**QUARANTINE CHECKER AND MEDICAL DATA BASE USING RFID**

**A SENIOR DESIGN PROJECT REPORT**

*Submitted in partial fulfillment of the*

*requirement for the award of the*

*Degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**SPECIALIZATION IN VLSI**

*By*

**S.Balaji (17BEV7004)**

*Under the Guidance of*

**Dr.Bappadittya Roy**



SCHOOL OF ELECTRONICS ENGINEERING

VIT-AP UNIVERSITY

AMARAVATI- 522237

*JUNE 2021*

**CERTIFICATE**

This is to certify that the Senior Design Project work titled “**QUARANTINE CHECKER AND MEDICAL DATA BASE USING RFID**” that is being submitted by **S.Balaji (17BEV7004)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Dr.Bappadittya Roy

Guide

**The thesis is satisfactory / unsatisfactory**

**Internal Examiner External Examiner**

**Approved by**

**PROGRAM CHAIR DEAN**

B. Tech. ECE School of Electronics Engineering

**ACKNOWLEDGEMENTS**

Initially, I really like to specific my honest thanks to my guide Dr. Bappadittya Roy for the

regular assist of my study and associated research, for his endurance, motivation, and huge

information. His guidance help me in my all of the time of project and data of this these. I

couldn’t imagine to have a better guide for my project.

Along with my guide, I really like to thank the all of my committee: Dr. Jayendra Kumar and Dr.

Khairnar Vikas Vishnu for their precious feedback and boost towards my project, however

the hard doubt which incanted me to extern my venture from all the perspectives. I must need to

respect steering given with the resource of them in my task presentation that has stepped forward

our presentation abilities manner to their observation and advices.

**ABSTRACT**

In this project I have built a quarantine monetizer which detect the face of the target and compare with the location to check whether the target is in proper isolation or not. If the target is in isolation, then the module will send an email to target regarding the remaining medicine, if not in isolation, then it will send warning in mail. I have also built a medical database which have all the data of the patients in MySQL database. It will sore all the patients name, documents, appointment status, medical data. I have also added a remainder will remain the target regarding remaining medicine which he or she need to take in that time. We can also book an appointment in that app by selecting the doctor, hospital, date and time. We can access those data using RFID. If a person lost or not has his RFID, we can also access it using facial recognition. We can access those data in any hospital because as per this project those RFID is like an Aadhar card which will be issued but government so those data will be stored in data that is governed by government, so that we can access those data in any hospital in India.

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Chapter** | **Title** | **Page Number** |
| **1.** |  | **Acknowledgement** | **3** |
| **2.** |  | **Abstract** | **4** |
| **3.** |  | **List of Figures and Table** | **6** |
| **4.** | **1**  **1.1**  **1.2**  **1.3** | **Introduction**  **Objectives**  **Background and Literature Survey**  **Organization of the Report** | **8**  **9**  **10**  **11** |
| **5.** | **2**  **2.1**  **2.2**  **2.21**  **2.2.2**  **2.2.3**  **2.4**  **2.5**  **2.4.4**  **2.5**  **2.5.1**  **2.5.2** | **quarantine checker and medical data base using RFID**  **Proposed System**  **Face recognition**  **Image acquisition**  **Dataset Creation**  **Face Detection and Extraction**  **RFID**  **MySQL**  **Sending E-mail**  **System Details**  **Software details**  **Hardware details** | **12**  **12**  **13**  **13**  **13**  **13**  **14**  **14**  **14**  **15**  **15**  **17** |
| **6.** | **3**  **3.1** | **Cost Analysis**  **List of components and their cost** | **22**  **22** |
| **7.** | **4** | **Results and Discussion** | **23** |
| **8.** | **5** | **Conclusion & Future Works** | **24** |
| **9.** | **6** | **Appendix** | **25** |
| **10.** | **7** | **References** | **39** |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| 1 | Covid 19 impact | 8 |
| 2 | Face recognition Block Diagram | 12 |
| 3 | SQL Block Diagram | 12 |
| 4 | Screen 2 | 12 |
| 5 | face detection | 13 |
| 6 | Raspberry pi 4 | 17 |
| 8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | Arduino UNO  RFID  UI -1  UI - 2  Registration form  SQl  table  person table  GPS  Map  Output 1  Output 2  Output 3  Output 4  Output 5 | 18  18  19  19  20  20  21  21  21  22  22  22  23  23  23  23  23 |

**CHAPTER 1**

**INTRODUCTION**

There have been many diseases and disorders reported in this world, but recently a term COVID-19 has changed this world emotionally and economically. We can say a term quarantine has been mostly used due to this. More than 700 million has been pushed to poverty, around 3.5 billion people are pushed into work risk globally, this world has lost about 3% of its gross domestic product (GDP). Can we prevent its? Yes, we can control it in some extent. If we were in proper social distancing and good quarantine technique, we can prevent it at some distance. So here I have proposed a project using facial recognition to recognize the face and compare it with google map location to check weather he is in proper isolation or not, and I also added an medical data base to keep all the medical records in proper format.

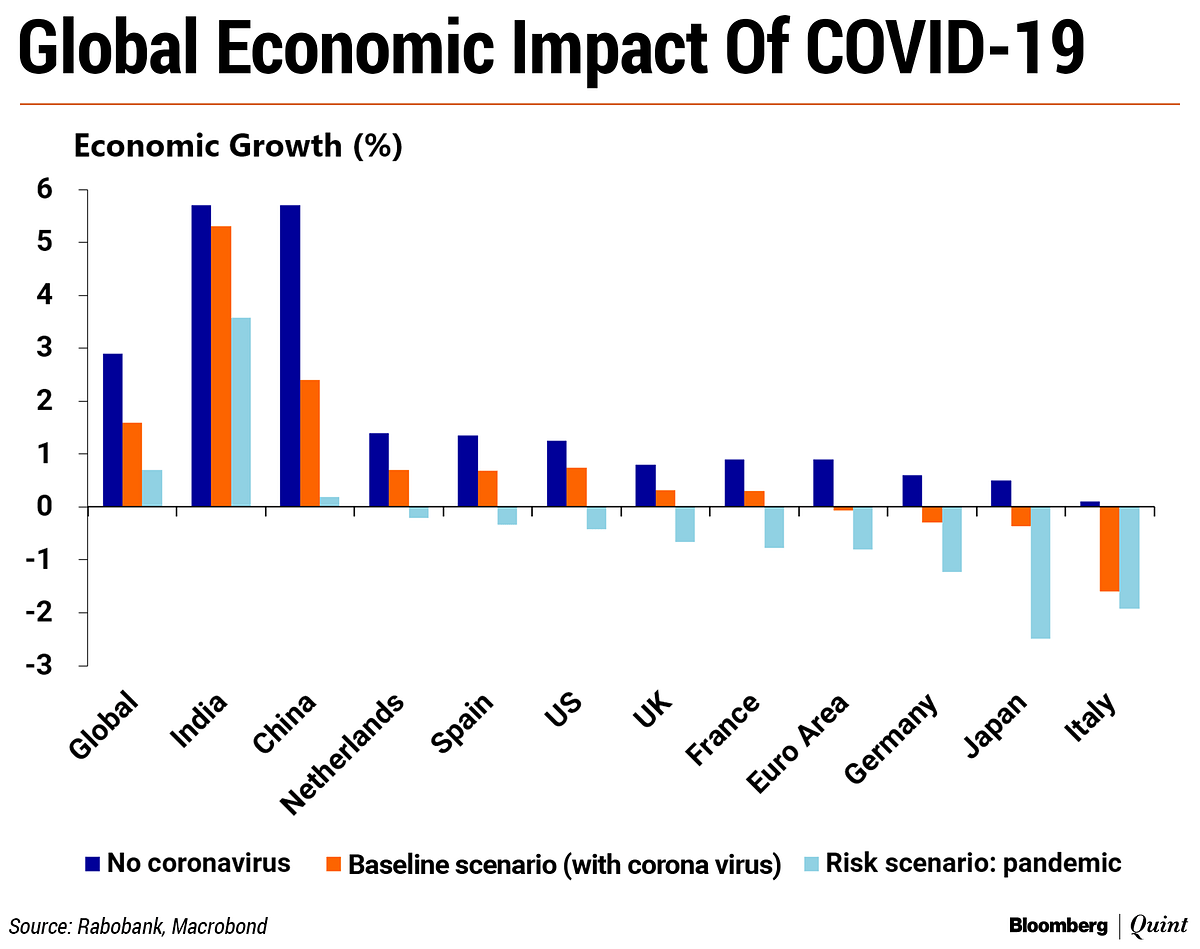


Figure 1

The medical database is developed using MySQL. It’s a database which is mostly utilized to start a web form application. We all have our own medical records. If we go for a sophisticated hospitals they give us a bunch of mesical files and tell us to carry for all there hospital appointment. Which is more complex in mobility and usability to make easier I have proposed a medical data base using MySQL which contains all targets medical history and their profile. 40% of deaths in India is due to lack of medical history and profile, using facial recognition we can overcome this.

Since we have all the medical history and facial data, we can get all the medical data using facial recognition.

* 1. **Objectives**

The following are the objectives of this project:

* To design a facial recognition system to detect the faces effectively using machine.
* Developing a blink detector inside facial detector to find image or any video.
* Tracking the target location for random time in a day to keep the target in isolation.
* To store all the medical data in MySQL database and separating all those data in tables.
  1. **Background and Literature Survey**

A similar proposal is done by Michael Donver, Albit Douhgt, Tim Diboi, in year 2008 have given a concept of facial recognition device. The device contains facial landmark, datasets, light detector and face pinchers. This is published under the usage of attendance recorder using machine learning and Open cv2.

The 2nd proposal is authored by H. Sherow and O.P Idewo, as improvement in device control using fingerprint. Here the data is stored by packages and can be accessed by the form of packages. This is built to reduce the performance of the computer and less heat generation of PC. This technique also used in many software developing company to reduce manpower.

The concept of utilizing all the user data into the form of tables and colloem to reduce the searching efficiency of the processor of done by K. operta and Y. kim john in year 1978 has done a good improving in usage of MySQL data. This is presented to make search fast in server for better result.

The 3rd paper was proposed by JMTZ Bee healthcare, Inc in year 2000, for better patients record section and using those records for diagnose. The project is titled by “Database Management Systems”. This proposal is done by a private medical hospital for give next generation of medical support for patient with them. Here first the system gathers the information for analysis for the hospital by sing the entity – “Doctors, Paycheck, patient, Insurance, Invoice, CPT, Diagnosis, Prescription, Medicine”. They also introduce a registration form explicitly for their patient for easy usage of this software.

Robust 30D Facial recognition: this proposal delivers a facial identification module that satisfies the problem of changing in face expression in external condition.

**1.3 Organization of the Report**

The remaining chapters of the project report are described as follows:

* Chapter 2 contains the proposed design, Facial recognition, RFID, Sending Email, SQL, Software and Hardware.
* Chapter 3 gives the cost involved in the execution of the project.
* Chapter 4 discusses the results obtained after the project was implemented.
* Chapter 5 concludes the report.
* Chapter 6 consists of codes.
* Chapter 7 gives references.

**CHAPTER 2**

**QUARANTINE CHECKER AND MEDICAL DATA BASE USING RFID**

* This chapter contains the proposed design, Facial recognition, RFID, Sending Email, SQL, Software and Hardware.

**2.1 Proposed System**

The following block diagram (figure 2) shows the system flow graph of this project.

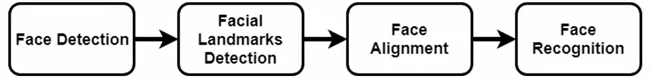


Figure 2

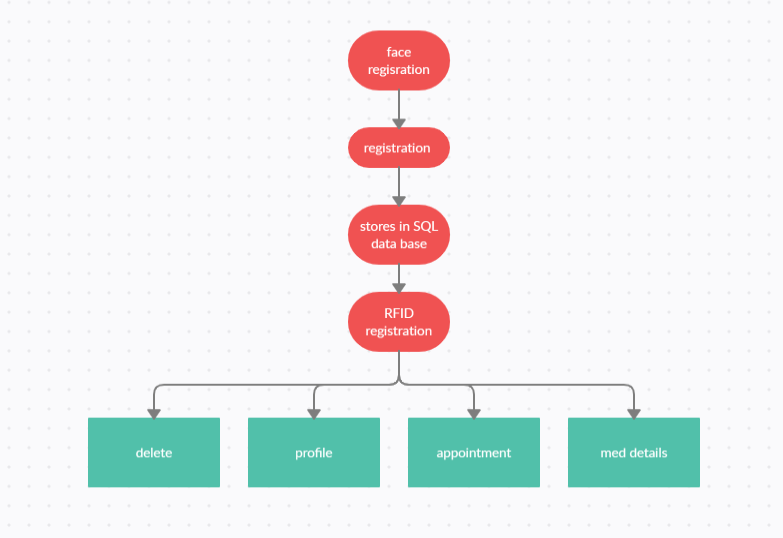


Figure 3

**2.2 Face recognition**

Face recognition is most crucial section in the device. If the target is ready the instructor

should turn the camera on, so device will constantly locate and detect the target face. If the task is

completed a text file will be generated with all the data like name, date, time.

**2.2.1 Image acquisition**

The input data is taken from the digital pi-cam which is placed in front of class. This will process the image and will be sent as input to device.

**2.2.2 Dataset Initiation**

Dataset of the target will be generated all the methods use in the device. It is generated just to make learn the device. It will generate dataset of 2 students with includes their Id number, branch and photos of each target in unique version. If you want higher accuracy at least 50 photos of each target has to be given for training. Every time when you check in target’s data and photos in device to generate dataset, machine learning applied to every input data to process 521-d face functions, keep in target facial records folder to remember the facial recognition technique.

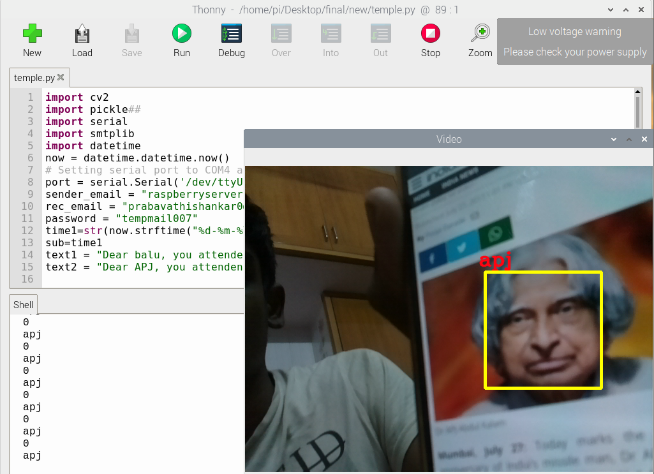


Figure 4

**2.2.3 Facial recognition and processing**

It is essential as the data input given via the pi-cam attached to device, facial recognition code technique applied to detect the target faces on the captures photo, the total no. of input data preparing code is established to detecting facial in an input data, although the location of the area where the face is detected. We have got use Head on Generation technique to capture human face and facial positions. To be precise 57 specific regions in a person face. The primary work of function is to find the landmark of face and to put the photo. Python IDE is utilized to finding athe facial marks, to place the input as a lot viable as of not disturbing the input. Facial Encode: as soon as the face is found in the given input data, very next process is to filter out the finding the faces for every picture. As when we get location of facial, these 64 key face factors is extract for specific photo give enter which will be mostly correct and those 64-d face location are saved in records for face reputation. Facial match: this is final process of facial reputation method.

**2.3 RFID**

RFID full form is “radio-frequency identification detector” and term to technology where by data that’s are present in the RFID tag are recorded using a sensor through radio waves. Radio-frequency identification detector is just similar to the barcodes where the all data are collected by using a toll that’s stores the recorded in database. RFID is made of technologies known as AIDC. AIDC techniques detects targets, collects the data about them, then saves the collected data inside the database directly into computer systems with very little human intervention. RFID techniques make use of radio signal to perform this. At an easy method, RFID structures include three parts: an RFID card, an RFID detector, and an antenna.

**2.4 MySQL**

My SQL is a Data base that is mostly used to manage and maintain a cloud database for storing data and using those. Those data can be stored in form of rows and columns so access it will be easier and more comfortable. The language which used MySQL is called as SQL which is build using C and C++. MySQL is first appeared in 1974 by ISO/IEC, which is developed by Donald D. Chamberlin Raymond and F. Boyce. Since its more unstable first its get relaunched in 2016 in the month of December.

**2.5 Sending E-mail**

As the SD card module produce the txt file of the persons who are present in class. SO the Email will be sent to the students with their present percentage so attendance and the verification so that they can know there status

**2.5 System Details**

This section describes the software and hardware details of the system:

**2.5.1 Software Details**

The simulations are run on a personal computer (PC) basically a laptop and its specifications are as follows:

* Processor – Intel core i5- CPU @ 2.5GHz
* Installed RAM – 16.00 GB
* Hardware – 1 GB

Operating System – Windows 10 (Home)

**1) Open CV**

The present project use Open CV library. It belongs to open source Library that is easy and powerful to work. It works in cpp, Java and Python network, works in Linux, Marcos, IOS, Windows and Android. It have powerful attention on real world usage. This class consists of more than 2600 optimized code, those code can be used to find and recognized face, detect object or items, and so on. Face Recognizer lib is use to detect face in Open CV. These find and changes face in python3.8 or python 2.7. This is very easy class constructed the use of dlib’s 49a2d564f1275e1c4e633abc331547db facial recognize constructed with machine learning (ML) state. This device has a correct state 97.87%. This offers an easy facial recognize coding line device that helps you to do face recognition on a file to state pix from the command. The dlib is platform open source software library this is applied on more than one computing systems. Open CV uses grayscale class to do face recognition easily so the system process fast and easily.

**2) Arduino IDE Software**

Arduino software (IDE) or The Arduino integrated development environment - consists of a cpp code writing platform. The code is written in cpp or c language and the software runs in Linux, macOS and windows. It gets paired to the Arduino and Genuine hardware to update executed code and transmit with the hardware. The Arduino integrated development environment helps the languages c and cpp using unique rule of code constructing. The Arduino integrated development environment elements a software library from the Wiring task, which provides many common input and output procedures. Source code for the IDE is released under the GNU well known Public License, version 2.

This is used to code RFID module, OLED display, SD card module, RTC module.

Library used are:

* RFID – MFRC522
* SD card – SD
* RTC –DS3231
* OLED – ug821

**3) VNC viewer and VNC server**

Client working on one kind of operating system can’t connect to the VNC server that operates on a unique kind of operating system. There are numerous distinctive types of clients and servers available for unique GUI-based totally systems. Further, a virtual network computing system is available for Java. several the VNC programs only work for the windows operating system. VNC became initially developed via an AT&T research group, however virtual network computing systems are extremely famous in handling commercial and personal uses.

**4) Anger IP scanner**

Angry IP scanner is an IP address finder. It is fast and accurate, It is used to scan an Ip address in a device. In the published project I have use Anger IP scanner for identifying the IP address of the raspberry pi, so that we can communicate using putty TV.

**5) SQL workbench**

SQL workbench is a open, easy and free interface that’s used to access the data in side the data server. The version that I have used is built on 127. Its mostly used to insert, delete and edit the data inside the server directly. It is more sophisticated such away that the all the tables in side it can be exported in a one line of commend.

**2.5.2 Hardware Details**

I have numerous hardware components getting used in this system. The details of every component is as follows

**1) Raspberry pi 4**

Raspberry Pi is cost efficient, small size computer that can be connect to a display or TV, and makes use of a normal keyboard and mouse. I can do anything we want, as like normal computer it can search in browser, we can play games, even we can code in raspberry pi default language used in that was python2. It far a successful small component that allows person of all age to discover networking and computing, and it’s also used in data collection and modification.



Figure 5

The device uses the ARM processor which does almost all the difficult work so that you can

run the Raspberry Pi. The primary purpose of Raspberry Pi is to acquire the data from numerous nodes i.e. Arduino and upload them to the firebase database.

**2) Arduino**

Arduino come into the market in 2005 his platform was introduced for giving a less cost and easy usage of professionals, experts, students and children. Arduino is a microcontroller as shown in figure which can be programed effortlessly, deleted and reprogrammed at any immediately at any number of times. Arduino can get all the inputs from sensors and collect those data for further uses.



Figure 6

**3) RC522 RFID module**

RC522 is a RTC module, it uses MFRC522 library in Arduino IDE. It is available in pair of RFID tags. Those tags contain a series of number that is unique. So when the card is shown in front of the module the radio wave produced by the module detects the series of numbers and send to Arduino. So then it will be used for further process.

****

Figure 7

**2.5.3 Software development**

Development of software plays a major role in this project because it consists of 80% of whole project. It’s separated in to 3 sections – user interface, registration, data allocation and location tracking.

**1)User Interface (UI)**

user interface is build using tkinter which is a library in python3. This consist of button, canvas, background, color and much more that are required for UI developing.



Figure 8

As you see in above picture the software is divided into 2 sections. First section is for quarantine and section is for medical data base.

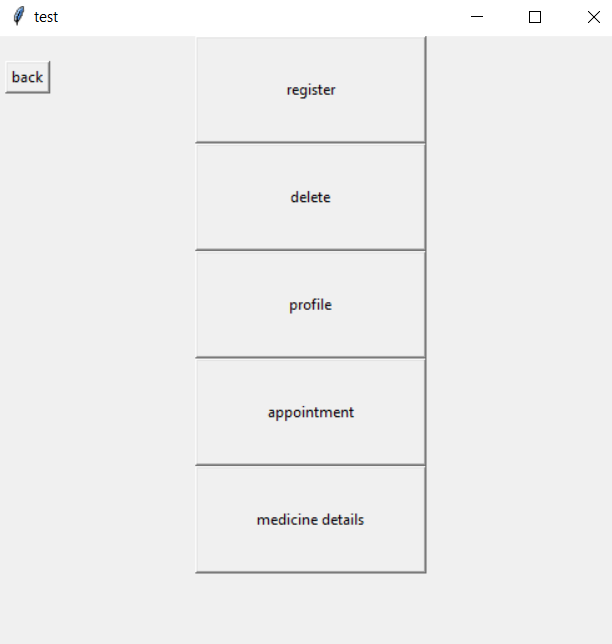


Figure 9

If you go further inside the data section its subdivided into 5 section such, first section is for registering, then for deleting the target data, then to see target profile, further for appointment status and finally medicine details.

**2) Registration**

This the one of the critical part of the project. Here the user will register their data and get allocated RFID tag to it. If it done successfully then the entered data will be store inside the MySQL data base.

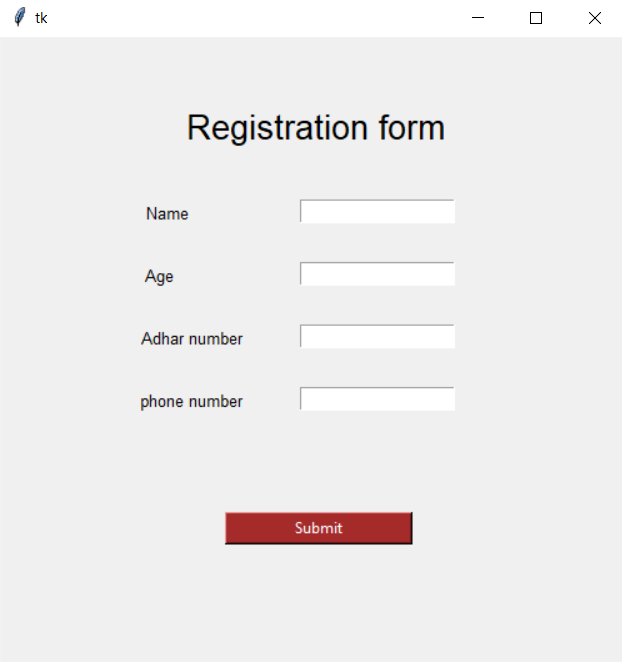


Figure 10

If the patient enters the name, age, Aadhar number and phone number then he or she need to click that red submit button to move further. A window with saying “place your tag” if the target places his allocated tag to it. It will be register under his profile.

**3) data allocation**

There all the data will be stored in rows and columns. So I have created 3 table with name drugs which will store all the drugs name, med will store all the medicine details, patient will sore all records of patients.

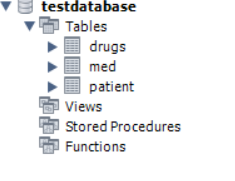


Figure 11

All those data will be subdivided and tagged with a personID so this will be used as reference to their table to get details of its target. For example, this profile table is linked with personID to a table in med and drugs.

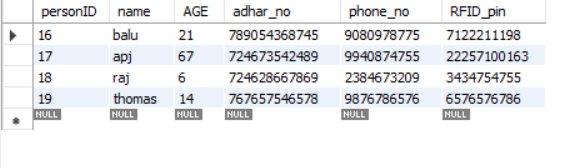


Figure 12

**4) Location tracking**

This is achieved using the python library known as geopy. This function returns the targets latitude and longitude. So, we can compare it with google map to get the exact location of the target.

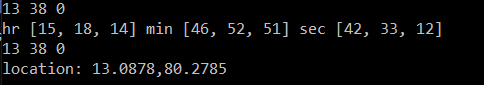


Figure 13

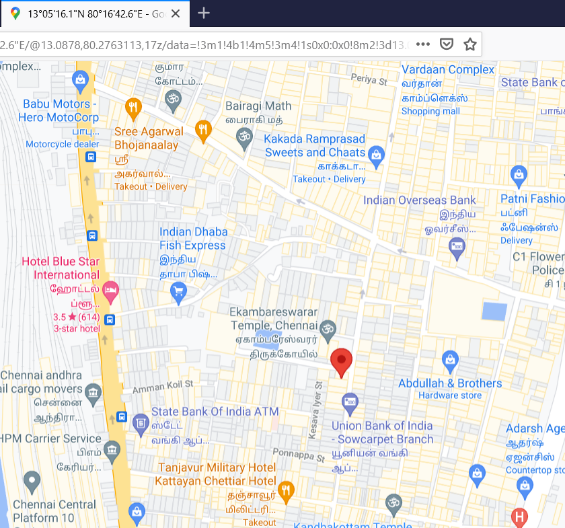


Figure 14

**CHAPTER 3**

**COST ANALYSIS**

**3.1 List of components and their cost**

The costs of the various modules used in this project are given below in List 3.1.

**List 3.1 List of modules and their costs**

|  |  |
| --- | --- |
| **MODULES** | **COST** |
| Raspberry Pi 4 | ₹ 3000 |
| Arduino | ₹ 500 |
| RFID Module | ₹ 150 |
| Miscellaneous | ₹ 200 |
| TOTAL | ₹ 3850 |

**CHAPTER 4**

**RESULTS AND DISCUSSIONS**

when the patient shows the face near the camera, the face detector module will detect the face and display his name near it. Then the blink detector will detect his blink to verify its not a photo or video.

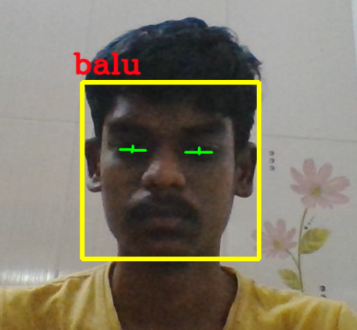
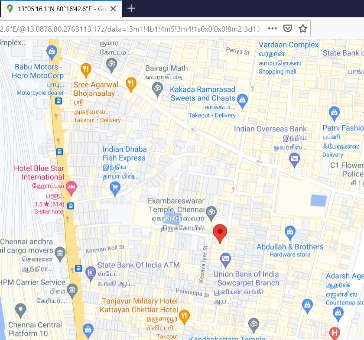
 

Figure 15

If the face match with the data sets, then the location tracking module starts to get the longitude and latitude of the target. Then its compare with the google maps to get exact location if its gets matched then email will be pop out with remain medicine

Figure 16

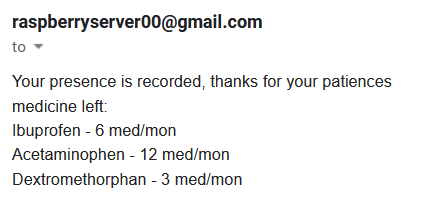
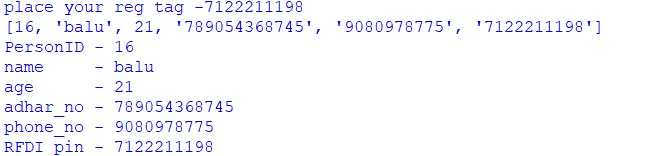
 

Figure 17

In medical database we can get all the details of the patients inside the server using the interface build by python by just clinking the buttons. have also added a remainder for patients so at particular time the medicine remainder will pop out remain about the left out medicine and count.

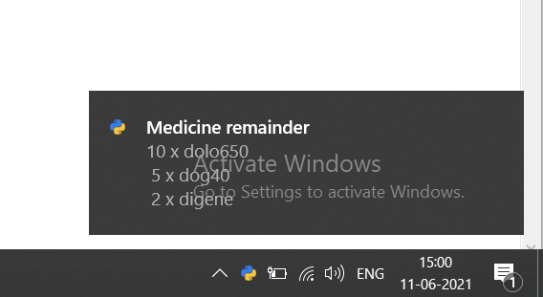


Figure 18

**CHAPTER 5**

**CONCLUSION AND FUTURE WORK**

We all know face the primary and most imported identity of every people, so using vintage techniques in using medical recodes is not secure and time consuming. Using the technique face recognition in our project tell the significance of evolving of technology. This device uses Open- Cv, python, raspberry-pi, Arduino as a primary platform. This device has an accuracy percent of 97.38%. The further version of this device needs to reduce the error and increase its features. The overall working of this device is when it detects the face it send to data base to get the its belonging RFID ID number. When the user places his tag in module, if the tag match with any tag in SQL it will bring out the desired output.

It uses facial detection algorithm from Open-cv2. I used Haar cascade, for facial detection. It has all the faces with different lighting conditions so we don’t need to send 1000s of face image to train our system. Email is send using SMT protocol in python.

**CHAPTER 6**

**APPENDIX**

**Arduino UNO code**

#include <SPI.h>

#include <MFRC522.h>

MFRC522 rfid(CS\_RFID, RST\_RFID);

String uidString;

String person;

void setup()

{

Serial.begin(9600);

Serial.begin(9600);

while(!Serial); // for Leonardo/Micro/Zero

// Init SPI bus

SPI.begin();

rfid.PCD\_Init();

if(!SD.begin(CS\_SD)) {

return;

}

if(!rtc.begin()) {

while(1);

}

else {

// following line sets the RTC to the date & time this sketch was compiled

rtc.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_)));

}

if(!rtc.isrunning()) {

}

u8g.firstPage();

do {

u8g.drawStr(1, 10, "face not detected");

} while (u8g.nextPage());

delay(50);

}

void loop()

{

if(rfid.PIC\_IsNewCardPresent()) {

readRFID();}

}

void readRFID() {

rfid.PICC\_ReadCardSerial();

uidString = String(rfid.uid.uidByte[0]) + " " + String(rfid.uid.uidByte[1]) + " " +

String(rfid.uid.uidByte[2]) + " " + String(rfid.uid.uidByte[3]);

delay(100);

}

**Face recognition and blink detection code**

**1) Data sets:**

impor cv2

video = cv2.VideoCaptur(0)

# load "haarcascade\_frontalface\_default.xml" by creating a CascadeClassifier

# object as cascade

cascad = cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")

while True:

check,frame = video.rea()

# Image from webacm is in the format of BGR i.e combination of 3 colours

# which will basicall require more amount of computation.

# so we convert it into a gray scale image which is only single colour

# and requires less computation.

gra = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRA)

# stream. Which will return x,y,w,h which are basically the positions

# with which we create a rectangle box.

face = cascade.detectMultiScal(gray, scaleFactor = 1.0, minNeighbor = 6)

# using for loop to go through the locations

for a,b,c,d in face:

fram = cv2.rectangle(frame, (a,b), (a+c,b+d), (0,255,255), 3)

cv2.imshow("Video",fram)

key = cv2.waitKey(1)

if(key == ord('k')):

break

video.release()

cv2.destroyAllWindows()

**2) Trainer:**

# import the required libraries

from PIL import Image

import cv2

import pickle

import numpy as np

import os

cascad = cv2.CascadeClassifie("haarcascade\_frontalface\_default.xml")

recognise = cv2.face.LBPHFaceRecognize\_creat()

# Created a function

def getdata():

current\_id = 0

label\_id = {} #dictionanary

face\_train = [] # list

face\_label = [] # list

# Finding the path of the base directory i.e path were this file is placed

BASE\_DIR = os.path.dirname(os.path.abspat(\_\_file\_\_))

# We have created "image\_data" folder that contains the data so basically

# we are appending its path to the base path

my\_face\_dir = os.path.join(BASE\_DIR,'image\_data')

# Finding all the folders and files inside the "image\_data" folder

for root, dirs, files in os.walk(my\_face\_dir):

for file in files:

# Checking if the file has extention ".png" or ".jpg"

if file.endswith("png") or file.endswith("jpg"):

# Adding the path of the file with the base path

# so you basically have the path of the image

path = os.path.join(root, file)

# Taking the name of the folder as label i.e his/her name

label = os.path.basename(root).lower()

# providing label ID as 1 or 2 and so on for different persons

if not label in label\_id:

label\_id[label] = current\_id

current\_id += 1

ID = label\_id[label]

# converting the image into gray scale image

# you can also use cv2 library for this action

pil\_imag = Image.open(path).conver("L")

# converting the image data into numpy array

image\_arra = np.array(pil\_imag, "uint8")

# identifying the faces

face = cascade.detectMultiScale(image\_arra)

# finding the Region of Interest and appending the data

for x,y,w,h in face:

img = image\_arra[y:y+h, x:x+w]

#image\_array = cv2.rectangle(im

cv2.imshow("Test",img)

cv2.waitKey(1)

face\_train.append(img)

face\_label.append(ID)

# string the labels data into a file

Wit open("labels.pickle", 'wb') as f:

pickle.dump(labe\_id, f)

return face\_train,face\_label

# creating ".yml" file

face,ids = getdata()

recognise.train(face, np.array(ids))

recognise.save("trainner.yml")

**3) Face recognizer:**

import cv2

import pickle

vide = cv2.VideoCaptur(0)

cascad = cv2.CascadeClassifie("haarcascade\_frontalface\_default.xml")

# Loaading the face recogniser and the trained data into the program

recognise = cv2.face.LBPHFaceRecognizer\_create()

recognise.read("trainner.yml")

labels = {} # dictionary

# Opening labels.pickle file and creating a dictionary containing the label ID

# and the name

with ope("labels.pickle", 'rb') as f:##

og\_label = pickle.load(f)##

labels = {v:k for k,v in og\_label.items()}##

print(labels)

while True:

check,frame = video.read()

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

face = cascade.detectMultiScale(gray, scaleFactor = 1.2, minNeighbors = 5)

#print(face)

for x,y,w,h in face:

face\_save = gra[y:y+h, x:x+w]

# Predicting the face identified

ID, conf = recognise.predict(face\_save)

#print(ID,conf)

if conf >= 20 and conf <= 115:

print(ID)

print(labels[ID])

cv2.putText(frame,labels[ID],(x-10,y-10),cv.FONT\_HERSHEY\_COMPLEX ,1, (18,5,255), 2, cv2.LINE\_AA )

frame = cv2.rectangle(frame, (x,y), (x+w,y+h),(0,255,255),4)

cv2.imshow("Video",frame)

key = cv2.waitKey(1)

if(key == ord('q')):

break

video.release()

cv2.destroyAllWindows()

**4) Final code:**

# import the required libraries

import cv2

import pickle

import numpy as np

import dlib

import time

import datetime

import smtplib

from math import hypot

import random

def main():

now = datetime.datetime.now()

day=now.day

day=30-day

s=day

a=day\*2

d=day//2

sender\_email = "raspberryserver00@gmail.com"

rec\_email = "prabavathishankar0@gmail.com"

password = "tempmail007"

time1=str(now.strftime("%d-%m-%Y %H:%M:%S"))

sub=time1

text1 = "Your presence is recorded, thanks for your patiences\r\n"+"medicine left:\r\n"+"Ibuprofen - %s med/mon\r\n"%s+"Acetaminophen - %a med/mon\r\n"%a+"Dextromethorphan - %d med/mon\r\n"%d

text2 = "your failed to be in quarantine"

video = cv2.VideoCapture(0)

cascade = cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml")

# Loaading the face recogniser and the trained data into the program

recognise = cv2.face.LBPHFaceRecognizer\_create()

recognise.read("trainner.yml")

labels = {} # dictionary

# Opening labels.pickle file and creating a dictionary containing the label ID

# and the name

with open("labels.pickle", 'rb') as f:##

og\_label = pickle.load(f)##

labels = {v:k for k,v in og\_label.items()}##

print(labels)

k=random.randint(2,3)

print(k)

detector = dlib.get\_frontal\_face\_detector()

predictor = dlib.shape\_predictor("shape\_predictor\_68\_face\_landmarks.dat")

def midpoint(p1 ,p2):

return int((p1.x + p2.x)/2), int((p1.y + p2.y)/2)

i=0

font = cv2.FONT\_HERSHEY\_PLAIN

def get\_blinking\_ratio(eye\_points, facial\_landmarks):

left\_point = (facial\_landmarks.part(eye\_points[0]).x, facial\_landmarks.part(eye\_points[0]).y)

right\_point = (facial\_landmarks.part(eye\_points[3]).x, facial\_landmarks.part(eye\_points[3]).y)

center\_top = midpoint(facial\_landmarks.part(eye\_points[1]), facial\_landmarks.part(eye\_points[2]))

center\_bottom = midpoint(facial\_landmarks.part(eye\_points[5]), facial\_landmarks.part(eye\_points[4]))

hor\_line = cv2.line(frame, left\_point, right\_point, (0, 255, 0), 2)

ver\_line = cv2.line(frame, center\_top, center\_bottom, (0, 255, 0), 2)

hor\_line\_lenght = hypot((left\_point[0] - right\_point[0]), (left\_point[1] - right\_point[1]))

ver\_line\_lenght = hypot((center\_top[0] - center\_bottom[0]), (center\_top[1] - center\_bottom[1]))

ratio = hor\_line\_lenght / ver\_line\_lenght

return ratio

while True:

check,frame = video.read()

gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

faces = detector(gray)

face = cascade.detectMultiScale(gray, scaleFactor = 1.2, minNeighbors = 5)

#print(face)

for x,y,w,h in face:

face\_save = gray[y:y+h, x:x+w]

# Predicting the face identified

ID, conf = recognise.predict(face\_save)

#print(ID,conf)

if conf >= 20 and conf <= 115:

cv2.putText(frame,labels[ID],(x-10,y-10),cv2.FONT\_HERSHEY\_COMPLEX ,1, (18,5,255), 2, cv2.LINE\_AA )

frame = cv2.rectangle(frame, (x,y), (x+w,y+h),(0,255,255),4)

for fac in faces:

#x, y = face.left(), face.top()

#x1, y1 = face.right(), face.bottom()

#cv2.rectangle(frame, (x, y), (x1, y1), (0, 255, 0), 2)

landmarks = predictor(gray, fac)

left\_eye\_ratio = get\_blinking\_ratio([36, 37, 38, 39, 40, 41], landmarks)

right\_eye\_ratio = get\_blinking\_ratio([42, 43, 44, 45, 46, 47], landmarks)

blinking\_ratio = (left\_eye\_ratio + right\_eye\_ratio) / 2

if blinking\_ratio > 5.7:

i=i+1

cv2.putText(frame, "", (50, 150), font, 7, (255, 0, 0))

print(i)

time.sleep(0.25)

cv2.imshow("Video",frame)

key = cv2.waitKey(1)

if key == 27 or i==k:

break

if ID==2:

print("----FACE MATCHED----")

server = smtplib.SMTP('smtp.gmail.com', 587)

server.starttls()

server.login(sender\_email, password)

print("Login success")

message1 = 'Subject: {}\n\n{}'.format(sub, text1)

server.sendmail(sender\_email, rec\_email, message1)

print("Email has been sent to ", rec\_email)

print(message1)

video.release()

cv2.destroyAllWindows()

if \_\_name\_\_ == "\_\_main\_\_":

main()

**location tracking**

import geocoder

import datetime

import time

import random

import cv2

import pickle

import os

import temple

#sys contains command line arguments

from sys import argv

from pyperclip import paste

def main():

current\_time = datetime.datetime.now()

hr=current\_time.hour

mi=current\_time.minute

sec=current\_time.second

print(hr,mi,sec)

hour=[]

minute=[]

second=[]

for i in range(3):

hr\_r=random.randint(hr,24)

mi\_r=random.randint(mi,60)

sec\_r=random.randint(sec,60)

hour.append(hr\_r)

minute.append(mi\_r)

second.append(sec\_r)

print("hr",hour,"min",minute,"sec",second)

y=0

from webbrowser import open

while True:

p=0

current\_time = datetime.datetime.now()

hr=current\_time.hour

mi=current\_time.minute

sec=current\_time.second

print(hr,mi,sec)

time.sleep(1)

if True or (hour[0]==hr and minute[0]==mi and second[0]==sec) or (hour[1]==hr and minute[1]==mi and second[1]==sec) or (hour[2]==hr and minute[2]==mi and second[2]==sec):

y=y+1

g = geocoder.ip('me')

address=g.latlng

k=""

for i in range(len(address)):

address[i]=str(address[i])

if i==len(address)-1:

k=k+address[i]

else:

k=k+address[i]+","

print("location:",k)

open("http://www.google.com/maps/place/"+k)

time.sleep(5)

if k=="13.0878,80.2785":

print("--LOCATION MATCHED--")

temple.main()

time.sleep(3)

else:

print("--LOCATION NOT MATCHED--")

if y==1:

break

if \_\_name\_\_=="\_\_main\_\_":

main()

**Accessing SQL**

import mysql.connector

import serial

def main():

arduino = serial.Serial('COM4',9600)

mydb = mysql.connector.connect(host='localhost',user='root', passwd='balaji',

database="testdatabase")

mycursor=mydb.cursor()

num=int(input())

datalist=["PersonID -","name -","age -","adhar\_no -","phone\_no -","RFDI\_pin -"]

arr=[]

drugs\_data=["dolo650","paracetamol","dog40","digene"]

med\_data=["PersonID -","name -","hospital -","doctor -","Appointment -"]

if num==1:

print("enter name")

str1=input()

print("enter age")

num1=int(input())

print("enter adhar number")

num2=int(input())

print("phone number")

num3=int(input())

print("register your tag")

for i in range(2):

data = arduino.readline()[0:-2]

data=str(data)[2:-1]

data=data.replace(" ","")

print(data)

mycursor.execute("INSERT INTO patient(name,age,adhar\_no,phone\_no,RFID\_pin) VALUES(%s,%s,%s,%s,%s)",(str1,num1,num2,num3,data))

mydb.commit()

print("updated successfully")

sql = ("SELECT \* FROM patient WHERE RFID\_pin='%s'"%data)

result = mycursor.execute(sql)

for i in mycursor:

i=list(i)

for j in range(len(i)):

print(datalist[j],i[j])

if num==2:

print("enter ID")

num1=int(input())

mycursor.execute("DELETE FROM patient WHERE personID=%d"%(num1))

mydb.commit()

print("deleted successfully")

if num==3:

print("place your reg tag -",end="")

for i in range(2):

data = arduino.readline()[0:-2]

data=str(data)[2:-1]

data=data.replace(" ","")

print(data)

sql = ("SELECT \* FROM patient\

WHERE RFID\_pin='%s'"%data)

data=data.replace(" ","")

result = mycursor.execute(sql)

for i in mycursor:

i=list(i)

print(i)

for j in range(len(i)):

print(datalist[j],i[j])

if num==4:

print("place your reg tag -",end="")

for i in range(2):

data = arduino.readline()[0:-2]

data=str(data)[2:-1]

data=data.replace(" ","")

print(data)

mycursor.execute("SELECT patient.PersonID,name,hospital,doctor,Appointment\

FROM patient JOIN med ON patient.PersonID=med.id \

WHERE RFID\_pin='%s'"%data)

myresult = mycursor.fetchall()

for i in myresult:

print(i)

i=list(i)

for j in range(len(i)):

print(med\_data[j],i[j])

if num==5:

print("place your reg tag -",end="")

for i in range(2):

data = arduino.readline()[0:-2]

data=str(data)[2:-1]

data=data.replace(" ","")

print(data)

mycursor.execute("SELECT patient.PersonID,name,dolo650,paracetamol,dog40,digene\

FROM patient JOIN drugs ON patient.PersonID=drugs.id \

WHERE RFID\_pin=%s"%data)

myresult = mycursor.fetchall()

for i in myresult:

i=list(i)

print("name -",i[1])

print("drugs:")

for j in range(2,len(i)):

if i[j]!=None:

print(i[j],"x",drugs\_data[j-2])

if \_\_name\_\_=="\_\_main\_\_":

main()

**button code**

1. **Button’s code**

from tkinter import \*

import date\_loc

import data

import data\_button

import profile\_button1

import mysql.connector

import serial

import profile\_button

import med\_button

mydb = mysql.connector.connect(host='localhost',user='root', passwd='balaji',

database="testdatabase")

mycursor=mydb.cursor()

datalist=["PersonID -","name -","age -","adhar\_no -","phone\_no -","RFDI\_pin -"]

arr=[]

drugs\_data=["dolo650","paracetamol","dog40","digene"]

med\_data=["PersonID -","name -","hospital -","doctor -","Appointment -"]

root=Tk()

root.geometry('500x500')

root.title("test")

def click():

print(entry\_1.get(),entry\_2.get(),entry\_3.get(),entry\_4.get())

def fun\_q():

button\_q.forget()

button\_data.forget()

date\_loc.main()

button3.place(x=5, y=20)

def fun\_data():

button\_q.place(x=1000, y=1000)

button\_data.place(x=1000, y=1000)

button3.forget()

button\_lvl1.pack()

button\_lvl2.pack()

button\_lvl3.pack()

button\_lvl4.pack()

button\_lvl5.pack()

button3.place(x=5, y=20)

def fun\_back():

button\_lvl1.forget()

button\_lvl2.forget()

button\_lvl3.forget()

button\_lvl4.forget()

button\_lvl5.forget()

button3.place(x=1000,y=1000)

button\_q.place(x=70,y=100)

button\_data.place(x=70,y=200)

def fun\_1():

data\_button.main()

def fun\_2():

print("enter ID")

num1=int(input())

mycursor.execute("DELETE FROM patient WHERE personID=%d"%(num1))

mydb.commit()

print("deleted successfully")

def fun\_3():

profile\_button.main()

def fun\_4():

profile\_button.main()

def fun\_5():

med\_button.main()

button\_q=Button(root,text="quarantine",command=fun\_q,width=50,height=5)

button\_data=Button(root,text="data",command=fun\_data,width=50,height=5)

button\_lvl1=Button(root,text="register",command=fun\_1,width=25,height=5)

button\_lvl2=Button(root,text="delete",command=fun\_2,width=25,height=5)

button\_lvl3=Button(root,text="profile",command=fun\_3,width=25,height=5)

button\_lvl4=Button(root,text="appointment",command=fun\_4,width=25,height=5)

button\_lvl5=Button(root,text="medicine details",command=fun\_5,width=25,height=5)

button3=Button(root,text="back",command=fun\_back)

button\_q.place(x=70,y=100)

button\_data.place(x=70,y=200)

root.mainloop()

1. **buttons for data**

from tkinter import\*

import serial

import mysql.connector

def main():

root = Tk()

arduino = serial.Serial('COM4',9600)

mydb = mysql.connector.connect(host='localhost',user='root', passwd='balaji',

database="testdatabase")

root.geometry('500x500')

mycursor=mydb.cursor()

datalist=["PersonID -","name -","age -","adhar\_no -","phone\_no -","RFDI\_pin -"]

def click():

global k,age,adh,phone

k,age,adh,phone=entry\_1.get(),entry\_2.get(),entry\_3.get(),entry\_4.get()

print(entry\_1.get(),entry\_2.get(),entry\_3.get(),entry\_4.get())

root.destroy()

label\_0 = Label(root, text="Registration form",width=20,font=("bold", 20))

label\_0.place(x=90,y=53)

label\_1 = Label(root, text="Name",width=14,font=("bold", 10))

label\_1.place(x=75,y=130)

entry\_1 = Entry(root)

entry\_1.place(x=240,y=130)

label\_2 = Label(root, text="Age",width=15,font=("bold", 10))

label\_2.place(x=65,y=180)

entry\_2 = Entry(root)

entry\_2.place(x=240,y=180)

label\_3 = Label(root, text="Adhar number",width=14,font=("bold", 10))

label\_3.place(x=95,y=230)

entry\_3 = Entry(root)

entry\_3.place(x=240,y=230)

label\_4 = Label(root, text="phone number",width=20,font=("bold", 10))

label\_4.place(x=70,y=280)

entry\_4 = Entry(root)

entry\_4.place(x=240,y=280)

Button(root, text='Submit',width=20,command=click,bg='brown',fg='white').place(x=180,y=380)

root.mainloop()

print("register your tag")

for i in range(2):

data = arduino.readline()[0:-2]

data=str(data)[2:-1]

data=data.replace(" ","")

print(data)

mycursor.execute("INSERT INTO patient(name,age,adhar\_no,phone\_no,RFID\_pin) VALUES(%s,%s,%s,%s,%s)",(k,age,adh,phone,data))

mydb.commit()

print("updated successfully")

sql = ("SELECT \* FROM patient WHERE RFID\_pin='%s'"%data)

result = mycursor.execute(sql)

for i in mycursor:

i=list(i)

for j in range(len(i)):

print(datalist[j],i[j])

if \_\_name\_\_=="\_\_main\_\_":

main()

1. **profile**

import serial

import mysql.connector

def main():

arduino = serial.Serial('COM4',9600)

mydb = mysql.connector.connect(host='localhost',user='root', passwd='balaji',

database="testdatabase")

mycursor=mydb.cursor()

med\_data=["PersonID -","name -","hospital -","doctor -","Appointment -"]

print("place your reg tag -",end="")

for i in range(2):

data = arduino.readline()[0:-2]

data=str(data)[2:-1]

data=data.replace(" ","")

print(data)

mycursor.execute("SELECT patient.PersonID,name,hospital,doctor,Appointment\

FROM patient JOIN med ON patient.PersonID=med.id \

WHERE RFID\_pin='%s'"%data)

myresult = mycursor.fetchall()

for i in myresult:

print(i)

i=list(i)

for j in range(len(i)):

print(med\_data[j],i[j])

if \_\_name\_\_=="\_\_main\_\_":

main()

1. **medicine button**

def main():

import serial

import mysql.connector

arduino = serial.Serial('COM4',9600)

mydb = mysql.connector.connect(host='localhost',user='root', passwd='balaji',

database="testdatabase")

mycursor=mydb.cursor()

drugs\_data=["dolo650","paracetamol","dog40","digene"]

print("place your reg tag -",end="")

for i in range(2):

data = arduino.readline()[0:-2]

data=str(data)[2:-1]

data=data.replace(" ","")

print(data)

mycursor.execute("SELECT patient.PersonID,name,dolo650,paracetamol,dog40,digene\

FROM patient JOIN drugs ON patient.PersonID=drugs.id \

WHERE RFID\_pin=%s"%data)

myresult = mycursor.fetchall()

for i in myresult:

i=list(i)

print("name -",i[1])

print("drugs:")

for j in range(2,len(i)):

if i[j]!=None:

print(i[j],"x",drugs\_data[j-2])

if \_\_name\_\_=="\_\_main\_\_":

main()

**REFERENCES**

[1] Nel Ramakat Borkr, Sona Kuwelka -“Real-time execution of a facial detection device” IEEE 2017.

[2] Harries Wechslersr - “easiest facial detection methods and prosses” 2005

[3] Masimo Banoizi – “how to use and advance audio ” 2007

[4] R. Wanter intel development USA – “start with RFID and its technology”, tell about working of RFID and technology used, IEEE2006

[5] Mark Lutzer – “get started with python”, this is one of the best published book of python and its application. Published in October 1998

[6] SittinnKongsontank and YuttapngRangsansriy, “Face detection in 3 dimentinal space”, IEEE. 2014

[7] oxerfel and defter, has published a book knows as” all you need to know about blink detection” which says about all type of blink detection and processing.

[8] Ivan Culja; David Abrm; Tomislav Pribnic; Hrvoje Dzpo; Mariof Cifek – “open CV2 the next generation of facial detection”, 2014

**BIODATA**



Name : S.Balaji

Mobile Number : 9080978773

E-mail : s.balaji@vitap.ac.in

Permanaent Address : No 20, F1, Ambattur flats, Redhills road,

Ambattur OT, Chennai - 600053