User Manual

Software Engineering 2025

AssistU

Intelligent Voice-Enabled Student Assistant

Name	Roll Number
Bazil Suhail	Bscs22072
Abdullah Masood	Bscs22054

Contents

1 Introduction **2**

2 Technical Overview & Prerequisites **2**
2.1 General Prerequisites 2

3 Server Setup & Installation (Backend) **2**
3.1 Prerequisites 2
3.2 Installation Steps 2
3.2.1 1. Clone the Server Repository 2
3.2.2 2. Create and Activate Virtual Environment (Windows) 3
3.2.3 3. Install Dependencies 3
3.2.4 4. Environment Configuration 3
3.2.5 5. Run the Server 3

4 Client Setup & Installation (Frontend) **3**
4.1 Prerequisites 3
4.2 Installation Steps 3
4.2.1 1. Clone the Client Repository 3
4.2.2 2. Install Dependencies 4
4.2.3 3. Configure Environment Variables 4
4.2.4 4. Run the Client Application 4

5 Using the Assistu Application (Client) **4**
5.1 General Usage 4
5.2 Core Feature Modules 4
5.3 Voice Command Interface 5
5.3.1 Demonstrating Example Voice Commands 5

1 Introduction

Assistu is a modern, intelligent voice-enabled student assistant designed to simplify academic life. It allows students to manage tasks, schedule events, take notes, and generate study plans using intuitive voice commands. This manual provides a step-by-step guide to installing the server backend and the client web application, followed by instructions on utilizing its core features.

2 Technical Overview & Prerequisites

Assistu consists of two main components:

1. **Assistu Server (Backend):** Built with **Django REST Framework** and **MongoDB** for data management and features like semantic search via the **all-MiniLM-L6-v2** embedding model.
2. **Assistu Client (Frontend):** A responsive web application built with **Next.js** and **React** that provides the user interface and voice interaction capabilities.

2.1 General Prerequisites

To successfully set up the entire system, you will need:

- **Git** (for cloning the repositories)
- **Python 3.8 or higher** (for the Server)
- **Node.js (v18 or higher)** (for the Client)
- **MongoDB** installed and running locally on the default port (`mongodb://localhost:27017`).

3 Server Setup & Installation (Backend)

The server processes all voice commands, handles data, and performs AI-powered functions. It must be running *before* starting the client.

3.1 Prerequisites

- Python 3.8+
- MongoDB running on `mongodb://localhost:27017`

3.2 Installation Steps

3.2.1 1. Clone the Server Repository

Open your terminal and execute the following commands.

```
git clone https://github.com/BazilSuhail/Assistu-Server.git
cd Assistu-Server
```

The Server Repository can be accessed here: https://github.com/BazilSuhail/Assistu-Server

3.2.2 2. Create and Activate Virtual Environment (Windows)

This step isolates the server dependencies from your system's global Python packages.

```
python -m venv venv
venv\Scripts\activate
```

3.2.3 3. Install Dependencies

Install all required Python packages, including Django, DRF, and ML/AI libraries.

```
pip install -r requirements.txt
```

3.2.4 4. Environment Configuration

Create a file named `.env` in the `Assistu-Server` directory and populate it with the necessary API keys and secrets.

```
SECRET_KEY=your-django-secret-key
GROQ_API_KEY=your-groq-api-key
GROQ_LLM_URL=https://api.groq.com/openai/v1/chat/completions
```

3.2.5 5. Run the Server

Ensure your **MongoDB** instance is running. Then, start the backend server using Uvicorn.

```
uvicorn assistu_project.asgi:application --reload
```

The server should now be running at `http://localhost:8000`.

4 Client Setup & Installation (Frontend)

The client is the web application you will interact with. It connects to the running Assistu Server.

4.1 Prerequisites

- Node.js (v18+)
- The **Assistu Server** must be running on `http://localhost:8000`.

4.2 Installation Steps

4.2.1 1. Clone the Client Repository

Open a *new* terminal window and navigate to your desired directory.

```
git clone https://github.com/BazilSuhail/Assistu-Client.git
cd Assistu-Client
```

The Client Repository can be accessed here: https://github.com/BazilSuhail/Assistu-Client

4.2.2 2. Install Dependencies

Install all required JavaScript packages using `npm`.

```
npm install
```

4.2.3 3. Configure Environment Variables

Create a file named `.env.local` in the `Assistu-Client` root directory to link the client to the running server.

```
NEXT_PUBLIC_SERVER_API_URL=http://localhost:8000/api
```

4.2.4 4. Run the Client Application

Start the Next.js development server.

```
npm run dev
```

The web application will launch at **`http://localhost:3000`**. Open this address in your browser to begin.

5 Using the Assistu Application (Client)

This section details the primary features and modules accessible in the Assistu client application.

5.1 General Usage

- **Authentication:** Navigate to the **`auth/register`** page to create an account, or **`auth/login`** to sign in. The system uses secure JWT authentication.
- **Voice Commands:** The core feature. Use the **microphone icon** (Voice Command Bar) in the client interface to initiate voice interaction. This allows you to navigate the application and create content hands-free.
- **Audio Feedback:** The application provides audible confirmations and responses using your browser's Speech Synthesis API.

5.2 Core Feature Modules

The client is organized into several key modules, accessible via the sidebar or bottom navigation.

- **Dashboard:**
  - Provides a comprehensive **overview** of your academic status.
  - Displays key **statistics** such as tasks due, upcoming events, and study hours.

- **Tasks:**
  - **View, filter, and manage** all your academic tasks (assignments, exams, projects).
  - Tasks can be created using **voice commands** which are intelligently processed by the backend LLM.
  - Supports priority levels, status tracking, and subject-based organization.
- **Calendar:**
  - An **interactive calendar view** for managing all scheduled academic events.
  - Events, such as classes, meetings, and study sessions, can be created and managed via **voice input**.
- **Notes:**
  - Create notes from **text or PDF uploads**, or by **recording voice notes**.
  - Features automatic **transcription and summarization** for quick review.
  - Utilizes **Semantic Search** to find similar notes based on content, powered by the `all-MiniLM-L6-v2` model.
- **Planner:**
  - **AI-Generated Study Plans:** Input your subjects, purpose, and deadlines, and the system will generate a detailed, customizable study plan.
  - **View and manage** your created study plans, broken down into sessions.
- **Settings:**
  - Manage your **User Profile** settings.
  - Customize themes and avatars.

5.3 Voice Command Interface

The application is built around voice interaction. To use this feature:

1. **Click the Microphone Icon** in the interface to start listening.
2. **Speak your command** clearly and naturally.

5.3.1 Demonstrating Example Voice Commands

The following examples show the flexibility of the voice command system:

- **Task Creation:** "Create an assignment task for Math due tomorrow."
- **Event Scheduling:** "Schedule a study session on Friday at 2 PM."
- **Note Creation:** "Start a new voice note on Physics."
- **Navigation:** "Go to the Planner page."