scheduler (via .bat and .ps1)
Authentication	Custom login with SHA-256 hashed passwords
Audio Input	sounddevice (records user's voice)
Audio Output	pygame audio playback

🔹 What Are System Modules?

Module	Description
Authentication Module	Handles registration, login, password hashing, and session state management.
Profile Management	Stores user-specific details and displays in sidebar.
Memory Manager	Stores and retrieves personal memories using per-user JSON files.
Voice Interface	Captures voice via microphone, transcribes using Whisper, replies using Edge TTS.
AI Assistant	Interacts using Gemini 1.5, maintains context, and uses custom commands to store/retrieve memory.
Face Recognition Module	Captures face images, associates them with memories, and verifies using DeepFace.
Reminder System	Schedules alarms using Windows Task Scheduler with audio and popup messages.

Module	Description
UI Layer	Uses Streamlit to render interactive chat UI, inputs, buttons, and expansions.

2. System Environment

Environment Element	Description
Operating System	Windows 10 or later (required for reminder system)
Python Version	3.10 or above
Libraries Used	streamlit, google.generativeai, opencv-python, deepface, whisper, sounddevice, pygame, edge-tts, playsound, wave, json, asyncio, datetime, platform, subprocess, hashlib
Execution Mode	Local execution via streamlit run sample.py
Storage	JSON files in local directories (memory_*.json, face_memory_*.json, users.json)
Dependencies	Requires Whisper model download and Google Gemini API key in Streamlit secrets

3. System Testing

◆ Types of Testing Performed

Testing Type	Description
Unit Testing	Individual functions like save_memory(), authenticate(), get_time() tested with valid and invalid inputs.
Integration Testing	Modules like voice input + AI + voice output were tested in sequence.
Functional Testing	Verified user flows: login, face registration, memory storage, voice interaction, reminders.
UI Testing	Ensured responsiveness and correct rendering in Streamlit environment.
Security Testing	Verified password hashing and session isolation per user.

🔹 Test Cases

Test Case ID	Description	Input	Expected Output	Status
TC01	Register new user	Valid form data	User registered, session started	✅
TC02	Login with wrong password	Incorrect password	Login failure message	✅
TC03	Store memory via chat	"Remember my medication is at 6 PM"	Memory stored and acknowledged	✅
TC04	Retrieve memory	"What is my medication time?"	Fetched stored memory	✅
TC05	Register face	Name and image	Face captured and saved	✅
TC06	Recognize face	Live webcam	Matches and shows memory	✅
TC07	Voice input	Speech recording	Transcribed and replied correctly	✅
TC08	Set reminder	Voice command with time	Alarm scheduled in Windows	✅
TC09	Logout	Logout button click	Session cleared	✅

🔹 Results and Discussions

- ✅ **All critical features function as expected.**
- 🎯 **Voice, vision, and AI models work in harmony**, offering a unique assistive experience.
- 🧠 **User memories are persistent** across sessions, and face recognition is personalized per user.
- ⏰ **Reminder system only works on Windows**, which is a known limitation.
- 🧪 **Stress tested voice chat**, and the system handles continuous input well.
- 📉 **Limitations** include performance on lower-end systems and lack of cross-platform support for reminders.