PROJECT REPORT

PERSONAL ASSISTANCE FOR SENIORS WHO ARE SELF-RELIANT

Submitted by,

Team ID : PNT2022TMID37590

A. Nandhini-(962819205020)
B. Pakkiya-(962819205023)
C. Surya-(962819205033)
D. Theresa Lino Coasta-(962819205034)

TABLE OF CONTENT

INTRODUCTION

• Project Overview

This is an Android-based application in which an automatic alarm ringing system is implemented. Elders need not remember their medicine dosage timings as they can set an alarm on their dosage timings. The alarm can be set for multiple medicines and timings including date, time and medicine description. A notification will be sent to them through email or message inside the system preferably chosen by the patients. They can search doctor disease wise. The patients will get the contact details of doctors as per their availability. Also the users can see different articles related to medical fields and health care tips. The system focuses on easy navigation and good user interface. Many such Medical Reminder Systems have been developed where a new hardware is required but in our work we have made an attempt to develop a system which is economical, time-saving and supports medication adherence.

• Purpose

- Customers would go to the site to create reminders and later receive them as simple text messages that work even on older mobile phone models.
- Call-based reminders will work perfectly for people with poor eyesight.
- Depending on how robust our medication reminder is, we might want to track medicine delivery from a pharmacy or find a route to the nearest pharmacy.
- The app would show interaction warnings in case it detects incompatible drugs.

LITERATURE SURVEY

Considered as elderly people suffer from an increasing number of problems, mainly due to social isolation and loneliness, requiring support from social agents. These problems, related to loneliness, social isolation, and reduced social activity are linked to the person's mental health, depression, and social bonds. Promoting the social engagement motivates persons to have more complex interactions, mobilizing the cognitive faculties and helping to maintain a good mental health.

Proposed a model for the design of an autonomous system, based on the paradigm of the intelligent personal assistant, in order to support the elderly people in maintain their social bonds with the family, friends and colleagues groups. This proposal is focused on tailoring the digital assistant for the specific group of elderlies and for their specific life contexts, which has good perspectives, as the intelligent personal assistants are equipment's that are becoming more interactive and with a more natural language.

When so many staff, services, sectors and agencies are involved it was felt that it was all too easy for gaps in care, fragmentation of care, lack of co-ordination between services, or duplication of services to occur. Studied the most important related with the family role and privacy control, to issues related with the design of the user interface, the importance of multimodal interaction and adaptive solutions to compensate age-related declines, to several other focusing on the importance of groups, photos, cultural and health information.

Developed the Google Assistant is an Intelligent Personal Assistant that allows communication with the user through voice commands. It is capable of search online, set reminders and play music using Spotify.

2.1 Existing Problem

- Does not encourage cancellation or rescheduling in patients who cannot attend or who no longer wish to attend.
- People may not be willing to disclose their mobile phone numbers and record them in patient notes.
-

References

1. Fernandes A (2001), “Velhice, solidariedades familiares e política social: itinerário de pesquisa em torno do aumento da esperança de vida. Sociologia, Problemas e Práticas “[online], n.36, pp.39-52.

2. Reis A, Reis C, Morgado L, Borges J, Tavares F, Gonçalves R, Cruz J (2016) , “Management of surgery waiting lists in the Portuguese public healthcare network: The information system for waiting list recovery programs.”In Information Systems and Technologies(CISTI), 2016 11th Iberian Conference on (pp. 1-7).

3. Palmer D, Newsom J, Rook K (2016),”How does difficulty communicating affect the social relationships of older adults? An exploration using data from a national survey.” Journal of Communication Disorders, 62:131-146.

4. Rook K, Lains J, Paredes H, Filipe V, Abrantes C, Ferreira F, Barroso, J. (2016),”Developing a System for Post-Stroke Rehabilitation: An Exergames Approach. In International Conference on Universal Access in Human-Computer Interaction “Springer International Publishing (pp. 403-413).

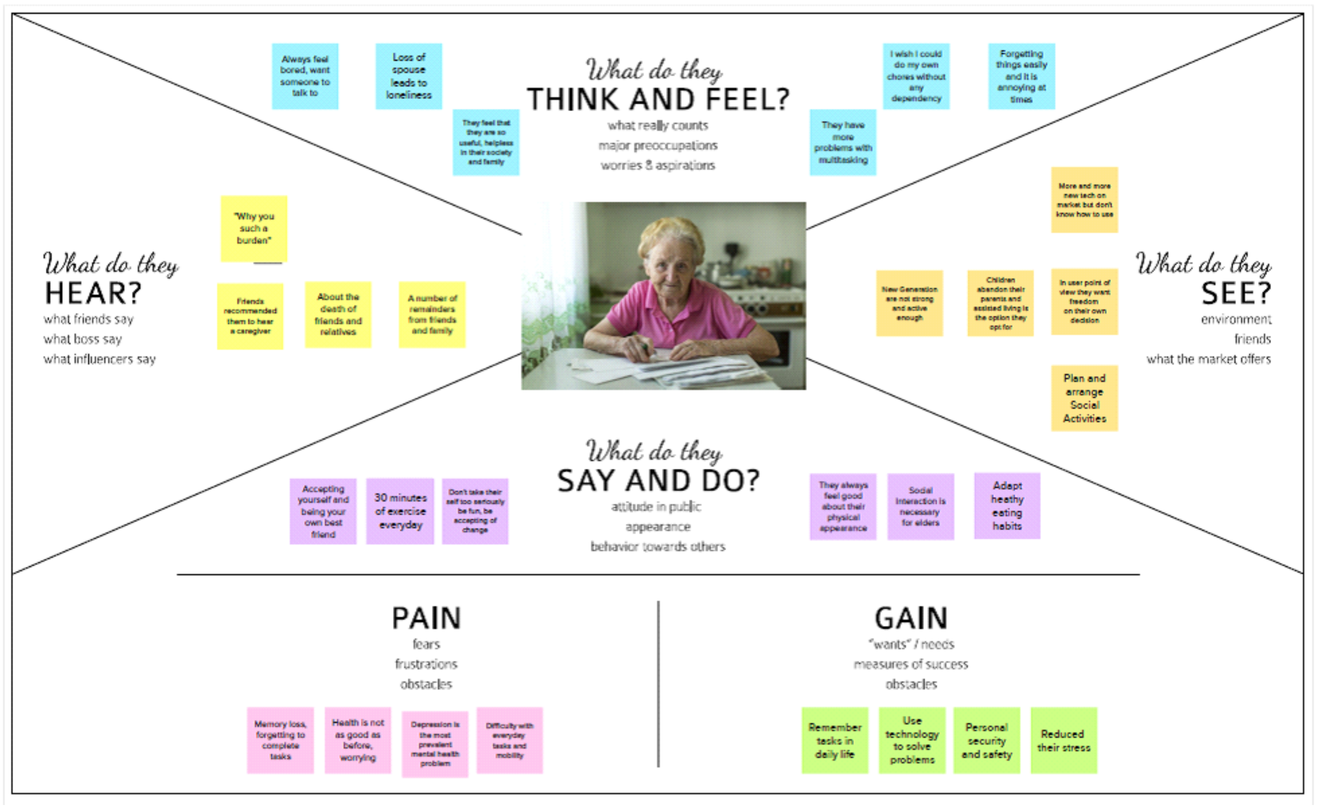
5. Stephen kopp & Karola pitsh,”Social disengagement and incident cognitive decline in community-dwelling older persons.”(pp.173-176)

Problem Statement Definition

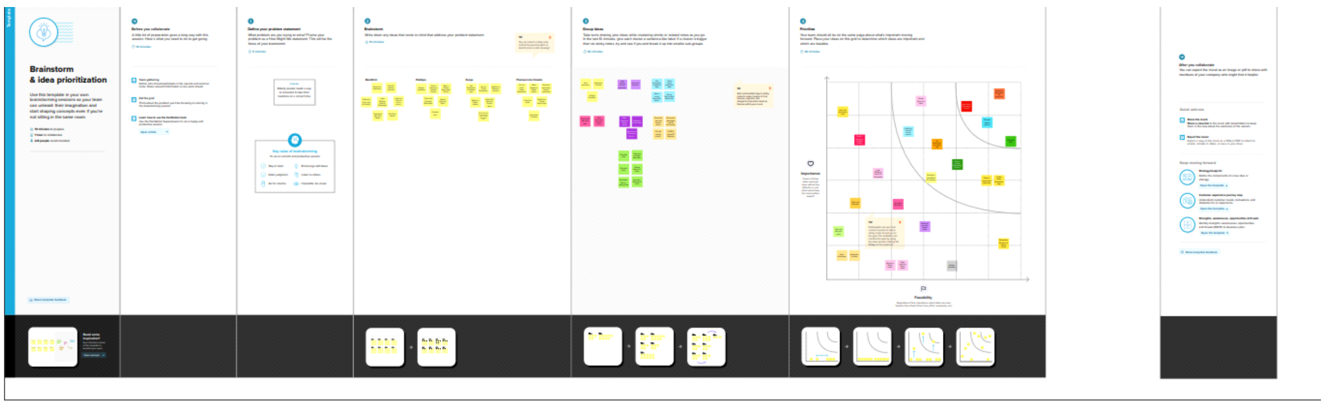
- Sometimes elderly people forget to take their medicine at the correct time.
- They also forget which medicine He / She should take at that particular time.
- And it is difficult for doctors/caretakers to monitor the patients around the clock.
- To avoid this problem, this medicine reminder system is developed.

IDEATION & PROPOSED SOLUTION

- **Empathy Map Canvas**


Always feel bored, want someone to talk to
Loss of spouse leads to loneliness
They feel that they are so useful, helpless in their society and family
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
They have more problems with multitasking
I wish I could do my own chores without any dependency
Forgetting things easily and it is annoying at times
"Why you such a burden"
What do they HEAR?
what friends say
what boss say
what influencers say
Friends recommended them to hear a caregiver
About the death of friends and relatives
A number of remainders from friends and family
More and more new tech on market but don't know how to use
New Generation are not strong and active enough
Children abandon their parents and assisted living is the option they opt for
In user point of view they want freedom on their own decision
What do they SEE?
environment
friends
what the market offers
Plan and arrange Social Activities
Accepting yourself and being your own best friend
30 minutes of exercise everyday
Don't take their self too seriously be fun, be accepting of change
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
They always feel good about their physical appearance
Social Interaction is necessary for elders
Adapt heathy eating habits
PAIN
fears
frustrations
obstacles
Memory loss, forgetting to complete tasks
Health is not as good as before, worrying
Depression is the most prevalent mental health problem
Difficulty with everyday tasks and mobility
GAIN
"wants" / needs
measures of success
obstacles
Remember tasks in daily life
Use technology to solve problems
Personal security and safety
Reduced their stress

- **Ideation & Brainstorming**



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

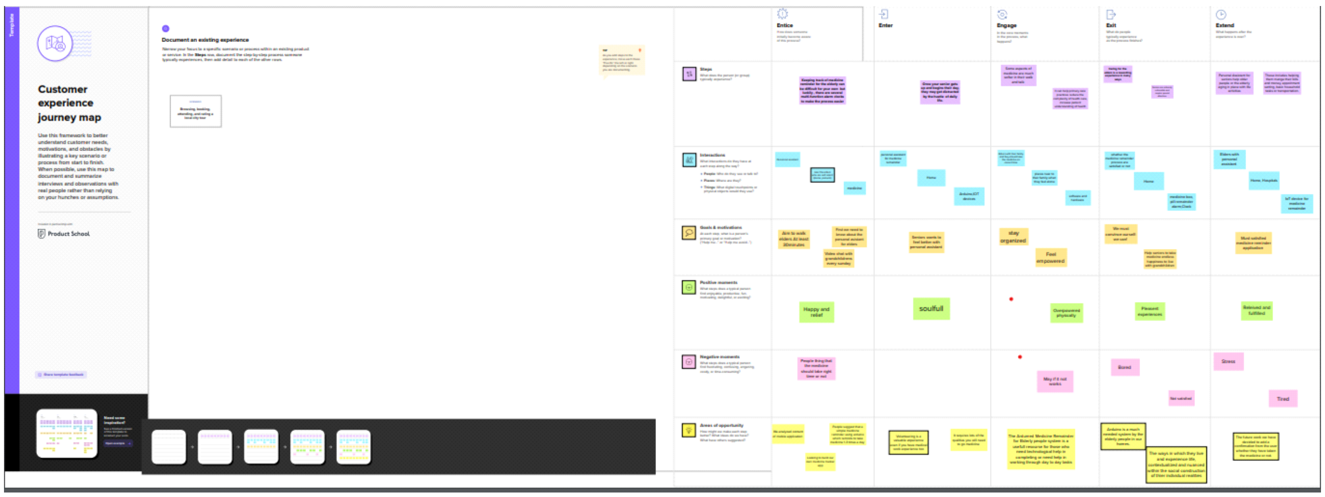
Key rules of brainstorming

Brainstorm

Group ideas

Prioritize

After you collaborate



Customer experience journey map

Use this framework to better understand customer needs, motivations, and obstacles by illustrating a key scenario or process from start to finish. When possible, use this map to document and summarize interviews and observations with real people rather than relying on your hunches or assumptions.

Product School

Document an existing experience

Narrow your focus to a specific scenario or process within an existing product or service. In the **Steps** row, document the step-by-step process someone typically experiences, then add detail to each of the other rows.

Browsing, booking, attending, and rating a local city tour

	Entice	Enter	Engage	Exit	Extend
Steps	Keeping track of medicine reminder for the elderly can be difficult for your own but luckily, there are several multi-function alarm clocks to make the process easier	Once your senior gets up and begins their day, they may get distracted by the hustle of daily life.	Some aspects of medicine are much softer in their walk animals; [illegible]	[illegible]	[illegible]
Interactions	[illegible]; medicine	personal assistant for medicine remainder; Home; Arduino,IOT devices	[illegible]; software and hardware	whether the medicine remainder process are satisfied or not; Home; medicine box, pill remainder alarm Clock	Elders with personal assistant; Home, Hospitals; IoT device for medicine remainder
Goals & motivations	Aim to walk elders At least 30minutes; Find we need to know about the personal assistant for elders; Video chat with grandchildrens every sunday	Seniors wants to feel better with personal assistant	stay organized; Feel empowered	We must convince ourself we can!; [illegible]	Must satisfied medicine reminder application
Positive moments	Happy and relief	soulfull	Overpowered physically	Pleasent experiences	Relieved and fulfilled
Negative moments	People thing that the mendicine should take right time or not		May if it not works	Bored; Not satisfied	Stress; Tired
Areas of opportunity	[illegible]	Volunteering is a valuable experience [illegible]; [illegible]	The Arduino Medicine Remainder for Elderly people system is a usefull resource for those who need technological help in completing or need help in working through day to day tasks	Arduino is a much needed system by the elderly people in our homes.; The ways in which they live and experience life, contextualized and nuanced within the social construction of their individual realities	The future work we have decided to add a confirmation from the user whether they have taken the medicine or not.

Proposed Solution

SL. No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	Sometimes elderly people forget to take their medicine at correct time. They also forget which he/she should take at that particular time. And it is difficult for doctor's/care takers to monitor the patients around the clock. To avoid this problem, the medicine remainder is developed.
2.	Idea / Solution description	To remind the users to take medicines on time based on Android Operating system, when notification and automatic alarm ringing system.
3.	Novelty / Uniqueness	The solution is about to remined the medicines. User can add details of his dosage schedules. Using the date field one can enter the starting and ending dates between which he has to take medicines.
4.	Social Impact / CustomerSatisfaction	The application gives reliable reminders, good user interface, nice user experience and it supports many new features supporting medication adherence.

5.	Business Model (Revenue Model)	Many Medication Reminder Systems have been developed on different platforms. Many of these systems require special hardware devices to remind the patients about the medicine in-take timings. Purchasing new hardware devices becomes costly and more time and money consuming. So in the given work an attempt has been made to implement a system which is economical, easily accessible and improves medication adherence.
6.	Scalability of the Solution	User can select them in either repeating or non-repeating alarm patterns. Any hourly time interval between alarms can be selected, starting from the minimum of 1 hour. At the scheduled time, application will produce a notification with an alarm, vibration or LED indication

Problem Solution fit

Develop a portable device to alert patients to take their medicine	Daily task remainder	It helps to take proper medicine at right time
High quality of care	Complex medicine schedule	Old age people
Audio signal will be necessary	Check pharmacy stores near by their current locations	Automated medicine remainder

4.REQUIREMENT ANALYSIS

- **Functional requirement**

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	Set Alarm	It helps in reminding about the medicines.All the information will be saved in the database.This makes any time availability of patients records.
FR-2	Get Notifiction	Once the alarm is set then the user gets the notification. User can Activate and Deactivate directly. .
FR-3	Sensor	This is used for keeping the record in medicine details the reminding the schedule of medicine.
FR-4	GPS Tracker	Medical equipment GPS tracking can also help large hospitals and clinics manage their inventory more effectively.
FR-5	Add Medication	We have to add medication with the medicine name and description.

- **Non Functional requirement**

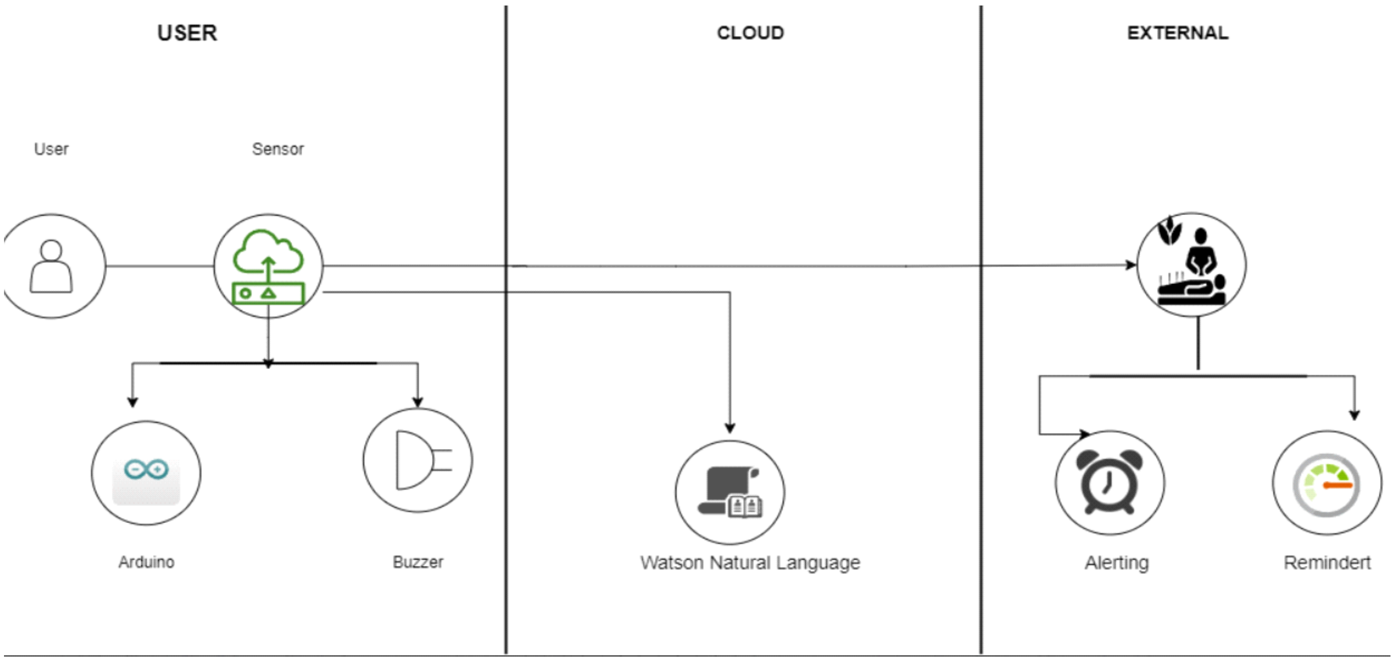
FR No.	**Non-Functional Requirement**	**Description**
NFR-1	Usability	If the medicine arrives the web application will send the medicine name to that device.
NFR-2	Security	To reduce the risk of serious problems, one may need to apply extra care in monitoring and extra care in checking for interactions when a new drug is prescribed.

NFR-3	Performance	These apps offer simple and user-friendly functionality enabling quick scheduling.
NFR-4	Availability	To remind the users to take medicines on time based on Android Operating system, when notification and automatic alarm ringing system.
NFR-5	Scalability	User can select them in either repeating or non-repeating alarm patterns.

5.PROJECT DESIGN

• Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows
within a system. A neat and clear DFD can depict the right amount of the system requirementgraphically. It shows how data enters and leaves the system, what changes the information,
and where data is stored.

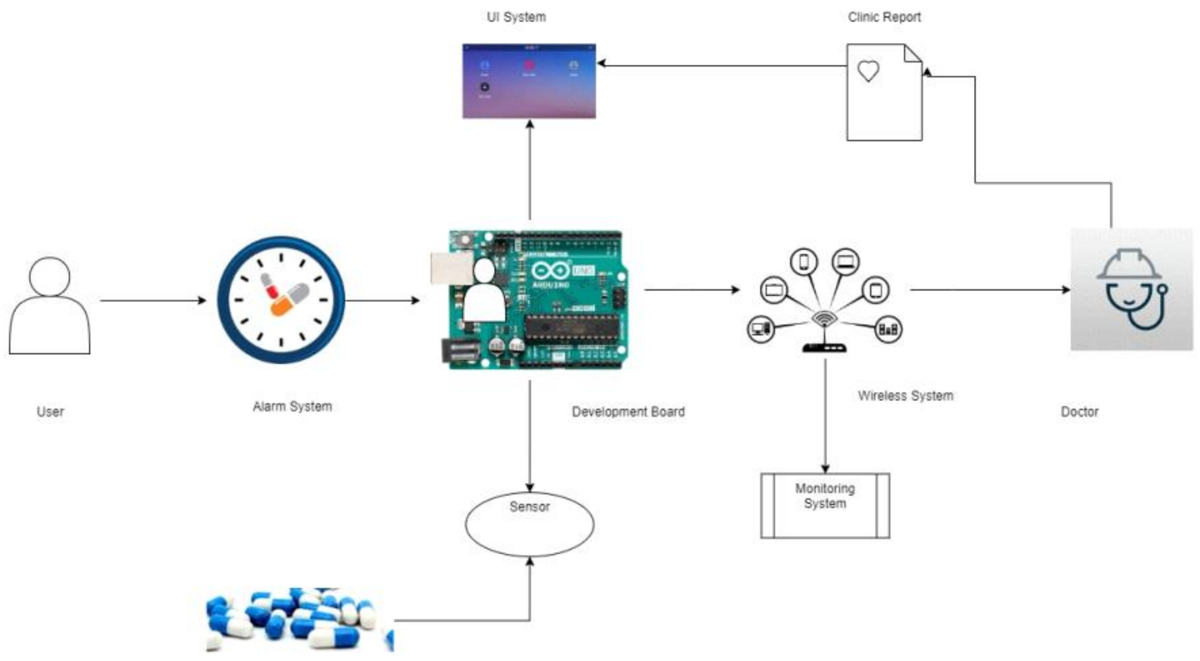


• Solution & Technical Architecture

Solution Architecture:

- User interacts with the UI (User Interface) to upload the image as input.
-

Technology Architecture Diagram:



5.3 User Stories

User Type	Functional Requirement (epic)	User Story No	User Story / Task	Acceptance criteria	Priority	Release
Customer (App user)	Register	USN-1	As a user, I can register for the application by entering my email, password, and confirming my	I can access my account / dashboard	High	Sprint-1

			password.			
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Alarming	USN-3	As a user, I can set an alarm to alerting a medicine through Medicine Remainder System	I can set an alarm by the remainder.	High	Sprint-2
		USN-4	As a user, I can Activate and Deactivate theAlarm	I can Access the alarm	Moderate	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can enter the details by desired information	High	Sprint-1
Customer (Web	Notification	USN-6	As a user once I can the set the	I can set an alarm to get	High	Sprint-2

user)			alarm then I gets the notification.	the notification		
		USN-7	As a user, If I requires this system then a notification will be sent into his device.	I can able to see a notification details	Moderate	Sprint-2
	Medication Detail	USN-8	As a user, I have multiple medications each day, I can put each pill in the box for the corresponding day.	I can access the pill box atany time for multiple medications taken.	Moderate	Sprint-3
		USN-9	As a user, Between setting an alarm and using a pillbox, I’ll be able to stay on top of your medications and not miss a dose.	I can’t miss a medication on setting alarm using pillbox	Low	Sprint-3

Customer (Patients)	Healthcare	USN-10	As a user, I can monitoring and update new medicine data of patients, which can be done by prescriber through web.	I can monitor and Update medicine through web.	Low	Sprint-4
		USN-11	As a user, I can store the name of the patient With its description	I can add the medicine with its description	High	Sprint-3

6.PROJECT PLANNING & SCHEDULING

- **Sprint Planning & Estimation**

User Story Number	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
USN-1	As a user, I can set an alarm to alerting a medicine through medicine remainder system	2	High	Nandhini , Pakkiya, Surya , Theresa Lino Coasta
USN-2	As a user once I can the set the alarm then I gets the notification.	1	High	Nandhini , Pakkiya, Surya , Theresa Lino Coasta
USN-3	As a user, between setting an alarm and using a pillbox, I'll be able to stay on top of your medications and not miss a dose.	2	Medium	Nandhini , Pakkiya, Surya , Theresa Lino Coasta

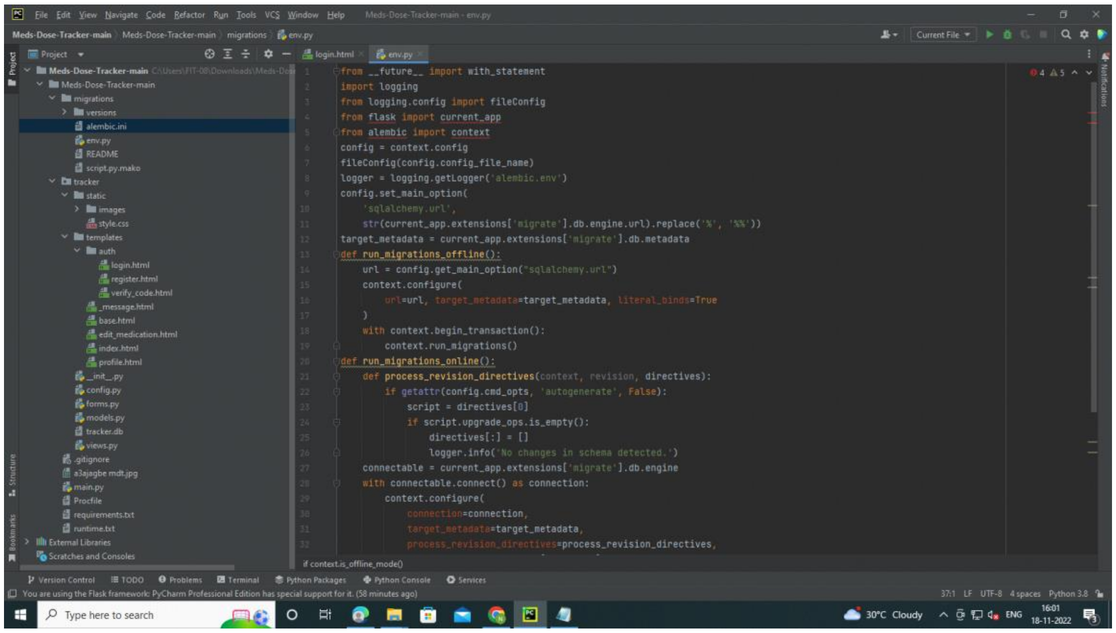
USN-4	As a user ,they used for keeping the record in medicine details the reminding the schedule of medicine. We have used the IoT enabled Arduino device for monitoring the System.	2	High	Nandhini , Pakkiya, Surya , Theresa Lino Coasta

- **Sprint Delivery Schedule**

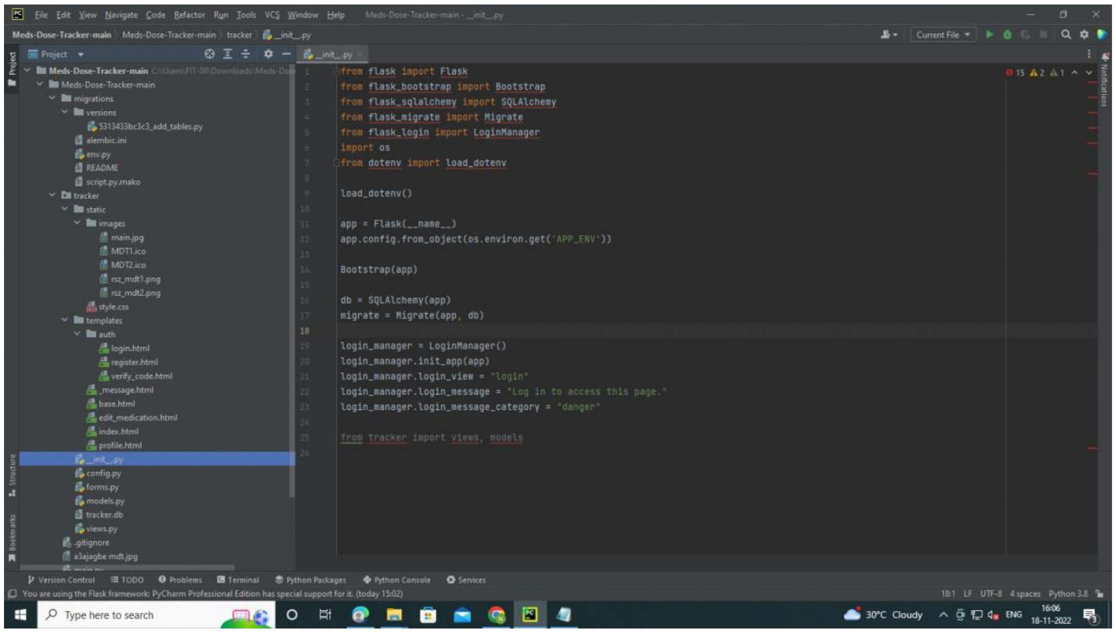
Sprint	**Sprint Topic**	**Start Date**	**Expected Delivery**
Sprint 1	Set alarm	28-10-2022	4-11-2022
Sprint 2	Notification	6-11-2022	13-11-2022
Sprint 3	Medication details	16-11-2022	23-11-2022
Sprint 4	GPS Tracking	23-11-2022	30-11-2022

7.CODING & SOLUTIONING

7.1 Feature



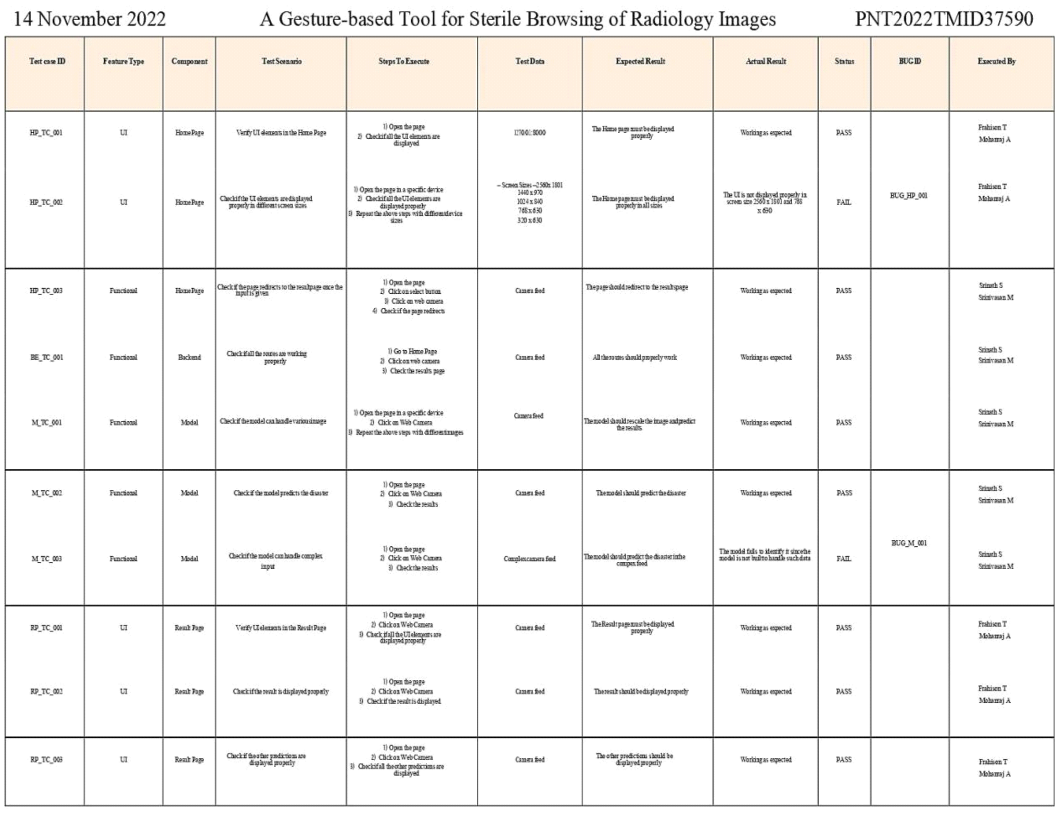
```python
from __future__ import with_statement
import logging
from logging.config import fileConfig
from flask import current_app
from alembic import context
config = context.config
fileConfig(config.config_file_name)
logger = logging.getLogger('alembic.env')
config.set_main_option(
    'sqlalchemy.url',
    str(current_app.extensions['migrate'].db.engine.url).replace('%', '%%'))
target_metadata = current_app.extensions['migrate'].db.metadata
def run_migrations_offline():
    url = config.get_main_option("sqlalchemy.url")
    context.configure(
        url=url, target_metadata=target_metadata, literal_binds=True
    )
    with context.begin_transaction():
        context.run_migrations()
def run_migrations_online():
    def process_revision_directives(context, revision, directives):
        if getattr(config.cmd_opts, 'autogenerate', False):
            script = directives[0]
            if script.upgrade_ops.is_empty():
                directives[:] = []
                logger.info('No changes in schema detected.')
    connectable = current_app.extensions['migrate'].db.engine
    with connectable.connect() as connection:
        context.configure(
            connection=connection,
            target_metadata=target_metadata,
            process_revision_directives=process_revision_directives,
```



```python
from flask import Flask
from flask_bootstrap import Bootstrap
from flask_sqlalchemy import SQLAlchemy
from flask_migrate import Migrate
from flask_login import LoginManager
import os
from dotenv import load_dotenv

load_dotenv()

app = Flask(__name__)
app.config.from_object(os.environ.get('APP_ENV'))

Bootstrap(app)

db = SQLAlchemy(app)
migrate = Migrate(app, db)

login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = "login"
login_manager.login_message = "Log in to access this page."
login_manager.login_message_category = "danger"

from tracker import views, models
```

TESTING

- ## Test Cases



14 November 2022 A Gesture-based Tool for Sterile Browsing of Radiology Images PNT2022TMID37590

Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	BUG ID	Executed By
HP_TC_001	UI	Home Page	Verify UI elements in the Home Page	1) Open the page 2) Check if all the UI elements are displayed	127.0.0.1:8000	The Home page must be displayed properly	Working as expected	PASS		Frahison T Moharraj A
HP_TC_002	UI	Home Page	Check if the UI elements are displayed properly in different screen sizes	1) Open the page in a specific device 2) Check if all the UI elements are displayed properly 3) Repeat the above steps with different device sizes	– Screen Sizes – 2560 x 1801 1440 x 970 1024 x 840 768 x 630 320 x 630	The Home page must be displayed properly in all sizes	The UI is not displayed properly in screen size 2560 x 1801 and 768 x 690	FAIL	BUG_HP_001	Frahison T Moharraj A
HP_TC_003	Functional	Home Page	Check if the page redirects to the result page once the input is given	1) Open the page 2) Click on select button 3) Click on web camera 4) Check if the page redirects	Camera feed	The page should redirect to the result page	Working as expected	PASS		Srinath S Srinivasan M
BE_TC_001	Functional	Backend	Check if all the routes are working properly	1) Go to Home Page 2) Click on web camera 3) Check the results page	Camera feed	All the routes should properly work	Working as expected	PASS		Srinath S Srinivasan M
M_TC_001	Functional	Model	Check if the model can handle various image	1) Open the page in a specific device 2) Click on Web Camera 3) Repeat the above steps with different images	Camera feed	The model should rescale the image and predict the results	Working as expected	PASS		Srinath S Srinivasan M
M_TC_002	Functional	Model	Check if the model predicts the disaster	1) Open the page 2) Click on Web Camera 3) Check the results	Camera feed	The model should predict the disaster	Working as expected	PASS		Srinath S Srinivasan M
M_TC_003	Functional	Model	Check if the model can handle complex input	1) Open the page 2) Click on Web Camera 3) Check the results	Complex camera feed	The model should predict the disaster in the compex feed	The model fails to identify it since the model is not built to handle such data	FAIL	BUG_M_001	Srinath S Srinivasan M
RP_TC_001	UI	Result Page	Verify UI elements in the Result Page	1) Open the page 2) Click on Web Camera 3) Check if all the UI elements are displayed properly	Camera feed	The Result page must be displayed properly	Working as expected	PASS		Frahison T Moharraj A
RP_TC_002	UI	Result Page	Check if the result is displayed properly	1) Open the page 2) Click on Web Camera 3) Check if the result is displayed	Camera feed	The result should be displayed properly	Working as expected	PASS		Frahison T Moharraj A
RP_TC_003	UI	Result Page	Check if the other predictions are displayed properly	1) Open the page 2) Click on Web Camera 3) Check if all the other predictions are displayed	Camera feed	The other predictions should be displayed properly	Working as expected	PASS		Frahison T Moharraj A

- ## User Acceptance Testing

PURPOSE OF THE DOCUMENT

The purpose of this document is to briefly explain the test coverage and open issues of the personal assistance for seniors who are self reliant project at the time of the release to User Acceptance Testing (UAT).

DEFECT ANALYSIS

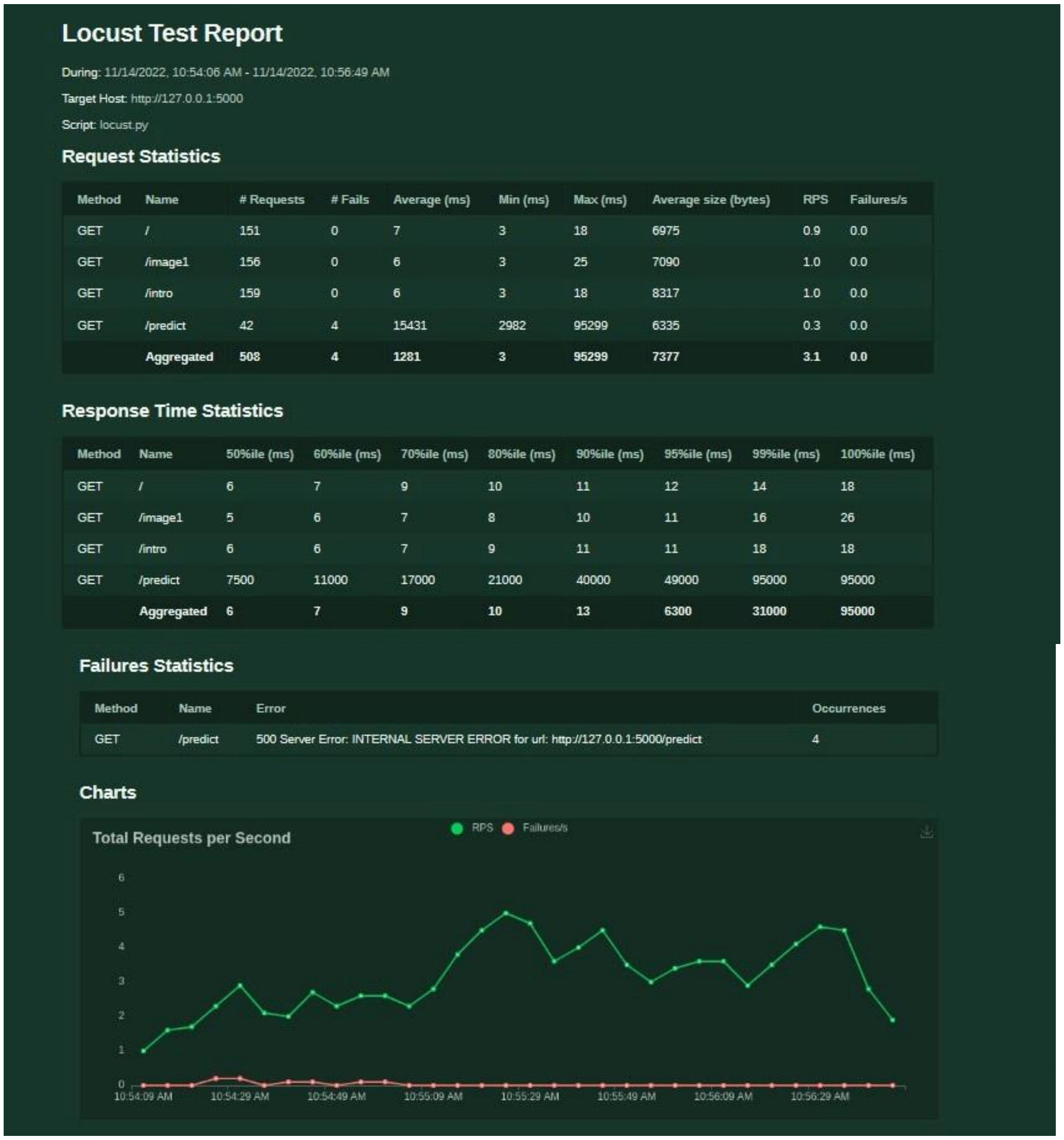
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Total
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won't Fix	1	0	1	0	2
Total	6	1	4	3	14

TEST CASE ANALYSIS

Section	Total Cases	Not Tested	Fail	Pass
Client Application	10	0	3	7
Security	2	0	1	1
Performance	3	0	1	2
Exception Reporting	2	0	0	2

RESULTS

- **Performance Metrics**



Locust Test Report

During: 11/14/2022, 10:54:06 AM - 11/14/2022, 10:56:49 AM

Target Host: http://127.0.0.1:5000

Script: locust.py

Request Statistics

Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	/	151	0	7	3	18	6975	0.9	0.0
GET	/image1	156	0	6	3	25	7090	1.0	0.0
GET	/intro	159	0	6	3	18	8317	1.0	0.0
GET	/predict	42	4	15431	2982	95299	6335	0.3	0.0
	Aggregated	**508**	**4**	**1281**	**3**	**95299**	**7377**	**3.1**	**0.0**

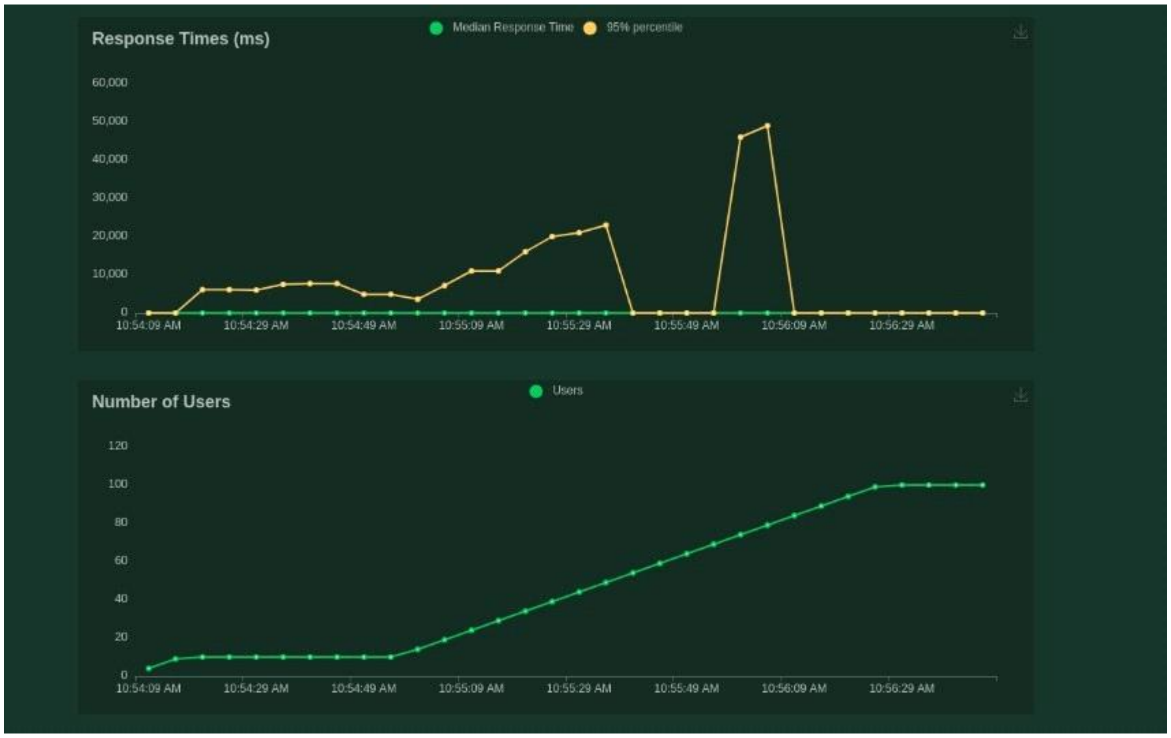
Response Time Statistics

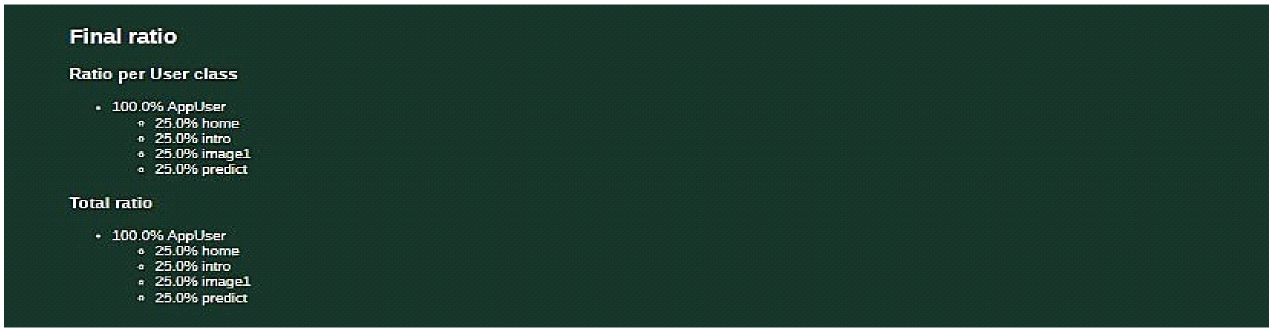
Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	/	6	7	9	10	11	12	14	18
GET	/image1	5	6	7	8	10	11	16	26
GET	/intro	6	6	7	9	11	11	18	18
GET	/predict	7500	11000	17000	21000	40000	49000	95000	95000
	Aggregated	**6**	**7**	**9**	**10**	**13**	**6300**	**31000**	**95000**

Failures Statistics

Method	Name	Error	Occurrences
GET	/predict	500 Server Error: INTERNAL SERVER ERROR for url: http://127.0.0.1:5000/predict	4

Charts





Final ratio

Ratio per User class

- 100.0% AppUser
  - 25.0% home
  - 25.0% intro
  - 25.0% image1
  - 25.0% predict

Total ratio

- 100.0% AppUser
  - 25.0% home
  - 25.0% intro
  - 25.0% image1
  - 25.0% predict

ADVANTAGES & DISADVANTAGES

Advantages:

- Major advantage of this tool is that it helps to maintain the sterility of theenvironment.
- It is also easy to use and is quicker than the existing methods to browseimages.
- It can also be performed even if the surgeon is a bit far away from thesystem, this helps to save time.
- The tool does not need the person using it to have an apparatus or anydevices on them to use it.
- They can simply move their hands to browse through the images.

Disadvantages:

- The tool can be quite expensive as it requires cameras and other expensivedevices to capture images and process it.
- Such systems are difficilt to develop because of the complexity and the costof implementation.
- As each gesture is assigned a specific control command , this system is not platform independent since certain control commands vary as the operating system varies.

CONCLUSION

In this project we developed a tool which recognises hand gestures and enables doctors to browse through radiology images using these gestures. This enables doctors and surgeons to maintain the sterility as they would not have to touch any mouse or keyboard to go through the images. This tool is also easy to use and is quicker than the regular method of using mouse/keyboard. It can be used regardless of the users location since they don’t have to be in contact with any device. It also does not require the user to have any device on them to use it. Further this technology can be extended to other industries like it can be used by presenters, by teachers for show images in the classroom, etc.

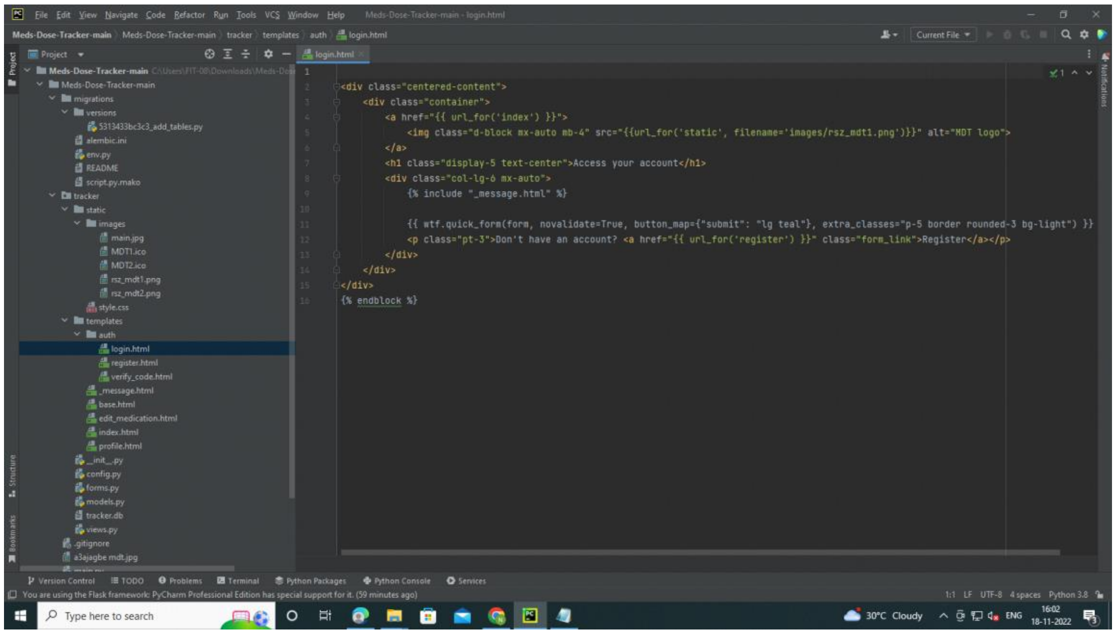
FUTURE SCOPE

- The tool can be made quicker by increasing the recognition speed.
- More number of gestures can be added thereby increasing this tool’sfunctionality and useability for different purposes.
- Tracking of both hands can be added to increase the set of commands.
- Voice commands can also be added to further increase the functionality.

APPENDIX

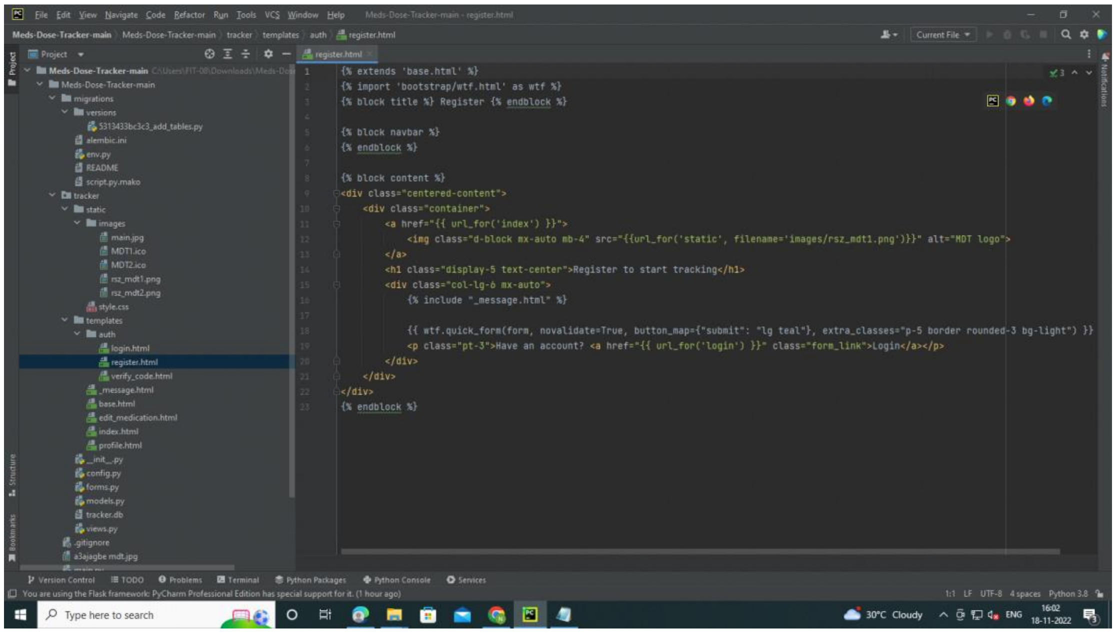
- ## Source Code

login.html



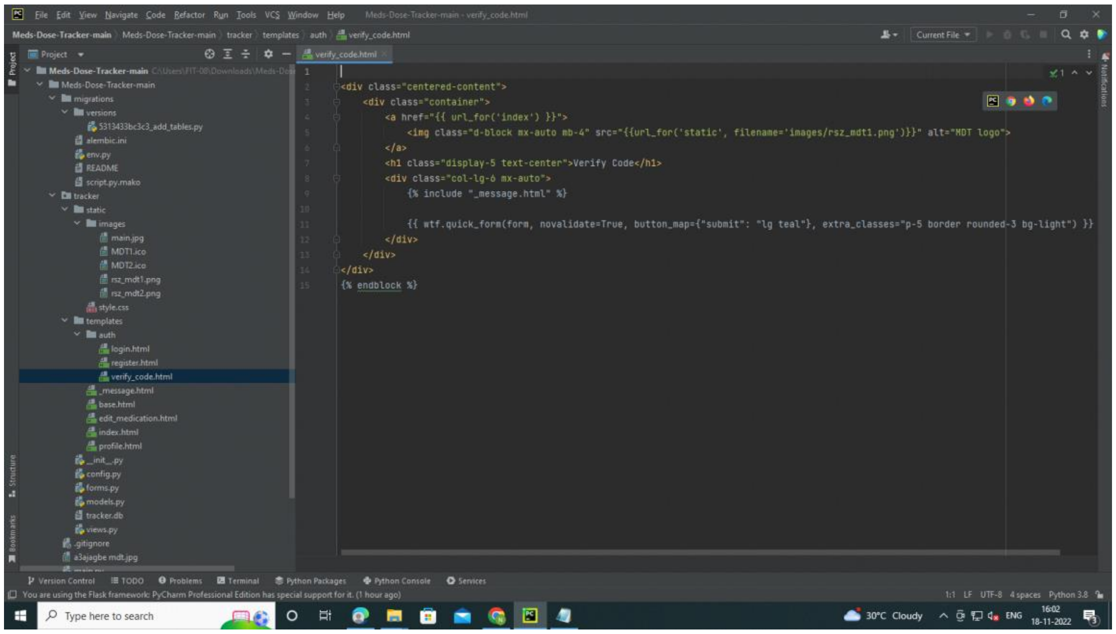
```html

<div class="centered-content">
    <div class="container">
        <a href="{{ url_for('index') }}">
            <img class="d-block mx-auto mb-4" src="{{url_for('static', filename='images/rsz_mdt1.png')}}" alt="MDT logo">
        </a>
        <h1 class="display-5 text-center">Access your account</h1>
        <div class="col-lg-6 mx-auto">
            {% include "_message.html" %}

            {{ wtf.quick_form(form, novalidate=True, button_map={"submit": "lg teal"}, extra_classes="p-5 border rounded-3 bg-light") }}
            <p class="pt-3">Don't have an account? <a href="{{ url_for('register') }}" class="form_link">Register</a></p>
        </div>
    </div>
</div>
{% endblock %}
```

register.html



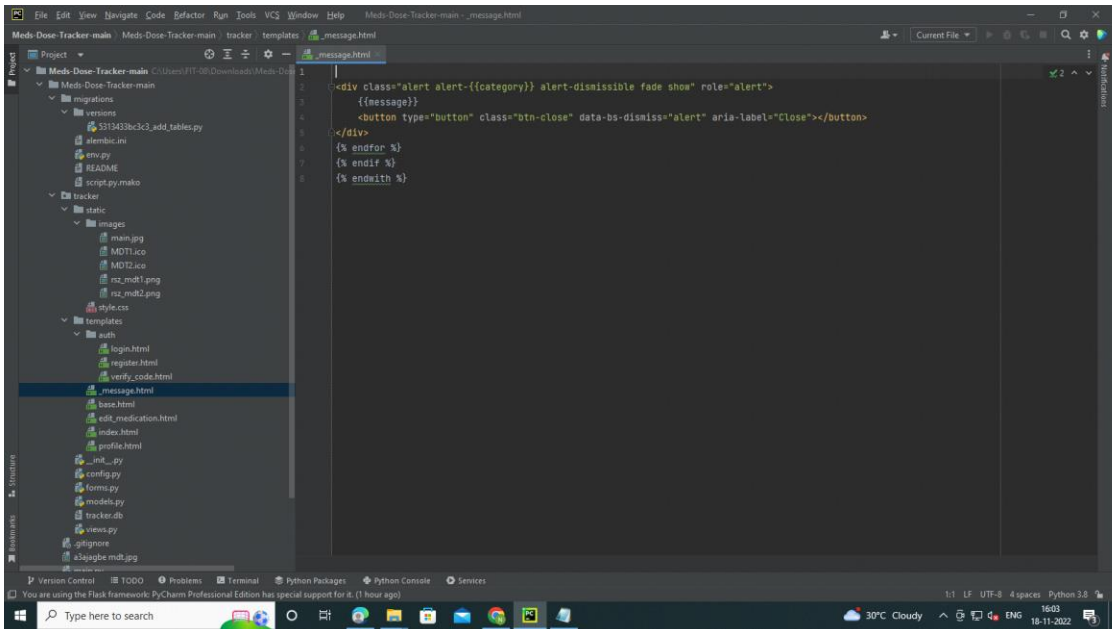
```
{% extends 'base.html' %}
{% import 'bootstrap/wtf.html' as wtf %}
{% block title %} Register {% endblock %}

{% block navbar %}
{% endblock %}

{% block content %}
<div class="centered-content">
    <div class="container">
        <a href="{{ url_for('index') }}">
            <img class="d-block mx-auto mb-4" src="{{url_for('static', filename='images/rsz_mdt1.png')}}" alt="MDT logo">
        </a>
        <h1 class="display-5 text-center">Register to start tracking</h1>
        <div class="col-lg-6 mx-auto">
            {% include "_message.html" %}

            {{ wtf.quick_form(form, novalidate=True, button_map={"submit": "lg teal"}, extra_classes="p-5 border rounded-3 bg-light") }}
            <p class="pt-3">Have an account? <a href="{{ url_for('login') }}" class="form_link">Login</a></p>
        </div>
    </div>
</div>
{% endblock %}
```

verify.html



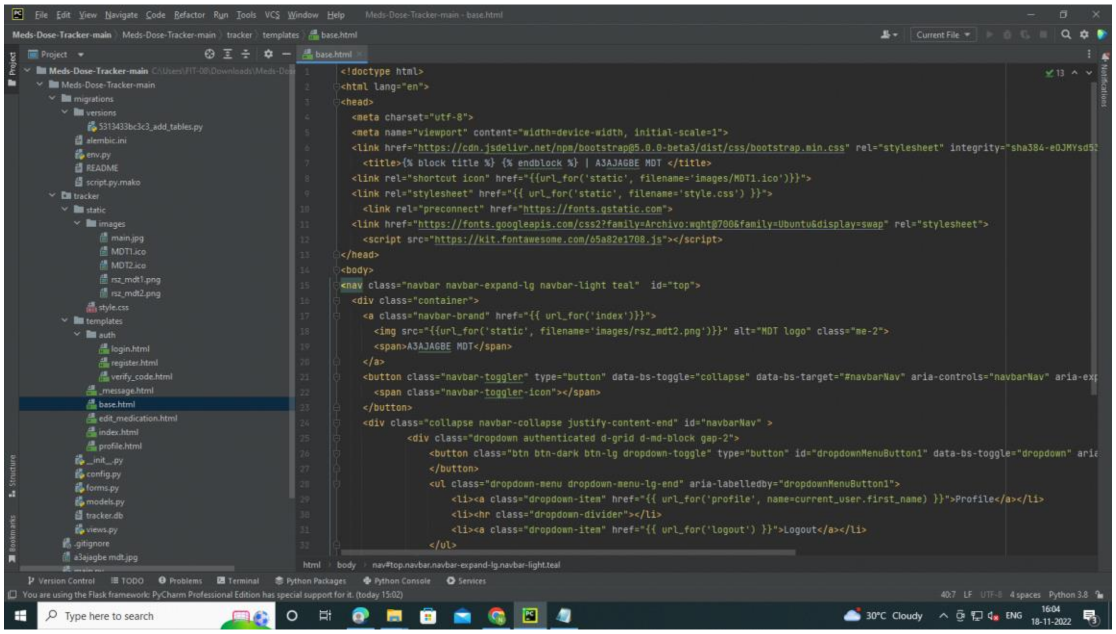
```

<div class="centered-content">
    <div class="container">
        <a href="{{ url_for('index') }}">
            <img class="d-block mx-auto mb-4" src="{{url_for('static', filename='images/rsz_mdt1.png')}}" alt="MDT logo">
        </a>
        <h1 class="display-5 text-center">Verify Code</h1>
        <div class="col-lg-6 mx-auto">
            {% include "_message.html" %}

            {{ wtf.quick_form(form, novalidate=True, button_map={"submit": "lg teal"}, extra_classes="p-5 border rounded-3 bg-light") }}
        </div>
    </div>
</div>
{% endblock %}
```

message.html



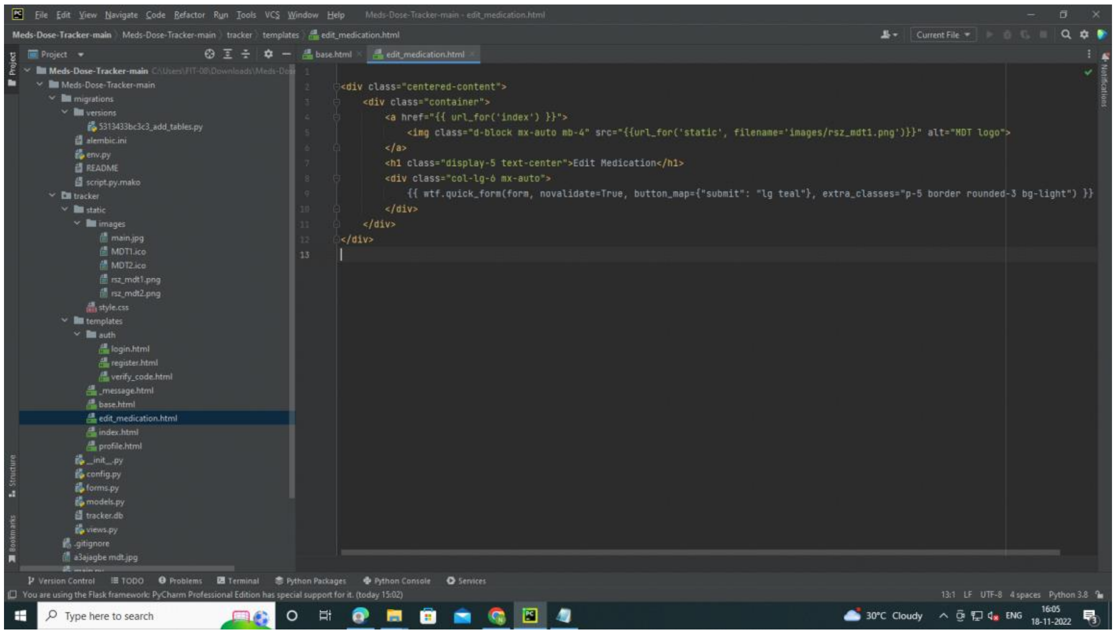
```
<div class="alert alert-{{category}} alert-dismissible fade show" role="alert">
    {{message}}
    <button type="button" class="btn-close" data-bs-dismiss="alert" aria-label="Close"></button>
</div>
{% endfor %}
{% endif %}
{% endwith %}
```

base.html



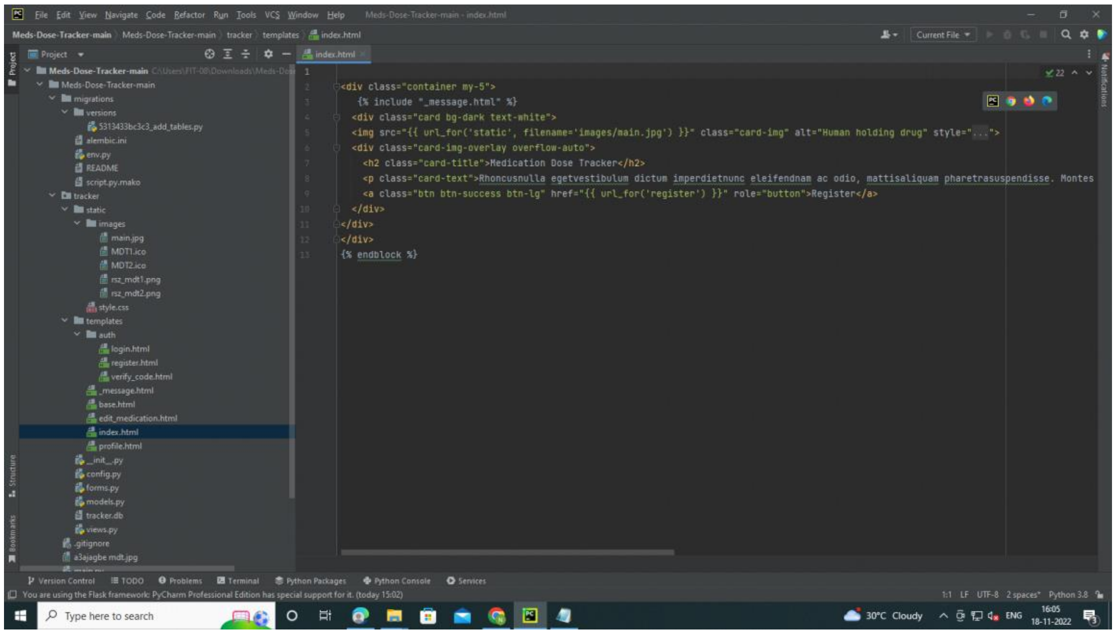
```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-eOJMYsd5
    <title>{% block title %} {% endblock %} | A3AJAGBE MDT </title>
  <link rel="shortcut icon" href="{{url_for('static', filename='images/MDT1.ico')}}">
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    <link rel="preconnect" href="https://fonts.gstatic.com">
  <link href="https://fonts.googleapis.com/css2?family=Archivo:wght@700&family=Ubuntu&display=swap" rel="stylesheet">
    <script src="https://kit.fontawesome.com/65a82e1708.js"></script>
</head>
<body>
<nav class="navbar navbar-expand-lg navbar-light teal"  id="top">
  <div class="container">
    <a class="navbar-brand" href="{{ url_for('index')}}">
      <img src="{{url_for('static', filename='images/rsz_mdt2.png')}}" alt="MDT logo" class="me-2">
      <span>A3AJAGBE MDT</span>
    </a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-ex
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse justify-content-end" id="navbarNav" >
            <div class="dropdown authenticated d-grid d-md-block gap-2">
                <button class="btn btn-dark btn-lg dropdown-toggle" type="button" id="dropdownMenuButton1" data-bs-toggle="dropdown" aria
                </button>
                <ul class="dropdown-menu dropdown-menu-lg-end" aria-labelledby="dropdownMenuButton1">
                    <li><a class="dropdown-item" href="{{ url_for('profile', name=current_user.first_name) }}">Profile</a></li>
                    <li><hr class="dropdown-divider"></li>
                    <li><a class="dropdown-item" href="{{ url_for('logout') }}">Logout</a></li>
                </ul>
```

edit medication.html



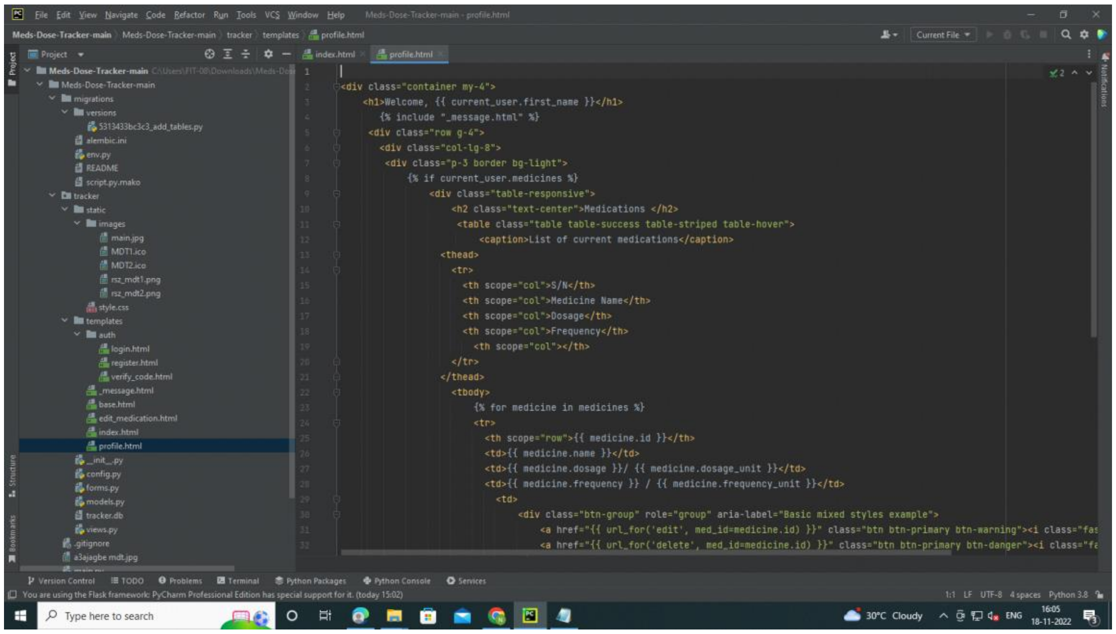
```
<div class="centered-content">
    <div class="container">
        <a href="{{ url_for('index') }}">
            <img class="d-block mx-auto mb-4" src="{{url_for('static', filename='images/rsz_mdt1.png')}}" alt="MDT logo">
        </a>
        <h1 class="display-5 text-center">Edit Medication</h1>
        <div class="col-lg-6 mx-auto">
            {{ wtf.quick_form(form, novalidate=True, button_map={"submit": "lg teal"}, extra_classes="p-5 border rounded-3 bg-light") }}
        </div>
    </div>
</div>
```

index.html

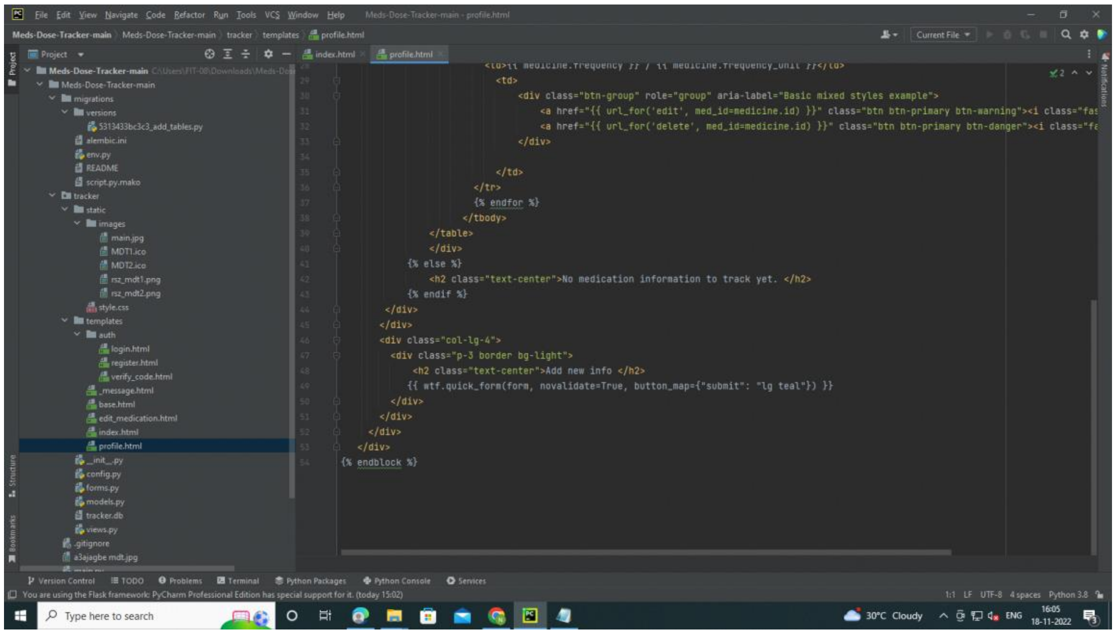


```
<div class="container my-5">
   {% include "_message.html" %}
  <div class="card bg-dark text-white">
  <img src="{{ url_for('static', filename='images/main.jpg') }}" class="card-img" alt="Human holding drug" style="...">
  <div class="card-img-overlay overflow-auto">
    <h2 class="card-title">Medication Dose Tracker</h2>
    <p class="card-text">Rhoncusnulla egetvestibulum dictum imperdietnunc eleifendnam ac odio, mattisaliquam pharetrasuspendisse. Montes
    <a class="btn btn-success btn-lg" href="{{ url_for('register') }}" role="button">Register</a>
  </div>
</div>
</div>
{% endblock %}
```

profile.html

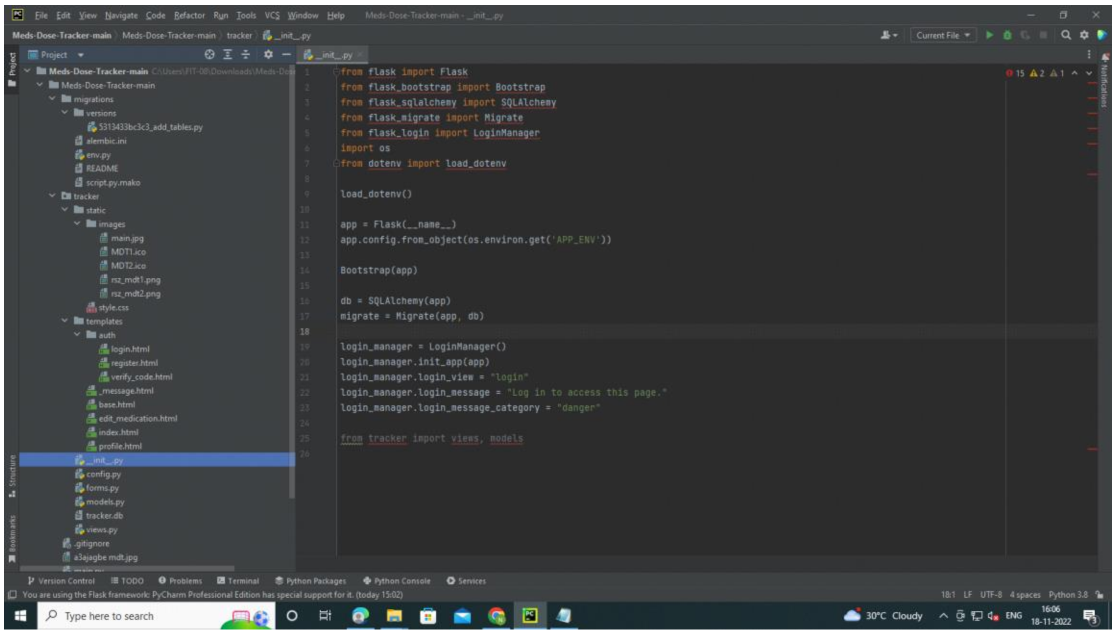


```
<div class="container my-4">
    <h1>Welcome, {{ current_user.first_name }}</h1>
      {% include "_message.html" %}
     <div class="row g-4">
       <div class="col-lg-8">
        <div class="p-3 border bg-light">
            {% if current_user.medicines %}
                <div class="table-responsive">
                    <h2 class="text-center">Medications </h2>
                     <table class="table table-success table-striped table-hover">
                         <caption>List of current medications</caption>
                  <thead>
                    <tr>
                      <th scope="col">S/N</th>
                      <th scope="col">Medicine Name</th>
                      <th scope="col">Dosage</th>
                      <th scope="col">Frequency</th>
                        <th scope="col"></th>
                    </tr>
                  </thead>
                    <tbody>
                        {% for medicine in medicines %}
                        <tr>
                          <th scope="row">{{ medicine.id }}</th>
                          <td>{{ medicine.name }}</td>
                          <td>{{ medicine.dosage }}/ {{ medicine.dosage_unit }}</td>
                          <td>{{ medicine.frequency }} / {{ medicine.frequency_unit }}</td>
                            <td>
                                <div class="btn-group" role="group" aria-label="Basic mixed styles example">
                                    <a href="{{ url_for('edit', med_id=medicine.id) }}" class="btn btn-primary btn-warning"><i class="fas
                                    <a href="{{ url_for('delete', med_id=medicine.id) }}" class="btn btn-primary btn-danger"><i class="fa
```



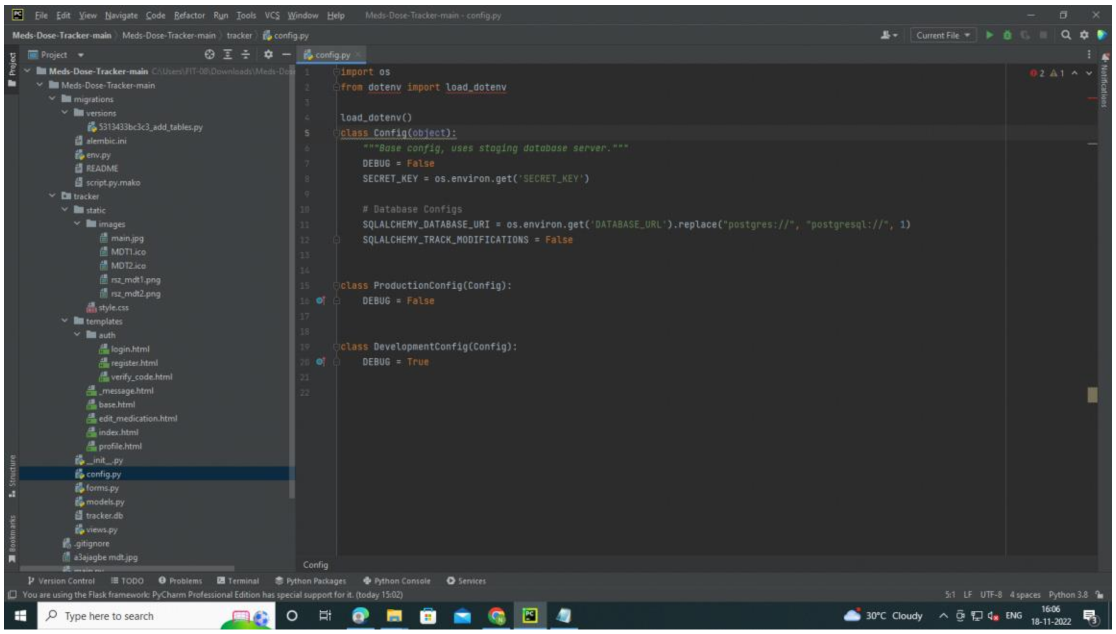
```
                            <td>
                                <div class="btn-group" role="group" aria-label="Basic mixed styles example">
                                    <a href="{{ url_for('edit', med_id=medicine.id) }}" class="btn btn-primary btn-warning"><i class="fas
                                    <a href="{{ url_for('delete', med_id=medicine.id) }}" class="btn btn-primary btn-danger"><i class="fa
                                </div>

                            </td>
                        </tr>
                        {% endfor %}
                      </tbody>
                </table>
                </div>
            {% else %}
                <h2 class="text-center">No medication information to track yet. </h2>
            {% endif %}
        </div>
       </div>
       <div class="col-lg-4">
         <div class="p-3 border bg-light">
             <h2 class="text-center">Add new info </h2>
            {{ wtf.quick_form(form, novalidate=True, button_map={"submit": "lg teal"}) }}
         </div>
       </div>
     </div>
   </div>
{% endblock %}
```

init.py



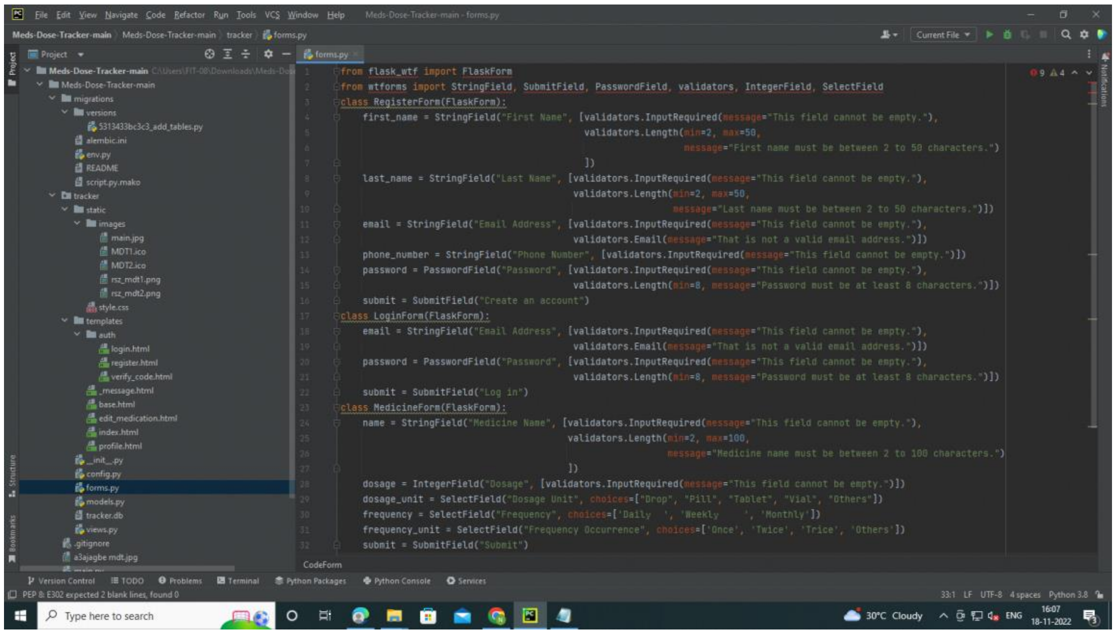
```python
from flask import Flask
from flask_bootstrap import Bootstrap
from flask_sqlalchemy import SQLAlchemy
from flask_migrate import Migrate
from flask_login import LoginManager
import os
from dotenv import load_dotenv

load_dotenv()

app = Flask(__name__)
app.config.from_object(os.environ.get('APP_ENV'))

Bootstrap(app)

db = SQLAlchemy(app)
migrate = Migrate(app, db)

login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = "login"
login_manager.login_message = "Log in to access this page."
login_manager.login_message_category = "danger"

from tracker import views, models
```

config.py



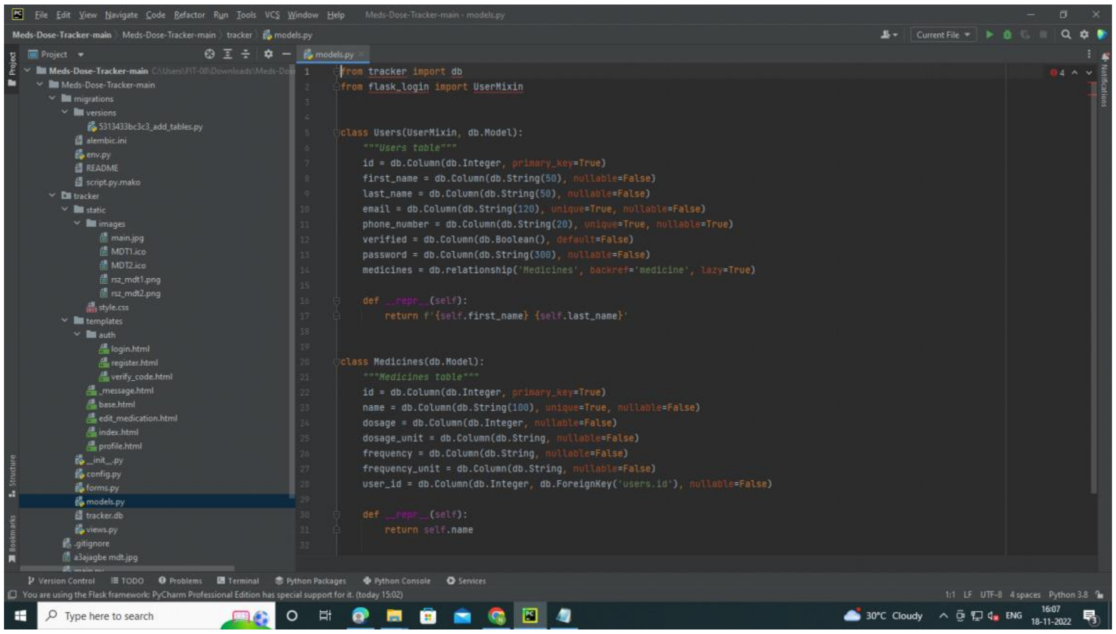
```python
import os
from dotenv import load_dotenv

load_dotenv()
class Config(object):
    """Base config, uses staging database server."""
    DEBUG = False
    SECRET_KEY = os.environ.get('SECRET_KEY')

    # Database Configs
    SQLALCHEMY_DATABASE_URI = os.environ.get('DATABASE_URL').replace("postgres://", "postgresql://", 1)
    SQLALCHEMY_TRACK_MODIFICATIONS = False


class ProductionConfig(Config):
    DEBUG = False


class DevelopmentConfig(Config):
    DEBUG = True
```

forms.py



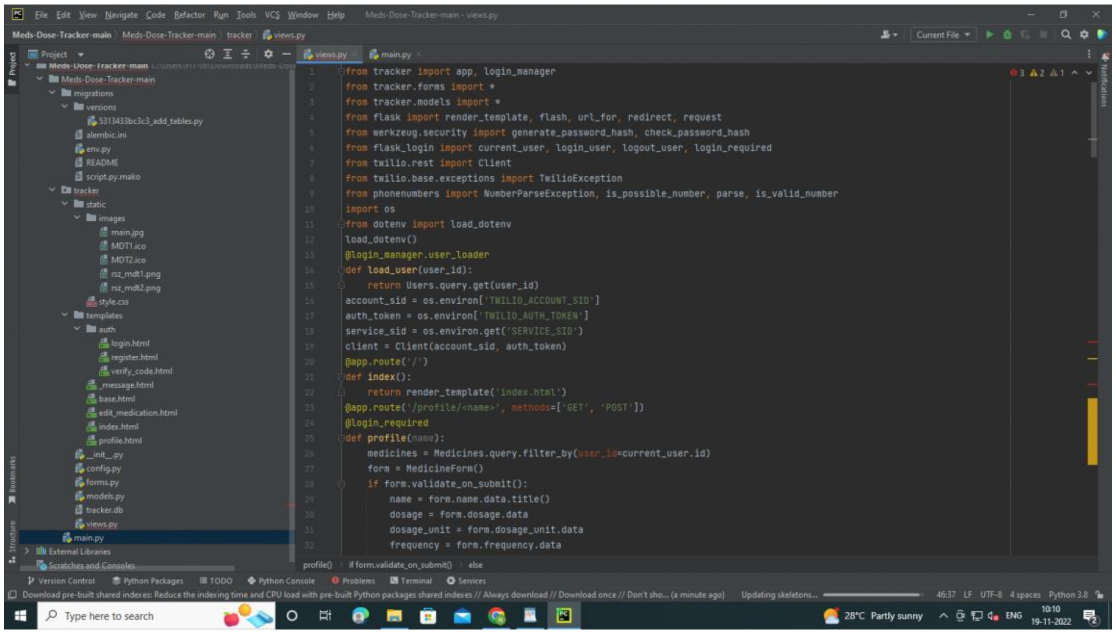
```python
from flask_wtf import FlaskForm
from wtforms import StringField, SubmitField, PasswordField, validators, IntegerField, SelectField
class RegisterForm(FlaskForm):
    first_name = StringField("First Name", [validators.InputRequired(message="This field cannot be empty."),
                                            validators.Length(min=2, max=50,
                                                              message="First name must be between 2 to 50 characters.")
                                            ])
    last_name = StringField("Last Name", [validators.InputRequired(message="This field cannot be empty."),
                                          validators.Length(min=2, max=50,
                                                            message="Last name must be between 2 to 50 characters.")])
    email = StringField("Email Address", [validators.InputRequired(message="This field cannot be empty."),
                                          validators.Email(message="That is not a valid email address.")])
    phone_number = StringField("Phone Number", [validators.InputRequired(message="This field cannot be empty.")])
    password = PasswordField("Password", [validators.InputRequired(message="This field cannot be empty."),
                                          validators.Length(min=8, message="Password must be at least 8 characters.")])
    submit = SubmitField("Create an account")
class LoginForm(FlaskForm):
    email = StringField("Email Address", [validators.InputRequired(message="This field cannot be empty."),
                                          validators.Email(message="That is not a valid email address.")])
    password = PasswordField("Password", [validators.InputRequired(message="This field cannot be empty."),
                                          validators.Length(min=8, message="Password must be at least 8 characters.")])
    submit = SubmitField("Log in")
class MedicineForm(FlaskForm):
    name = StringField("Medicine Name", [validators.InputRequired(message="This field cannot be empty."),
                                         validators.Length(min=2, max=100,
                                                           message="Medicine name must be between 2 to 100 characters.")
                                         ])
    dosage = IntegerField("Dosage", [validators.InputRequired(message="This field cannot be empty.")])
    dosage_unit = SelectField("Dosage Unit", choices=["Drop", "Pill", "Tablet", "Vial", "Others"])
    frequency = SelectField("Frequency", choices=['Daily  ', 'Weekly    ', 'Monthly'])
    frequency_unit = SelectField("Frequency Occurrence", choices=['Once', 'Twice', 'Trice', 'Others'])
    submit = SubmitField("Submit")
```

modes.py

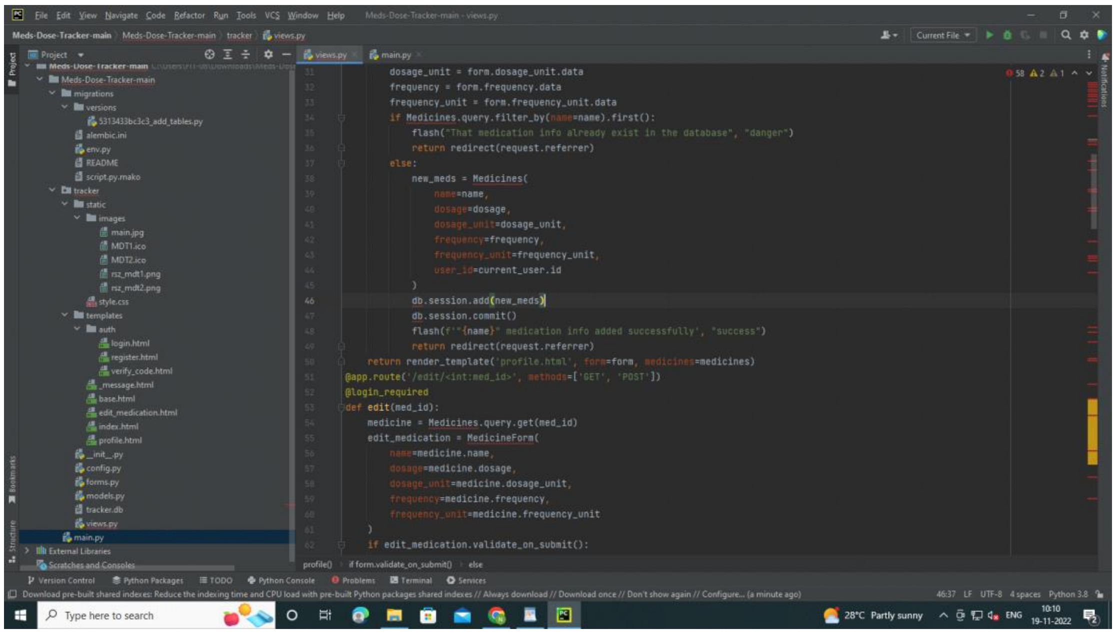


```python
from tracker import db
from flask_login import UserMixin


class Users(UserMixin, db.Model):
    """Users table"""
    id = db.Column(db.Integer, primary_key=True)
    first_name = db.Column(db.String(50), nullable=False)
    last_name = db.Column(db.String(50), nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)
    phone_number = db.Column(db.String(20), unique=True, nullable=True)
    verified = db.Column(db.Boolean(), default=False)
    password = db.Column(db.String(300), nullable=False)
    medicines = db.relationship('Medicines', backref='medicine', lazy=True)

    def __repr__(self):
        return f'{self.first_name} {self.last_name}'


class Medicines(db.Model):
    """Medicines table"""
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), unique=True, nullable=False)
    dosage = db.Column(db.Integer, nullable=False)
    dosage_unit = db.Column(db.String, nullable=False)
    frequency = db.Column(db.String, nullable=False)
    frequency_unit = db.Column(db.String, nullable=False)
    user_id = db.Column(db.Integer, db.ForeignKey('users.id'), nullable=False)

    def __repr__(self):
        return self.name
```

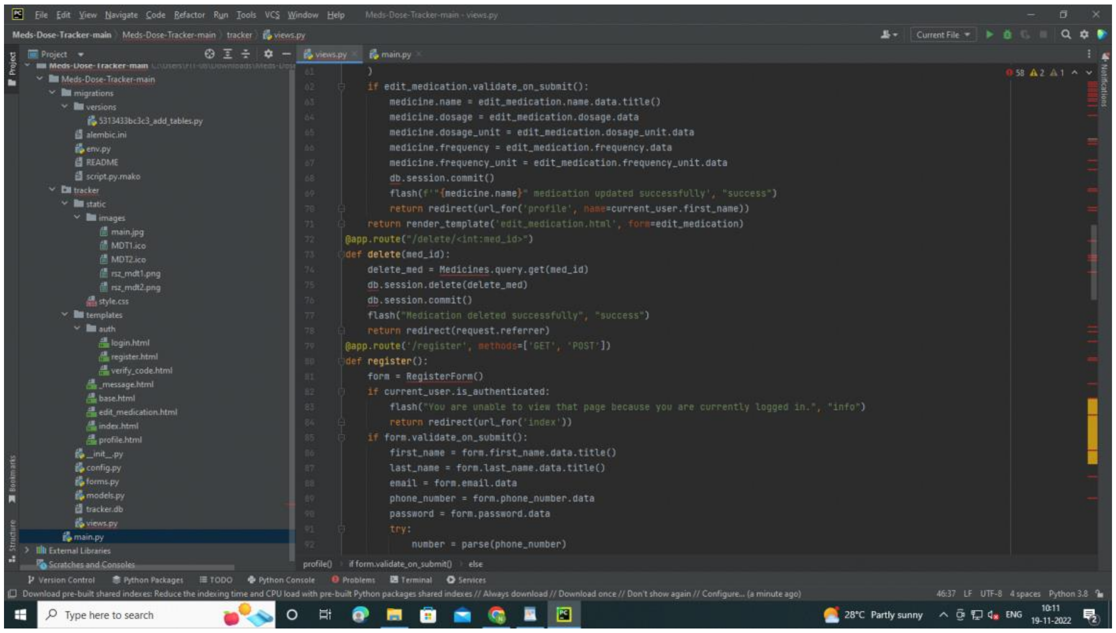
view.py



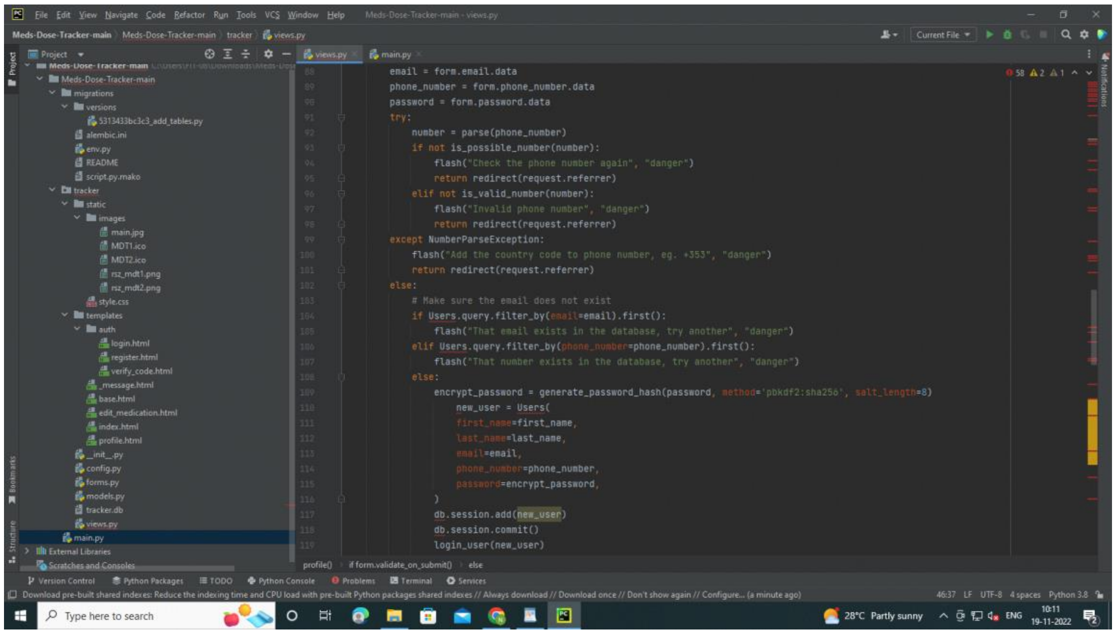
```python
from tracker import app, login_manager
from tracker.forms import *
from tracker.models import *
from flask import render_template, flash, url_for, redirect, request
from werkzeug.security import generate_password_hash, check_password_hash
from flask_login import current_user, login_user, logout_user, login_required
from twilio.rest import Client
from twilio.base.exceptions import TwilioException
from phonenumbers import NumberParseException, is_possible_number, parse, is_valid_number
import os
from dotenv import load_dotenv
load_dotenv()
@login_manager.user_loader
def load_user(user_id):
    return Users.query.get(user_id)
account_sid = os.environ['TWILIO_ACCOUNT_SID']
auth_token = os.environ['TWILIO_AUTH_TOKEN']
service_sid = os.environ.get('SERVICE_SID')
client = Client(account_sid, auth_token)
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/profile/<name>', methods=['GET', 'POST'])
@login_required
def profile(name):
    medicines = Medicines.query.filter_by(user_id=current_user.id)
    form = MedicineForm()
    if form.validate_on_submit():
        name = form.name.data.title()
        dosage = form.dosage.data
        dosage_unit = form.dosage_unit.data
        frequency = form.frequency.data
```



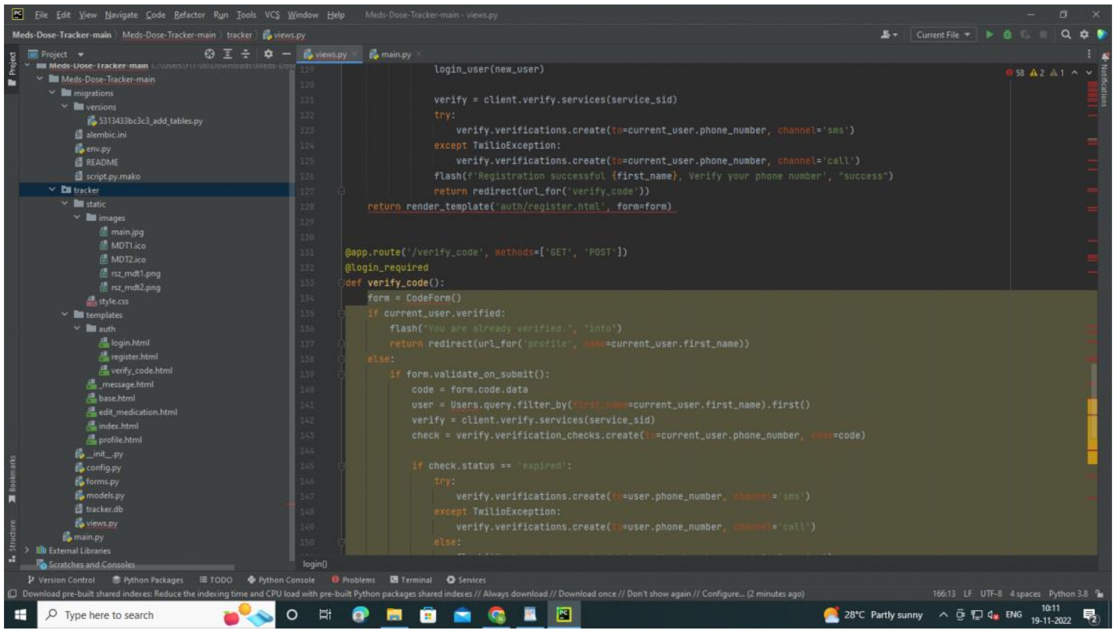
```python
        dosage_unit = form.dosage_unit.data
        frequency = form.frequency.data
        frequency_unit = form.frequency_unit.data
        if Medicines.query.filter_by(name=name).first():
            flash("That medication info already exist in the database", "danger")
            return redirect(request.referrer)
        else:
            new_meds = Medicines(
                name=name,
                dosage=dosage,
                dosage_unit=dosage_unit,
                frequency=frequency,
                frequency_unit=frequency_unit,
                user_id=current_user.id
            )
            db.session.add(new_meds)
            db.session.commit()
            flash(f'"{name}" medication info added successfully', "success")
            return redirect(request.referrer)
    return render_template('profile.html', form=form, medicines=medicines)
@app.route('/edit/<int:med_id>', methods=['GET', 'POST'])
@login_required
def edit(med_id):
    medicine = Medicines.query.get(med_id)
    edit_medication = MedicineForm(
        name=medicine.name,
        dosage=medicine.dosage,
        dosage_unit=medicine.dosage_unit,
        frequency=medicine.frequency,
        frequency_unit=medicine.frequency_unit
    )
    if edit_medication.validate_on_submit():
```



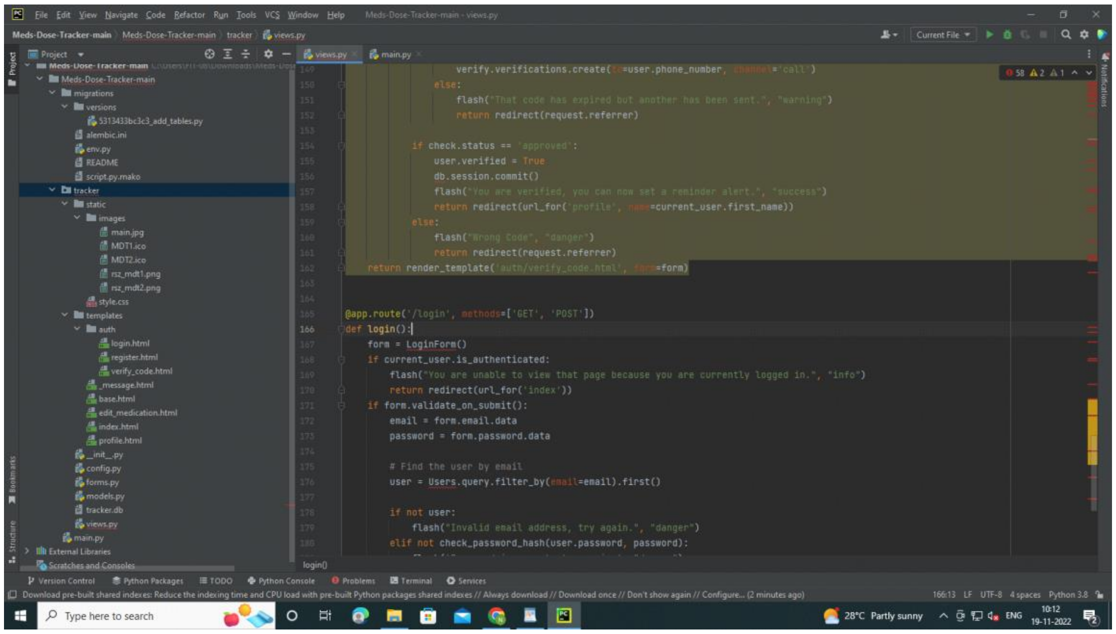
```python
    )
    if edit_medication.validate_on_submit():
        medicine.name = edit_medication.name.data.title()
        medicine.dosage = edit_medication.dosage.data
        medicine.dosage_unit = edit_medication.dosage_unit.data
        medicine.frequency = edit_medication.frequency.data
        medicine.frequency_unit = edit_medication.frequency_unit.data
        db.session.commit()
        flash(f'"{medicine.name}" medication updated successfully', "success")
        return redirect(url_for('profile', name=current_user.first_name))
    return render_template('edit_medication.html', form=edit_medication)
@app.route("/delete/<int:med_id>")
def delete(med_id):
    delete_med = Medicines.query.get(med_id)
    db.session.delete(delete_med)
    db.session.commit()
    flash("Medication deleted successfully", "success")
    return redirect(request.referrer)
@app.route('/register', methods=['GET', 'POST'])
def register():
    form = RegisterForm()
    if current_user.is_authenticated:
        flash("You are unable to view that page because you are currently logged in.", "info")
        return redirect(url_for('index'))
    if form.validate_on_submit():
        first_name = form.first_name.data.title()
        last_name = form.last_name.data.title()
        email = form.email.data
        phone_number = form.phone_number.data
        password = form.password.data
        try:
            number = parse(phone_number)
```



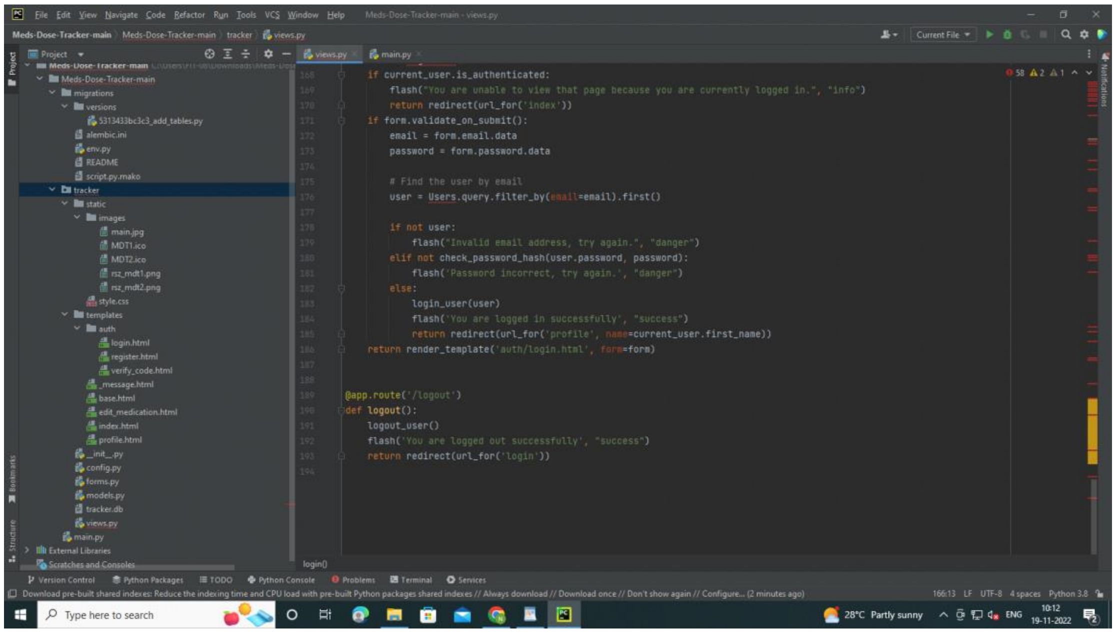
```python
        email = form.email.data
        phone_number = form.phone_number.data
        password = form.password.data
        try:
            number = parse(phone_number)
            if not is_possible_number(number):
                flash("Check the phone number again", "danger")
                return redirect(request.referrer)
            elif not is_valid_number(number):
                flash("Invalid phone number", "danger")
                return redirect(request.referrer)
        except NumberParseException:
            flash("Add the country code to phone number, eg. +353", "danger")
            return redirect(request.referrer)
        else:
            # Make sure the email does not exist
            if Users.query.filter_by(email=email).first():
                flash("That email exists in the database, try another", "danger")
            elif Users.query.filter_by(phone_number=phone_number).first():
                flash("That number exists in the database, try another", "danger")
            else:
                encrypt_password = generate_password_hash(password, method='pbkdf2:sha256', salt_length=8)
                    new_user = Users(
                    first_name=first_name,
                    last_name=last_name,
                    email=email,
                    phone_number=phone_number,
                    password=encrypt_password,
                )
                db.session.add(new_user)
                db.session.commit()
                login_user(new_user)
```



```python
                login_user(new_user)

                verify = client.verify.services(service_sid)
                try:
                    verify.verifications.create(to=current_user.phone_number, channel='sms')
                except TwilioException:
                    verify.verifications.create(to=current_user.phone_number, channel='call')
                flash(f'Registration successful {first_name}, Verify your phone number', "success")
                return redirect(url_for('verify_code'))
    return render_template('auth/register.html', form=form)


@app.route('/verify_code', methods=['GET', 'POST'])
@login_required
def verify_code():
    form = CodeForm()
    if current_user.verified:
        flash("You are already verified.", "info")
        return redirect(url_for('profile', name=current_user.first_name))
    else:
        if form.validate_on_submit():
            code = form.code.data
            user = Users.query.filter_by(first_name=current_user.first_name).first()
            verify = client.verify.services(service_sid)
            check = verify.verification_checks.create(to=current_user.phone_number, code=code)

            if check.status == 'expired':
                try:
                    verify.verifications.create(to=user.phone_number, channel='sms')
                except TwilioException:
                    verify.verifications.create(to=user.phone_number, channel='call')
                else:
```

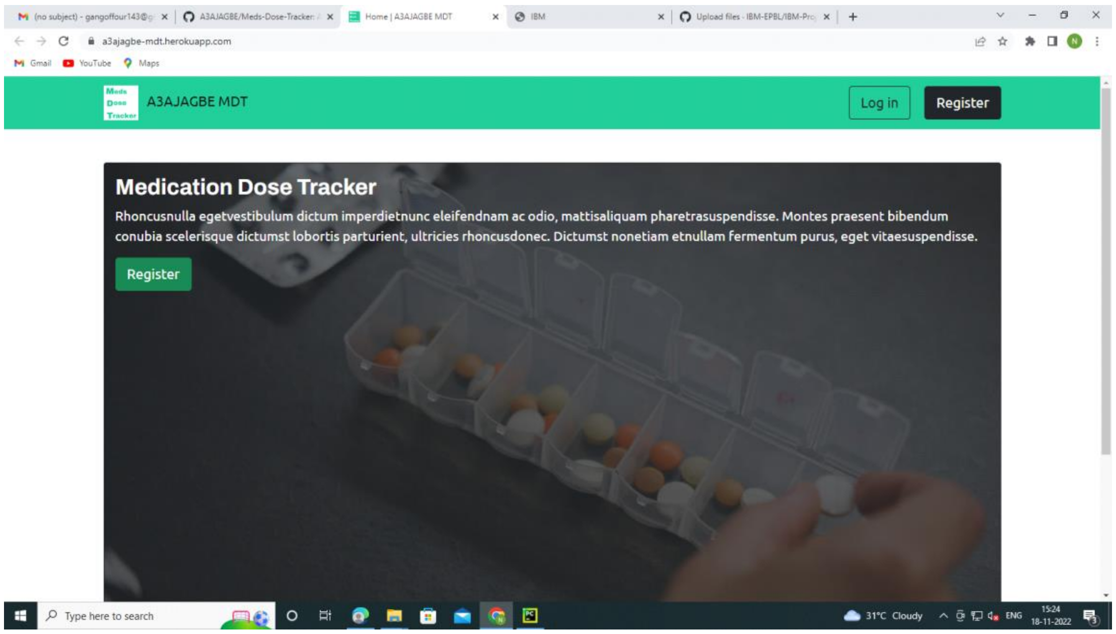


```python
                    verify.verifications.create(to=user.phone_number, channel='call')
                else:
                    flash("That code has expired but another has been sent.", "warning")
                    return redirect(request.referrer)

            if check.status == 'approved':
                user.verified = True
                db.session.commit()
                flash("You are verified, you can now set a reminder alert.", "success")
                return redirect(url_for('profile', name=current_user.first_name))
            else:
                flash("Wrong Code", "danger")
                return redirect(request.referrer)
    return render_template('auth/verify_code.html', form=form)


@app.route('/login', methods=['GET', 'POST'])
def login():
    form = LoginForm()
    if current_user.is_authenticated:
        flash("You are unable to view that page because you are currently logged in.", "info")
        return redirect(url_for('index'))
    if form.validate_on_submit():
        email = form.email.data
        password = form.password.data

        # Find the user by email
        user = Users.query.filter_by(email=email).first()

        if not user:
            flash("Invalid email address, try again.", "danger")
        elif not check_password_hash(user.password, password):
```



```python
    if current_user.is_authenticated:
        flash("You are unable to view that page because you are currently logged in.", "info")
        return redirect(url_for('index'))
    if form.validate_on_submit():
        email = form.email.data
        password = form.password.data

        # Find the user by email
        user = Users.query.filter_by(email=email).first()

        if not user:
            flash("Invalid email address, try again.", "danger")
        elif not check_password_hash(user.password, password):
            flash('Password incorrect, try again.', "danger")
        else:
            login_user(user)
            flash('You are logged in successfully', "success")
            return redirect(url_for('profile', name=current_user.first_name))
    return render_template('auth/login.html', form=form)


@app.route('/logout')
def logout():
    logout_user()
    flash('You are logged out successfully', "success")
    return redirect(url_for('login'))
```

OUTPUT



A3AJAGBE MDT

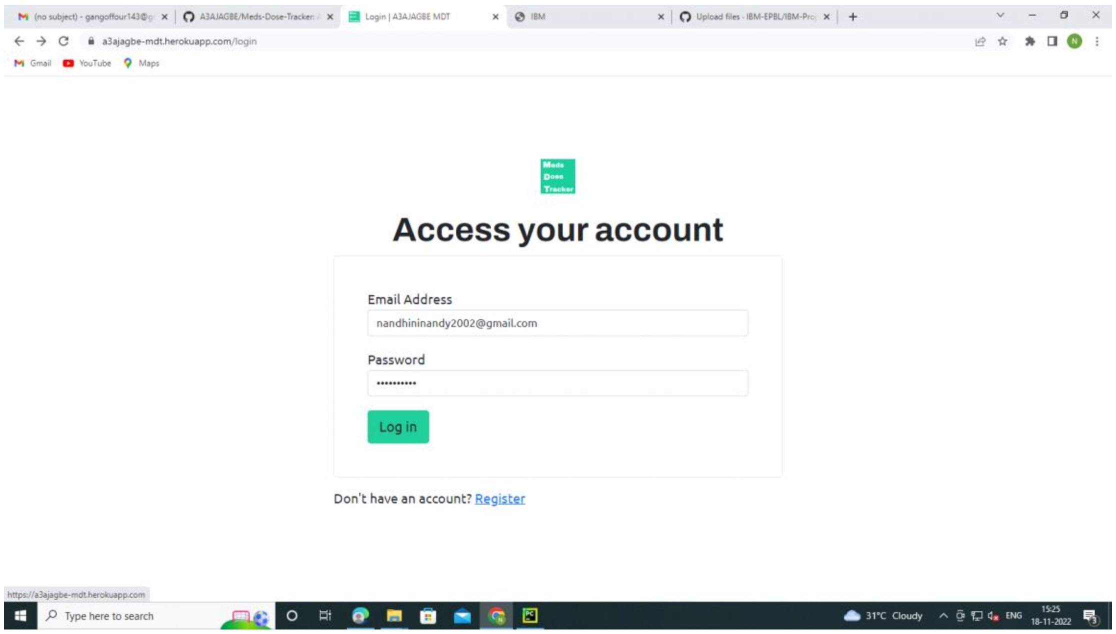
Log in

Register

Medication Dose Tracker

Rhoncusnulla egetvestibulum dictum imperdietnunc eleifendnam ac odio, mattisaliquam pharetrasuspendisse. Montes praesent bibendum conubia scelerisque dictumst lobortis parturient, ultricies rhoncusdonec. Dictumst nonetiam etnullam fermentum purus, eget vitaesuspendisse.

Register



Access your account

Email Address

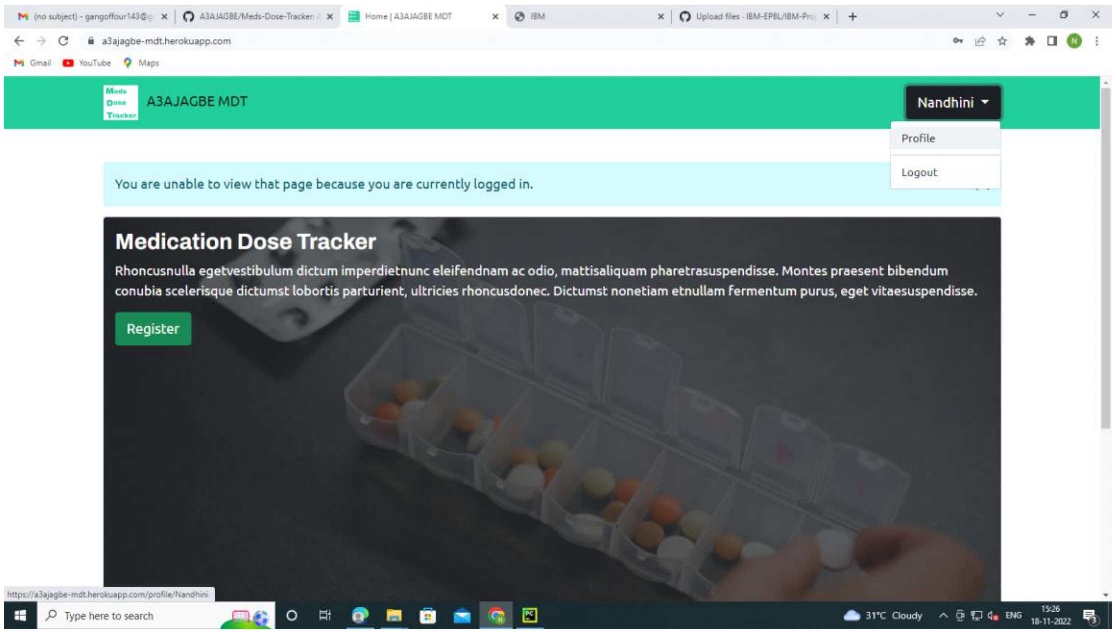
nandhininandy2002@gmail.com

Password

••••••••••

Log in

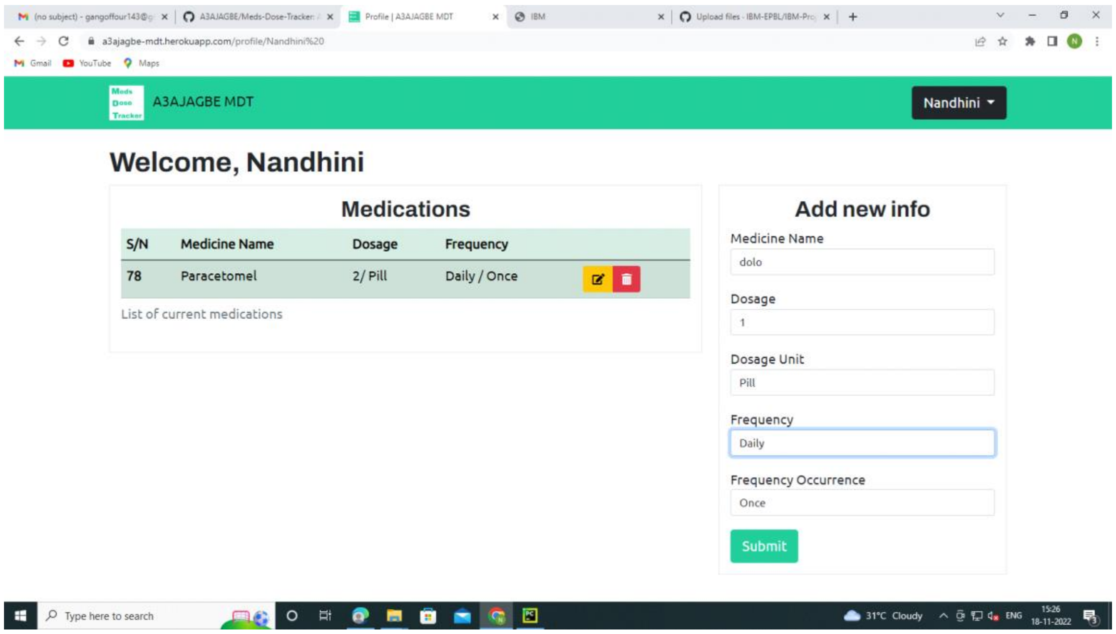
Don't have an account? Register



You are unable to view that page because you are currently logged in.

Medication Dose Tracker

Rhoncusnulla egetvestibulum dictum imperdietnunc eleifendnam ac odio, mattisaliquam pharetrasuspendisse. Montes praesent bibendum conubia scelerisque dictumst lobortis parturient, ultricies rhoncusdonec. Dictumst nonetiam etnullam fermentum purus, eget vitaesuspendisse.



Welcome, Nandhini

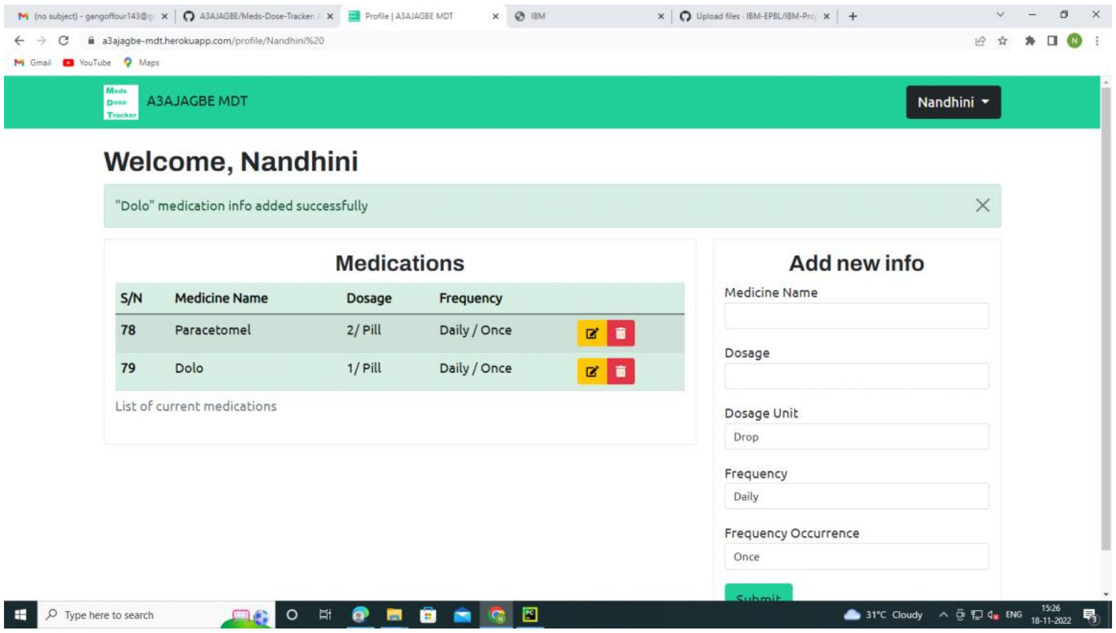
Medications

S/N	Medicine Name	Dosage	Frequency
78	Paracetomel	2/ Pill	Daily / Once

List of current medications

Add new info

Medicine Name	dolo
Dosage	1
Dosage Unit	Pill
Frequency	Daily
Frequency Occurrence	Once


A3AJAGBE MDT

Nandhini

Welcome, Nandhini

"Dolo" medication info added successfully

Medications

S/N	Medicine Name	Dosage	Frequency
78	Paracetomel	2/ Pill	Daily / Once
79	Dolo	1/ Pill	Daily / Once

List of current medications

Add new info

Medicine Name:

Dosage:

Dosage Unit: Drop

Frequency: Daily

Frequency Occurrence: Once

GitHub & Project Demo Link

GitHub LInk
https://github.com/IBM-EPBL/IBM-Project-36460-1664355807

Project Demo Video Link
https://drive.google.com/file/d/18LfRgkp3wfV7mI4rqGbQqOPljyaxXv4l/view