# Learning Assistant System Documentation

## 1. Project Overview

### 1.1 Introduction

The Learning Assistant System is an intelligent learning tool designed to help students manage assignments, record mistakes, analyze learning progress, and interact via text or voice. Built with Python and Gradio, it offers an intuitive web interface, integrating a database, speech synthesis, and statistical analysis. The system comprises two core modules: homework\_assistant.py (main program) and voice\_assistant.py (speech processing and API interaction), catering to middle and high school students’ academic needs.

### 1.2 Key Features

* **Text-Based Interaction**: Users input academic questions or commands, select tones (Strict Teacher, Gentle Sister, Funny Classmate), and receive text and audio responses.
* **Assignment Management**: Add, delete, view assignments, and automatically clear expired ones.
* **Mistake Tracking**: Record mistakes with optional images, filter by subject, and generate practice questions.
* **Learning Analytics**: Generate mistake distribution charts and analyze weak subjects.
* **Continuous Voice Interaction**: Conduct multi-turn conversations via microphone or audio files, processed by the voice\_assistant module.

### 1.3 Technology Stack

* **Programming Language**: Python 3.8+
* **Interface Framework**: Gradio 5.33.1
* **Database**: SQLite
* **Data Visualization**: Matplotlib 3.10.3
* **Image Processing**: Pillow 11.2.0
* **Audio Processing**: Pygame 2.6.1, Edge-TTS 7.0.0
* **Speech Recognition**: SpeechRecognition 3.14.3, PyAudio 0.2.14 (voice\_assistant module)
* **API Interaction**: OpenAI 1.70.0 (DeepSeek API client)
* **Logging**: Python Logging
* **Modules**:
  + homework\_assistant.py: Main program, handling UI and business logic.
  + voice\_assistant.py: Speech processing, integrating SpeechRecognition and DeepSeek API.

### 1.4 Project Value

* **User-Friendly**: Intuitive Gradio interface with multi-tone voice interaction enhances engagement.
* **Data-Driven**: Mistake analytics identify learning gaps.
* **Modular**: Separated voice\_assistant module simplifies maintenance and extension.

## 2. Installation and Setup

### 2.1 System Requirements

* **Operating System**: Windows 10/11, macOS 10.15+, or Linux (e.g., Ubuntu 20.04+)
* **Python Version**: 3.8 or higher
* **Disk Space**: 100MB for dependencies, database, and logs
* **Network**: Required for speech synthesis (Edge-TTS), speech recognition (Google API), and DeepSeek API
* **Hardware**: 4GB RAM, 1GHz dual-core CPU, microphone (optional for voice interaction)

### 2.2 Dependency Installation

Install dependencies for both modules:

pip install gradio==5.33.1 sqlite3 matplotlib==3.10.3 pillow==11.2.0 pygame==2.6.1 edge-tts==7.0.0 speechrecognition==3.14.3 pyaudio==0.2.14 openai==1.70.0

**Note**: - Windows users may need to install wheel before pyaudio: bash pip install wheel pip install pyaudio - Linux users may require portaudio: bash sudo apt-get install portaudio19-dev

Generate a requirements file:

pip freeze > requirements.txt

### 2.3 Configuration

* **Database**: homework\_assistant.py auto-creates homework.db. Ensure the directory is writable.
* **Log Files**:
  + app\_debug.log (main program): Debugging, max 5MB, 5 backups.
  + user\_log.txt (main program): User actions.
  + voice\_assistant.log (voice module): Speech and API errors.
* **DeepSeek API**: Set DEEPSEEK\_API\_KEY in voice\_assistant.py (replace default key).
* **Matplotlib Fonts**: Windows users need SimHei font for Chinese charts or modify plt.rcParams["font.sans-serif"].
* **Speech Recognition**: Google Speech Recognition API requires no key but needs internet.
* **PyCharm Setup** (for developers):
  + To resolve SQL warnings:
    1. Go to Settings > Tools > Database > SQL Dialects.
    2. Select SQLite.
    3. Or add # @language=SQLite at the top of homework\_assistant.py.

### 2.4 Running the Program

1. Ensure dependencies are installed and the directory is writable.
2. Run the main program:

* python homework\_assistant.py

1. Access the Gradio interface at http://127.0.0.1:7861 in a browser.
2. If port 7861 is occupied, modify server\_port (e.g., 7862) in demo.launch.

## 3. System Features

### 3.1 Text-Based Interaction

* **Description**: Users input questions or commands, select a tone, and receive text and audio responses. Supports commands like “添加作业” (add assignment) or “记录错题” (record mistake).
* **Implementation**:
  + homework\_assistant.py: handle\_input parses input, calls voice\_assistant.chat\_with\_deepseek.
  + voice\_assistant.py: chat\_with\_deepseek uses DeepSeek API for responses.
* **Example**:
  + Input: “什么是牛顿第二定律？” (What is Newton’s Second Law?)
  + Tone: Gentle Sister
  + Output: Text and audio response: “牛顿第二定律是…” (Newton’s Second Law is…).

### 3.2 Assignment Management

* **Description**: Add, delete, view assignments, and clear expired ones.
* **Implementation**:
  + homework\_assistant.py: add\_homework, delete\_homework, refresh\_homework manage the homework table.
  + reset\_ids ensures continuous IDs.
* **Example**:
  + Add: “数学练习” (Math Practice), due “2025-06-30”.
  + Delete: ID 1.
  + Refresh: Clear assignments before today.

### 3.3 Mistake Tracking

* **Description**: Record mistakes with images, filter by subject, view details, and generate practice questions.
* **Implementation**:
  + homework\_assistant.py: add\_mistake, delete\_mistake, recommend\_practice.
  + voice\_assistant.py: chat\_with\_deepseek generates practice content.
* **Example**:
  + Record: “二次方程解错了” (Quadratic equation solved incorrectly), subject “数学” (Math), upload image.
  + Filter: View math mistakes.
  + Recommend: Practice for ID 1.

### 3.4 Learning Analytics

* **Description**: Generate mistake distribution charts and analyze weak subjects.
* **Implementation**:
  + homework\_assistant.py: get\_stats\_data (Matplotlib), analyze\_learning (DeepSeek).
* **Example**:
  + Chart: Math mistakes 70%, Physics 20%.
  + Analysis: Suggest “加强二次函数练习” (Focus on quadratic functions).

### 3.5 Continuous Voice Interaction

* **Description**: Multi-turn conversations via microphone or audio files (WAV, MP3), converting speech to text and generating responses.
* **Implementation**:
  + homework\_assistant.py: continuous\_voice\_chat calls voice\_assistant.process\_voice\_input.
  + voice\_assistant.py:
    - process\_voice\_input: Processes Gradio or microphone audio using SpeechRecognition.
    - recognize\_with\_speech\_recognition: Uses Google API for Chinese recognition.
    - speak: Synthesizes speech with Edge-TTS.
* **Example**:
  + Input: Audio “如何解一元二次方程” (How to solve a quadratic equation).
  + Output: Text “使用求根公式…” (Use the quadratic formula…).

## 4. Usage Guide

### 4.1 Gradio Interface Navigation

* **Layout**:
  + **Left Sidebar**: Text Interaction, Assignment Management, Mistake Tracking, Learning Analytics, Voice Interaction.
  + **Right Panel**: Input fields, buttons, lists, or charts.
* **Switching**: Click sidebar buttons, e.g., “Mistake Tracking” for mistake management.

### 4.2 Usage Examples

1. **Add Assignment**:
   * Go to “Assignment Management”.
   * Enter “数学复习” (Math Review), due “2025-06-31”.
   * Click “Add Assignment”, verify in “Assignment List”.
2. **Record Mistake**:
   * Go to “Mistake Tracking”.
   * Enter “二次函数错误” (Quadratic function error), subject “数学” (Math), upload image.
   * Click “Record Mistake”, check “Mistake List”.
3. **Voice Interaction**:
   * Go to “Continuous Voice Interaction”.
   * Record “三角函数公式” (Trigonometric formulas).
   * View response in Chatbot.
4. **Learning Analytics**:
   * Go to “Learning Analytics”.
   * View mistake chart, click “Analyze” for suggestions.

### 4.3 Notes

* **Date Format**: YYYY-MM-DD, not earlier than today.
* **Voice Input**: Requires internet, WAV/MP3 format, 16000Hz microphone.
* **Database**: Do not delete homework.db.
* **Images**: Keep below 5MB.
* **PyCharm**: Set SQLite dialect to avoid SQL warnings.

## 5. Code Structure

### 5.1 Module Breakdown

* **homework\_assistant.py**:
  1. **Configuration**: Logging, Matplotlib, chat\_history.
  2. **Utilities**: log\_action, get\_db\_connection, generate\_audio.
  3. **Database Operations**: reset\_ids, get\_homework\_list.
  4. **Business Logic**: add\_homework, recommend\_practice.
  5. **Gradio Interface**: create\_interface.
* **voice\_assistant.py**:
  1. **Configuration**: Logging, DeepSeek API.
  2. **Speech Processing**:
     + process\_voice\_input: Handles Gradio/microphone audio.
     + recognize\_with\_speech\_recognition: Google API recognition.
  3. **Text Processing**: clean\_text removes special characters.
  4. **API Interaction**: chat\_with\_deepseek calls DeepSeek.
  5. **Speech Synthesis**: speak uses Edge-TTS.

### 5.2 Core Functions

* **homework\_assistant.py**:
  + get\_db\_connection(): Initializes SQLite, creates tables.
  + generate\_audio(text: str, tone: str) -> bytes: Synthesizes speech.
  + reset\_ids(cursor, conn, table\_name: str): Resets table IDs.
  + handle\_input(question: str, tone: str): Processes text input.
  + create\_interface(): Builds Gradio UI.
* **voice\_assistant.py**:
  + process\_voice\_input(audio=None) -> Tuple[str, str]: Converts audio to text, returns response.
    - Example: text, response = await process\_voice\_input(audio\_data)
  + recognize\_with\_speech\_recognition(audio\_path) -> str: Recognizes speech.
  + chat\_with\_deepseek(text: str, history: list) -> str: Calls DeepSeek.
    - Example: reply = chat\_with\_deepseek("什么是牛顿定律")
  + speak(text: str, voice: str) -> bool: Synthesizes speech.
  + clean\_text(text: str) -> str: Cleans input.

### 5.3 Database Design

* **Table homework**:
* CREATE TABLE IF NOT EXISTS homework (  
   id INTEGER PRIMARY KEY,  
   task TEXT NOT NULL,  
   deadline TEXT NOT NULL  
  )
* **Table mistakes**:
* CREATE TABLE IF NOT EXISTS mistakes (  
   id INTEGER PRIMARY KEY,  
   question TEXT NOT NULL,  
   subject TEXT NOT NULL,  
   image BLOB  
  )

### 5.4 Logging & Error Handling

* **Logs**:
  + app\_debug.log, user\_log.txt (main program).
  + voice\_assistant.log (speech/API errors).
* **Error Handling**:
  + Try-except in all functions.
  + Example: logging.error(f"语音合成错误: {e}")
* **Analysis**:
  + Search ERROR in voice\_assistant.log for speech issues.

## 6. Development & Maintenance

### 6.1 Feature Extensions

* **Batch Operations**: Support multi-ID deletion in delete\_homework.
* **Multi-User**: Add users table with user\_id foreign key.
* **Cloud Storage**: Use AWS S3 for images.
* **Voice Enhancements**: Add multi-language support in speak.

### 6.2 Debugging & Logging

* **Steps**:
  1. Trigger error.
  2. Check app\_debug.log or voice\_assistant.log for ERROR.
  3. Verify user\_log.txt for actions.
  4. Use PyCharm debugger (e.g., process\_voice\_input).
* **Example**:
  + Error: SpeechRecognition 请求错误
  + Solution: Check network/Google API.

### 6.3 Common Issues

1. **Voice Recognition Failure**:
   * **Cause**: Network or invalid audio.
   * **Solution**:
     + Ensure internet.
     + Verify WAV, 16000Hz.
     + Check voice\_assistant.log for UnknownValueError.
2. **Database Errors**:
   * **Cause**: homework.db permissions/corruption.
   * **Solution**:
     + Delete homework.db (auto-recreates).
     + Ensure directory writable.
3. **Gradio Failure**:
   * **Cause**: Port 7861 occupied.
   * **Solution**: Use 7862 in demo.launch.
4. **PyCharm SQL Warnings**:
   * **Cause**: SQLite dialect unset.
   * **Solution**:
     + Set Settings > Tools > Database > SQL Dialects > SQLite.
     + Add # @language=SQLite.
     + Invalidate caches: File > Invalidate Caches / Restart.

## 7. Future Improvements

### 7.1 Optimizations

* **Async Database**: Use aiosqlite.
* **Indexes**: Add for homework.deadline, mistakes.subject.
* **UI**: Add delete confirmation prompts.
* **Voice**: Cache audio responses.

### 7.2 New Features

* **AI Test Generator**: Create exams from mistakes.
* **Study Planner**: Generate schedules.
* **Multilingual**: Support English, Japanese UI/voice.
* **Mobile**: Optimize Gradio for mobile.

## 8. Appendix

### 8.1 Dependency Versions

* Python: 3.8+
* Gradio: 5.33.1
* Matplotlib: 3.10.3
* Pillow: 11.2.0
* Pygame: 2.6.1
* Edge-TTS: 7.0.0
* SpeechRecognition: 3.14.3
* PyAudio: 0.2.14
* OpenAI: 1.70.0 > Run pip freeze > requirements.txt to confirm.

### 8.2 Test Cases

1. **Add Assignment**:
   * Input: “数学练习”, “2025-06-30”
   * Expected: List shows “1: 数学练习 (截止: 2025-06-30)”
2. **Record Mistake**:
   * Input: “二次方程解错了”, “数学”, “test.jpg”
   * Expected: List shows “1: 二次方程解错了 (数学)”
3. **Voice Interaction**:
   * Input: Audio “牛顿第二定律”
   * Expected: Chatbot shows “牛顿第二定律是…”
4. **Learning Analytics**:
   * Input: Click “Analyze”
   * Expected: “数学错题较多，建议复习二次函数”

### 8.3 References

* Gradio: https://www.gradio.org/docs/
* SQLite: https://www.sqlite.org/docs.html
* Edge-TTS: https://github.com/rany2/edge-tts
* SpeechRecognition: https://github.com/Uberi/speech\_recognition
* DeepSeek API: https://platform.deepseek.com/docs
* Matplotlib: https://matplotlib.org/stable/contents.html

### 8.4 Contact

* Developer: 虞澍，陈萨宇
* Github: https://github.com/yu2486789817/Learning-Assistant