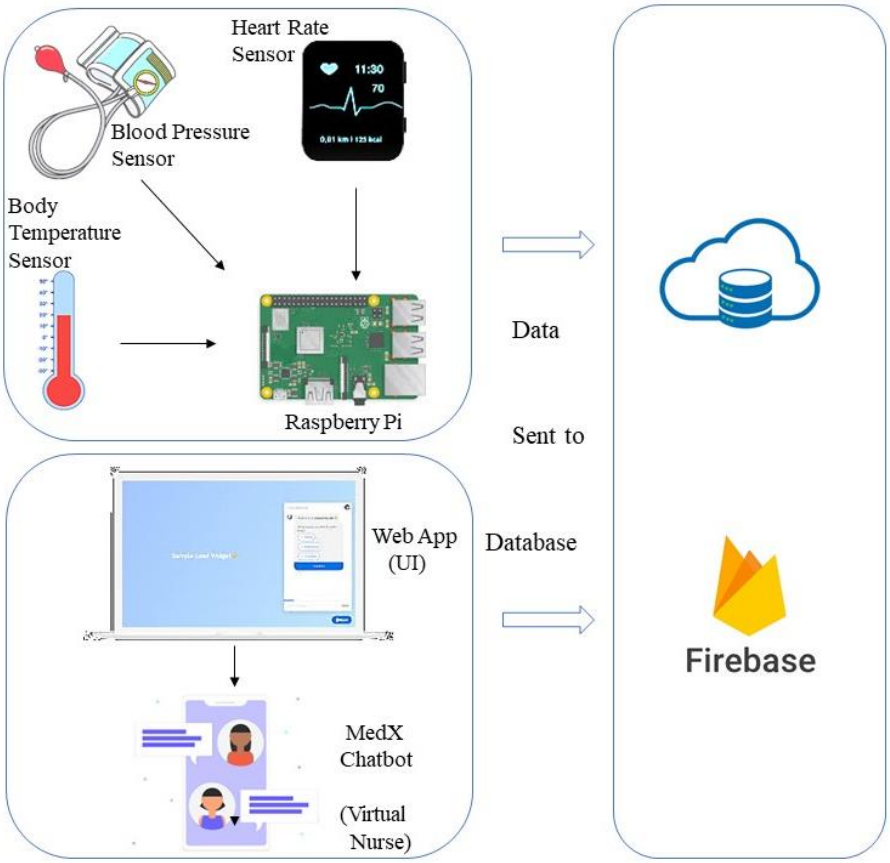
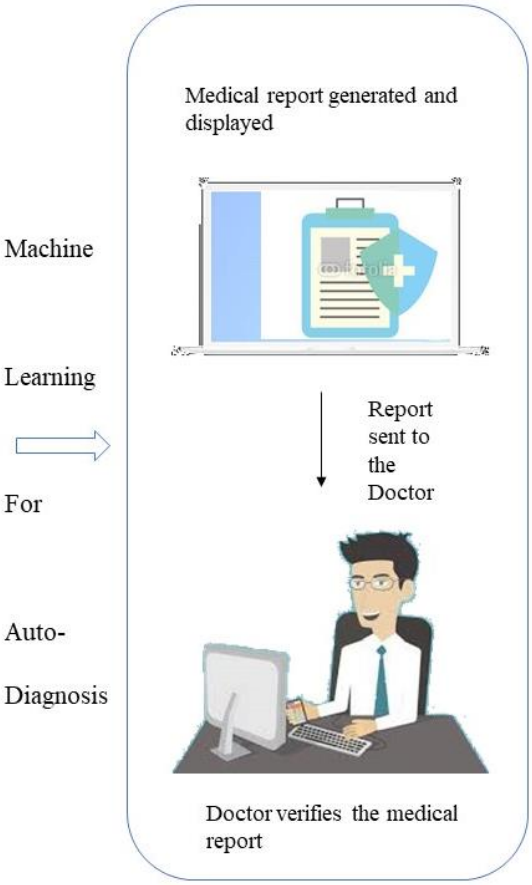


Machine Learning For Auto-Diagnosis



Hardware Specifications

1. Raspberry Pi Model 3B

2. LCD-

You can buy it from- https://www.rhydolabz.com/displays-c-88/43inch-hdmi-touch-screen-lcd-waveshare-p-2490.html?zenid=vnb07l3ejqmoep2ju7psfg7ho0

3. Blood Pressure and Heart Rate Monitor Sensor-

You can buy it from- https://www.sunrom.com/p/blood-pressure-sensor-serial-output

For interfacing- https://www.electronicwings.com/raspberry-pi/raspberry-pi-uart-communication-using-python-and-c

4. Temperature Sensor-

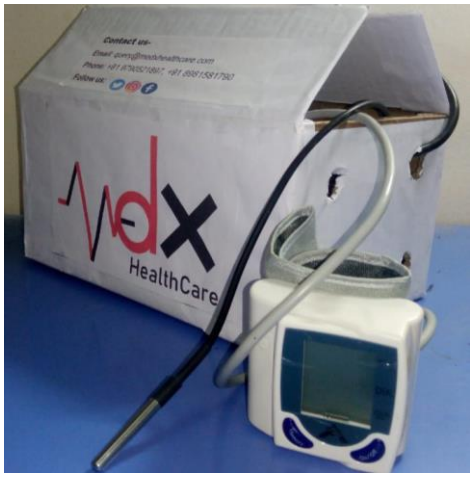
You can buy it from- https://robu.in/product/ds18b20-water-proof-temperature-probe-black-1m/

For interfacing- http://www.circuitbasics.com/raspberry-pi-ds18b20-temperature-sensor-tutorial/

Note- 10 K Ohm resistor and shouldering board to connect Temperature sensor to R pi

```
Blood pressure sensor with RPi-
SENSOR                  RPi
+5 V pin(white)         pin 4
GND pin(black)          pin 9
TX-OUT pin(grey)        pin 10

Temperature sensor with RPi-
SENSOR                  RPi
Vcc pin(red)            pin 1
GND pin(brown)          pin 6
Data pin(orange)        pin 7
```



The project can be extended by interfacing ECG sensor (AD8232) and Blood Cholesterol Sensors.

Software Specifications

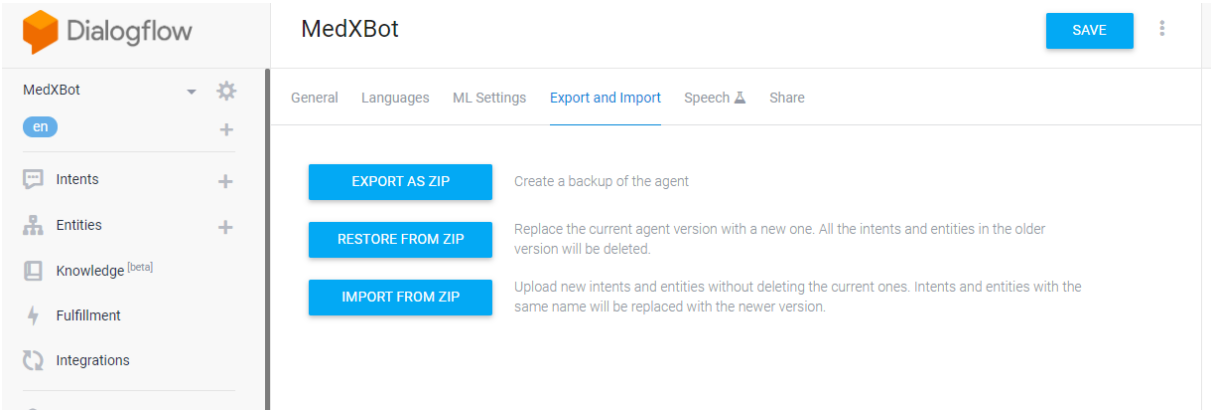
1. The sensors data is sent to **ThingSpeak**.
   (Code- `M2M_main.py`; Folder Name- ***Rpi***)
   In the code we need to change the ThingSpeak URL.

To send the same data to Firebase there is one HTML file in which Firebase URL is written.

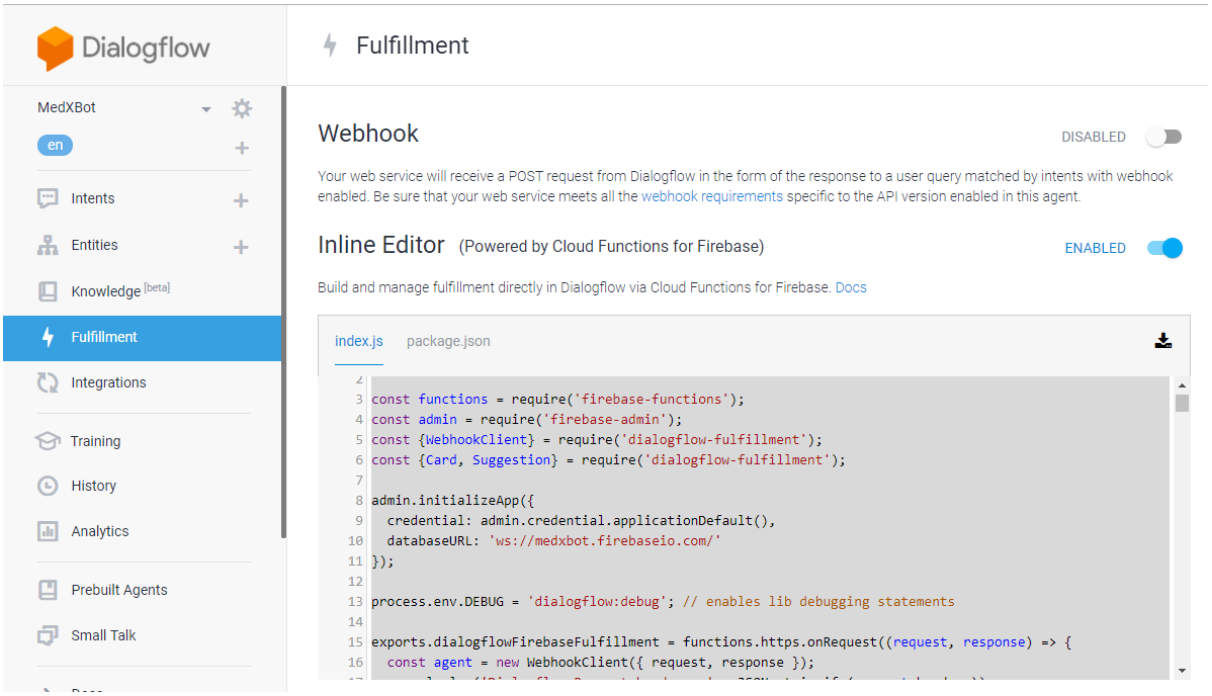
(Code- `index.html`; Folder Name- ***Rpi***)

2. The NLP based chatbot is created in Dialogflow. (**`MedXBot`** (ZIP file); Folder Name- ***Chatbot via Dialogflow***)

If we want to execute the same Bot we need to IMPORT this ZIP file in Dialogflow settings.



Dialogflow MedXBot data is also sent to Firebase. (for that **`index.js`** and **`package.json`** code files are attached in the same folder which needs to be changed in the Dialogflow Fulfillment Center).



The bot data is also sent to Firebase.

In the index.js section we need to put the Firebase URL to get the Real Time Database from Dialogflow Bot.

3. As we did not get any real time patient data for analysis, we used UCI ML repository data.

http://archive.ics.uci.edu/ml/datasets/heart+disease

https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/heart-disease.names

There are several algorithms used for the analysis. We implemented Random Forest in the project. Data .CSV files are also in the folder.

(Code- `heart_disease_prediction_RF.py`; Folder Name- ***Heart Disease Prediction System (RF)***)

Note- There is one folder ***7 ML Algos*** where other models are there which can also be implemented.

4. For UI one WebApp is developed. For Backend Python scripts are used.

(Folder Name- ***Frontend and Backend)***)

For WebApp HTML CSS is used and the scripts are in the folder.

Step 1: `home.html`

Step 2: `login.html`- the patient first logs into the website for further access.

`after_login.html`- after signing in the patient talks to the MedXBot as the bot asks about various health condition which will be needed for further analysis.

In both `login.html` and `after_login.html` Dialogflow for MedXBot (see point 2) URL is given to connect to the Firebase.

B/E- To get these data from MedXBot in the backend `medxbot_firebase.py` is run and this outputs to a data .CSV file- `MedxBot_Data.csv`

Step 3: `generate.html-` After answering all the questions, the patient goes to the Virtual Nurse section to input few more parameters related to cardiac health.

In this page, the token no., heart rate & blood pressure was taken from ThingSpeak and in the same script ThingSpeak API was written.

After submitting the details, Virtual Nurse data is also sent to Firebase and for that in the html script Firebase URL was written.

B/E- To get these data for Virtual Nurse in the backend `test-firebase.py` is run and this outputs to a data .CSV file- `Virtual_Nurse_Data.csv`

Step 4: This is the **B/E** step before generating the report. So this has to be done before clicking the generate button in the website.

To get the analysis heart_disease_prediction.py is run and this code uses `Virtual_Nurse_Data.csv` as the test sample.

It results in- `HeartCondition_Data.csv`

Step 5: The generate button is clicked. This uses `csv_to_web1.html` where all three CSV files are attached as the report.

For this particular step we need any server to host the report. We have used Wamp Server.

After installing Wamp in Local Disk (C:\wamp64) there will be one folder named **www**. We have to Copy paste everything from ***Frontend and Backend*** to ***www*** folder **(C:\wamp64\www)**.

Step 6: To send the report to the doctor we have used EasyPHP. And for that code is written in `csv_to_web1.html` script and it can be modified further.

Project by-

Saurav Mohapatra (saurabhmohapatra23@gmail.com | +91 97905 21897)

Arka Provo Mukhopadhyay (arkaprovo.mukherjee@gmail.com | +91 89815 81790)

Shivam Narula (shivam.narula7@gmail.com | +91 97906 13769)

Contact any of us for future development and improvement of MedX Healthcare system.