# Smart Medicine Storage Device

**Madhav** [redacted]

**Abstract—**This project introduces an IoT-based medicine inventory device with an integrated AI neural network chatbot and medication reminder system. Built around a Raspberry Pi 4, the device utilizes TensorFlow for the chatbot. Medicine inventory data is managed through an HTML Java application. The chatbot provides health tips and remedies, while the reminder system alerts users to take their medication. A conveyer belt system enhances accessibility, allowing users to move medicine containers easily.

**keywords—***Raspberry Pi 4, Tensorflow, Neural Network, HTML, Java, Python*

## Contents

## 1. Introduction

Welcome to the future of medication management: our cutting-edge medicine inventory device combines an AI neural network chatbot and a medicine reminder system to revolutionize how individuals organize and adhere to their medication regimens. Powered by IoT technology such as the Raspberry Pi 4 and leveraging TensorFlow for neural network development, our device offers personalized health tips and remedies through the chatbot, while also providing customizable medication reminders. With a user-friendly design featuring a central unit and conveyor belt system, accessing and organizing medications has never been easier. This comprehensive solution empowers users to take control of their health effortlessly, ensuring they never miss a dose again.

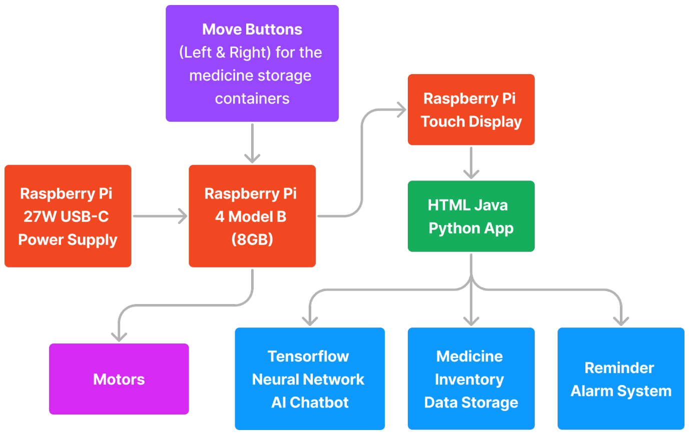

Move Buttons (Left & Right) for the medicine storage containers
Raspberry Pi Touch Display
Raspberry Pi 27W USB-C Power Supply
Raspberry Pi 4 Model B (8GB)
HTML Java Python App
Motors
Tensorflow Neural Network AI Chatbot
Medicine Inventory Data Storage
Reminder Alarm System

**Figure 1.** System Architecture

## 2. System Architecture

The system architecture of the Medicine Inventory Device with AI Chatbot and Reminder System is depicted in the diagram **(Figure 1)**. This section provides a detailed description of each component and its functionality, as well as an explanation of how the Raspberry Pi 4, TensorFlow, HTML, and Java work together to create the system.

### 2.1. Components and Functionality:

#### *2.1.1. Raspberry Pi 4 Model B (8GB)*

- Acts as the central processing unit of the device.
- Responsible for controlling the movement of the medicine storage container system and interacting with other components.
- Executes python scripts for the App.

#### *2.1.2. Medicine Storage Container Movement System*

- Physical mechanism for moving medicine storage containers.
- Controlled by Raspberry Pi GPIO Pins to move left right through the central unit using a a motor-gear mechanism.
- Allows users to move medicine containers by clicking corresponding buttons on device dashboard.

#### *2.1.3. AI Neural-Network Chatbot*

- Developed using TensorFlow framework for natural language processing.
- Runs on the touch display in the App.
- Provides general health tips and remedies based on user queries and concerns.

#### *2.1.4. App*

- User interface developed using HTML Java and Python for seamless interaction.
- Displays the chatbot interface, inventory status, and reminder settings.
- Allows users to chat with the chatbot, add/remove/edit medicine inventory data storage system, set medicine reminders.

### 2.2. Integration

- The Raspberry Pi 4 serves as the core of the system, orchestrating the interactions between different components.

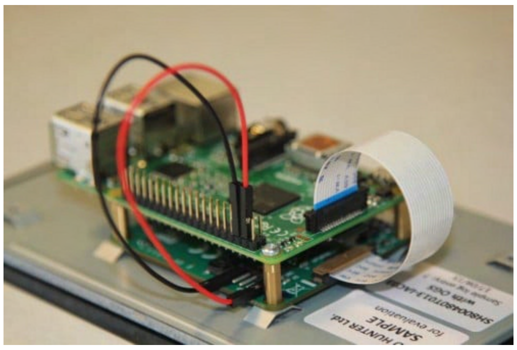


**Figure 2.** Mounting the touch display

- TensorFlow is integrated into the Python scripts to enable the AI chatbot functionality.
- HTML and Java are utilized to create the user interface for the app which will run on the touch display on the dashboard of the device.
- User can move the medicine storage containers using the move buttons on the dashboard to access the medicines in storage.

## 3. Hardware Requiremnts

### 3.1. Raspberry Pi 4 Model B (8GB)

- Raspberry Pi 4 Model B (8GB) acts as the central processing unit of the entire device.
- Required Accessories: Power adapter (Raspberry Pi 15W USB-C Power Supply), microSD Card (with Raspbian OS installed).

### 3.2. Raspberry Pi Touch Display

- Raspberry Pi Touch Display is a 7 Inch, 800 (RGB) x 480 pixels touch display which will act as the main display of this device.
- The display connects to Raspberry Pi via an adapter board that handles power and signal conversion.
- No separate power supply is required.
- Raspberry Pi OS provides touchscreen drivers with support for ten-finger touch and an on-screen keyboard.

#### *3.2.1. Mount the Touch Display*

- Mount Raspberry Pi to the back of the Touch Display using stand-offs **(Figure 2)**.
- Connect Flat Flexible Cable (FFC) to the RPI-DISPLAY port on the Touch Display PCB.
- Connect the other end of the FFC to the DISPLAY port on the Raspberry Pi.

#### *3.2.2. Power the Touch Display*

- Connect two jumper wires between the 5V and `GND` pins on Raspberry Pi's GPIO and the 5V and `GND` pins on the display **(Figure 3)**.
- One end of the black jumper wire must be connected to pin six (`GND`) on the Raspberry Pi and one end of the red jumper wire to pin four (5V)
- The other end of the black wire must be connected to the `GND` pin on the display and the other end of the red wire to the 5V pin on the display.

#### *3.2.3. Onscreen Keyboard Setup*

We can enhance Raspberry Pi OS with the convenient wvkbd on-screen keyboard. Simply execute the command below to install wvkbd and unlock seamless input capabilities:

```
$ sudo apt install wvkbd
```

**Code 1.** Installing wvkbd using apt.

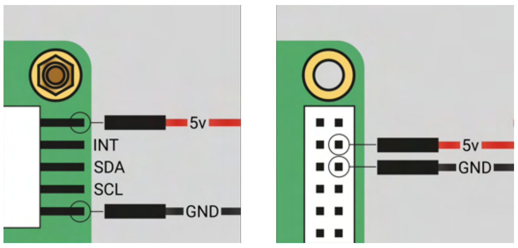


**Figure 3.** Location of display's 5V and GND pins and location of the Raspberry Pi headers

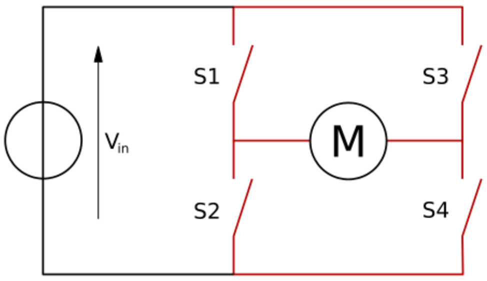


**Figure 4.** H Bridge

### 3.3. Raspberry Pi 15W USB-C Power Supply

- Intended for powering Raspberry Pi 4
- Equipped with a 1.5m fixed USB cable
- Provides stable 5.1V / 3.0A DC output
- Utilizes a USB-C connector
- Ensures reliable performance for this device.

### 3.4. Motors

A motor can't be controlled directly from the Raspberry Pi's GPIO pins, because it needs a variable supply of 5 volts. This means you need to power it separately. However, motor controller boards like L298N Dual H Bridge DC Stepper Motor Driver Controller Board can be used to provide this functionality.

#### *3.4.1. H Bridge*

- A motor can be driven forwards or backwards depending on which way around current flows through it.
- However, it would be awkward to have to rewire a motor, every time we want to change the direction it spins. To overcome this issue, motor controller boards include an H bridge **(Figure 4)**.
- An H bridge uses 4 transistors to allow digital control of which way current flows through the motor. Most H bridges also contain *flyback diodes*.
- A flyback diode prevents the voltage spike that is generated by the motor when it is no longer powered (but still spinning) from damaging delicate electronics.

#### *3.4.2. Battery Connections with Motor Board*

The motors require more power than the Raspberry Pi can provide. Therefore, we will use four AA batteries to power them **(Figure 5)**.

- Connect the positive (red) battery wire to the positive (+) power terminal on the motor controller board.
- Connect the negative (black) battery wire to the negative (-) power terminal on the motor controller board.

#### *3.4.3. Motor connections with Motor controller board*

- Using a small screwdriver, loosen the screws in the terminal blocks labeled OUT1, OUT2, OUT3, and OUT4 on the motor controller board. **(Figure 6)**
- Strip the ends of the wires attached to the motors and insert them into the corresponding terminal blocks.

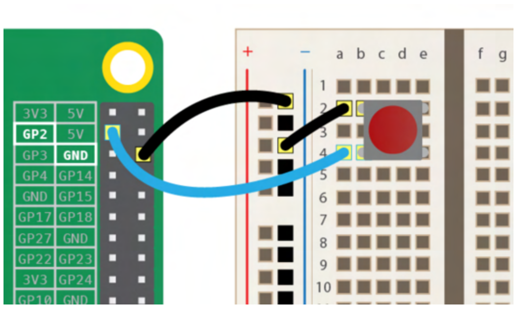


**Figure 5.** External Battery for the Motor Controller Board

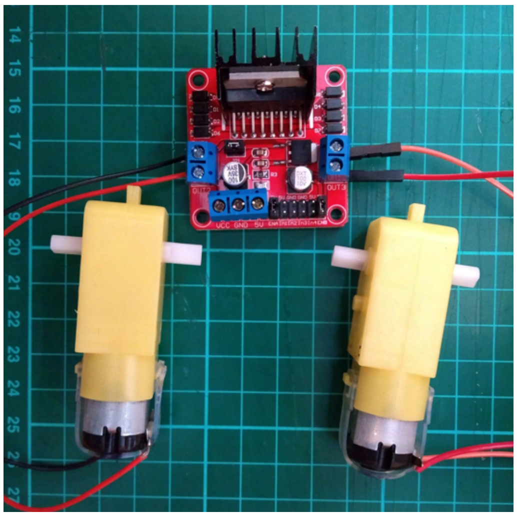


**Figure 6.** Motor connections with Motor Controller Board

- Tighten the screws to securely hold the wires in place.

#### 3.4.4. Motor Controller Board connections with Raspberry Pi

Two motor controller boards are required as we are using 4 motors in total. The connections of the two motor controller boards are as follows:

| GPIO Pin | Board Pin |
|---|---|
| 7 | ln1 |
| 8 | ln2 |
| 9 | ln3 |
| 10 | ln4 |

**Table 1.** Motor Controller Board-1 Connections with Raspberry Pi

| GPIO Pin | Board Pin |
|---|---|
| 11 | ln1 |
| 12 | ln2 |
| 13 | ln3 |
| 14 | ln4 |

**Table 2.** Motor Controller Board-2 Connections with Raspberry Pi

**Caution**

When making connections to a Raspberry Pi, do so while the Pi is off to avoid electrical risks. If connecting while powered on, ensure proper connections to prevent damage.

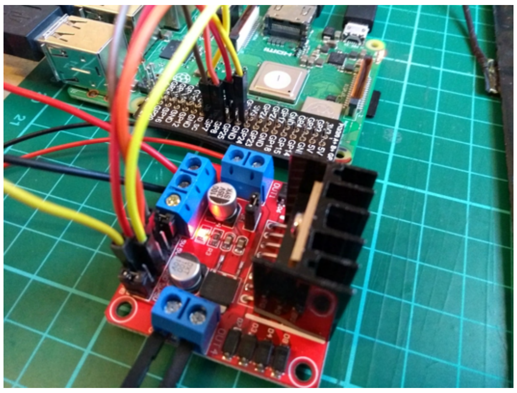


**Figure 7.** Motor Board Connections with Raspberry Pi

#### 3.4.5. Move Buttons

**Left Button**

- Connect one pin of the left button to GPIO pin 17.
- Connect the other pin of the left button to any GND pin on the Raspberry Pi

**Right Button**

- Connect one pin of the right button to GPIO pin 18.
- Connect the other pin of the right button to any GND pin on the Raspberry Pi.

### 3.5. Get Medicine Button

- Connect one pin of the left button to GPIO pin 19.
- Connect the other pin of the left button to any GND pin on the Raspberry Pi

### 3.6. 12V Solenoid

To connect a 12V Solenoid we need to use a relay to control the high-power solenoid with the low-power Raspberry Pi and to protect the Raspberry Pi from electrical issues. **(Figure 8)**

- The positive terminal of the 12v power supply is connected to the "FL-3FF-S-Z 5VDC" terminal of the relay.
- The negative terminal of the 12v power supply is connected to the ground pin of the Raspberry Pi.
- The normally open (NO) pin of the relay is connected to the positive terminal of the 12v solenoid.
- The common (COM) pin of the relay is connected to the positive voltage (5v) pin of the Raspberry Pi.
- The ground pin of the solenoid is connected to the ground pin of the Raspberry Pi.

When the Raspberry Pi sends a signal to the relay, the relay switches and allows current to flow to the solenoid, activating it.

## 4. Software Requirements

#### 4.0.1. Raspbian OS

Raspberry Pi OS, formerly known as Raspbian, is the official operating system for Raspberry Pi single-board computers. It's based on Debian Linux and offers a lightweight desktop environment, pre-installed software packages, easy configuration tools, hardware compatibility, and basic security features. It's designed to be user-friendly and optimized for Raspberry Pi hardware.

#### 4.0.2. Flask

A micro web framework for Python used to develop the backend of the application.

#### 4.0.3. NLTK Tensorflow

Python libraries utilized for natural language processing (NLP) and machine learning model training.

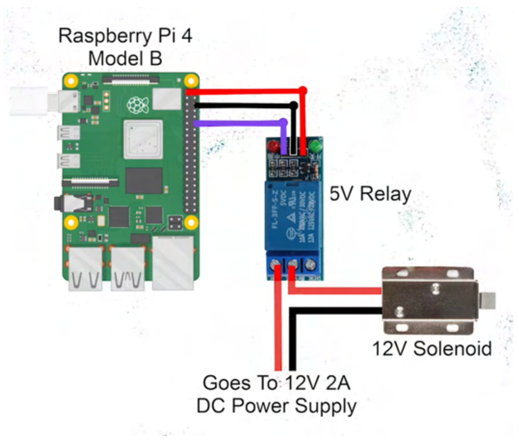


**Figure 8.** Solenoid Connections

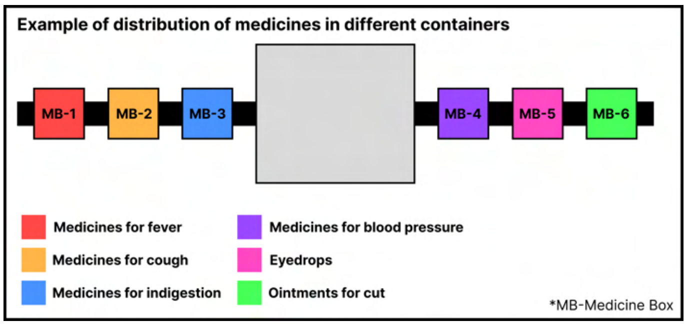

Example of distribution of medicines in different containers
MB-1
MB-2
MB-3
MB-4
MB-5
MB-6
🟥 Medicines for fever
🟧 Medicines for cough
🟦 Medicines for indigestion
🟪 Medicines for blood pressure
🟪 Eyedrops
🟩 Ointments for cut
*MB-Medicine Box

**Figure 9.** Medicine Distribution Example

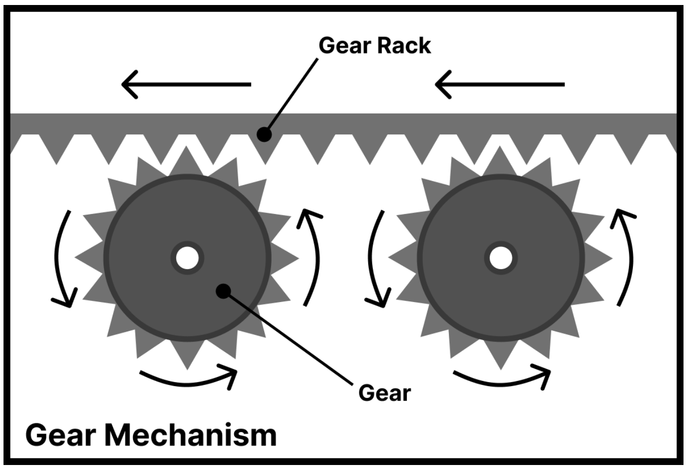


**Figure 10.** Gear Mechanism

*4.0.4. SQLite*

Database management system for storing medication and reminder data.

*4.0.5. HTML, CSS, and JavaScript*

Frontend technologies employed for creating the user interface and enhancing interactivity.

## 5. Design and Implementation

### 5.1. Medicine Inventory System

#### *5.1.1. Medicine Storage*

- The medication storage system offers users the flexibility to organize their medicines according to their preferences. Whether it's categorizing by diagnosis, medication type, or personal schedule, the system adapts to their needs.
- Each medication is housed in separate containers, allowing for efficient organization. Users can designate containers for specific days of the week or any other criteria they choose. These containers are interconnected by tubes, ensuring they remain in place during movement **(Figure 8)**.
- The user can add as many containers as per their need, as more containers can simply be connected to the gear rack beam like blocks.

#### *5.1.2. Motorised Movement Mechanism*

- The heart of the system lies in its motorised movement mechanism. Motors, situated centrally, are connected to a rack gear system within the tubes. A Raspberry Pi, equipped with movement buttons and a touch display dashboard, controls the motors.
- When users press the movement buttons, the motors engage, driving the rack gear in the desired direction. This synchronised movement causes all medication containers to shift accordingly. **(Figure 10 and Figure 11)**

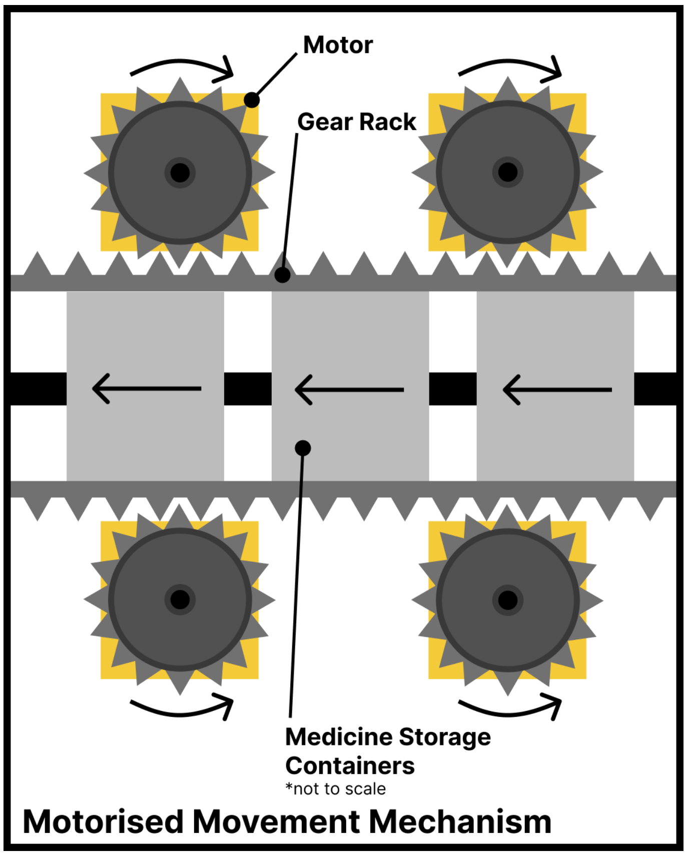


**Figure 11.** Motorised Movement Mechanism

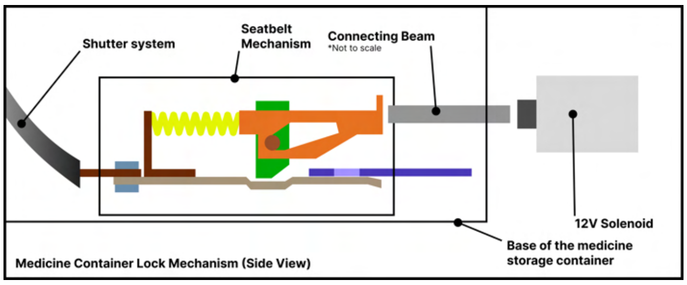

Shutter system
Seatbelt Mechanism
Connecting Beam
*Not to scale
12V Solenoid
Base of the medicine storage container
Medicine Container Lock Mechanism (Side View)

**Figure 12.** Medicine Container Lock Mechanism (Side View)

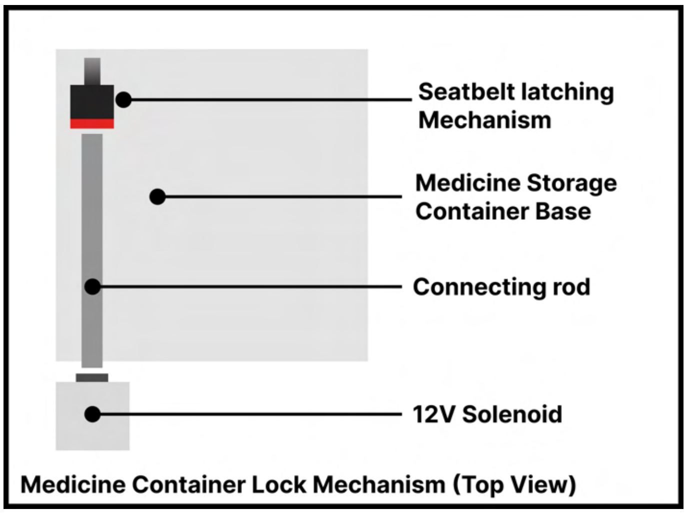

Seatbelt latching Mechanism
Medicine Storage Container Base
Connecting rod
12V Solenoid
Medicine Container Lock Mechanism (Top View)

**Figure 13.** Medicine Container Lock Mechanism (Top View)

### 5.2. Medicine Storage Container

- The Medicine Container features a secure locking mechanism that allows access only when the designated container is positioned within the central unit of the device. **(Figure 12)**
- Its components include a 12V Solenoid, a shutter system, and a seat-belt mechanism. **(Figure 14 and Figure 15)**
- Activation of the solenoid initiates the movement of the connecting beam towards the seat-belt mechanism, releasing it and thereby opening the container via the shutter, granting user access to its contents.
- Users can effortlessly close the container by simply shutting the shutter and securing the seatbelt mechanism from the outside.

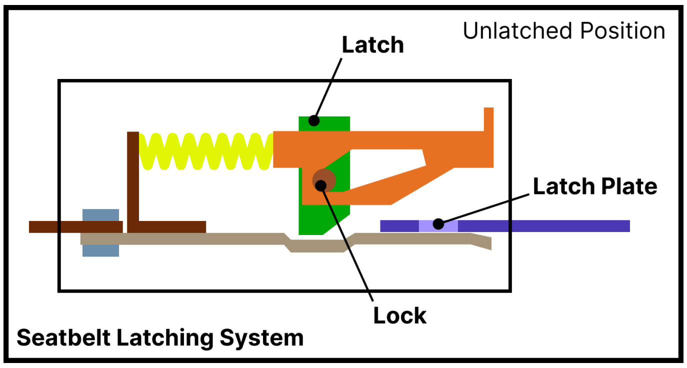
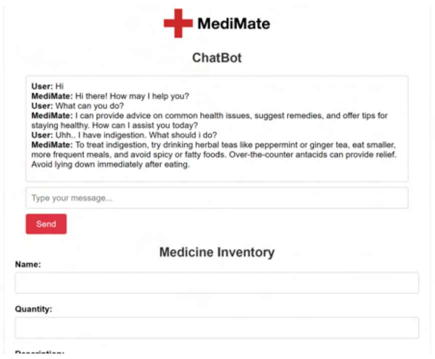


**Figure 14.** Seatbelt Mechanism

**Figure 15.** Shutter System



**User:** Hi
**MediMate:** Hi there! How may I help you?
**User:** What can you do?
**MediMate:** I can provide advice on common health issues, suggest remedies, and offer tips for staying healthy. How can I assist you today?
**User:** Uhh.. I have indigestion. What should i do?
**MediMate:** To treat indigestion, try drinking herbal teas like peppermint or ginger tea, eat smaller, more frequent meals, and avoid spicy or fatty foods. Over-the-counter antacids can provide relief. Avoid lying down immediately after eating.

**Figure 16.** Chatbot

### 5.3. App

#### *5.3.1. Tensorflow Neural Network Chatbot*

#### *5.3.2. Medicine Inventory Management*

- Allows users to add, remove, and update medication details such as name, description, and quantity.
- Displays inventory data in a tabular format for easy access and management.
- Provides search functionality to filter medication inventory based on name or description.

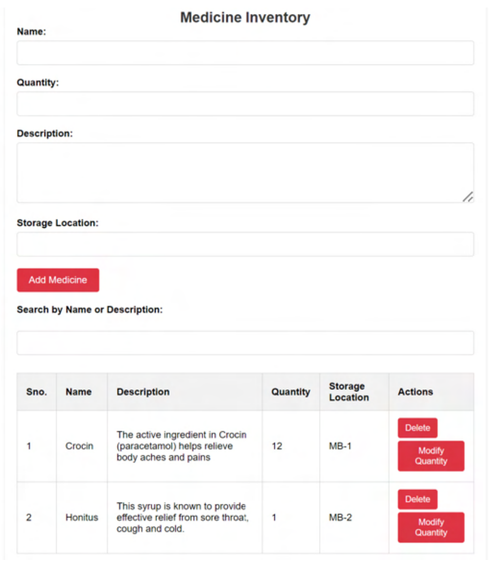
#### *5.3.3. Reminder System*

- Enables users to set reminders for medication intake, appointments, and other health-related tasks.
- Supports one-time and recurring reminders with customizable frequency options (daily, weekly, monthly).

## 6. Tensorflow Neural Network Chatbot

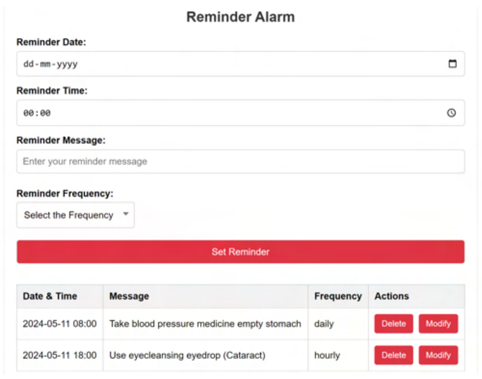
### 6.1. Chatbot functionality

1. **Preprocessing user input:**
    - When a user sends a message to the chatbot, the input text is preprocessed using the clean_up_sentence() function.
    - This function tokenizes the input text, lemmatizes the words (reducing them to their base or dictionary form), and converts them to lowercase for uniformity.
2. **Bag of Words (BoW) Representation:**
    - After preprocessing, the input sentence is converted into a bag of words representation using the bow() function.
    - This function converts the preprocessed sentence into a numerical vector, where each element represents the presence or absence of a word from a predefined vocabulary (words.json).



| Sno. | Name | Description | Quantity | Storage Location | Actions |
|---|---|---|---|---|---|
| 1 | Crocin | The active ingredient in Crocin (paracetamol) helps relieve body aches and pains | 12 | MB-1 | Delete Modify Quantity |
| 2 | Honitus | This syrup is known to provide effective relief from sore throat, cough and cold. | 1 | MB-2 | Delete Modify Quantity |

**Figure 17.** Medicine Inventory



| Date & Time | Message | Frequency | Actions |
|---|---|---|---|
| 2024-05-11 08:00 | Take blood pressure medicine empty stomach | daily | Delete Modify |
| 2024-05-11 18:00 | Use eyecleansing eyedrop (Cataract) | hourly | Delete Modify |

**Figure 18.** Reminder Alarm System

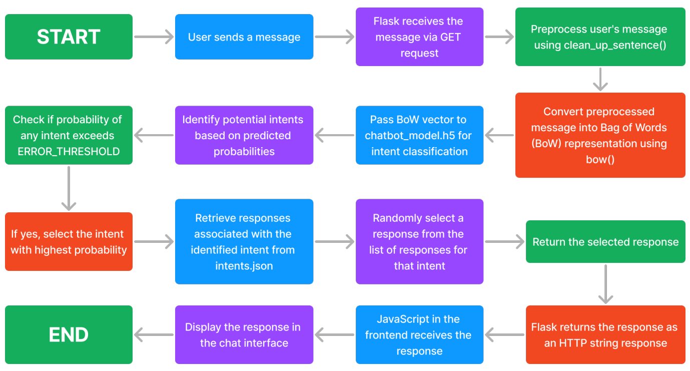

START
User sends a message
Flask receives the message via GET request
Preprocess user's message using clean_up_sentence()
Convert preprocessed message into Bag of Words (BoW) representation using bow()
Pass BoW vector to chatbot_model.h5 for intent classification
Identify potential intents based on predicted probabilities
Check if probability of any intent exceeds ERROR_THRESHOLD
If yes, select the intent with highest probability
Retrieve responses associated with the identified intent from intents.json
Randomly select a response from the list of responses for that intent
Return the selected response
Flask returns the response as an HTTP string response
JavaScript in the frontend receives the response
Display the response in the chat interface
END

**Figure 19.** Chatbot Workflow

3. **Intent Classification:**
   - The bag of words vector is then passed to the trained deep learning model (chatbot_model.h5) for intent classification. The model predicts the probabilities of different intents based on the input sentence.
   - The predict_class() function returns a list of intents sorted by their probabilities. If the probability of an intent exceeds a predefined threshold (ERROR_THRESHOLD), it is considered as a potential intent for the input sentence.
4. **Response Generation**:
   - Once the intents are identified, the chatbot selects a response based on the identified intent from the intents.json file.
   - The get_response() function retrieves a response corresponding to the identified intent from the JSON file. It randomly selects a response from the list of responses associated with the identified intent.
   - If the identified intent has multiple responses, the chatbot randomly selects one to provide variety in its interactions.

### 6.2. Integration with Flask

1. **Routing User Queries:**
   - Flask is used to define routes that handle user requests. In this case, the route /get is defined to handle GET requests containing user messages.
   - When a user sends a message through the chat interface, the JavaScript code makes an asynchronous GET request to the /get route with the user's message as a parameter.
2. **Handling User Queries:**
   - In the Flask application, the /get route is associated with the get_bot_response() function.
   - This function extracts the user's message from the request parameters and passes it to the chatbot for processing.
3. **Returning Bot Response:**
   - Once the chatbot processes the user's message and generates a response, the Flask application returns the response as a string in the HTTP response.
   - The JavaScript code in the frontend then receives the response and displays it in the chat interface for the user to see.

### 6.3. Identifying tags in intents.json

- The intents.json file contains a list of intents, each associated with a unique tag and a list of training phrases and responses.
- When a user sends a message, the chatbot identifies potential intents by classifying the input text against the training data associated with each intent.
- The tag associated with the intent that best matches the user's message is then used to retrieve a response from the JSON file.

### 6.4. Generating Responses

1. Responses in the intents.json file are structured as a list of responses for each intent.
2. When an intent is identified, the chatbot randomly selects a response from the list of responses associated with that intent.
3. This random selection adds variability to the chatbot's responses, making interactions with users more engaging and natural.

## 7. 3D Model and Measurements

Central Unit: 13.6in x 13.6in x 10in (LxBxH) Medicine Container: 6.5in x 6.5in x 7in (LxBxH)

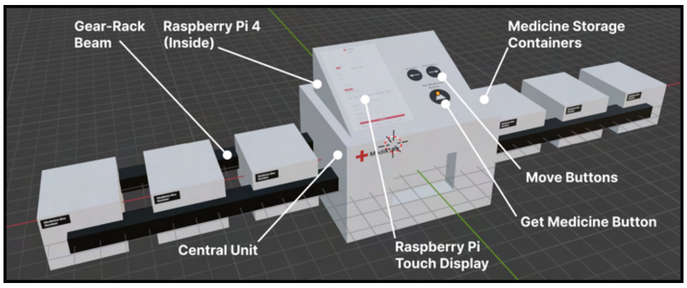


**Figure 20.** 3d Model

## 8. Code Link

Our code: https://github.com/fall-blue/Smart-Medicine-Inventory-Device