Software Requirements Specification for

AURA – Voice Controlled Intelligent Assistant

Contents

1 Introduction

1.1 Purpose

This document specifies the software requirements for **AURA – Voice Controlled Intelligent Assistant**. It defines how the system recognizes voice commands, performs automation (system, web, and IoT), and generates natural responses.

1.2 Document Conventions

- **Bold text** for major sections and requirements.
- *Italic text* for important notes and terminology.
- Requirements are numbered sequentially as REQ-1, REQ-2, etc.

1.3 Intended Audience and Reading Suggestions

This SRS is intended for:

- **Developers:** To understand implementation details.
- **Testers:** To design relevant test cases.
- **Project Guides and Evaluators:** To review the project's functionality and scope.

Readers are advised to start with the *Overall Description*, followed by *System Features* and *Nonfunctional Requirements.*

1.4 Product Scope

AURA is an AI-powered voice assistant that performs:

- Voice command recognition and processing.
- System automation (open apps, control brightness/volume).
- Web automation (Google search, YouTube, Gmail).
- IoT device control via Sinric Pro.
- Natural language interaction using OpenAI GPT.

The goal is to create a **hands-free intelligent assistant** that increases productivity and provides a smart interaction experience.

1.5 References

- OpenAI GPT Documentation, 2024
- Sinric Pro Developer Docs, 2024
- Python SpeechRecognition Library
- IEEE SRS Format Guidelines

2 Overall Description

2.1 Product Perspective

AURA is an independent desktop-based application integrated with IoT services. It uses a layered modular architecture: *Voice Input → Processing → Action Execution → Voice Output.*

2.2 Product Functions

- Recognize and process voice commands.
- Understand intent using OpenAI GPT.
- Automate system and web tasks.
- Control IoT devices via Sinric Pro API.
- Provide feedback using text-to-speech.
- Run continuously in the background.

2.3 User Classes and Characteristics

- **End Users:** Individuals using AURA for automation.
- **Developers:** Programmers enhancing modules or adding new features.
- **IoT Device Users:** Users controlling home devices via AURA voice commands.
- **Admin (Developer):** Manages configuration, API keys, and environment setup.

2.4 Operating Environment

- **Operating System:** Windows 10 or higher
- **Programming Language:** Python 3.10+
- **Libraries:** SpeechRecognition, PyAutoGUI, OpenAI, Selenium, SinricPro, pyttsx3
- **Hardware:** Standard PC with microphone and internet connectivity

2.5 Design and Implementation Constraints

- Must run locally with offline fallback (Vosk/Sphinx).
- Internet required for OpenAI and Sinric Pro operations.
- Compatible only with Windows OS.
- Limited memory usage for continuous background operation.

2.6 User Documentation

- User Manual (Quick Start Guide)
- Installation Guide (requirements, setup steps)
- Command Reference Sheet (voice examples)

2.7 Assumptions and Dependencies

- User has an active internet connection for online operations.
- A working microphone is connected.
- API keys for OpenAI and Sinric Pro are configured properly.

3 External Interface Requirements

3.1 User Interfaces

- **Voice Interface:** Microphone input for command recognition.
- **Speech Output:** Voice response using pyttsx3.

- **Graphical Interface:** Minimal GUI or notifications.
- **System Tray Notifications:** Visual alerts for actions.

3.2 Hardware Interfaces

- Microphone for capturing user input.
- Speakers for audio output.
- IoT devices controlled via network (Sinric Pro).

3.3 Software Interfaces

- OpenAI GPT API for NLP and conversation generation.
- Sinric Pro SDK for IoT control.
- SpeechRecognition library for ASR.
- PyAutoGUI for desktop automation.
- Selenium for web automation.

3.4 Communications Interfaces

- HTTP/HTTPS protocols for API communications.
- WebSockets for IoT device connectivity.
- Local system calls for OS-level automation.

4 System Features

4.1 Voice Command Recognition

- Detects wake word ("Hey AURA") and converts speech to text.
- **REQ-1:** System must process commands from microphone input.
- **REQ-2:** Offline recognition fallback must be available.

4.2 Intent Understanding and Processing

- Analyzes commands to identify user intent.
- **REQ-3:** Must differentiate between system, web, IoT, and chat commands.

4.3 System Automation

- Executes local tasks such as opening/closing apps or adjusting volume.
- **REQ-4:** Must support OS commands like "open Chrome", "mute system".

4.4 Web Automation

- Performs Google searches, plays YouTube videos, sends emails.
- **REQ-5:** Must handle multiple web platforms using Selenium.

4.5 IoT Control

- Controls smart devices via Sinric Pro (lights, fans, etc.).
- **REQ-6:** Must support REST and MQTT communication.

4.6 AI Conversational Response

- Generates human-like responses using OpenAI GPT.
- **REQ-7:** Must produce relevant spoken responses.

4.7 Background Operation

- Runs silently and wakes up with a voice or hotkey.
- **REQ-8:** Must consume less than 10% CPU while idle.

5 Other Nonfunctional Requirements

5.1 Performance Requirements

- Response time must be below 1.5 seconds for common commands.
- System must handle continuous listening without lag or memory leak.

5.2 Safety Requirements

- The system shall not execute unsafe or unauthorized OS commands.
- Proper error-handling mechanisms to prevent crashes or infinite loops.

5.3 Security Requirements

- All API keys stored securely in .env files.
- User data and logs must not be exposed externally.

5.4 Software Quality Attributes

- **Reliability:** Continuous stable operation.
- **Usability:** Simple, intuitive voice interaction.
- **Efficiency:** Optimized for background performance.
- **Maintainability:** Modular code design.
- **Scalability:** Supports new AI or IoT features.

5.5 Business Rules

- Only authorized users can configure or extend AURA's settings.
- Internet connection required for tasks automation.

6 Other Requirements

- The system should be portable to future OS versions.
- Must support future AI integrations such as GPT Realtime or Vision API.

Appendix A: Glossary

- ASR – Automatic Speech Recognition
- TTS – Text-to-Speech
- AI – Artificial Intelligence
- IoT – Internet of Things
- NLP – Natural Language Processing

Appendix B: Analysis Models

- Use Case Diagram – Depicts user and system interactions.
- Flowchart – Represents AURA's command processing flow.
- System Architecture Diagram – Shows layered modular design.

Appendix C: To Be Determined (TBD)

- Integration of scheduling and reminder modules.
- Addition of multilingual speech support.
- Emotion-based conversational tone and sentiment detection.