**Project Documentation: Multilingual Bus Info Agent**

**1. Project Overview**

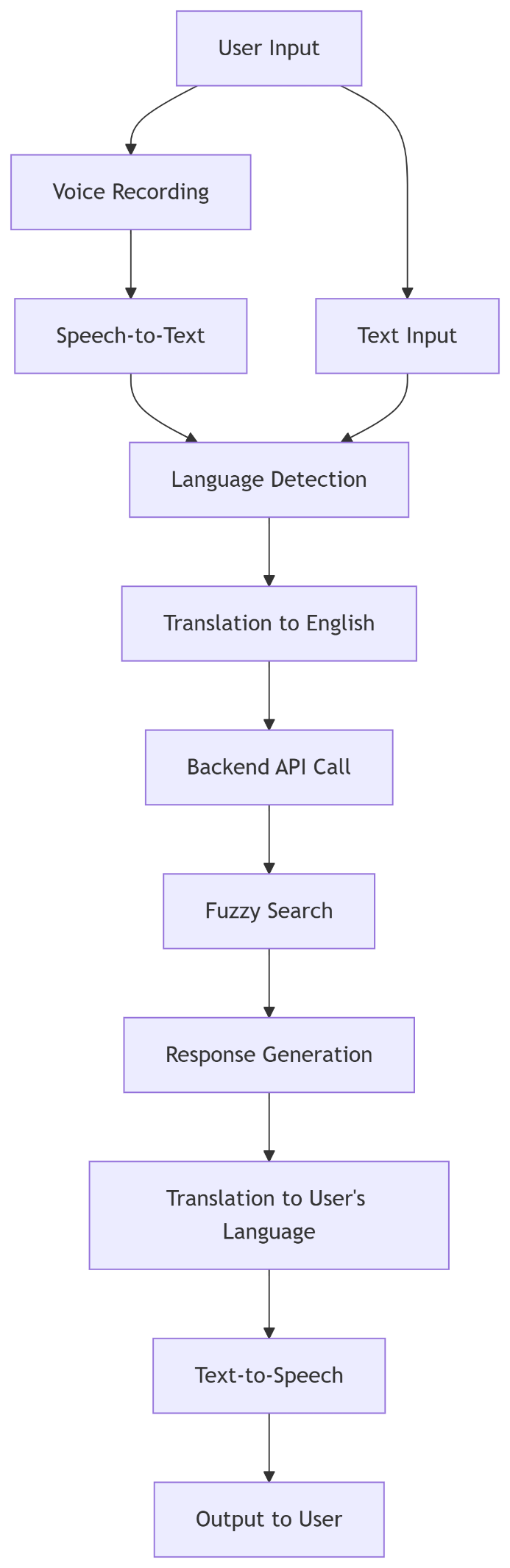
A web application that provides bus schedule information from Madanapalli to various destinations in 7 Indian languages (Telugu, Hindi, Urdu, Tamil, Kannada, Malayalam, English). Features include:

* Voice/Text input
* Automatic language detection
* Real-time translation
* Text-to-speech responses
* Fuzzy search for destinations

**2. Technical Pipeline**

Diagram

Code



**3. Key Components & APIs**

| **Component** | **Technology/Library** | **Purpose** |
| --- | --- | --- |
| **Frontend** | Streamlit (Python) | Web UI with voice recorder and input controls |
| **Backend** | FastAPI (Python) | REST API for bus schedule queries |
| **Speech-to-Text** | SpeechRecognition + PyAudio | Convert voice recordings to text |
| **Text-to-Speech** | gTTS (Google Text-to-Speech) | Convert responses to audio in user's language |
| **Translation** | deep\_translator + langdetect | Detect language and translate between English and 6 Indian languages |
| **Search Engine** | fuzzywuzzy (Python) | Find matching bus routes using fuzzy string matching |
| **Data Storage** | JSON (bus\_timings.json) | Store bus schedule data |

**4. Core Function Workflow**

1. **Input Handling** (app.py):
   * Voice: Record audio → Convert to text using speech\_recognition
   * Text: Directly capture user input
2. **Language Processing** (translate.py):

python

detect(text) *# Detect language code (e.g., 'te' for Telugu)*

LANG\_MAP[code] *# Get language metadata*

translate\_to\_english() *# Convert non-English queries*

1. **Backend Query** (main.py):

python

GET /bus?destination={query} *# FastAPI endpoint*

1. **Fuzzy Search** (search.py):

python

fuzz.token\_set\_ratio("Chenai", "Chennai") *# Returns 100% match*

*# Returns best match above 60% similarity threshold*

1. **Response Generation**:

python

*# Successful response format:*

{

"route": "Hyderabad",

"departure\_times": ["01:05", "15:40", ...],

"bus\_type": "SL"

}

1. **Output Delivery**:
   * Translate response to user's language (translate\_from\_english())
   * Convert text to speech using gTTS()
   * Display text + play audio

**5. Key Skillsets Required**

| **Category** | **Specific Skills** |
| --- | --- |
| **Languages** | Python (Advanced) |
| **Web Frameworks** | Streamlit (Frontend), FastAPI (Backend) |
| **NLP Libraries** | langdetect, deep\_translator, fuzzywuzzy, SpeechRecognition, gTTS |
| **APIs/Integrations** | REST API design, HTTP requests (requests library) |
| **Audio Processing** | PyAudio, WAV/MP3 handling, streamlit-audio-recorder |
| **Data Management** | JSON handling, fuzzy search algorithms |
| **Deployment** | Dependency management (requirements.txt), CORS configuration |

**6. Data Flow Example**

1. User says: "హైదరాబాద్ బస్సు సమయాలు" (Telugu for "Hyderabad bus times")
2. System detects language as Telugu (te)
3. Translates to: "Hyderabad bus times"
4. Backend searches bus\_timings.json → Finds 9 departures
5. Translates response to Telugu
6. Speaks: "హైదరాబాద్ కు బస్సులు: 01:05, 15:40, ..."

**7. Error Handling**

* **Translation Errors**: Fallback to English
* **API Failures**: User-friendly messages
* **Voice Recognition**: Retry prompt on failure
* **No Results**: "No buses found for [destination]"

**8. Setup & Execution**

1. Install dependencies:  
   pip install -r requirements.txt
2. Start backend:  
   uvicorn main:app --reload
3. Start frontend:  
   streamlit run app.py

**9. Customization Points**

* Add more languages in LANG\_MAP
* Modify bus\_timings.json with updated schedules
* Adjust fuzzy match threshold (currently 60%)
* Integrate real bus APIs instead of static JSON

This documentation provides a comprehensive overview of the system architecture, workflow, and technical components. The project demonstrates strong integration of NLP, voice processing, and web technologies to solve real-world multilingual information retrieval challenges.