INTEGRATED HEALTH RECORDS MANAGEMENT SYSTEM

SSWT ZG628T DISSERTATION

by

D CHARAN KUMAR

2020WA15718

Dissertation Work carried out at

Wipro Technologies, Chennai

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

Pilani (Rajasthan) India

September 2024

SSWT ZG628T DISSERTATION

INTEGRATED HEALTH RECORDS MANAGEMENT SYSTEM

Submitted in partial fulfilment of the requirements of

M. Tech in Software Systems

by

D CHARAN KUMAR

2020WA15718

Under the supervision of

Muthu Kumar M, (Senior Software Test Engineer)

Dissertation work carried out at

Wipro Technologies, Chennai

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

PILANI (RAJASTHAN)

September 2024

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

<u>CERTIFICATE</u>

This is to certify that the Dissertation entitled “**INTEGRATED HEALTH RECORDS MANAGEMENT SYSTEM**” and submitted by **D CHARAN KUMAR, ID No: 2020WA15718** in partial fulfillment of the requirements of SSWT ZG628T Dissertation, embodies the work done by him under my supervision.

[illegible]

Signature of the Supervisor

Name: Muthu Kumar M

Designation: Senior Software Test Engineer

Date: 12-11-2024

ACKNOWLEDGEMENT

I express my gratitude to **Muthu Kumar M**, my project supervisor, for his tireless work, wise counsel, assistance, and patience. His advice and assistance with this endeavour is indisputable in my opinion. His insightful remarks were helpful in raising the calibre of the output.

I want to sincerely thank **Saraneya Boopathi** and **Jeya Priya** M for taking the time out of their busy schedules to review my progress.

I want to express my gratitude to my **project's partners** for their assistance and cooperation.

I also want to express my gratitude to my friends and family for their encouragement and support throughout the dissertation.

I am incredibly grateful to **WIPRO Technologies** for providing me with the **M. Tech. Software Systems** program training opportunity and all the materials I needed for this dissertation

I appreciate everything that **BITS, Pilani,** and the Training department did to make this course possible.

Sincerely,
D CHARAN KUMAR

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

First Semester 2022-23

SSWT ZG628T DISSERTATION

Dissertation Title: INTEGRATED HEALTH RECORDS MANAGEMENT SYSTEM

Name of Supervisor: MUTHU KUMAR M

Name of Student: D CHARAN KUMAR

Bits ID of Student: 2020WA15718

ABSTRACT

Patients and medical professionals alike can access the centralized INTEGRATED HEALTH RECORDS MANAGEMENT SYSTEM. In the event that the patient needs an emergency procedure, the website aims to deliver vital and significant information about their medical history. This means that the doctors will get the information without having to wait for the patient's relatives to arrive at the hospital. Additionally, this website would serve as a central hub for information exchange between patients and medical experts, and vice versa.

TABLE OF CONTENTS

List of Tables / Figures

1. Chapter 1

1.1 Introduction

In the medical field, stabilizing the patient before medically operating on them is very important and essential when they are admitted during an emergency. But due to lack of the medical history of the patient, stabilising the patient turns difficult. This project introduces a sophisticated system that merges the way the doctors, patients, pharmacists and lab technicians share the data. The main objective of this website is to help doctors and patients view the medical history of the patients simply by inputting the unique id of the patient. All the medical data from patients' diagnostics to prescriptions to lab tests are stored in the website.

2. Chapter 2

2.1 Existing system

Presently, the hospitals do not keep any records of patients who have not yet visited; only those who visit frequently are kept up to date. Patients rarely take their medical records from the hospital they usually visit when they are transferred to another facility.

2.2 Resolution

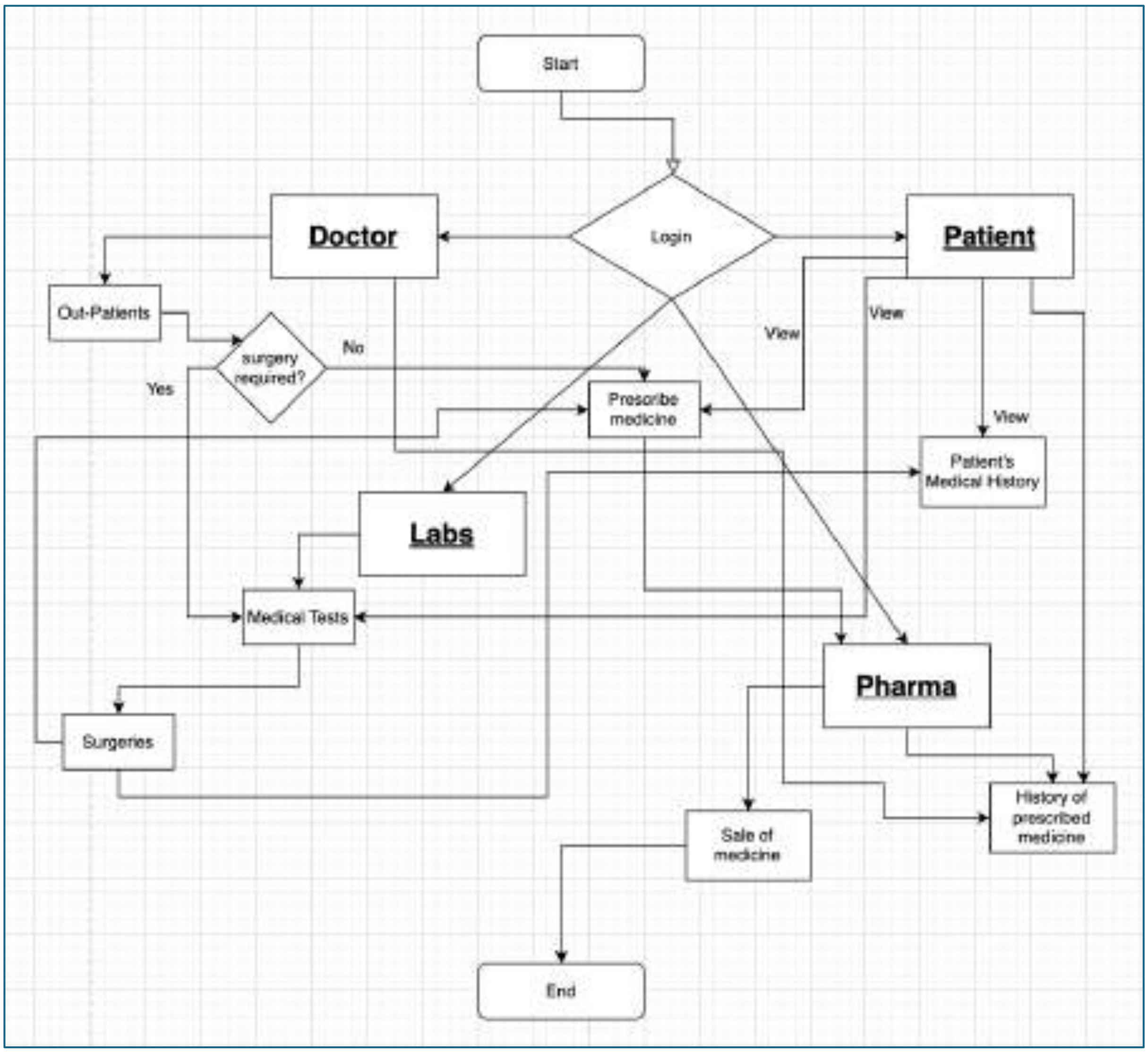
We want to resolve this issue of unavailability of medical history of the patient by creating a centralised website that can display the patient's medical history to the operating doctor. We will combine all the branches like Doctors, Pharmacies, Labs and patients in this project and provide a centralized website where the required data can be viewed by the medical professions as required.

3. Chapter 3

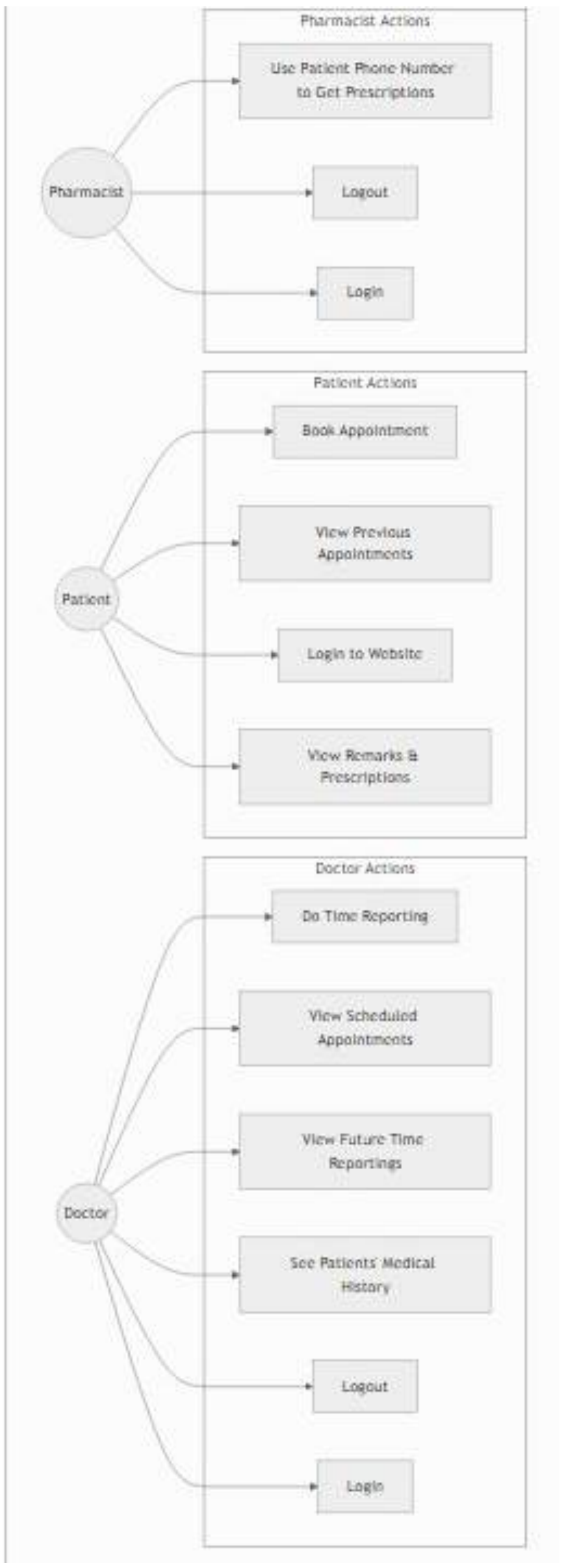
3.1. Estimate of the Project

Serial Number of Task	Tasks or subtasks to be done (be precise and specific)	Planned duration in weeks	Specific Deliverables in terms of the project
1	Requirement Gathering	Week 1	Sprint 1
2	Documentation of design for the website	Week 2 - 3	Sprint 2
3	Development of Dashboard of the website	Week 4	Sprint 3
4	Development of Doctor’s module	Week 5 - 6	Sprint 4
5	Development of Pharmacist’s module	Week 7 - 8	Sprint 5
6	Development of Lab’s module	Week 9 - 10	Sprint 6
7	Development of Patient’s module	Week 11 -12	Sprint 7
8	Customizations	Week 13	Sprint 8
9	Integration of Dashboard, Doctor’s module, Pharmacist’s module, Lab’s module and Patient’s module	Week 14 -15	Sprint 9
10	Integration of Database at the back end	Week 16	Sprint 10

3.2. Flow Chart

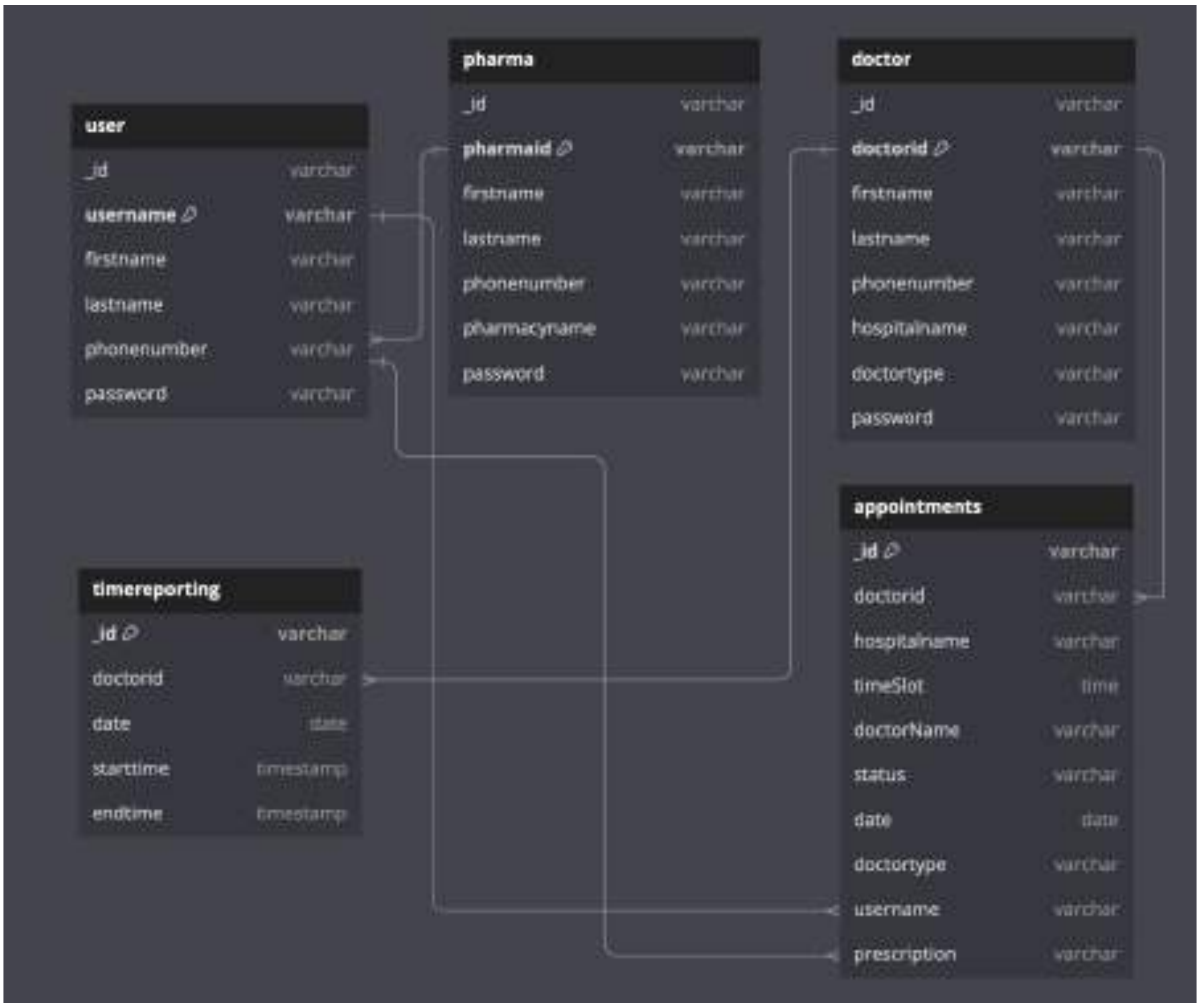


3.3 Use Case Diagram



Pharmacist Actions
Use Patient Phone Number to Get Prescriptions
Logout
Login
Pharmacist
Patient Actions
Book Appointment
View Previous Appointments
Login to Website
View Remarks & Prescriptions
Patient
Doctor Actions
Do Time Reporting
View Scheduled Appointments
View Future Time Reportings
See Patients' Medical History
Logout
Login
Doctor

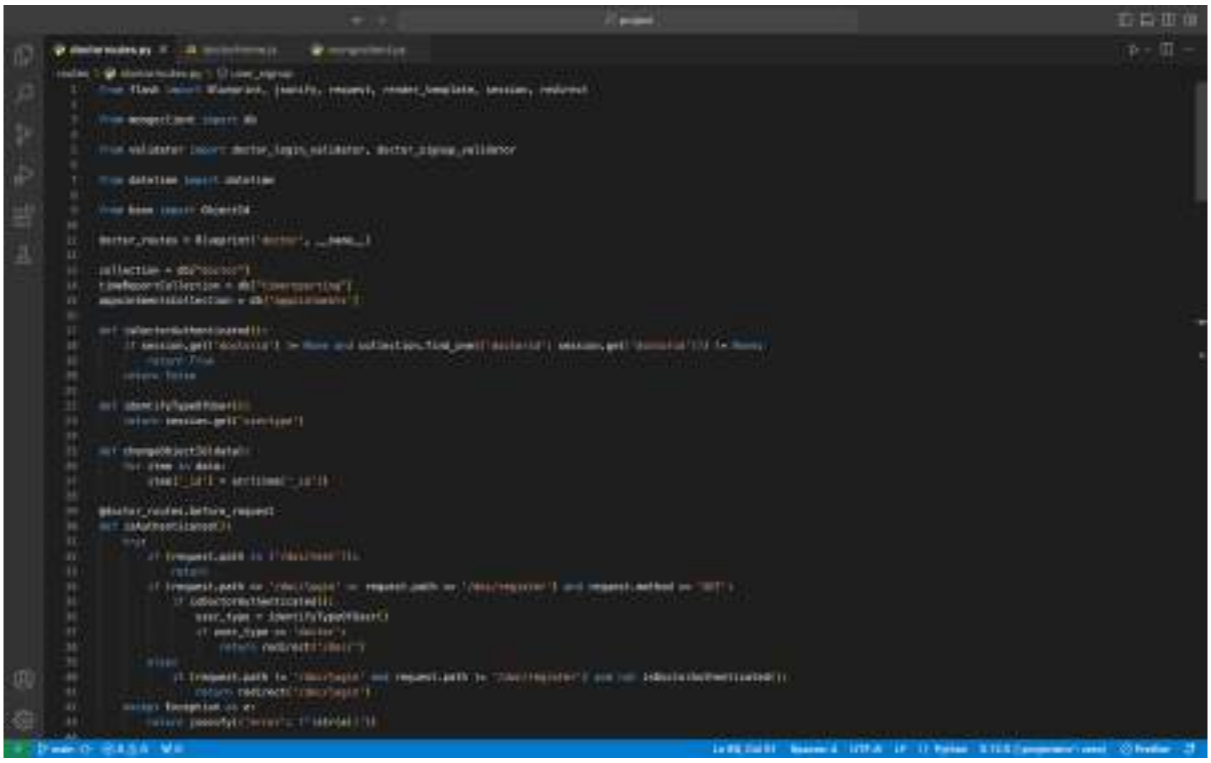
3.4 Database Schema Diagram



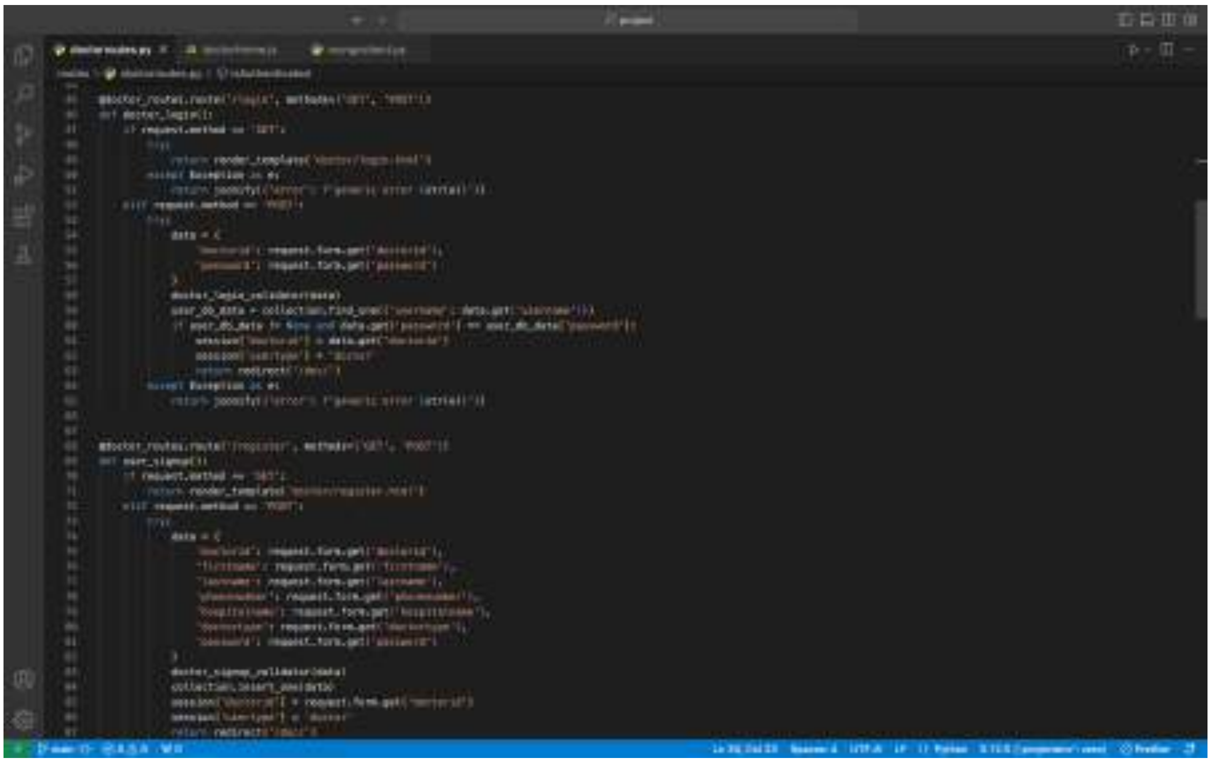
4. Chapter 4

4.1 Doctor Routes

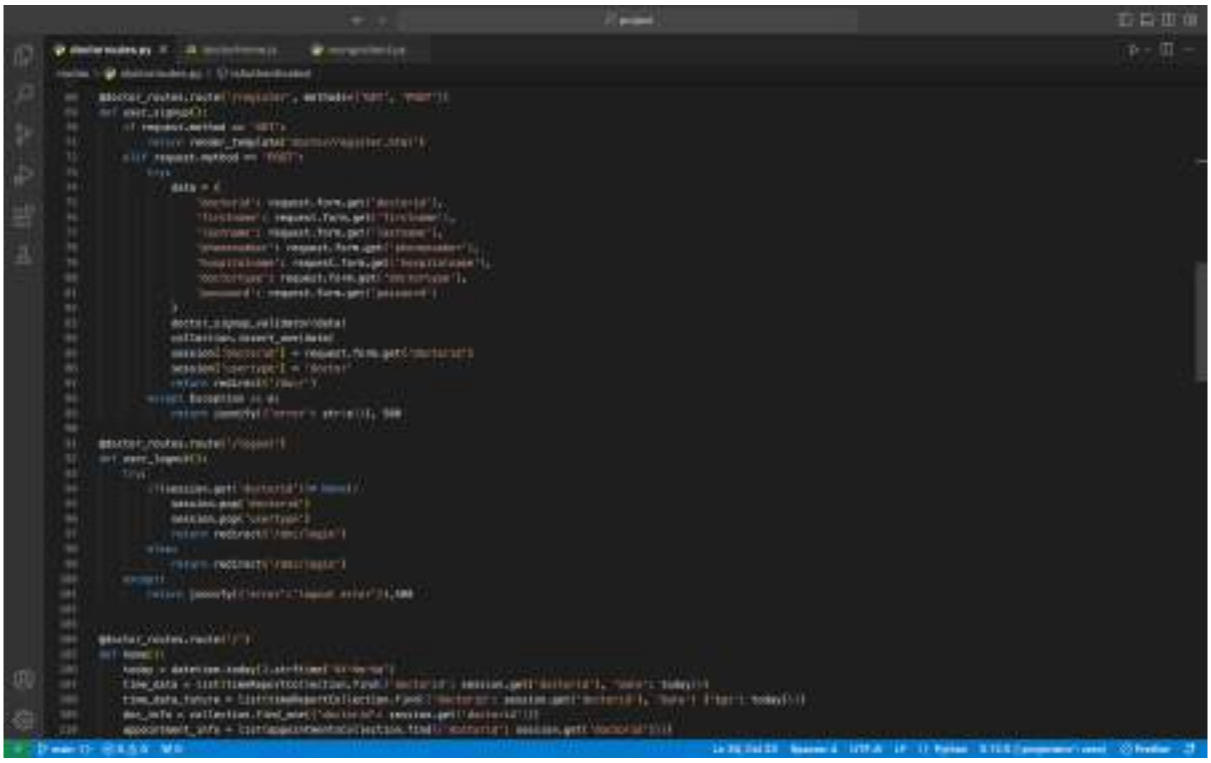
This is a python file that contains all the functions related to doctor flow. Below is the code for the same.



```
from flask import Blueprint, jsonify, request, render_template, session, redirect

from mongoclient import db

from validator import doctor_login_validator, doctor_signup_validator

from datetime import datetime

from bson import ObjectId

doctor_routes = Blueprint('doctor', __name__)

collection = db["doctor"]
timeReportCollection = db["[illegible]"]
appointmentsCollection = db["appointments"]

def isDoctorAuthenticated():
    if session.get('doctorId') != None and collection.find_one({'doctorId': session.get('doctorId')}) != None:
        return True
    return False

def identifyTypeOfUser():
    return session.get('userType')

def changeObjectId(data):
    for item in data:
        item['_id'] = str(item['_id'])

@doctor_routes.before_request
def isAuthenticated():
    try:
        if (request.path in ['/doc/home']):
            return
        if (request.path == '/doc/login' or request.path == '/doc/register') and request.method == 'GET':
            if isDoctorAuthenticated():
                user_type = identifyTypeOfUser()
                if user_type == 'doctor':
                    return redirect('/doc/')
        else:
            if (request.path != '/doc/login' and request.path != '/doc/register') and not isDoctorAuthenticated():
                return redirect('/doc/login')
    except Exception as e:
        return jsonify({'error': f'{str(e)}'})
```

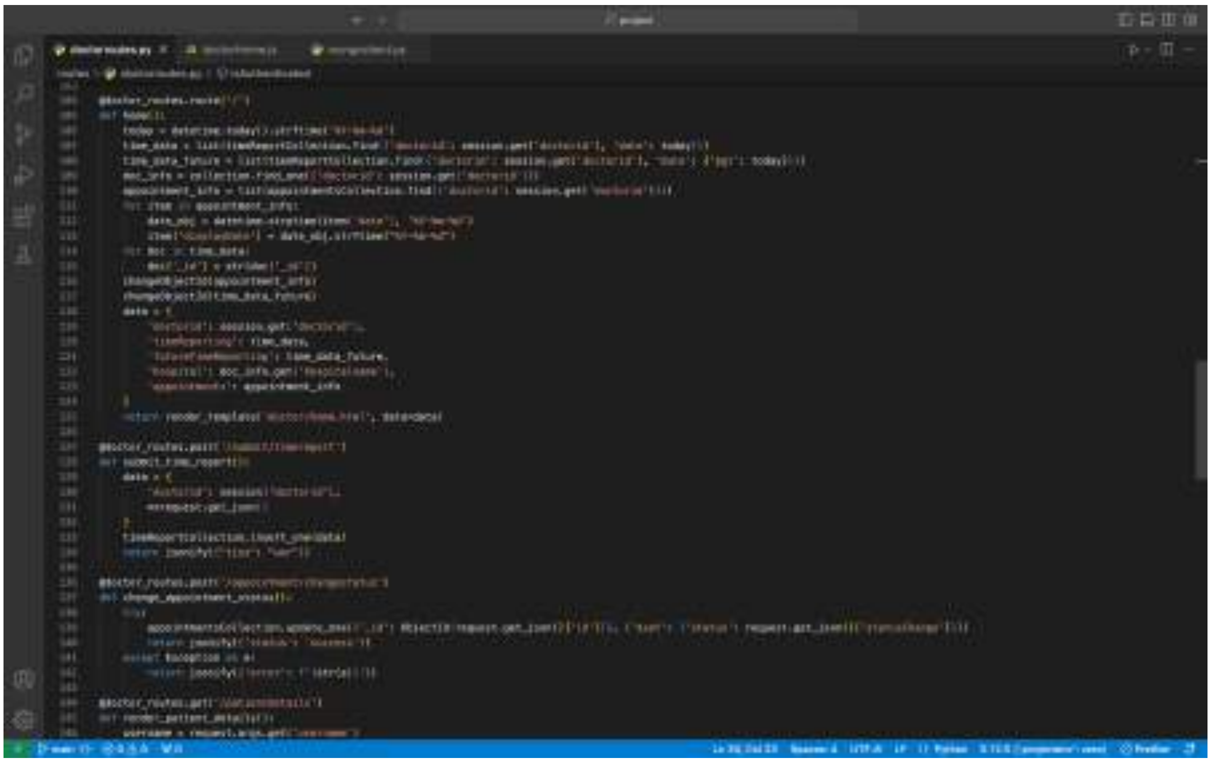


```
@doctor_routes.route('/login', methods=['GET', 'POST'])
def doctor_login():
    if request.method == 'GET':
        try:
            return render_template('doctor/login.html')
        except Exception as e:
            return jsonify({'error': f'generic error {str(e)}'})
    elif request.method == 'POST':
        try:
            data = {
                'doctorid': request.form.get('doctorid'),
                'password': request.form.get('password')
            }
            doctor_login_validator(data)
            user_db_data = collection.find_one({'username': data.get('username')})
            if user_db_data != None and data.get('password') == user_db_data['password']:
                session['doctorid'] = data.get('doctorid')
                session['usertype'] = 'doctor'
                return redirect('/doc/')
        except Exception as e:
            return jsonify({'error': f'generic error {str(e)}'})


@doctor_routes.route('/register', methods=['GET', 'POST'])
def user_signup():
    if request.method == 'GET':
        return render_template('doctor/register.html')
    elif request.method == 'POST':
        try:
            data = {
                'doctorid': request.form.get('doctorid'),
                'firstname': request.form.get('firstname'),
                'lastname': request.form.get('lastname'),
                'phonenumber': request.form.get('phonenumber'),
                'hospitalname': request.form.get('hospitalname'),
                'doctortype': request.form.get('doctortype'),
                'password': request.form.get('password')
            }
            doctor_signup_validator(data)
            collection.insert_one(data)
            session['doctorid'] = request.form.get('doctorid')
            session['usertype'] = 'doctor'
            return redirect('/doc/')
```

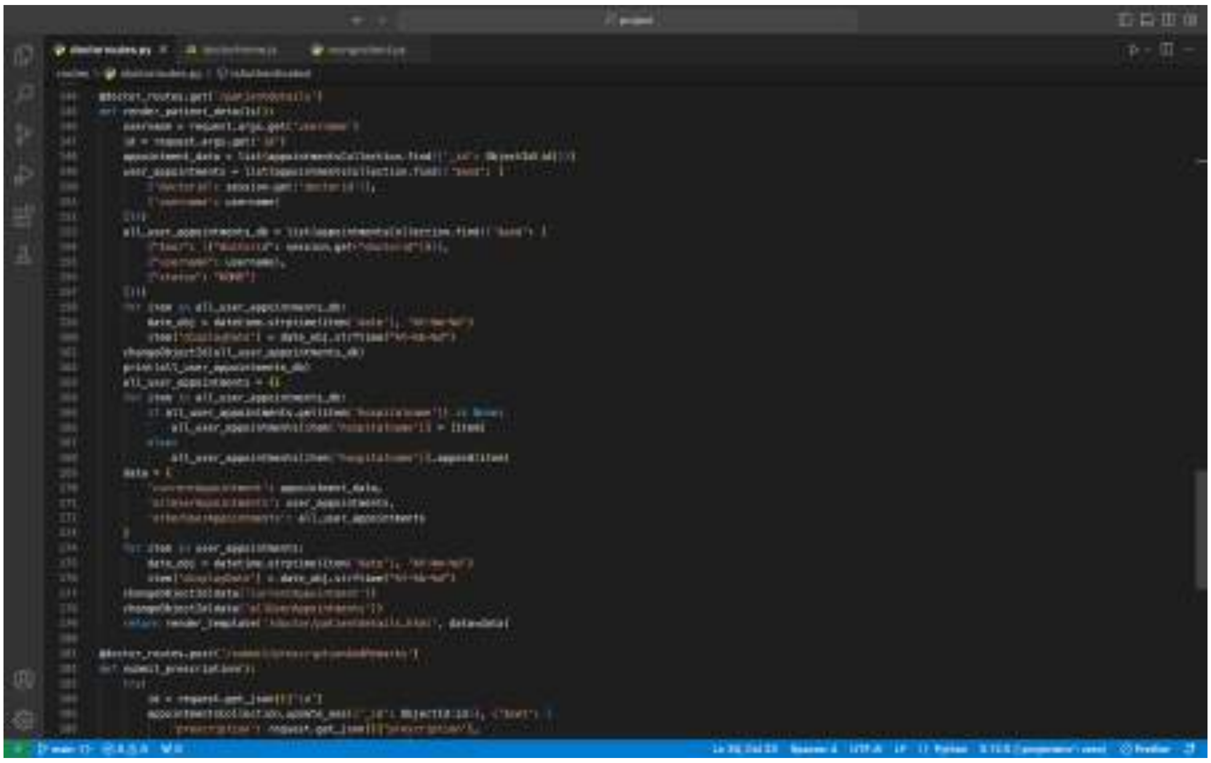


```
@doctor_routes.route('/register', methods=['GET', 'POST'])
def user_signup():
    if request.method == 'GET':
        return render_template('doctor/register.html')
    elif request.method == 'POST':
        try:
            data = {
                'doctorid': request.form.get('doctorid'),
                'firstname': request.form.get('firstname'),
                'lastname': request.form.get('lastname'),
                'phonenumber': request.form.get('phonenumber'),
                'hospitalname': request.form.get('hospitalname'),
                'doctortype': request.form.get('doctortype'),
                'password': request.form.get('password')
            }
            doctor_signup_validator(data)
            collection.insert_one(data)
            session['doctorid'] = request.form.get('doctorid')
            session['usertype'] = 'doctor'
            return redirect('/doc/')
```

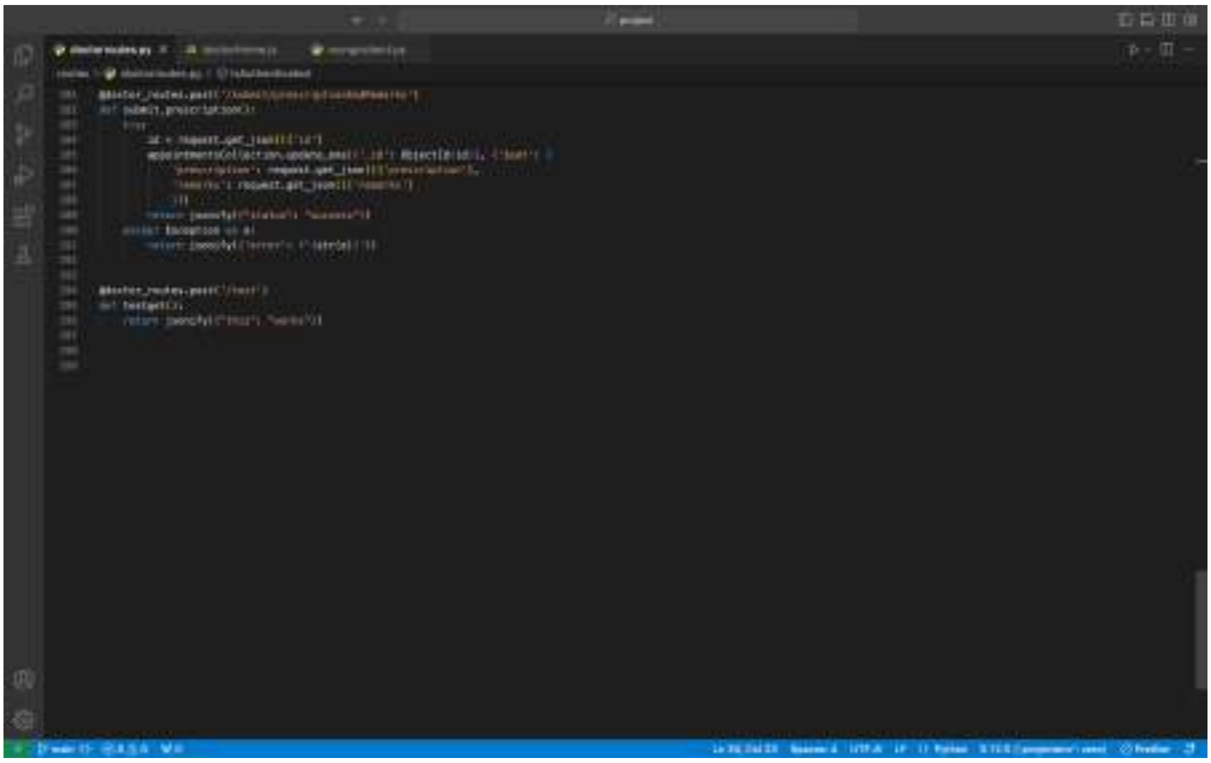
```
        except Exception as e:
            return jsonify({'error': str(e)}), 500

@doctor_routes.route('/logout')
def user_logout():
    try:
        if(session.get('doctorid')!= None):
            session.pop('doctorid')
            session.pop('usertype')
            return redirect('/doc/login')
        else:
            return redirect('/doc/login')
    except:
        return jsonify({'error':'logout error'}),500


@doctor_routes.route('/')
def home():
    today = datetime.today().strftime('%Y-%m-%d')
    time_data = list(timeReportCollection.find({'doctorid': session.get('doctorid'), 'date': today}))
    time_data_future = list(timeReportCollection.find({'doctorid': session.get('doctorid'), 'date': {'$gt': today}}))
    doc_info = collection.find_one({'doctorid': session.get('doctorid')})
    appointment_info = list(appointmentsCollection.find({'doctorid': session.get('doctorid')}))
```



```python
@doctor_routes.route('/')
def home():
    today = datetime.today().strftime('%Y-%m-%d')
    time_data = list(timeReportCollection.find({'doctorId': session.get('doctorId'), 'date': today}))
    time_data_future = list(timeReportCollection.find({'doctorId': session.get('doctorId'), 'date': {'$gt': today}}))
    doc_info = collection.find_one({'doctorId': session.get('doctorId')})
    appointment_info = list(appointmentsCollection.find({'doctorId': session.get('doctorId')}))
    for item in appointment_info:
        date_obj = datetime.strptime(item['date'], '%Y-%m-%d')
        item['displayDate'] = date_obj.strftime("%d-%b-%Y")
    for doc in time_data:
        doc['_id'] = str(doc['_id'])
    changeObjectId(appointment_info)
    changeObjectId(time_data_future)
    data = {
        'doctorId': session.get('doctorId'),
        'timeReporting': time_data,
        'futureTimeReporting': time_data_future,
        'hospital': doc_info.get('hospitalName'),
        'appointments': appointment_info
    }
    return render_template('doctorHome.html', data=data)

@doctor_routes.post('/submitTimeReport')
def submit_time_report():
    data = {
        'doctorId': session['doctorId'],
        **request.get_json()
    }
    timeReportCollection.insert_one(data)
    return jsonify({"tiss": "sar"})

@doctor_routes.post('/appointments/changeStatus')
def change_appointment_status():
    try:
        appointmentsCollection.update_one({'_id': ObjectId(request.get_json()['id'])}, {'$set': {'status': request.get_json()['statusChange']}})
        return jsonify({'status': 'success'})
    except Exception as e:
        return jsonify({'error': f'{str(e)}'})

@doctor_routes.get('/patientdetails')
def render_patient_details():
    username = request.args.get('username')
```



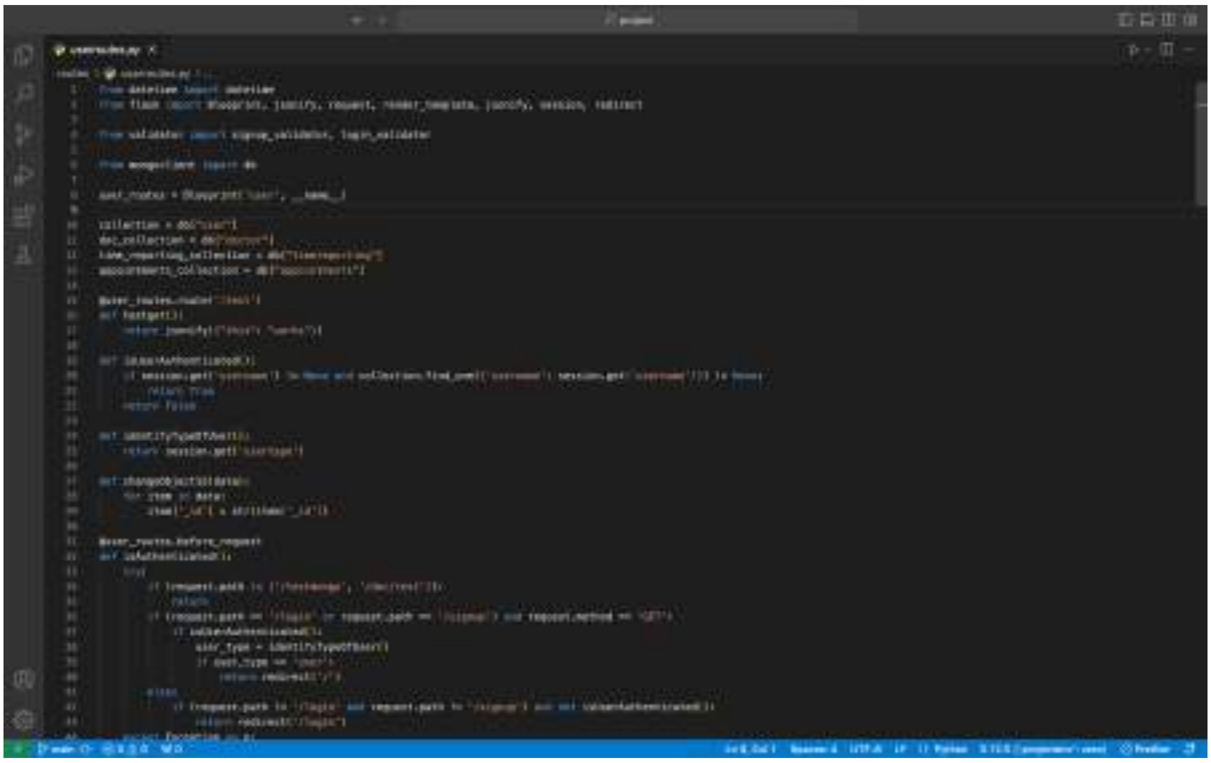
```python
@doctor_routes.get('/patientdetails')
def render_patient_details():
    username = request.args.get('username')
    id = request.args.get('id')
    appointment_data = list(appointmentsCollection.find({'_id': ObjectId(id)}))
    user_appointments = list(appointmentsCollection.find({'$and': [
        {'doctorId': session.get('doctorId')},
        {'username': username}
    ]}))
    all_user_appointments_db = list(appointmentsCollection.find({'$and': [
        {"$ne": {"doctorId": session.get("doctorId")}}},
        {"username": username},
        {"status": "DONE"}
    ]}))
    for item in all_user_appointments_db:
        date_obj = datetime.strptime(item['date'], '%Y-%m-%d')
        item['displayDate'] = date_obj.strftime("%d-%b-%Y")
    changeObjectId(all_user_appointments_db)
    print(all_user_appointments_db)
    all_user_appointments = {}
    for item in all_user_appointments_db:
        if all_user_appointments.get(item['hospitalName']) is None:
            all_user_appointments[item['hospitalName']] = [item]
        else:
            all_user_appointments[item['hospitalName']].append(item)
    data = {
        'currentAppointment': appointment_data,
        'allUserAppointments': user_appointments,
        'otherUserAppointments': all_user_appointments
    }
    for item in user_appointments:
        date_obj = datetime.strptime(item['date'], '%Y-%m-%d')
        item['displayDate'] = date_obj.strftime("%d-%b-%Y")
    changeObjectId(data['currentAppointment'])
    changeObjectId(data['allUserAppointments'])
    return render_template('doctor/patientDetails.html', data=data)

@doctor_routes.post('/submitPrescriptionAndRemarks')
def submit_prescription():
    try:
        id = request.get_json()['id']
        appointmentsCollection.update_one({'_id': ObjectId(id)}, {'$set': {
            'prescription': request.get_json()['prescription'],
```



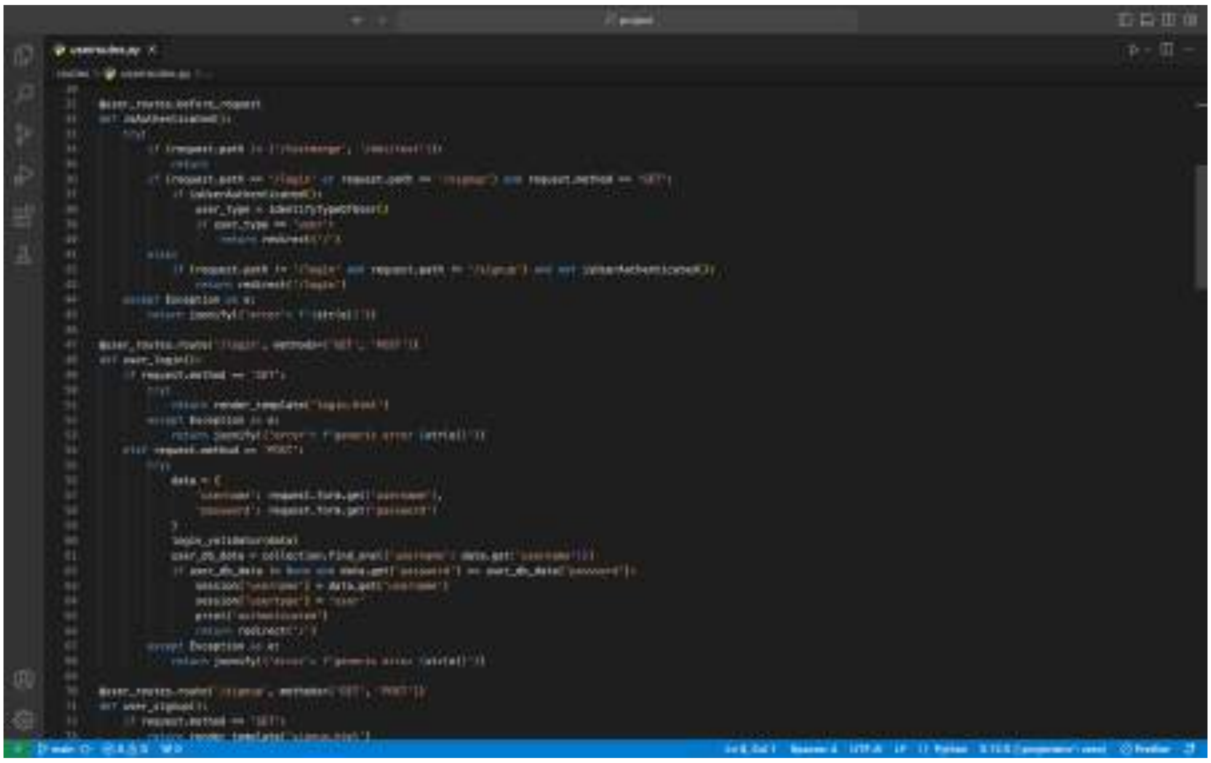
```python
@doctor_routes.post('/submitprescriptionAndRemarks')
def submit_prescription():
    try:
        id = request.get_json()['id']
        appointmentsCollection.update_one({'_id': ObjectId(id)}, {'$set': {
            'prescription': request.get_json()['prescription'],
            'remarks': request.get_json()['remarks']
            }})
        return jsonify({"status": "success"})
    except Exception as e:
        return jsonify({'error': f'{str(e)}'})


@doctor_routes.post('/test')
def testget():
    return jsonify({"this": "works"})
```

4.2 User Routes

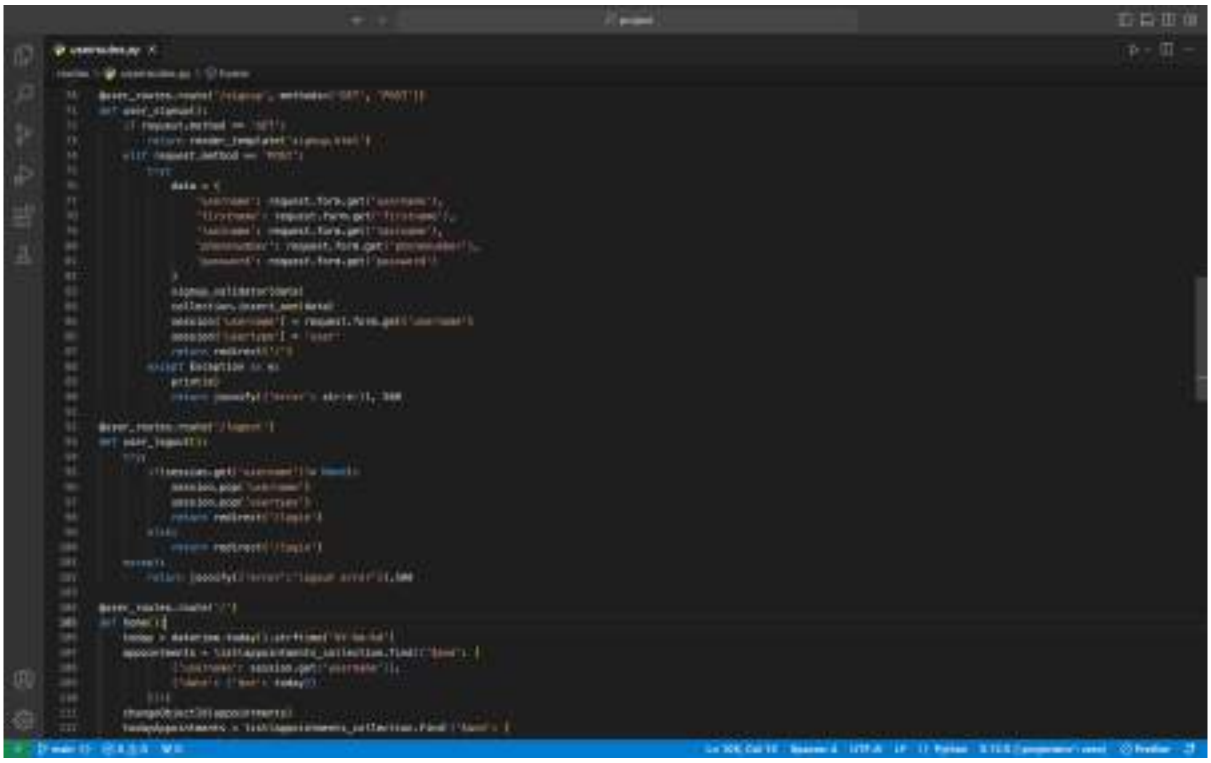
This is a python file that contains all the functions related to user flow. Below is the code for the same.



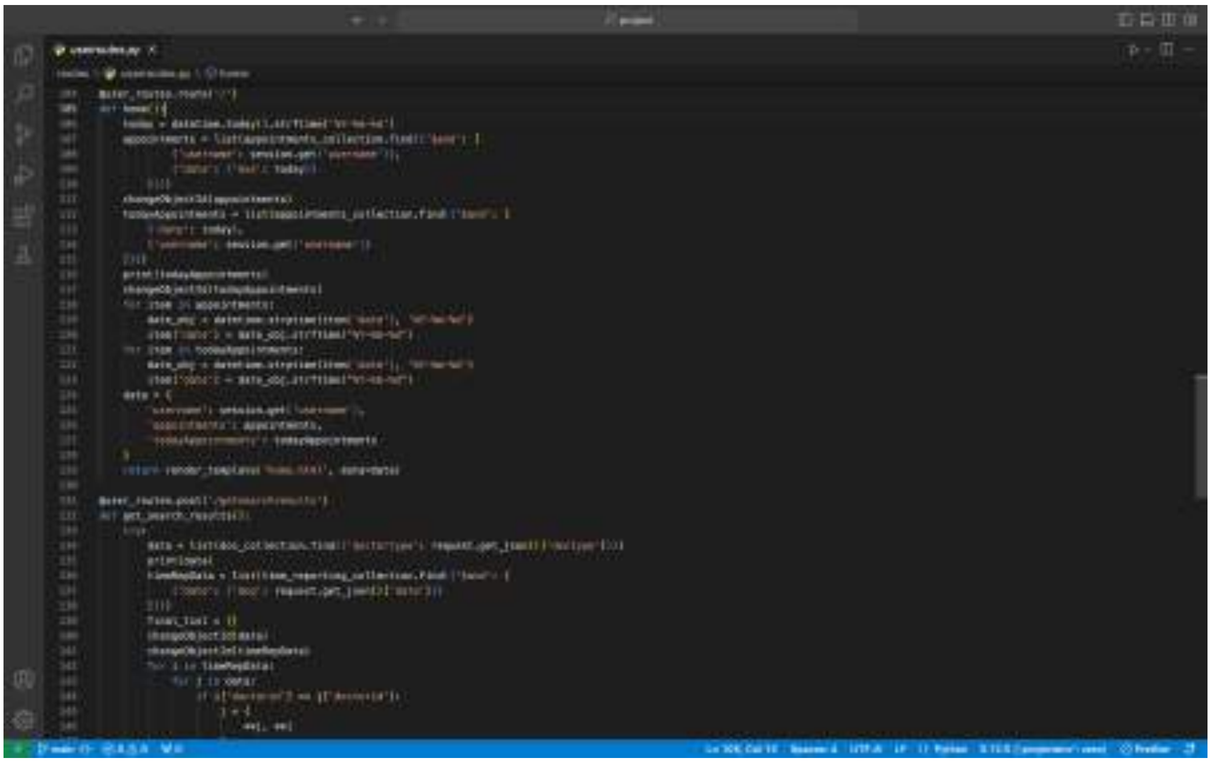
```python
from datetime import datetime
from flask import Blueprint, jsonify, request, render_template, jsonify, session, redirect

from validator import signup_validator, login_validator

from mongoclient import db

user_routes = Blueprint('user', __name__)

collection = db["user"]
doc_collection = db["doctor"]
time_reporting_collection = db["timereporting"]
appointments_collection = db["appointments"]

@user_routes.route('/test')
def testget():
    return jsonify({"this": "works"})

def isUserAuthenticated():
    if session.get('username') != None and collection.find_one({'username': session.get('username')}) != None:
        return True
    return False

def identifyTypeOfUser():
    return session.get('usertype')

def changeObjectId(data):
    for item in data:
        item['_id'] = str(item['_id'])

@user_routes.before_request
def isAuthenticated():
    try:
        if (request.path in ['/checkmongo', '/doctest']):
            return
        if (request.path == '/login' or request.path == '/signup') and request.method == 'GET':
            if isUserAuthenticated():
                user_type = identifyTypeOfUser()
                if user_type == 'user':
                    return redirect('/')
        else:
            if (request.path != '/login' and request.path != '/signup') and not isUserAuthenticated():
                return redirect('/login')
    except Exception as e:
```



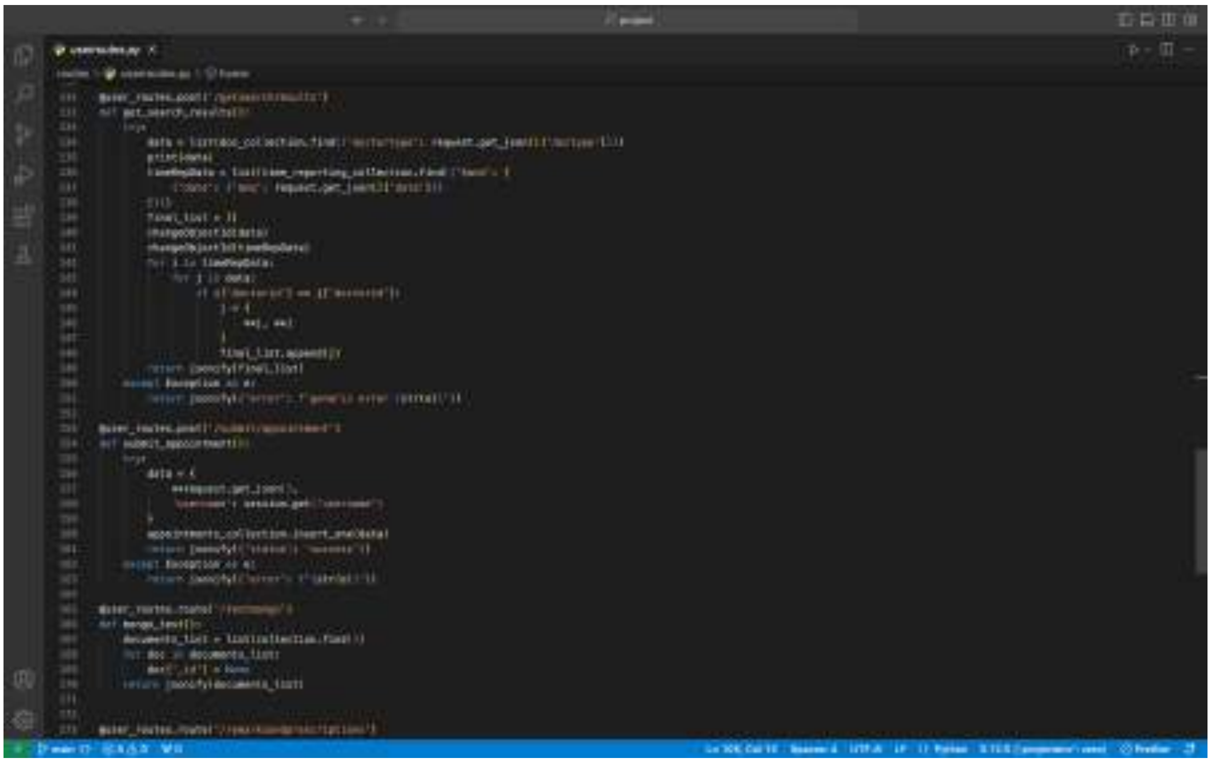
```
@user_routes.before_request
def isAuthenticated():
    try:
        if (request.path in ['/[illegible]', '[illegible]']):
            return
        if (request.path == '/login' or request.path == '/signup') and request.method == 'GET':
            if isUserAuthenticated():
                user_type = identifyTypeOfUser()
                if user_type == 'user':
                    return redirect('/')
        else:
            if (request.path != '/login' and request.path != '/signup') and not isUserAuthenticated():
                return redirect('/login')
    except Exception as e:
        return jsonify({'error': f'{str(e)}'})

@user_routes.route('/login', methods=['GET', 'POST'])
def user_login():
    if request.method == 'GET':
        try:
            return render_template('login.html')
        except Exception as e:
            return jsonify({'error': f'generic error {str(e)}'})
    elif request.method == 'POST':
        try:
            data = {
                'username': request.form.get('username'),
                'password': request.form.get('password')
            }
            login_validator(data)
            user_db_data = collection.find_one({'username': data.get('username')})
            if user_db_data is None and data.get('password') == user_db_data['password']:
                session['username'] = data.get('username')
                session['usertype'] = 'user'
                print('authenticated')
                return redirect('/')
        except Exception as e:
            return jsonify({'error': f'generic error {str(e)}'})

@user_routes.route('/signup', methods=['GET', 'POST'])
def user_signup():
    if request.method == 'GET':
        return render_template('signup.html')
```



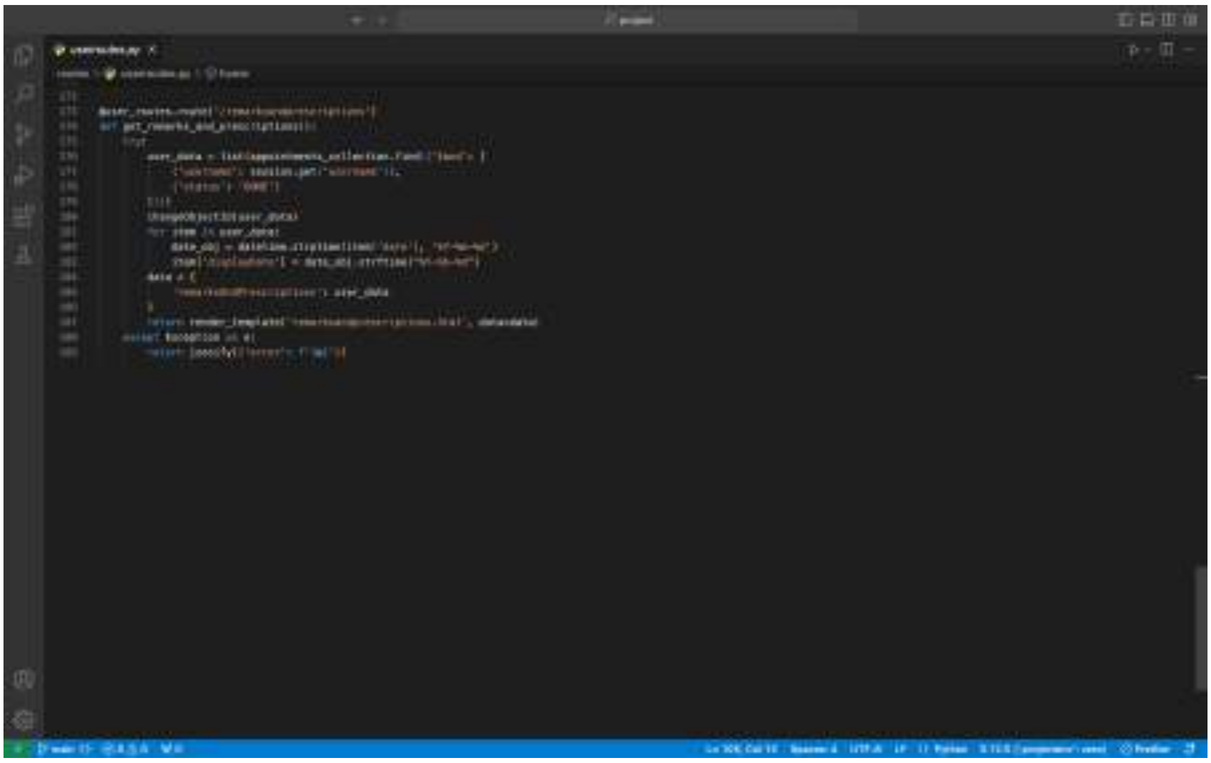
```
@user_routes.route('/signup', methods=['GET', 'POST'])
def user_signup():
    if request.method == 'GET':
        return render_template('signup.html')
    elif request.method == 'POST':
        try:
            data = {
                'username': request.form.get('username'),
                'firstname': request.form.get('firstname'),
                'lastname': request.form.get('lastname'),
                'phonenumber': request.form.get('phonenumber'),
                'password': request.form.get('password')
            }
            signup_validator(data)
            collection.insert_one(data)
            session['username'] = request.form.get('username')
            session['usertype'] = 'user'
            return redirect('/')
        except Exception as e:
            print(e)
            return jsonify({'error': str(e)}), 300

@user_routes.route('/logout')
def user_logout():
    try:
        if(session.get('username')!= None):
            session.pop('username')
            session.pop('usertype')
            return redirect('/login')
        else:
            return redirect('/login')
    except:
        return jsonify({'error':'logout error'}),500

@user_routes.route('/')
def home():
    today = datetime.today().strftime('%Y-%m-%d')
    appointments = list(appointments_collection.find({'$and': [
            {'username': session.get('username')},
            {'date': {'$gt': today}}
        ]}))
    changeObjectId(appointments)
    todayAppointments = list(appointments_collection.find({'$and': [
```



```
@user_routes.route('/')
def home():
    today = datetime.today().strftime('%Y-%m-%d')
    appointments = list(appointments_collection.find({'$and': [
            {'username': session.get('username')},
            {'date': {'$gt': today}}
        ]}))
    changeObjectId(appointments)
    todayAppointments = list(appointments_collection.find({'$and': [
        {'date': today},
        {'username': session.get('username')}
    ]}))
    print(todayAppointments)
    changeObjectId(todayAppointments)
    for item in appointments:
        date_obj = datetime.strptime(item['date'], '%Y-%m-%d')
        item['date'] = date_obj.strftime("%Y-%m-%d")
    for item in todayAppointments:
        date_obj = datetime.strptime(item['date'], '%Y-%m-%d')
        item['date'] = date_obj.strftime("%Y-%m-%d")
    data = {
        'username': session.get('username'),
        'appointments': appointments,
        'todayAppointments': todayAppointments
    }
    return render_template('home.html', data=data)

@user_routes.post('/getsearchresults')
def get_search_results():
    try:
        data = list(doc_collection.find({'doctortype': request.get_json()['doctype']}))
        print(data)
        timeRepData = list(time_reporting_collection.find({'$and': [
            {'date': {'$eq': request.get_json()['date']}}
        ]}))
        final_list = []
        changeObjectId(data)
        changeObjectId(timeRepData)
        for i in timeRepData:
            for j in data:
                if i['doctorid'] == j['doctorid']:
                    j = {
                        **j, **i
```



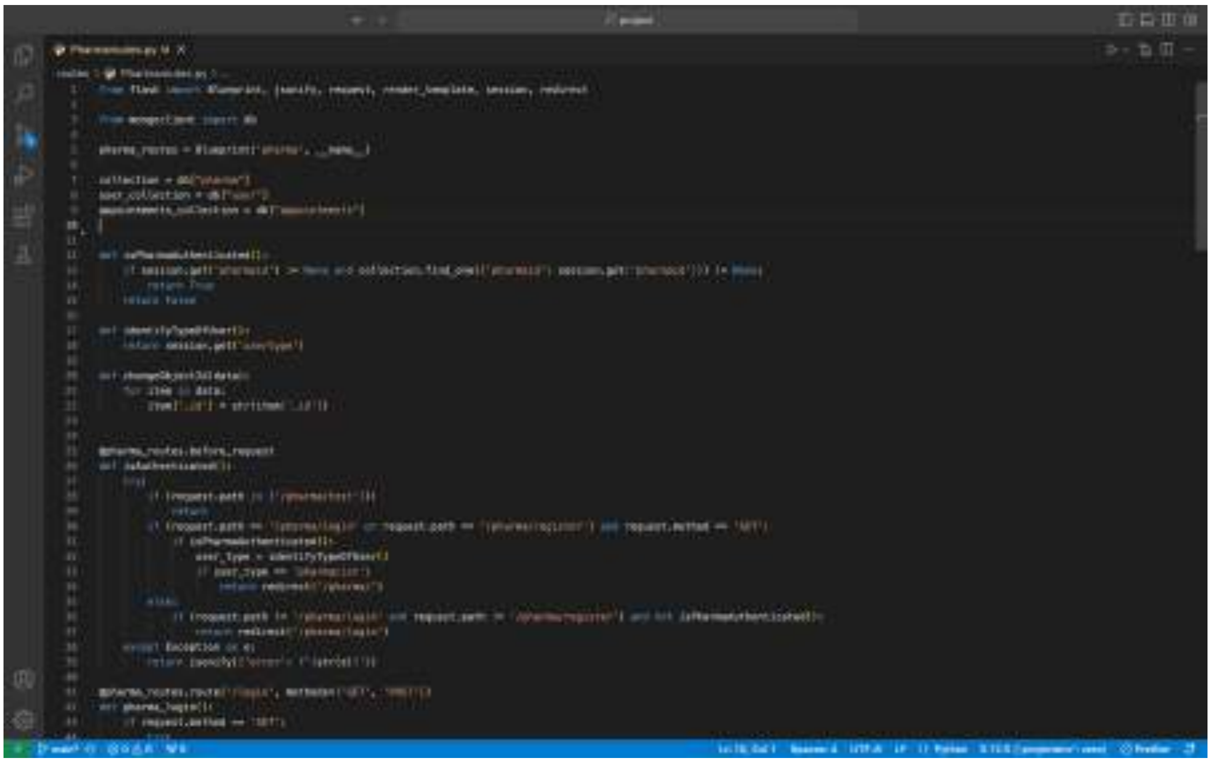
```
@user_routes.post('/getsearchresults')
def get_search_results():
    try:
        data = list(doc_collection.find({'doctortype': request.get_json()['doctype']}))
        print(data)
        timeRepData = list(time_reporting_collection.find({'$and': [
            {'date': {'$eq': request.get_json()['date']}}
        ]}))
        final_list = []
        changeObjectId(data)
        changeObjectId(timeRepData)
        for i in timeRepData:
            for j in data:
                if i['doctorid'] == j['doctorid']:
                    j = {
                        **j, **i
                    }
                    final_list.append(j)
        return jsonify(final_list)
    except Exception as e:
        return jsonify({'error': f'generic error {str(e)}'})

@user_routes.post('/submitappointment')
def submit_appointment():
    try:
        data = {
            **request.get_json(),
            'username': session.get('username')
        }
        appointments_collection.insert_one(data)
        return jsonify({'status': 'success'})
    except Exception as e:
        return jsonify({'error': f'{str(e)}'})

@user_routes.route('/testmongo')
def mongo_test():
    documents_list = list(collection.find())
    for doc in documents_list:
        doc['_id'] = None
    return jsonify(documents_list)


@user_routes.route('/searchandprescriptions')
```



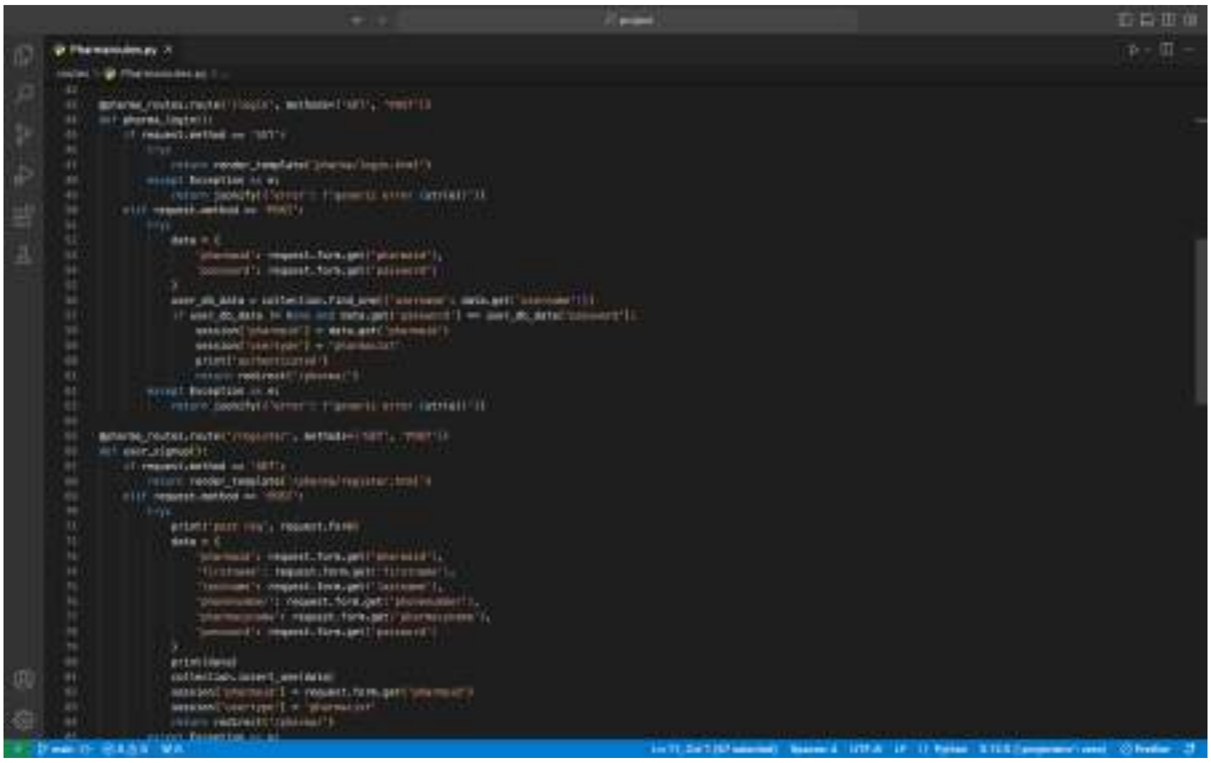
```python
@user_routes.route('/remarksandprescriptions')
def get_remarks_and_prescriptions():
    try:
        user_data = list(appointments_collection.find({'$and': [
            {'username': session.get('username')},
            {'status': 'DONE'}
        ]}))
        changeObjectId(user_data)
        for item in user_data:
            date_obj = datetime.strptime(item['date'], '%Y-%m-%d')
            item['displaydate'] = date_obj.strftime('%d-%m-%Y')
        data = {
            'remarksAndPrescriptions': user_data
        }
        return render_template('remarksandprescriptions.html', data=data)
    except Exception as e:
        return jsonify({'error': f'{e}'})
```

4.3 Pharma Routes

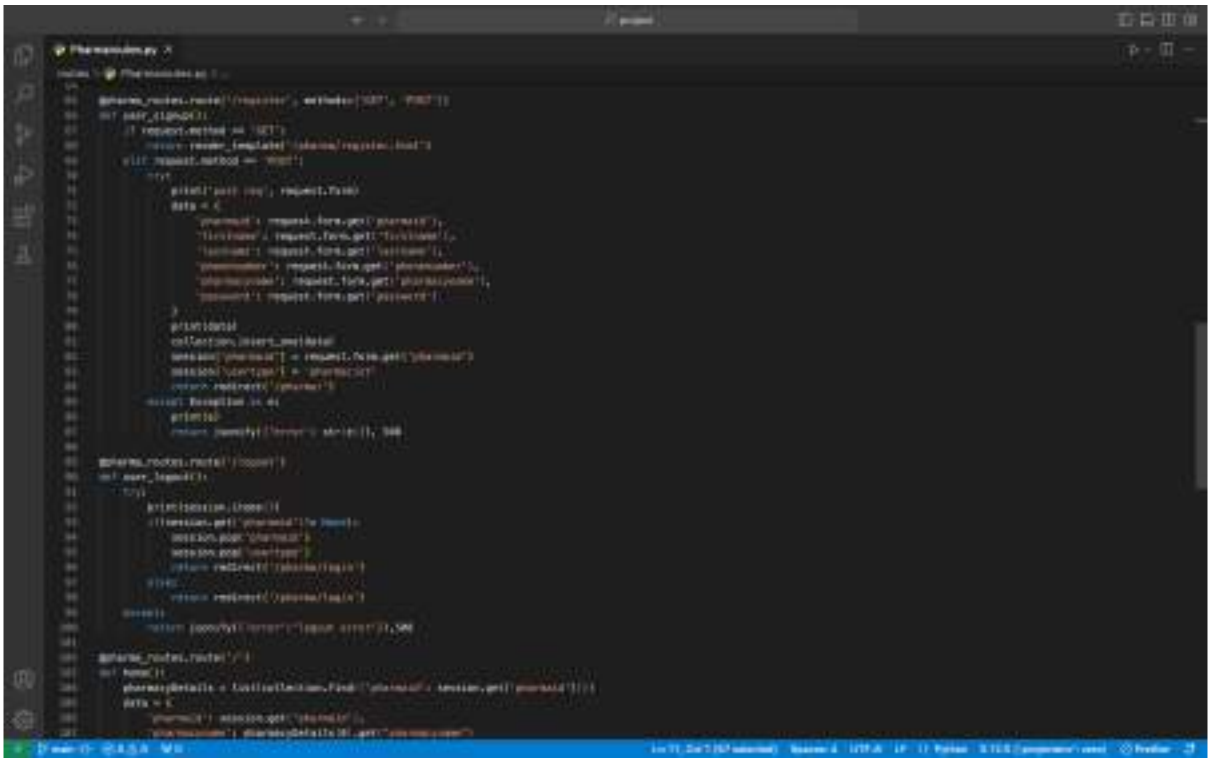
This is a python file that contains all the functions related to pharma flow. Below is the code for the same.



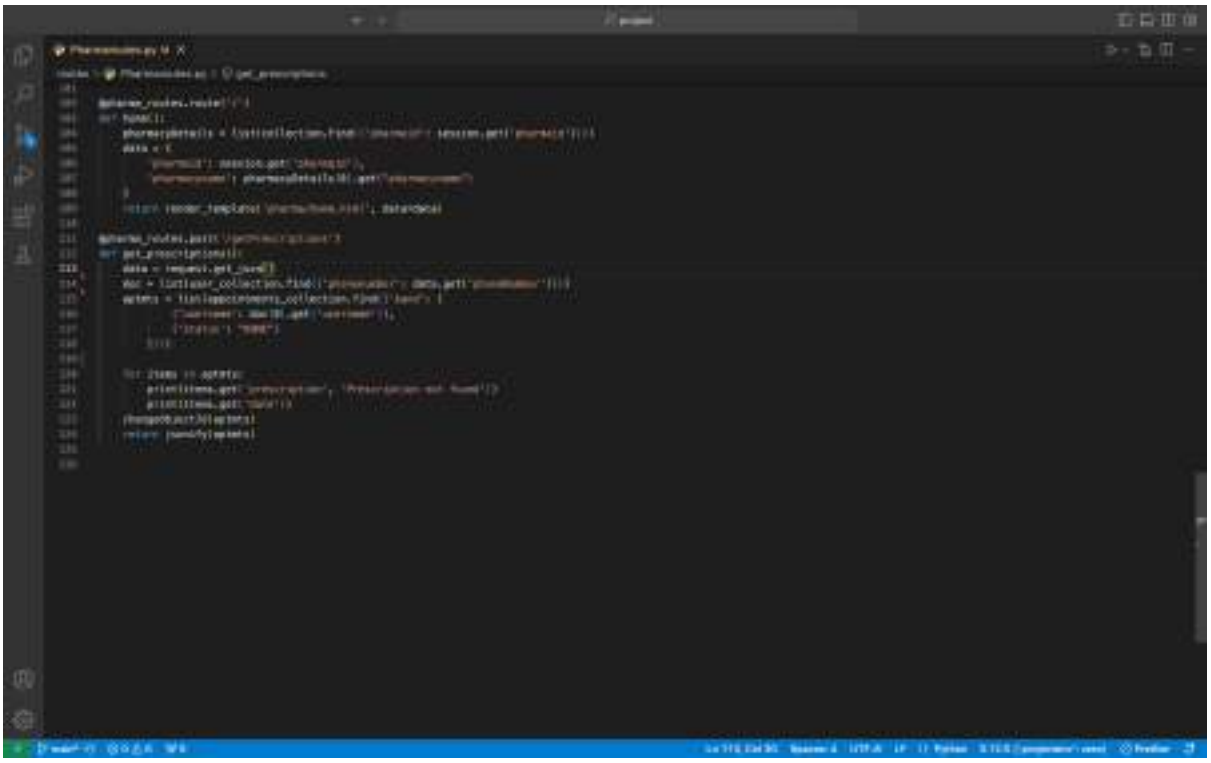
```python
from flask import Blueprint, jsonify, request, render_template, session, redirect

from mongoclient import db

pharma_routes = Blueprint('pharma', __name__)

collection = db['pharma']
user_collection = db['user']
appointments_collection = db['appointments']


def isPharmaAuthenticated():
    if session.get('pharmaId') != None and collection.find_one({'pharmaId': session.get('pharmaId')}) != None:
        return True
    return False

def identifyTypeOfUser():
    return session.get('userType')

def changeObjectId(data):
    for item in data:
        item['_id'] = str(item['_id'])


@pharma_routes.before_request
def isAuthenticated():
    try:
        if (request.path in ['/pharma/test']):
            return
        if (request.path == '/pharma/login' or request.path == '/pharma/register') and request.method == 'GET':
            if isPharmaAuthenticated():
                user_Type = identifyTypeOfUser()
                if user_type == 'pharmacist':
                    return redirect('/pharma/')
        else:
            if (request.path != '/pharma/login' and request.path != '/pharma/register') and not isPharmaAuthenticated():
                return redirect('/pharma/login')
    except Exception as e:
        return jsonify({'error': f'{str(e)}'})

@pharma_routes.route('/login', methods=['GET', 'POST'])
def pharma_login():
    if request.method == 'GET':
```



```
@pharma_routes.route('/login', methods=['GET', 'POST'])
def pharma_login():
    if request.method == 'GET':
        try:
            return render_template('pharma/login.html')
        except Exception as e:
            return jsonify({'error': f'generic error {e}'})
    elif request.method == 'POST':
        try:
            data = {
                'pharmaid': request.form.get('pharmaid'),
                'password': request.form.get('password')
            }
            user_db_data = collection.find_one({'username': data.get('username')})
            if user_db_data != None and data.get('password') == user_db_data['password']:
                session['pharmaid'] = data.get('pharmaid')
                session['usertype'] = 'pharmacist'
                print('authenticated')
                return redirect('/pharma/')
        except Exception as e:
            return jsonify({'error': f'generic error {e}'})

@pharma_routes.route('/register', methods=['GET', 'POST'])
def user_signup():
    if request.method == 'GET':
        return render_template('pharma/register.html')
    elif request.method == 'POST':
        try:
            print('post req', request.form)
            data = {
                'pharmaid': request.form.get('pharmaid'),
                'firstname': request.form.get('firstname'),
                'lastname': request.form.get('lastname'),
                'phonenumber': request.form.get('phonenumber'),
                'pharmacyname': request.form.get('pharmacyname'),
                'password': request.form.get('password')
            }
            print(data)
            collection.insert_one(data)
            session['pharmaid'] = request.form.get('pharmaid')
            session['usertype'] = 'pharmacist'
            return redirect('/pharma/')
        except Exception as e:
```



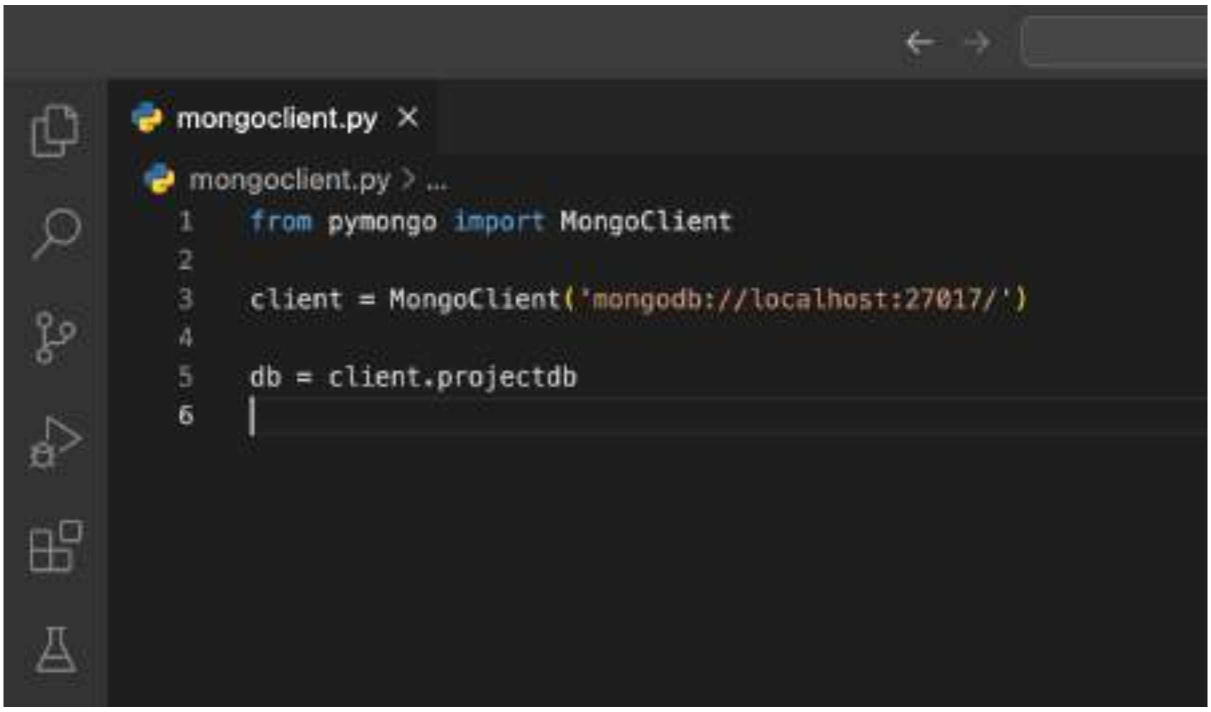
```
@pharma_routes.route('/register', methods=['GET', 'POST'])
def user_signup():
    if request.method == 'GET':
        return render_template('pharma/register.html')
    elif request.method == 'POST':
        try:
            print('post req', request.form)
            data = {
                'pharmaid': request.form.get('pharmaid'),
                'firstname': request.form.get('firstname'),
                'lastname': request.form.get('lastname'),
                'phonenumber': request.form.get('phonenumber'),
                'pharmacyname': request.form.get('pharmacyname'),
                'password': request.form.get('password')
            }
            print(data)
            collection.insert_one(data)
            session['pharmaid'] = request.form.get('pharmaid')
            session['usertype'] = 'pharmacist'
            return redirect('/pharma/')
        except Exception as e:
            print(e)
            return jsonify({'error': str(e)}), 500

@pharma_routes.route('/logout')
def user_logout():
    try:
        print(session.items())
        if(session.get('pharmaid')!= None):
            session.pop('pharmaid')
            session.pop('usertype')
            return redirect('/pharma/login')
        else:
            return redirect('/pharma/login')
    except:
        return jsonify({'error':'logout error'}),500

@pharma_routes.route('/')
def home():
    pharmacyDetails = list(collection.find({'pharmaid': session.get('pharmaid')}))
    data = {
        'pharmaid': session.get('pharmaid'),
        'pharmacyname': pharmacyDetails[0].get('pharmacyname')
```



```
@pharma_routes.route('/')
def home():
    pharmacyDetails = list(collection.find({'pharmaId': session.get('pharmaId')}))
    data = {
        'pharmaId': session.get('pharmaId'),
        'pharmacyname': pharmacyDetails[0].get('pharmacyname')
    }
    return render_template('pharma/home.html', data=data)

@pharma_routes.post('/getPrescriptions')
def get_prescriptions():
    data = request.get_json()
    doc = list(user_collection.find({'phonenumber': data.get('phoneNumber')}))
    apts = list(appointments_collection.find({'$and': [
            {'username': doc[0].get('username')},
            {'status': 'DONE'}
        ]}))

    for items in apts:
        print(items.get('prescription', 'Prescription not found'))
        print(items.get('date'))
    [illegible](apts)
    return jsonify(apts)
```

5. Chapter 5

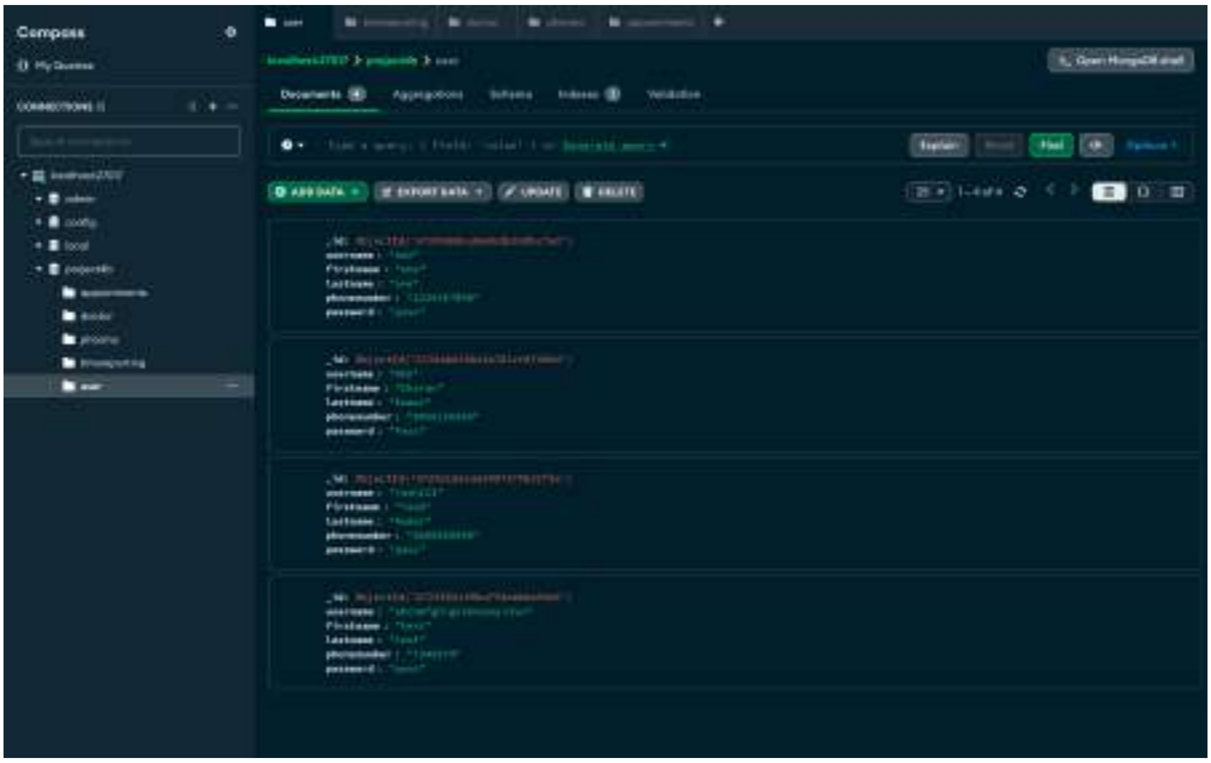
This chapter discusses about the connectivity of the mongo DB.



```
from pymongo import MongoClient

client = MongoClient('mongodb://localhost:27017/')

db = client.projectdb

```

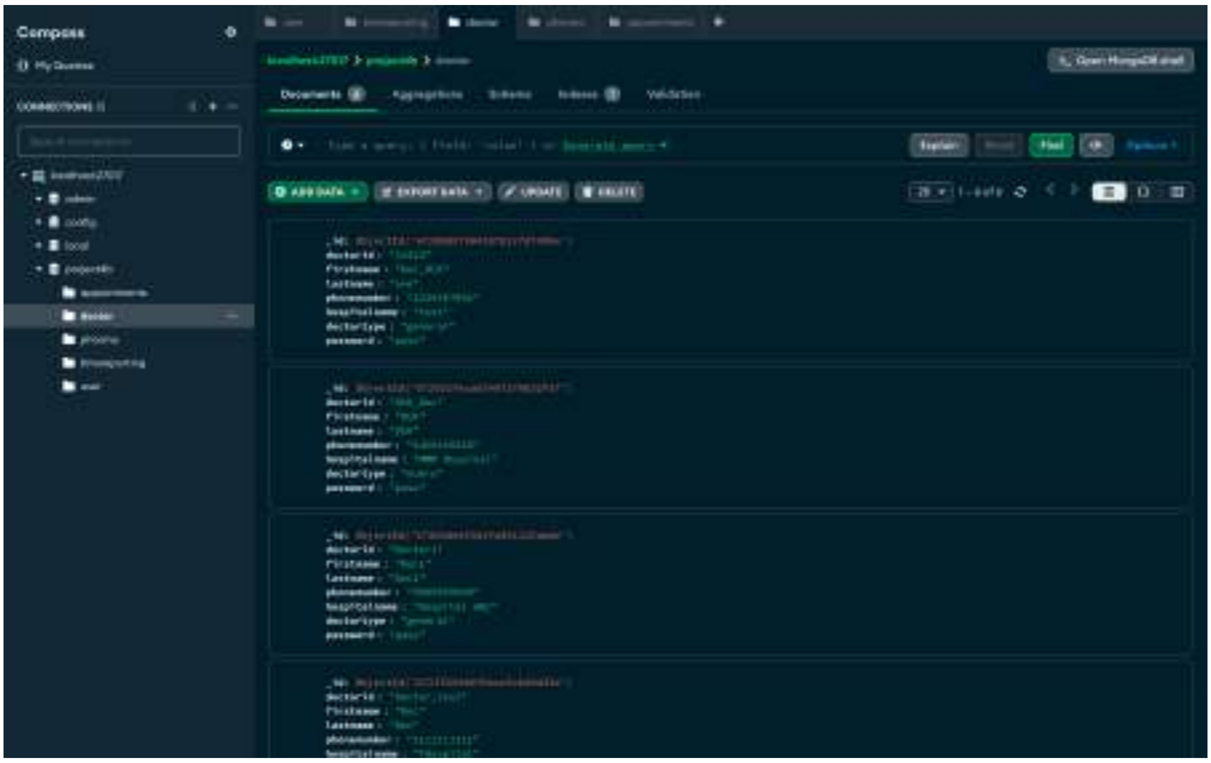
Here ‘mongodb://localhost:27017’ is the port where we can access the DB.

5.1 User Database collection



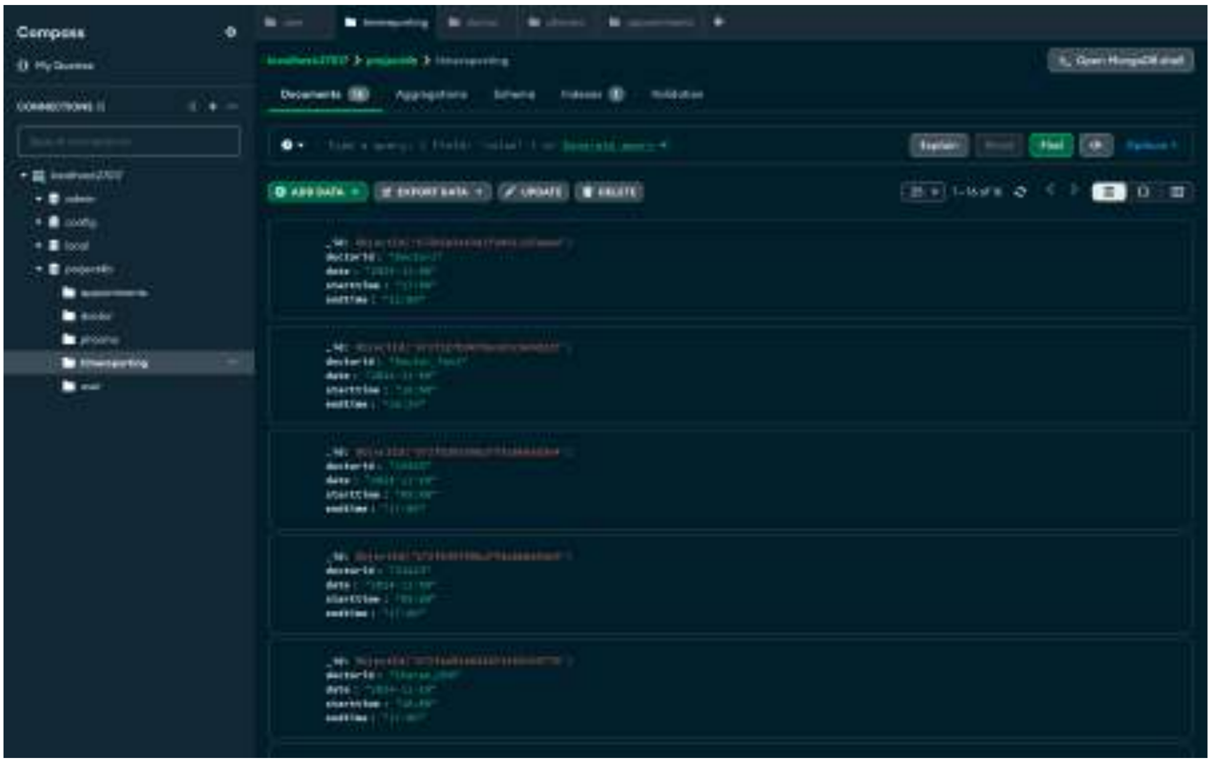
```
_id: ObjectId('[illegible]')
username : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
password : "[illegible]"

_id: ObjectId('[illegible]')
username : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
password : "[illegible]"

_id: ObjectId('[illegible]')
username : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
password : "[illegible]"

_id: ObjectId('[illegible]')
username : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
password : "[illegible]"
```

5.2 Doctor Database collection



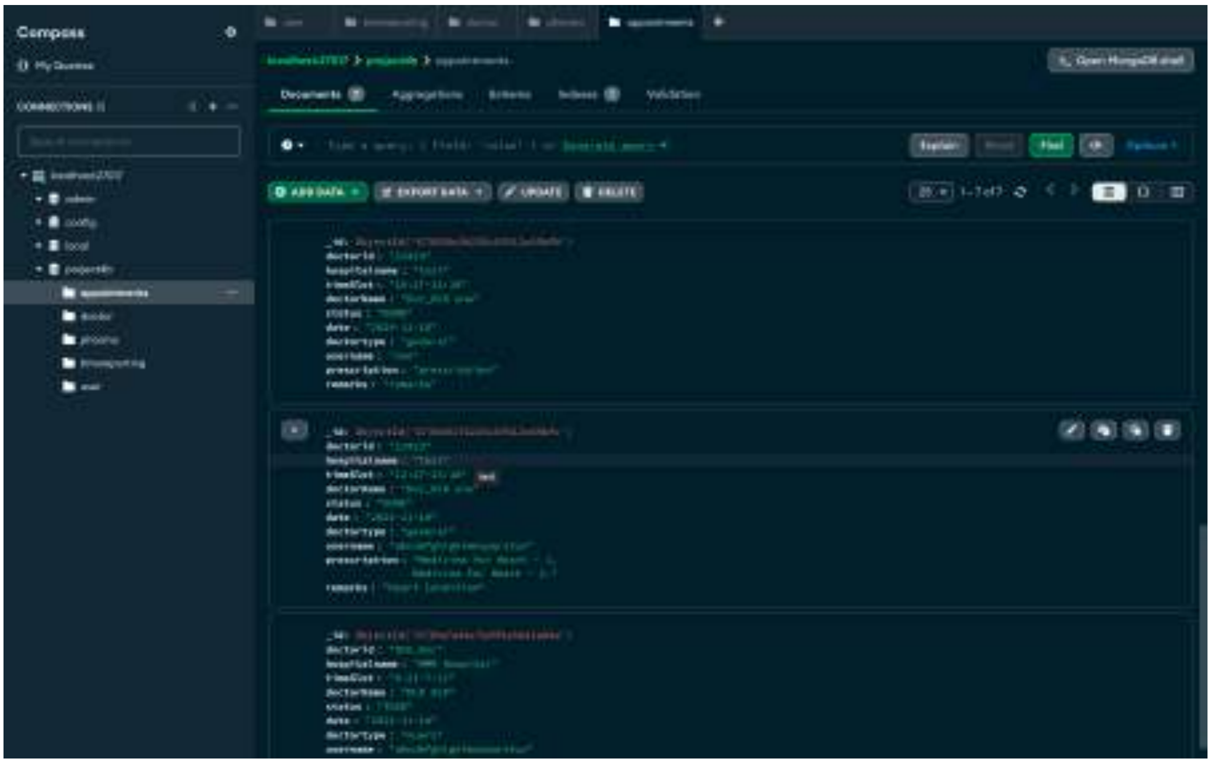
```
_id: ObjectId('[illegible]')
doctorId : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
hospitalname : "[illegible]"
doctortype : "[illegible]"
password : "[illegible]"

_id: ObjectId('[illegible]')
doctorId : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
hospitalname : "[illegible]"
doctortype : "[illegible]"
password : "[illegible]"

_id: ObjectId('[illegible]')
doctorId : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
hospitalname : "[illegible]"
doctortype : "[illegible]"
password : "[illegible]"

_id: ObjectId('[illegible]')
doctorId : "[illegible]"
firstname : "[illegible]"
lastname : "[illegible]"
phonenumber : "[illegible]"
hospitalname : "[illegible]"
```

5.3 Reporting Time Database collection



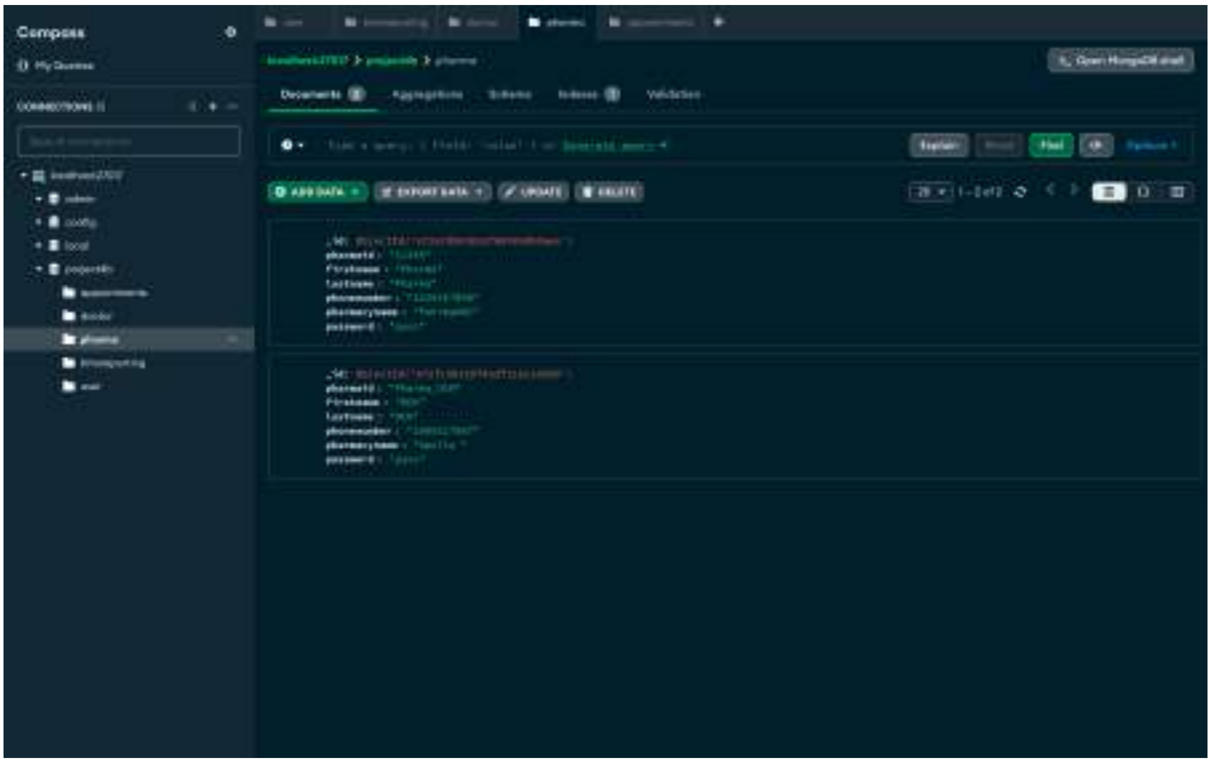
```
_id: ObjectId('[illegible]')
doctorId: "[illegible]"
date: "[illegible]"
starttime: "[illegible]"
endtime: "[illegible]"

_id: ObjectId('[illegible]')
doctorId: "[illegible]"
date: "[illegible]"
starttime: "[illegible]"
endtime: "[illegible]"

_id: ObjectId('[illegible]')
doctorId: "[illegible]"
date: "[illegible]"
starttime: "[illegible]"
endtime: "[illegible]"

_id: ObjectId('[illegible]')
doctorId: "[illegible]"
date: "[illegible]"
starttime: "[illegible]"
endtime: "[illegible]"

_id: ObjectId('[illegible]')
doctorId: "[illegible]"
date: "[illegible]"
starttime: "[illegible]"
endtime: "[illegible]"
```

5.4 Appointments Database collection



```
_id: ObjectId('[illegible]')
doctorId: "[illegible]"
hospitalname: "[illegible]"
timeSlot: "[illegible]"
doctorName: "[illegible]"
status: "[illegible]"
date: "[illegible]"
doctortype: "[illegible]"
username: "[illegible]"
prescription: "[illegible]"
remarks: "[illegible]"

_id: ObjectId('[illegible]')
doctorId: "[illegible]"
hospitalname: "[illegible]"
timeSlot: "[illegible]"
doctorName: "[illegible]"
status: "[illegible]"
date: "[illegible]"
doctortype: "[illegible]"
username: "[illegible]"
prescription: "[illegible]"
remarks: "[illegible]"

_id: ObjectId('[illegible]')
doctorId: "[illegible]"
hospitalname: "[illegible]"
timeSlot: "[illegible]"
doctorName: "[illegible]"
status: "[illegible]"
date: "[illegible]"
doctortype: "[illegible]"
username: "[illegible]"
```

5.5 Pharma Database collection

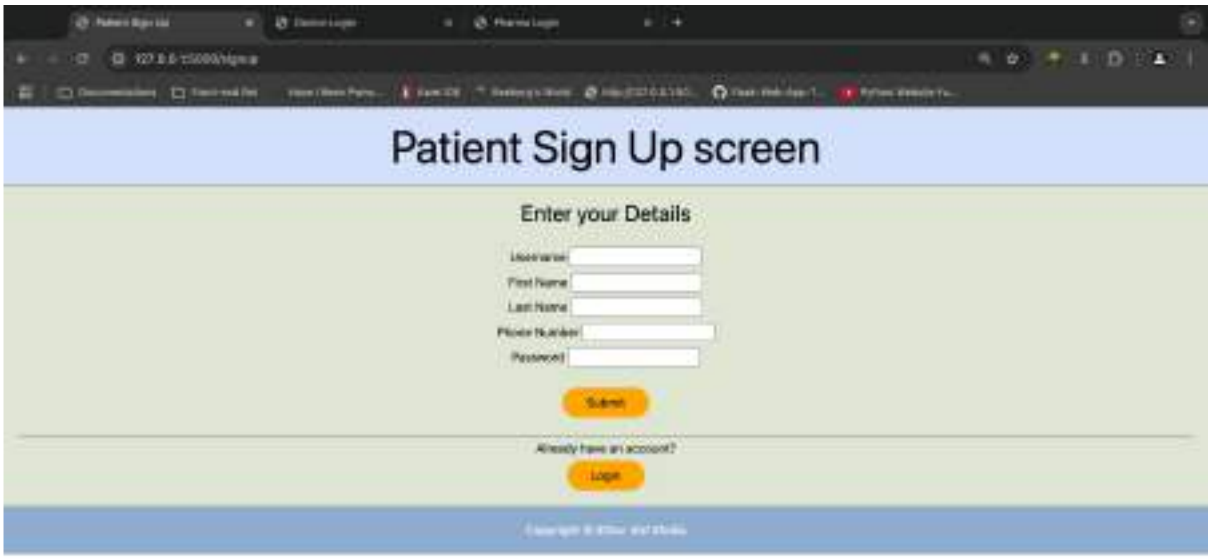


```
_id: [illegible]
pharmaId: [illegible]
firstname: [illegible]
lastname: [illegible]
phonenumber: [illegible]
pharmacyname: [illegible]
password: [illegible]

_id: [illegible]
pharmaId: [illegible]
firstname: [illegible]
lastname: [illegible]
phonenumber: [illegible]
pharmacyname: [illegible]
password: [illegible]
```

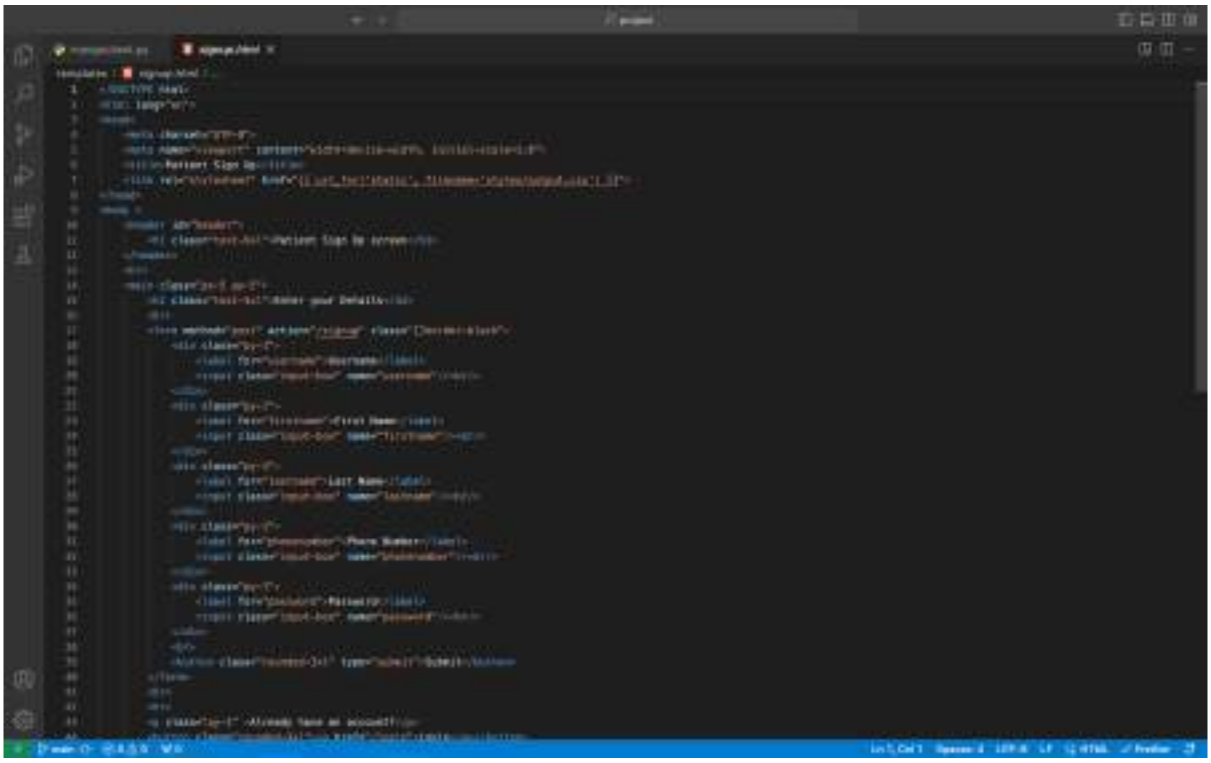
6 Chapter 6

6.1 User Sign-Up screen

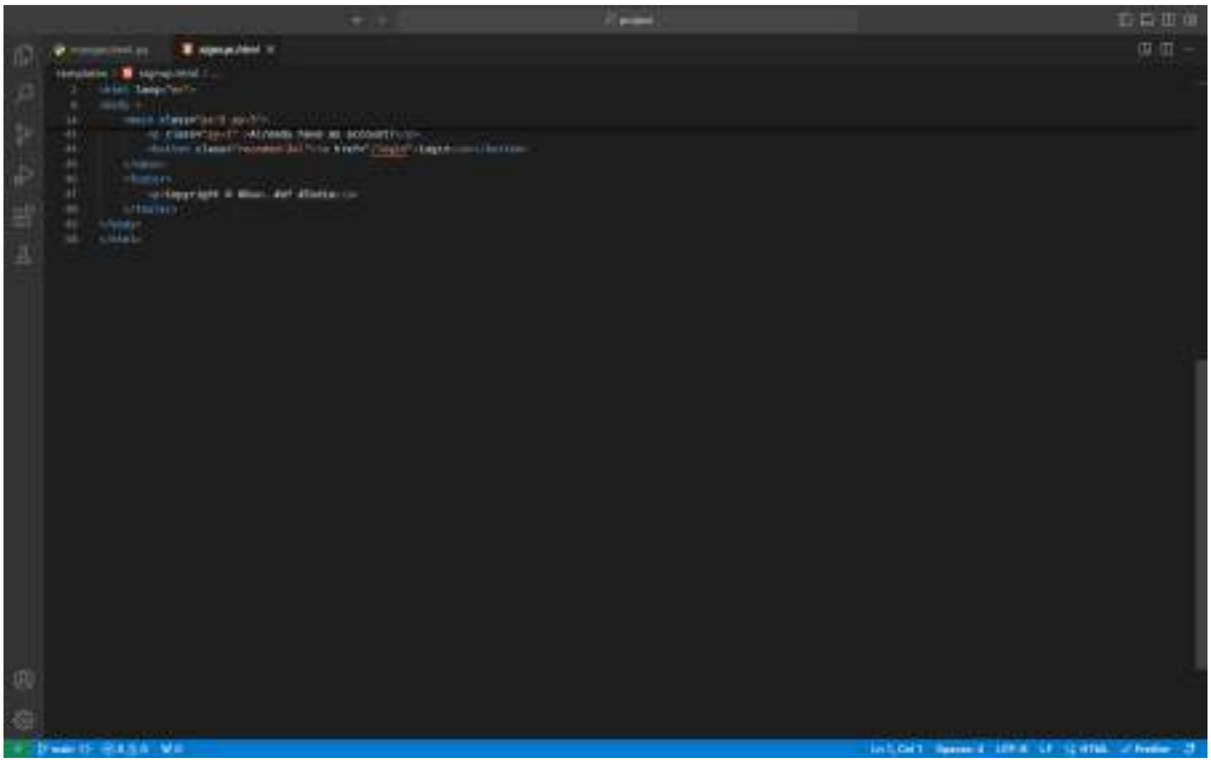
User can sign up here by providing the details on this screen. If the user already has an account, they can click on the Login button and login there.



Code for the above screen



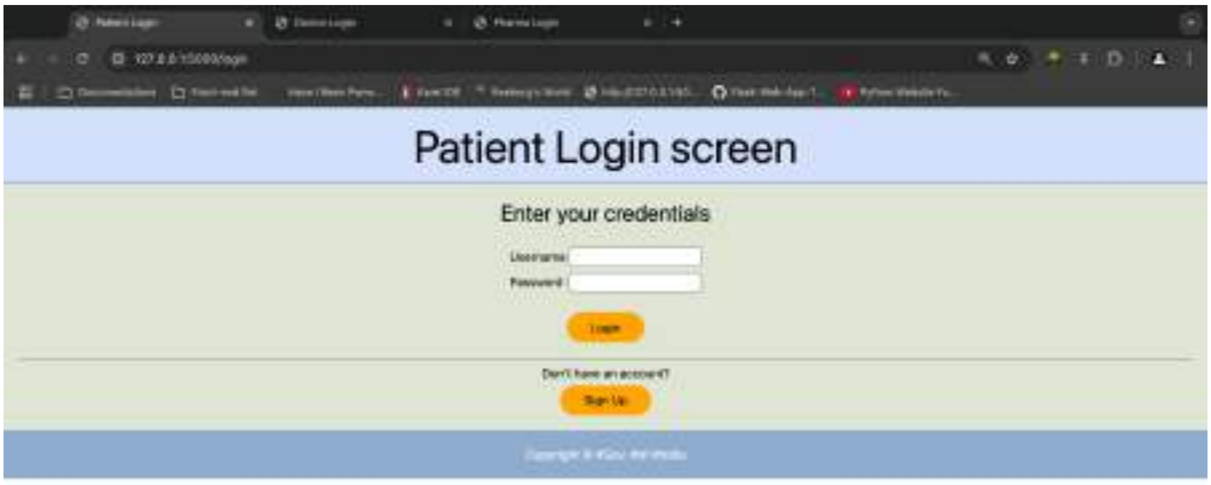
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Patient Sign Up</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='[illegible]') }}">
</head>
<body>
    <header id="header">
        <h1 class="[illegible]">Patient Sign Up screen</h1>
    </header>
    <hr>
    <main class="[illegible]">
        <h2 class="[illegible]">Enter your Details</h2>
        <hr>
        <form method="post" action="/signup" class="[illegible]">
            <div class="[illegible]">
                <label for="username">Username</label>
                <input class="input-box" name="username"><br>
            </div>
            <div class="[illegible]">
                <label for="firstname">First Name</label>
                <input class="input-box" name="firstname"><br>
            </div>
            <div class="[illegible]">
                <label for="lastname">Last Name</label>
                <input class="input-box" name="lastname"><br>
            </div>
            <div class="[illegible]">
                <label for="phonenumber">Phone Number</label>
                <input class="input-box" name="phonenumber"><br>
            </div>
            <div class="[illegible]">
                <label for="password">Password</label>
                <input class="input-box" name="password"><br>
            </div>
            <br>
            <button class="[illegible]" type="submit">Submit</button>
        </form>
        <hr>
        <hr>
        <p class="[illegible]">Already have an account?</p>
```



```
<html lang="en">
<body>
    <main class="[illegible]">
        <p class="[illegible]">Already have an account?</p>
        <button class="[illegible]"><a href="/login">Login</a></button>
    </main>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
</html>
```

6.2 User Login screen

User can log in here by providing the details on this screen. If the user doesn't have an account, they will have to click on the Sign Up button and login there.



Patient Login screen

Enter your credentials

Username

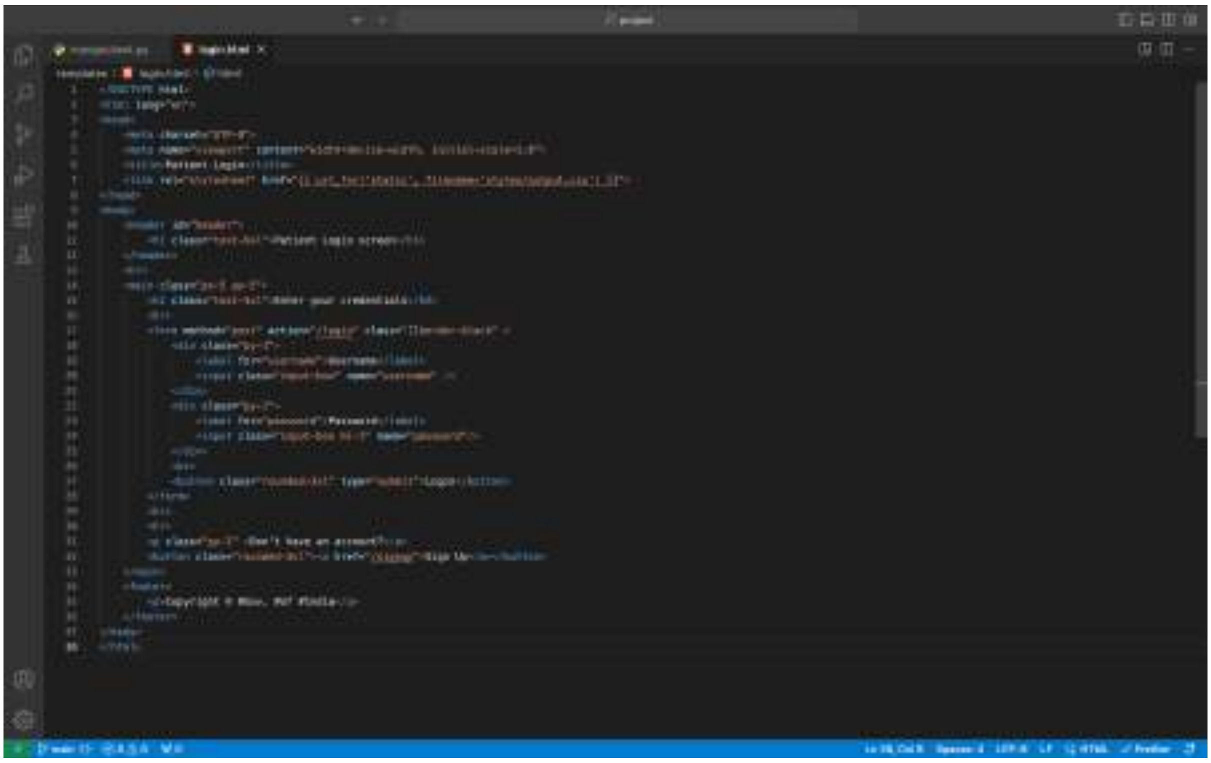
Password

Login

Don't have an account?

Sign Up

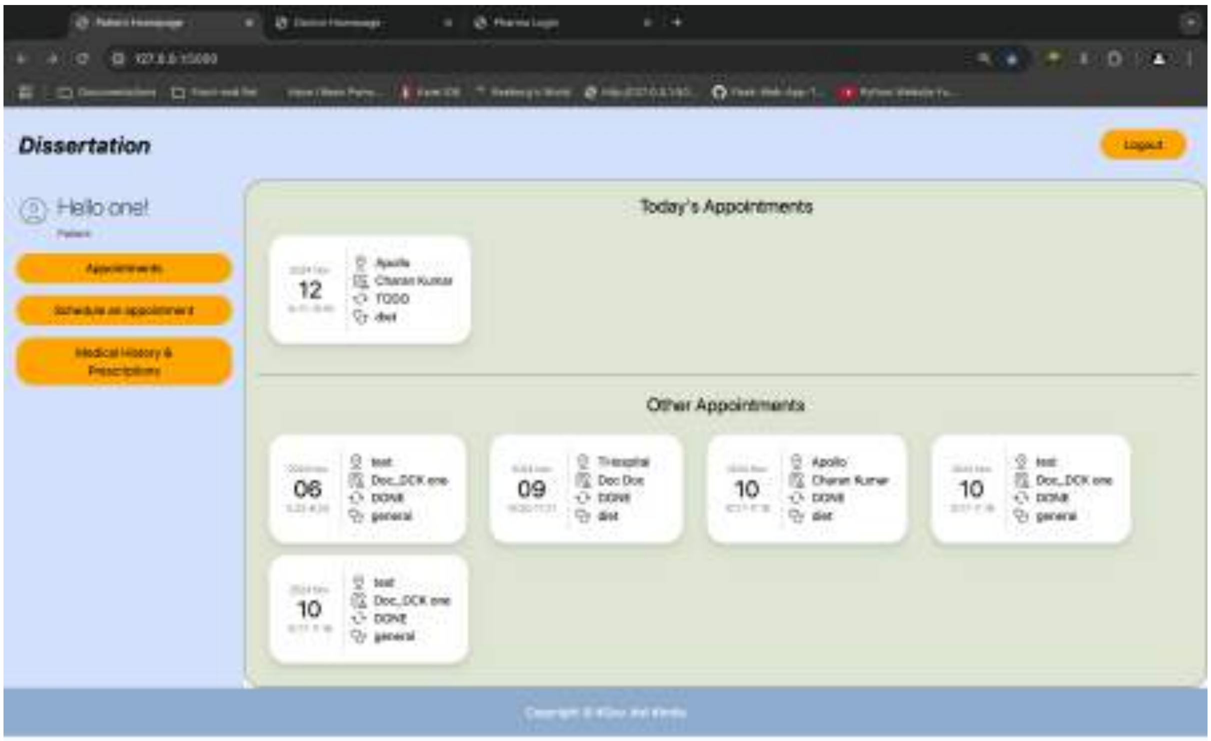
Code for the above screen



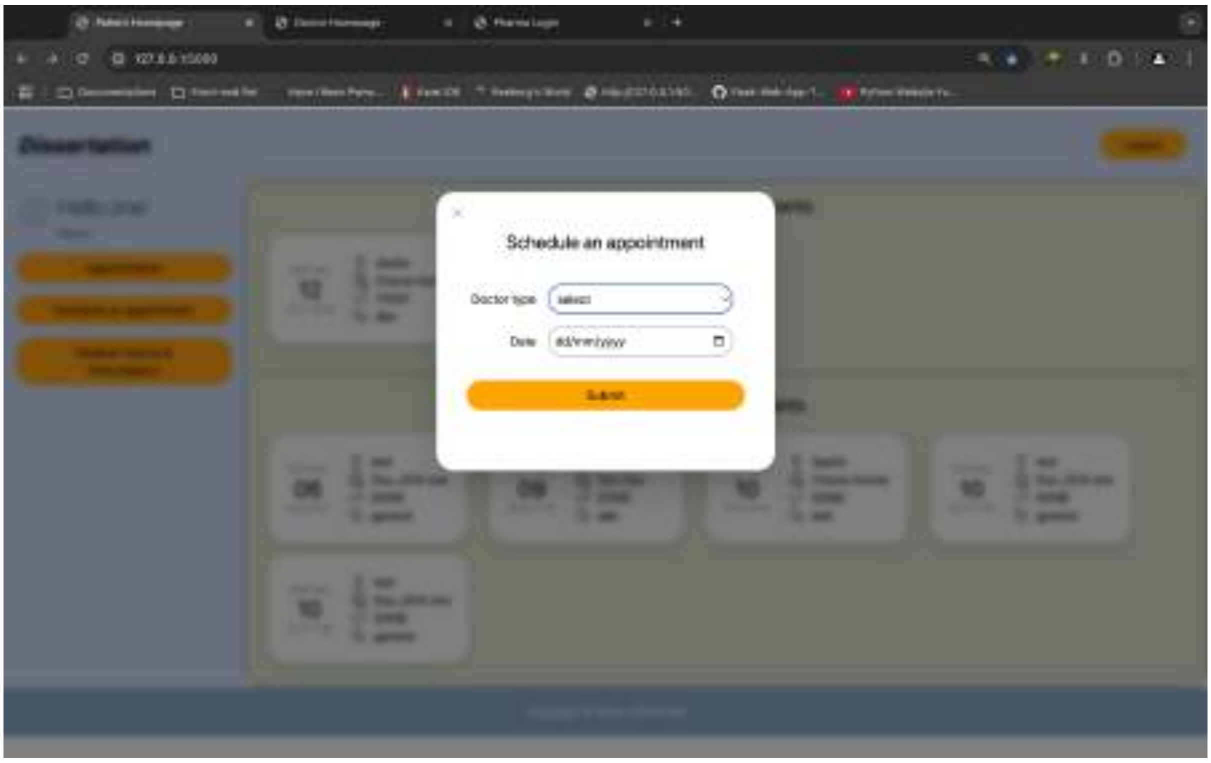
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Patient Login</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles/output.css') }}">
</head>
<body>
    <header id="header">
        <h1 class="text-4xl">Patient login screen</h1>
    </header>
    <br>
    <main class="px-5 py-5">
        <h2 class="text-xl">Enter your credentials</h2>
        <br>
        <form method="post" action="/login" class="[illegible]" >
            <div class="py-1">
                <label for="username">Username</label>
                <input class="input-box" name="username" />
            </div>
            <div class="py-1">
                <label for="password">Password</label>
                <input class="input-box ml-1" name="password"/>
            </div>
            <br>
            <button class="rounded-btn" type="submit">Login</button>
        </form>
        <br>
        <hr>
        <p class="py-1">Don't have an account?</p>
        <button class="rounded-btn"><a href="/signup">Sign Up</a></button>
    </main>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
</html>
```

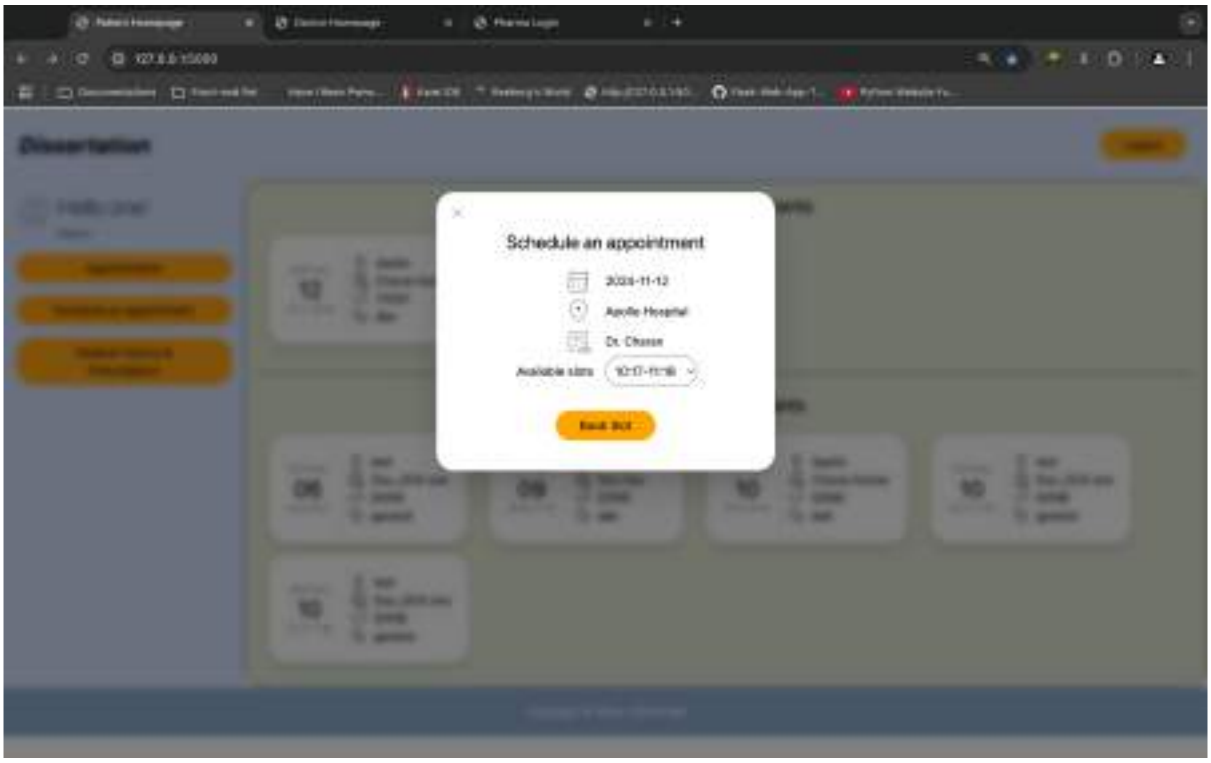
6.3 User Dashboard screen

On the Dashboard, user can view his existing and past appointments. Has an option to schedule appointments and view Medical History and Prescriptions.

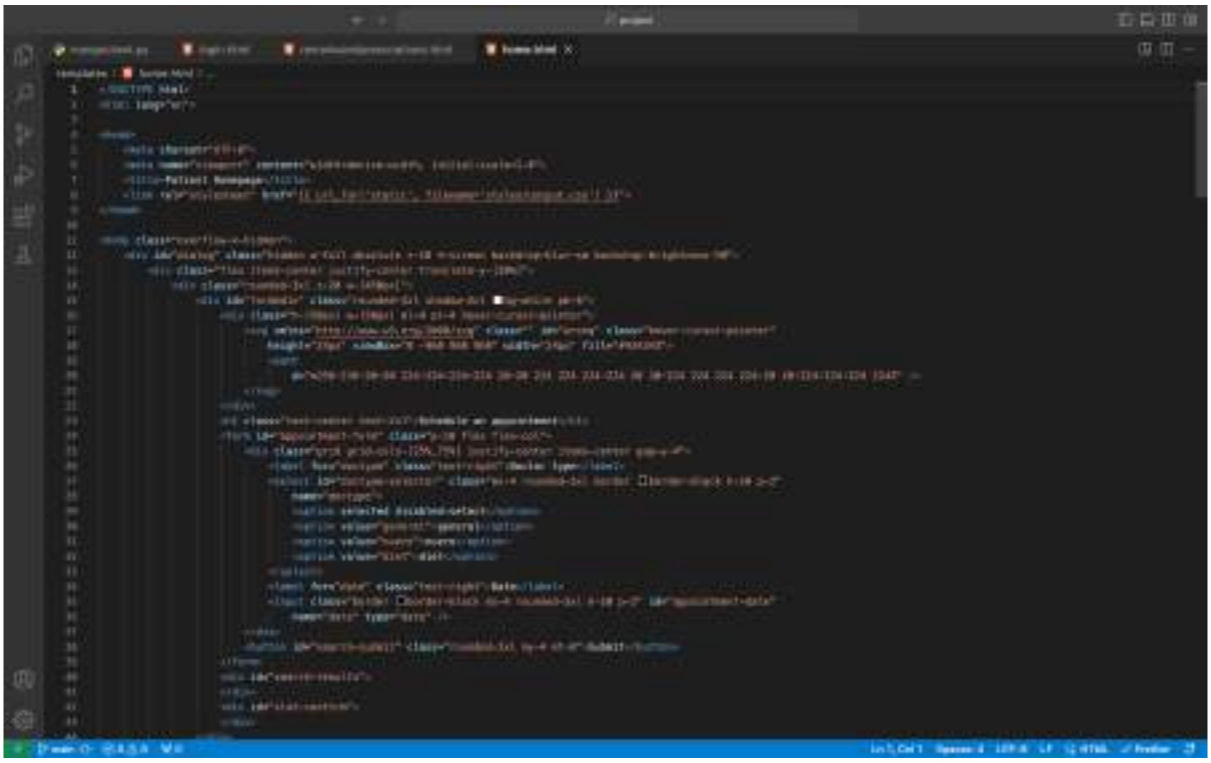


6.3.1 Appointments schedule page

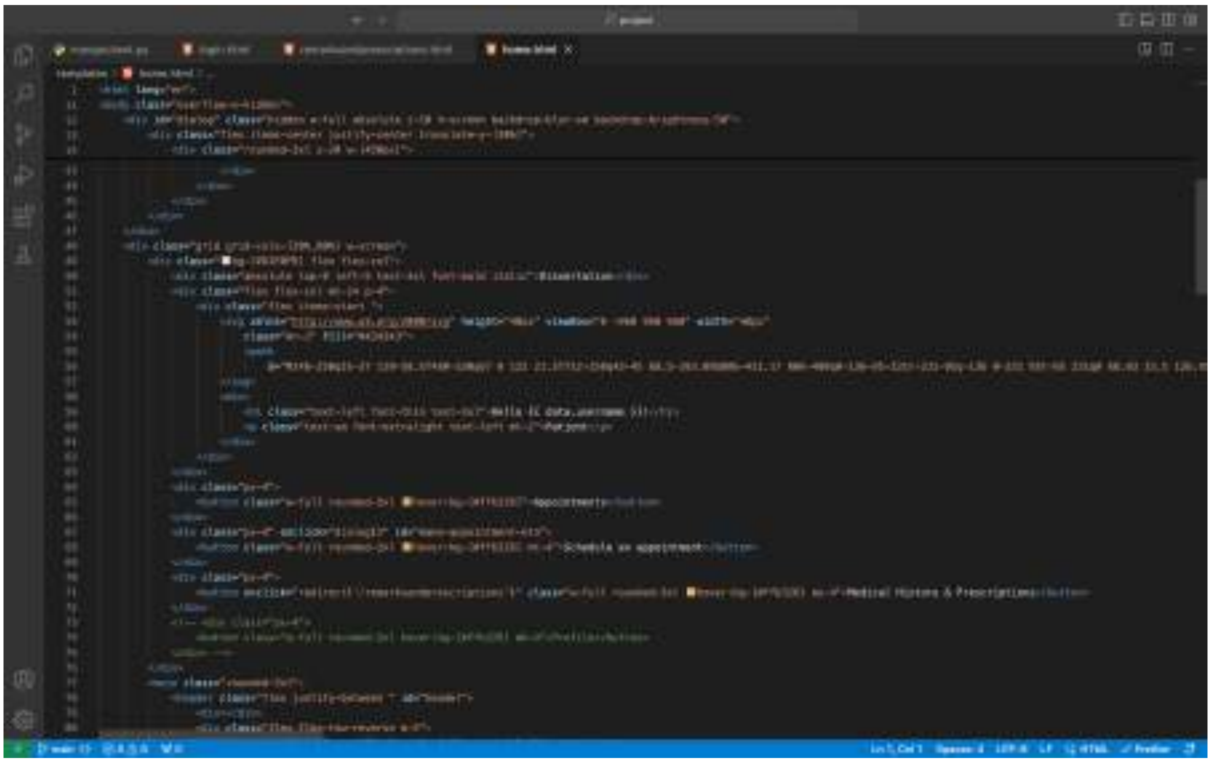




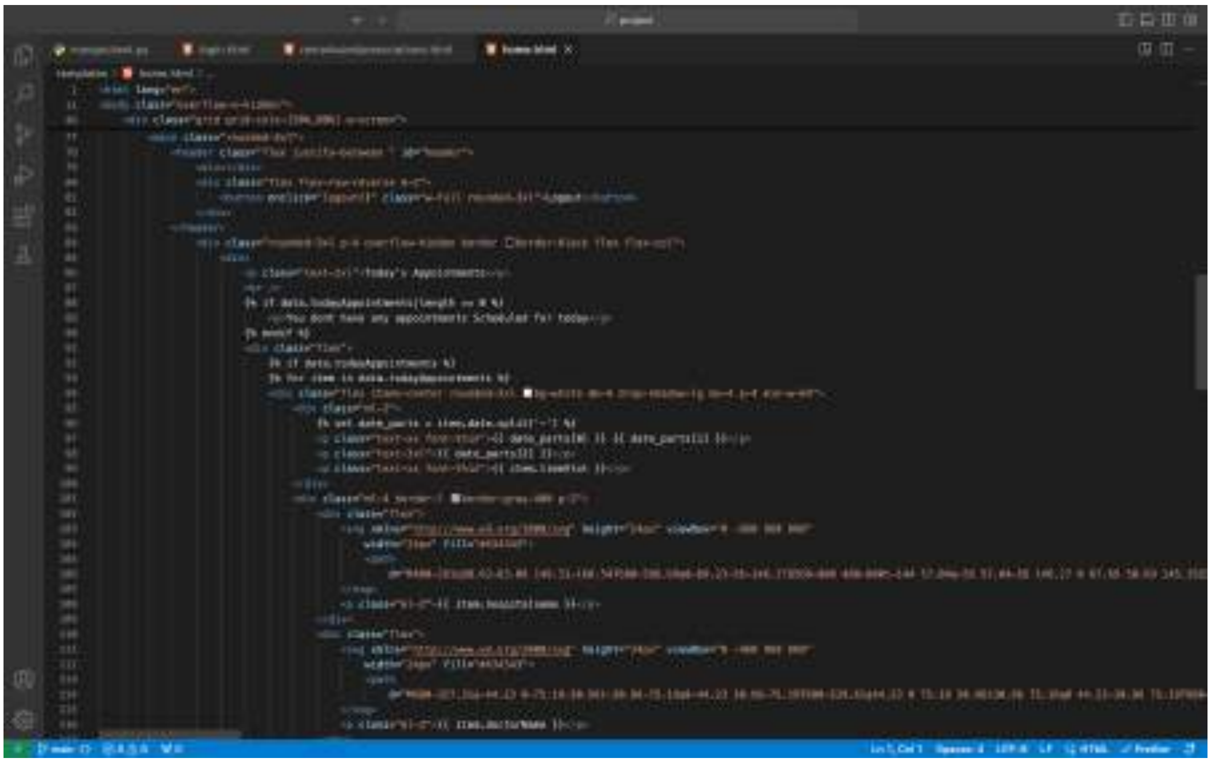
Code for the above screen



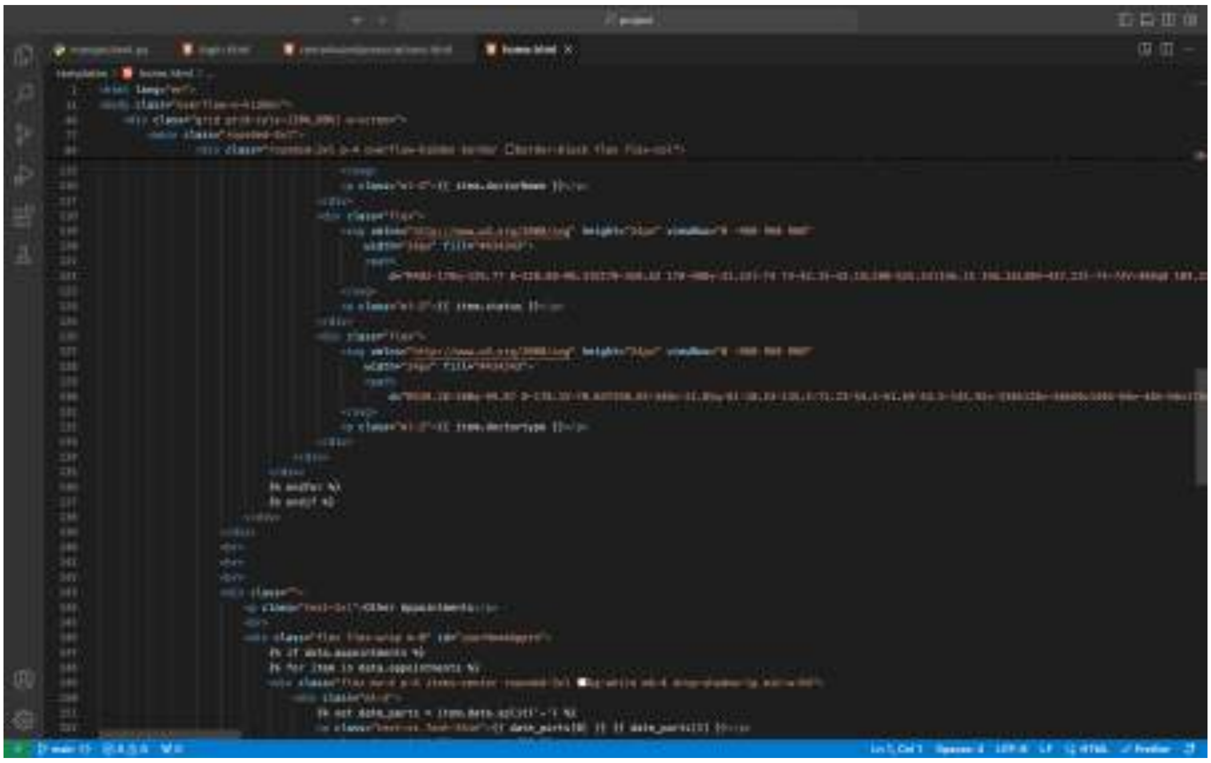
```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Patient Homepage</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='[illegible]') }}">
</head>

<body class="[illegible]">
    <div id="[illegible]" class="[illegible]">
        <div class="flex items-center justify-center [illegible]">
            <div class="[illegible]">
                <div id="[illegible]" class="[illegible]">
                    <div class="[illegible]">
                        <svg xmlns="http://www.w3.org/2000/svg" class="" id="[illegible]" class="[illegible]"
                            height="[illegible]" viewBox="[illegible]" width="[illegible]" fill="[illegible]">
                            <path
                                d="[illegible]" />
                        </svg>
                    </div>
                    <h2 class="[illegible]">Schedule an appointment</h2>
                    <form id="appointment-form" class="[illegible]">
                        <div class="[illegible]">
                            <label for="doctype" class="text-right">Doctor type</label>
                            <select id="[illegible]" class="[illegible]"
                                name="doctype">
                                <option selected disabled>select</option>
                                <option value="general">general</option>
                                <option value="[illegible]">[illegible]</option>
                                <option value="diet">diet</option>
                            </select>
                            <label for="date" class="text-right">Date</label>
                            <input class="[illegible]" id="appointment-date"
                                name="date" type="date" />
                        </div>
                        <button id="[illegible]" class="[illegible]">Submit</button>
                    </form>
                    <div id="search-results">
                    </div>
                    <div id="[illegible]">
                    </div>
```



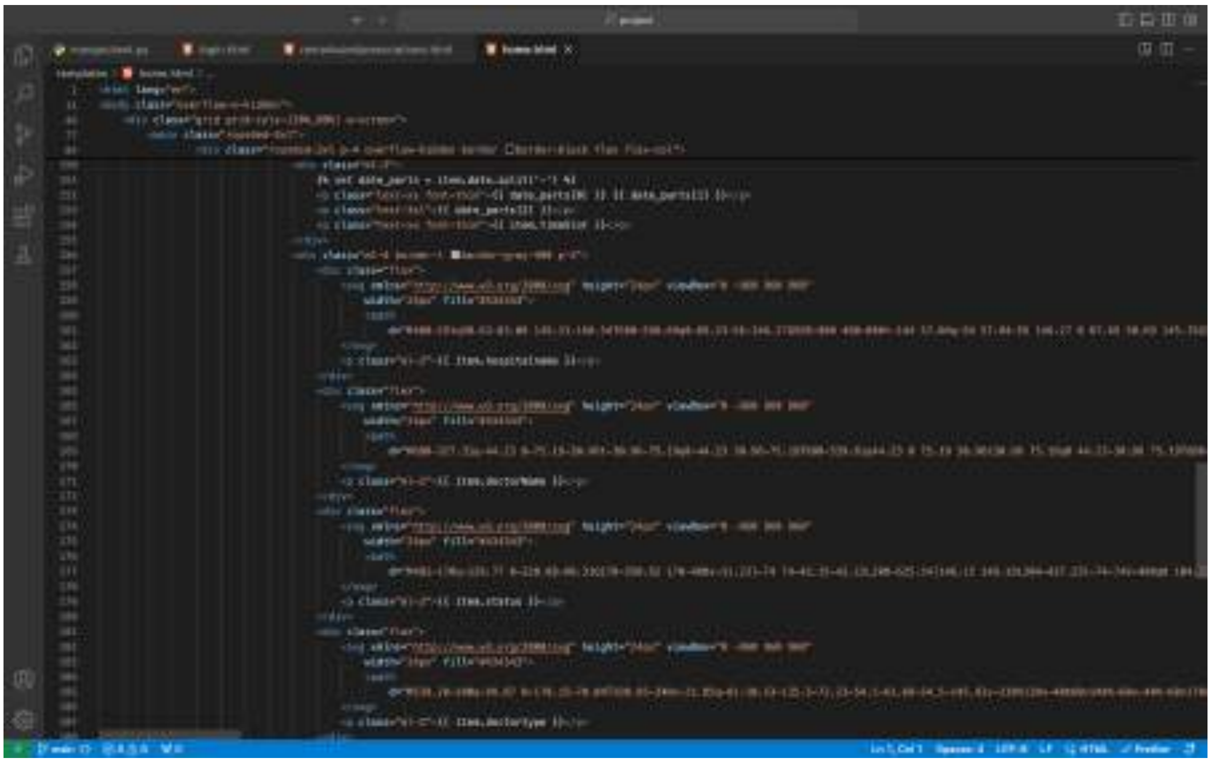
```
<html lang="en">
<body class="overflow-x-hidden">
    <div id="dialog" class="[illegible]">
        <div class="flex items-center justify-center [illegible]">
            <div class="rounded-2xl [illegible]">
                </div>
            </div>
        </div>
    </div>
    </div>
    <div class="grid grid-cols-[illegible] h-screen">
        <div class="bg-[illegible] flex flex-col">
            <div class="absolute top-0 left-0 [illegible]">Dissertation</div>
            <div class="flex flex-col [illegible]">
                <div class="flex items-start ">
                    <svg xmlns="http://www.w3.org/2000/svg" height="48px" viewBox="0 -960 960 960" width="48px"
                        class="[illegible]" fill="[illegible]">
                        <path
                            d="[illegible]"
                    </svg>
                    <div>
                        <h1 class="text-left [illegible]">Hello {{ data.username }}</h1>
                        <p class="[illegible]">Patient</p>
                    </div>
                </div>
            </div>
            <div class="px-4">
                <button class="w-full rounded-2xl [illegible]">Appointments</button>
            </div>
            <div class="px-4" onclick="[illegible]" id="[illegible]">
                <button class="w-full rounded-2xl [illegible]">Schedule an appointment</button>
            </div>
            <div class="px-4">
                <button onclick="[illegible]" class="w-full rounded-2xl [illegible]">Medical History & Prescriptions</button>
            </div>
            <!-- <div class="px-4">
                <button class="[illegible]">[illegible]</button>
            </div> -->
        </div>
        <main class="rounded-2xl">
            <header class="flex justify-between " id="header">
                <div></div>
                <div class="flex flex-row-reverse [illegible]">
```



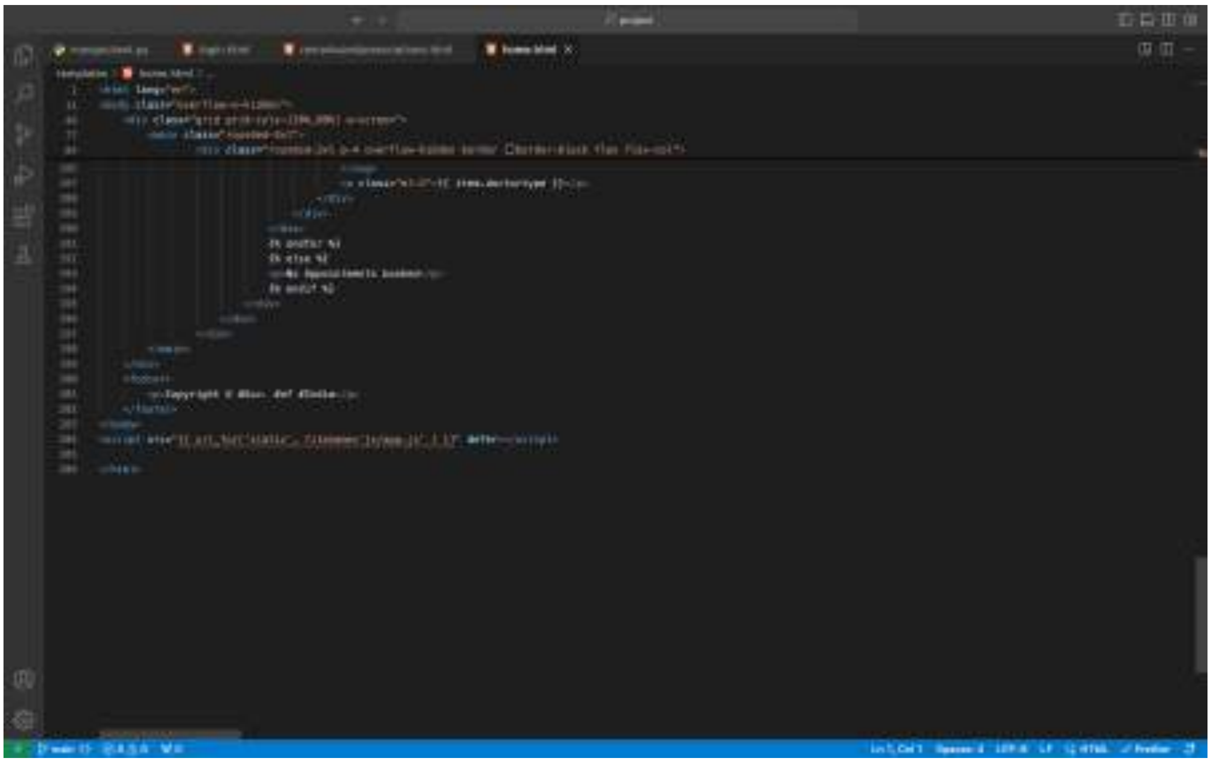
```
<html lang="en">
<body class="overflow-x-hidden">
    <div class="grid grid-cols-[illegible] h-screen">
        <main class="rounded-2xl">
            <header class="flex justify-between " id="header">
                <div></div>
                <div class="flex flex-row-reverse [illegible]">
                    <button onclick="logout()" class="w-full rounded-2xl">Logout</button>
                </div>
            </header>
                <div class="rounded-2xl p-4 overflow-hidden border [illegible] flex flex-col">
                <div>
                    <p class="text-2xl">Today's Appointments</p>
                    <hr />
                    {% if data.todayAppointments|length == 0 %}
                        <p>You dont have any appointments Scheduled for today</p>
                    {% endif %}
                    <div class="flex">
                        {% if data.todayAppointments %}
                        {% for item in data.todayAppointments %}
                        <div class="flex items-center rounded-2xl [illegible]">
                            <div class="[illegible]">
                                {% set date_parts = item.date.split('-') %}
                                <p class="[illegible]">{{ date_parts[0] }} {{ date_parts[1] }}</p>
                                <p class="[illegible]">{{ date_parts[2] }}</p>
                                <p class="[illegible]">{{ item.timeSlot }}</p>
                            </div>
                            <div class="[illegible] border-gray-400 [illegible]">
                                <div class="flex">
                                    <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                        width="24px" fill="[illegible]">
                                        <path
                                            d="[illegible]"
                                    </svg>
                                    <p class="[illegible]">{{ item.hospitalName }}</p>
                                </div>
                                <div class="flex">
                                    <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                        width="24px" fill="[illegible]">
                                        <path
                                            d="[illegible]"
                                    </svg>
                                    <p class="[illegible]">{{ item.doctorName }}</p>
```



```
<html lang="en">
<body class="overflow-x-hidden">
    <div class="grid grid-cols-[200,800] w-screen">
        <main class="[illegible]">
            <div class="[illegible] p-4 overflow-hidden border [illegible] flex flex-col">
                            </svg>
                            <p class="ml-2">{{ item.doctorName }}</p>
                        </div>
                        <div class="flex">
                            <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                width="24px" fill="#434343">
                                <path
                                    d="[illegible]
                            </svg>
                            <p class="ml-2">{{ item.status }}</p>
                        </div>
                        <div class="flex">
                            <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                width="24px" fill="#434343">
                                <path
                                    d="[illegible]
                            </svg>
                            <p class="ml-2">{{ item.doctortype }}</p>
                        </div>
                    </div>
                </div>
                {% endfor %}
                {% endif %}
            </div>
        </div>
        <br>
        <br>
        <br>
        <div class="">
            <p class="[illegible]">Other Appointments</p>
            <br>
            <div class="flex flex-wrap [illegible]" id="[illegible]">
                {% if data.appointments %}
                {% for item in data.appointments %}
                <div class="[illegible]">
                    <div class="[illegible]">
                        {% set date_parts = item.date.split('-') %}
                        <p class="[illegible]">{{ date_parts[0] }} {{ date_parts[1] }}</p>
```

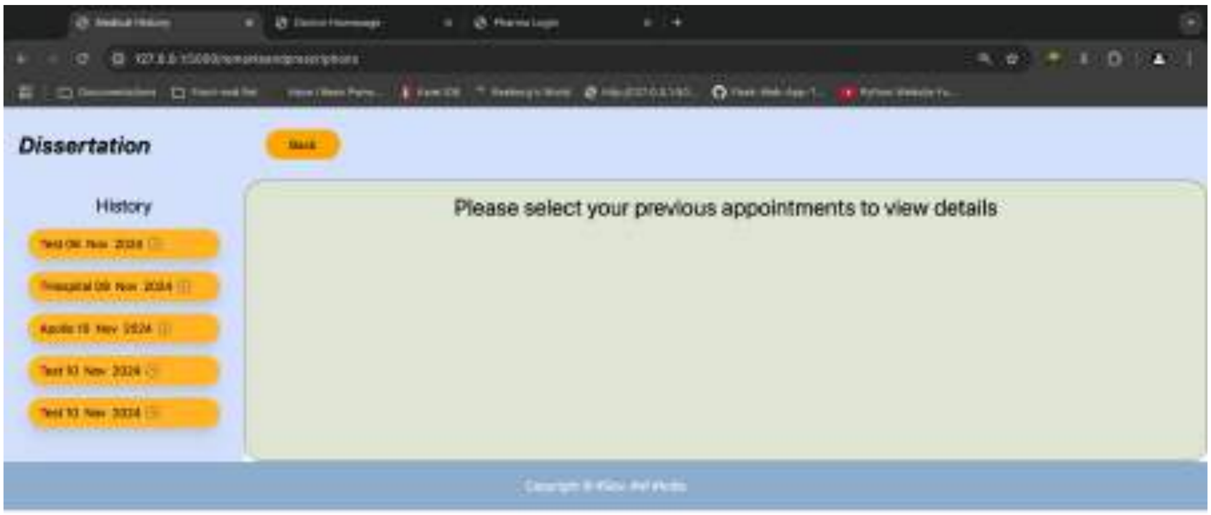


```
<html lang="en">
<body class="overflow-x-hidden">
    <div class="grid grid-cols-[200,800] w-screen">
        <main class="[illegible]">
            <div class="[illegible] p-4 overflow-hidden border [illegible] flex flex-col">
                    <div class="[illegible]">
                        {% set date_parts = item.date.split('-') %}
                        <p class="[illegible]">{{ date_parts[0] }} {{ date_parts[1] }}</p>
                        <p class="[illegible]">{{ date_parts[2] }}</p>
                        <p class="[illegible]">{{ item.timeslot }}</p>
                    </div>
                    <div class="[illegible]">
                        <div class="flex">
                            <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                width="24px" fill="#434343">
                                <path
                                    d="[illegible]
                            </svg>
                            <p class="ml-2">{{ item.hospitalname }}</p>
                        </div>
                        <div class="flex">
                            <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                width="24px" fill="#434343">
                                <path
                                    d="[illegible]
                            </svg>
                            <p class="ml-2">{{ item.doctorName }}</p>
                        </div>
                        <div class="flex">
                            <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                width="24px" fill="#434343">
                                <path
                                    d="[illegible]
                            </svg>
                            <p class="ml-2">{{ item.status }}</p>
                        </div>
                        <div class="flex">
                            <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                width="24px" fill="#434343">
                                <path
                                    d="[illegible]
                            </svg>
                            <p class="ml-2">{{ item.doctortype }}</p>
```



```
<html lang="en">
<body class="overflow-x-hidden">
    <div class="grid grid-rows-[illegible] h-screen">
        <main class="[illegible]">
            <div class="[illegible] p-4 overflow-hidden border border-black flex flex-col">
                                [illegible]
                                <p class="[illegible]">{{ item.doctortype }}</p>
                            </div>
                        </div>
                    </div>
                    {% endfor %}
                    {% else %}
                    <p>No Appointments booked.</p>
                    {% endif %}
                </div>
            </div>
        </div>
        </main>
    </div>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
<script src="{{ url_for('static', filename='js/app.js') }}" defer></script>

</html>
```

6.3.2 Medical History and prescription page



Dissertation

Back

History

Test 09 Nov 2024

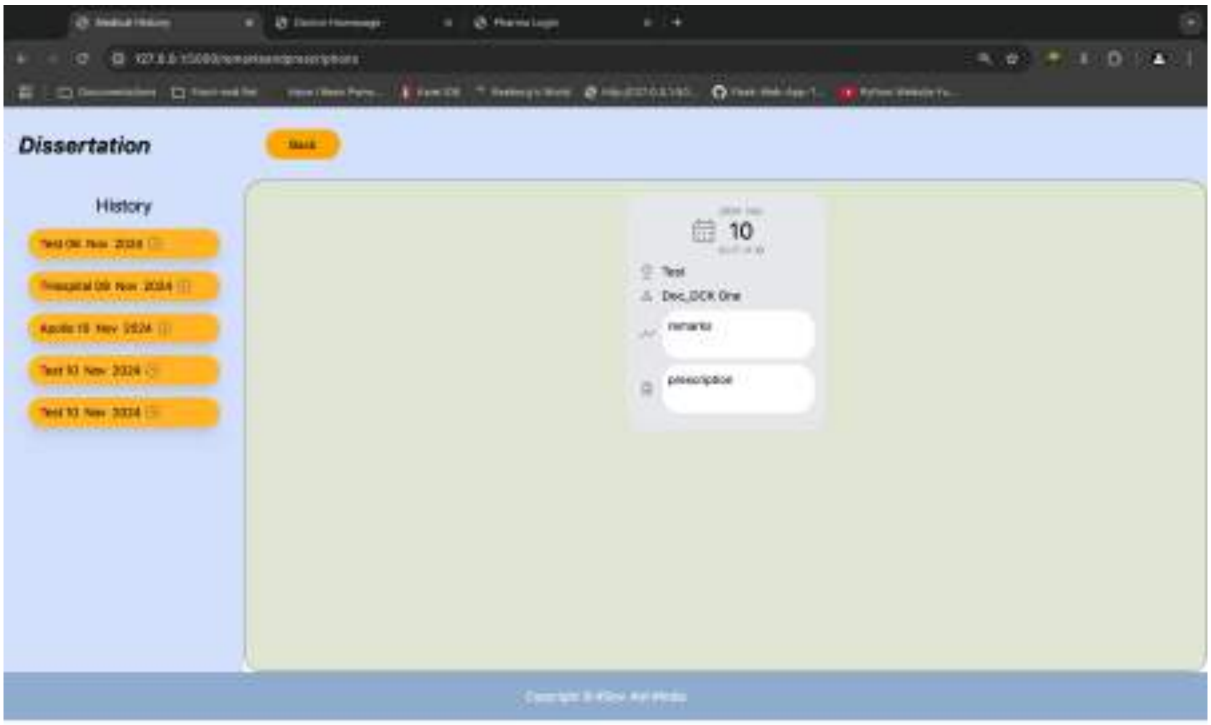
Hospital 09 Nov 2024

Apollo 10 Nov 2024

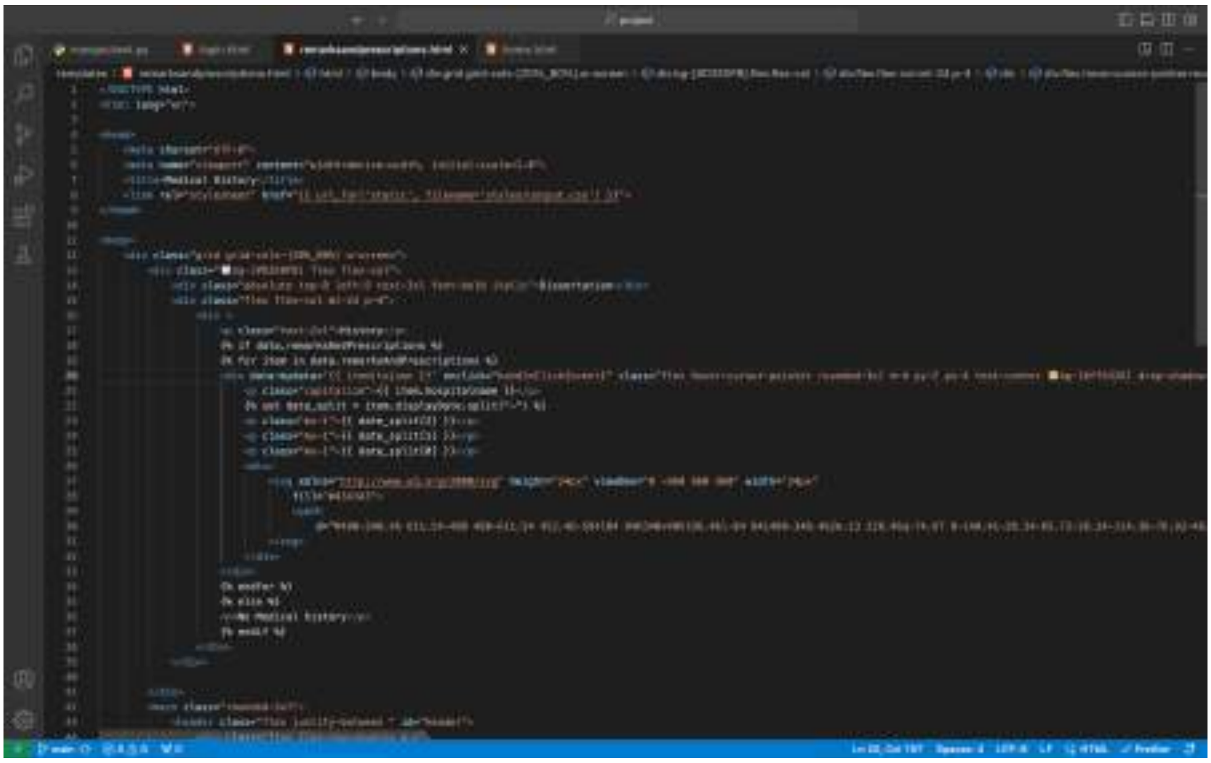
Test 10 Nov 2024

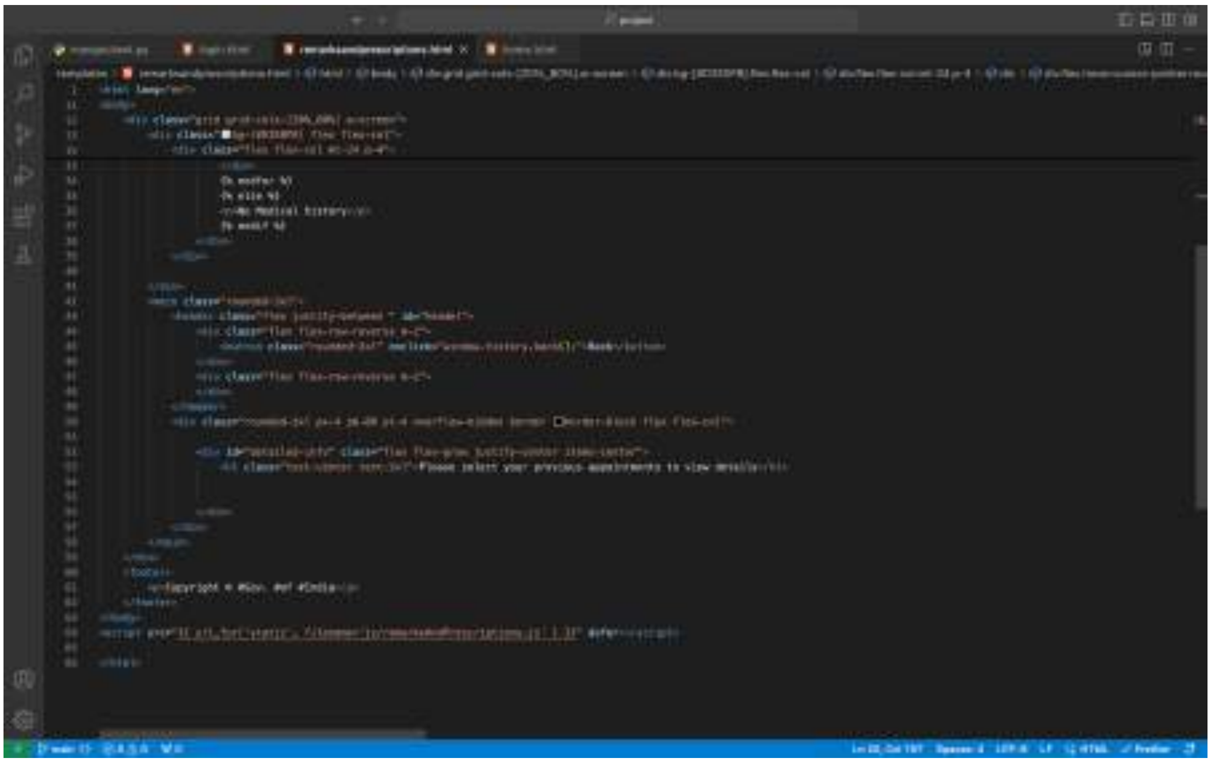
Test 10 Nov 2024

Please select your previous appointments to view details

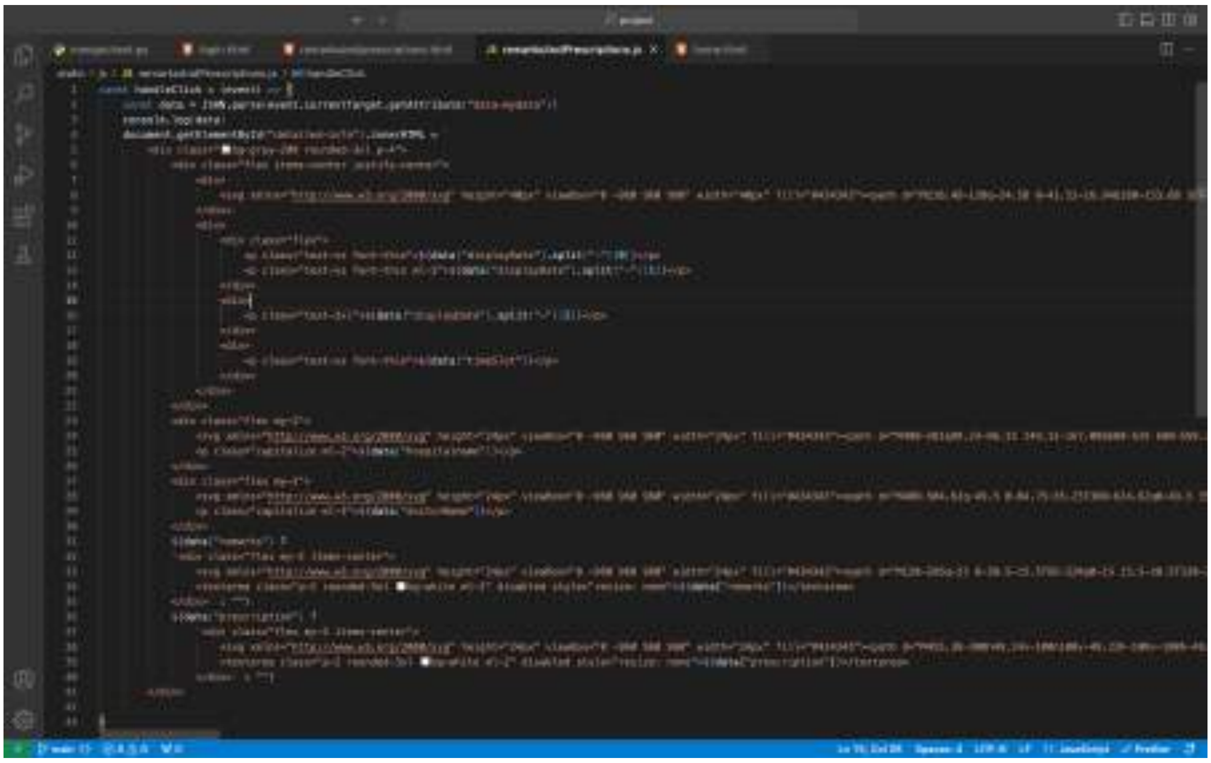


Code for the above screen


```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Medical History</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='[illegible]') }}">
</head>

<body>
    <div class="grid grid-cols-[20%_80%] h-screen">
        <div class="[illegible] flex flex-col">
            <div class="absolute top-0 left-0 text-2xl font-bold italic">Dissertation</div>
            <div class="flex flex-col [illegible]">
                <div >
                    <p class="text-2xl">History</p>
                    {% if data.remarksAndPrescriptions %}
                    {% for item in data.remarksAndPrescriptions %}
                    <div data-[illegible]="{{ item[illegible] }}" onclick="[illegible]" class="flex [illegible] cursor-pointer rounded-[illegible] [illegible]
                        <p class="[illegible]">{{ item.hospitalName }}</p>
                        {% set date_split = item.displayDate.split("-") %}
                        <p class="[illegible]">{{ date_split[2] }}</p>
                        <p class="[illegible]">{{ date_split[1] }}</p>
                        <p class="[illegible]">{{ date_split[0] }}</p>
                        <div>
                            <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px"
                                fill="[illegible]">
                                <path
                                    d="[illegible]
                            </svg>
                        </div>
                    </div>
                    {% endfor %}
                    {% else %}
                    <p>No medical history</p>
                    {% endif %}
                </div>
            </div>

        </div>
        <main class="[illegible]">
            <header class="flex justify-between [illegible]" id="header">
```



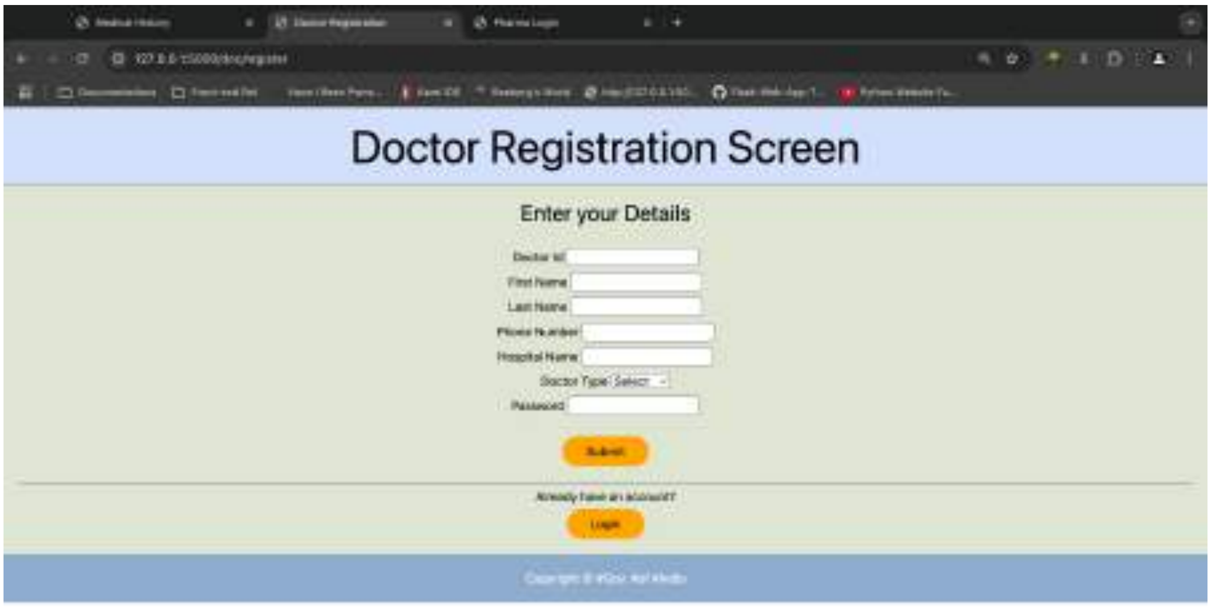
```
<html lang="en">
<body>
    <div class="[illegible]">
        <div class="[illegible] flex flex-col">
            <div class="flex flex-col [illegible]">
                [illegible]
                {% endfor %}
                {% else %}
                <p>No Medical history</p>
                {% endif %}
                [illegible]
            </div>

        [illegible]
        <main class="[illegible]">
            <header class="flex justify-between " id="header">
                <div class="flex flex-row-reverse [illegible]">
                <button class="[illegible]" onclick="[illegible]">Back</button>
                </div>
                <div class="flex flex-row-reverse [illegible]">
                </div>
            </header>
            <div class="[illegible] overflow-hidden [illegible] flex flex-col">

                <div id="detailed-info" class="flex flex-grow justify-center items-center">
                    <h1 class="[illegible]">Please select your previous appointments to view details</h1>

                </div>
            </div>
        </main>
    </div>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
<script src="{{ url_for('static', filename='[illegible]') }}" defer></script>

</html>
```



```
const handleClick = (event) => {
    const data = JSON.parse(event.currentTarget.getAttribute("data-[illegible]"))
    console.log(data)
    document.getElementById("detailed-info").innerHTML = `
        <div class="[illegible] rounded-lg p-4">
            <div class="flex items-center justify-center">
                [illegible]
```

7 Chapter 7

7.1 Doctor Registration screen



Doctor Registration Screen

Enter your Details

Doctor Id

First Name

Last Name

Phone Number

Hospital Name

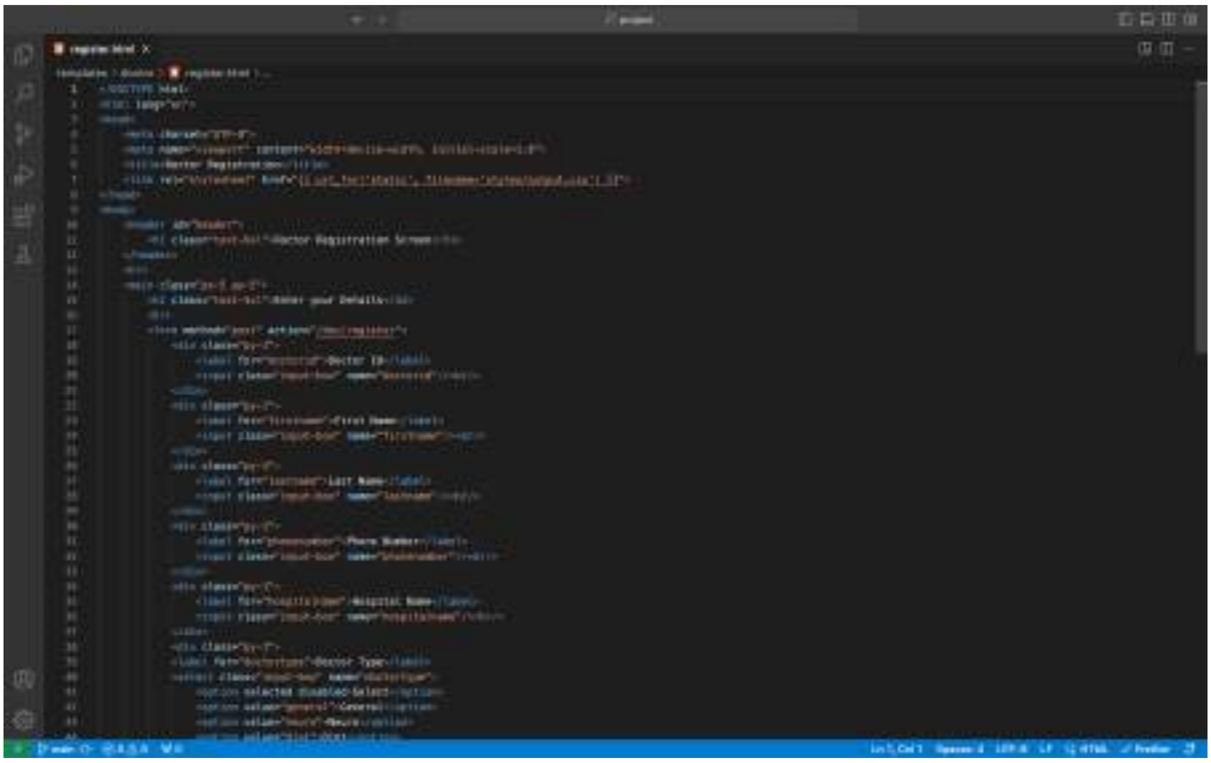
Doctor Type: Select

Password

Submit

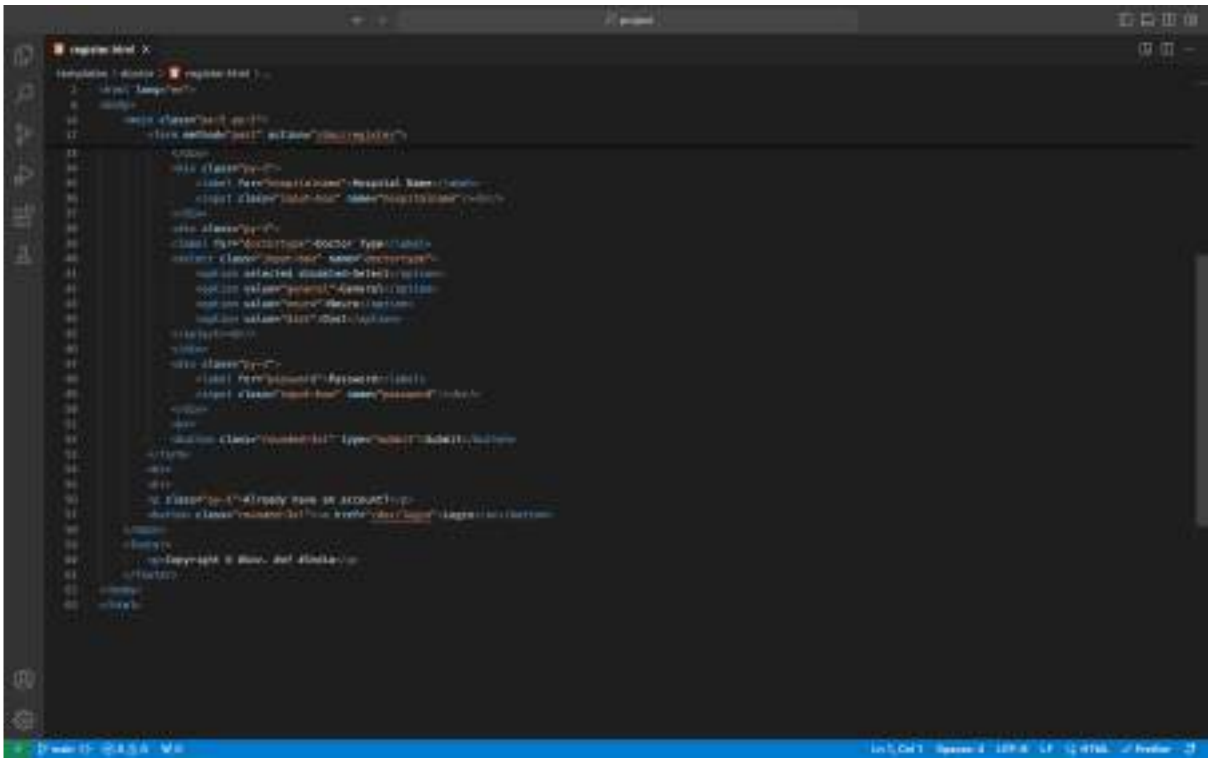
Already have an account?

Login



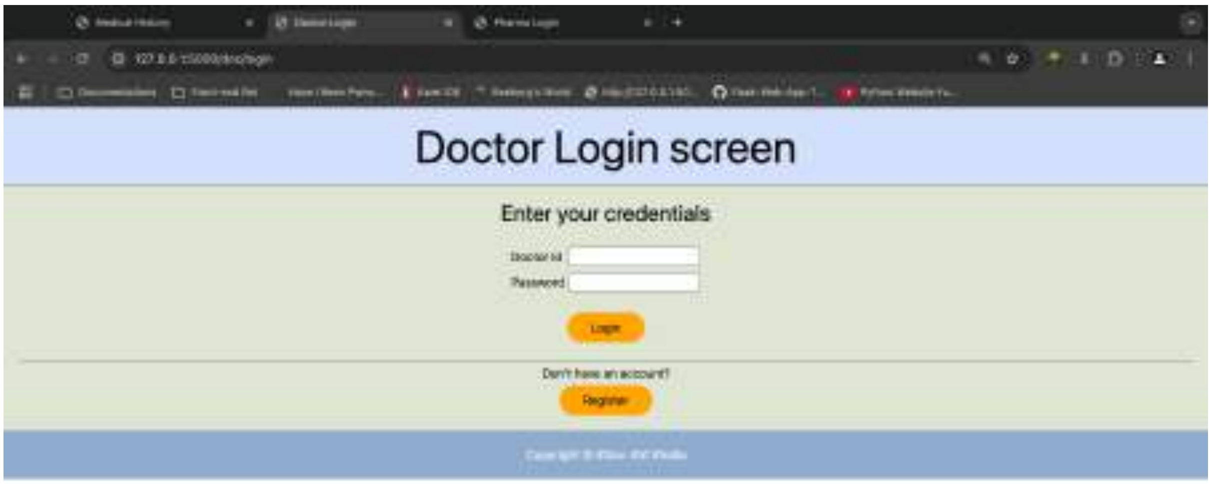
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Doctor Registration</title>
    <link rel="stylesheet" href="[illegible]">
</head>
<body>
    <header id="header">
        <h1 class="[illegible]">Doctor Registration Screen</h1>
    </header>
    <hr>
    <main class="[illegible]">
        <h2 class="[illegible]">Enter your Details</h2>
        <hr>
        <form method="post" action="/doc/register">
            <div class="[illegible]">
                <label for="[illegible]">Doctor ID</label>
                <input class="input-box" name="[illegible]"><br>
            </div>
            <div class="[illegible]">
                <label for="firstname">First Name</label>
                <input class="input-box" name="firstname"><br>
            </div>
            <div class="[illegible]">
                <label for="lastname">Last Name</label>
                <input class="input-box" name="lastname"><br>
            </div>
            <div class="[illegible]">
                <label for="phonenumber">Phone Number</label>
                <input class="input-box" name="phonenumber"><br>
            </div>
            <div class="[illegible]">
                <label for="hospitalname">Hospital Name</label>
                <input class="input-box" name="hospitalname"><br>
            </div>
            <div class="[illegible]">
            <label for="doctortype">Doctor Type</label>
            <select class="input-box" name="doctortype">
                <option selected disabled>Select</option>
                <option value="general">General</option>
                <option value="neuro">Neuro</option>
                <option value="[illegible]">[illegible]</option>
```

34



```
<html lang="en">
<body>
    <main class="px-5 py-5">
        <form method="post" action="/doc/register">
            <hr>
            <div class="py-2">
                <label for="hospitalname">Hospital Name</label>
                <input class="input-box" name="hospitalname"><br>
            </div>
            <div class="py-2">
            <label for="doctortype">Doctor Type</label>
            <select class="input-box" name="doctortype">
                <option selected disabled>Select</option>
                <option value="general">General</option>
                <option value="neuro">Neuro</option>
                <option value="diet">Diet</option>
            </select><br>
            </div>
            <div class="py-2">
                <label for="password">Password</label>
                <input class="input-box" name="password"><br>
            </div>
            <br>
            <button class="rounded-btn" type="submit">Submit</button>
        </form>
        <hr>
        <br>
        <p class="px-1">Already have an account?</p>
        <button class="rounded-btn"><a href="/doc/login">Login</a></button>
    </main>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
</html>
```

7.2 Doctor Login screen



127.0.0.1:5000/doc/login

Doctor Login screen

Enter your credentials

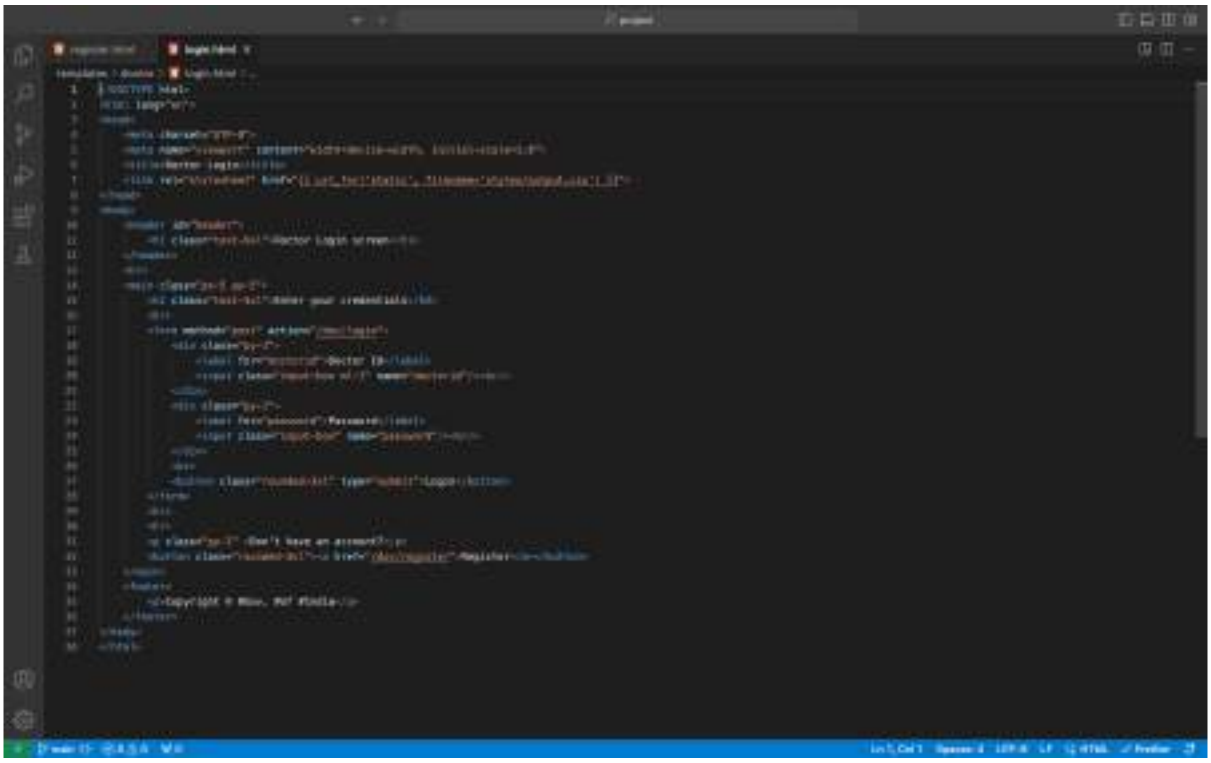
Doctor Id

Password

Login

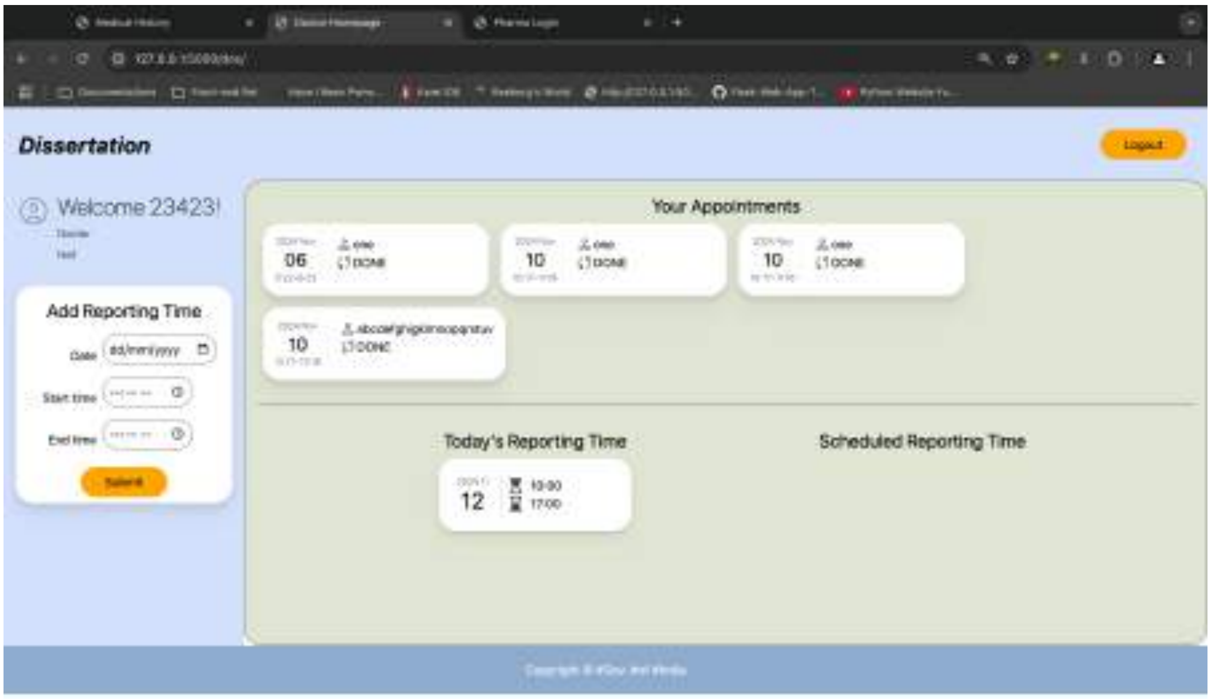
Don't have an account?

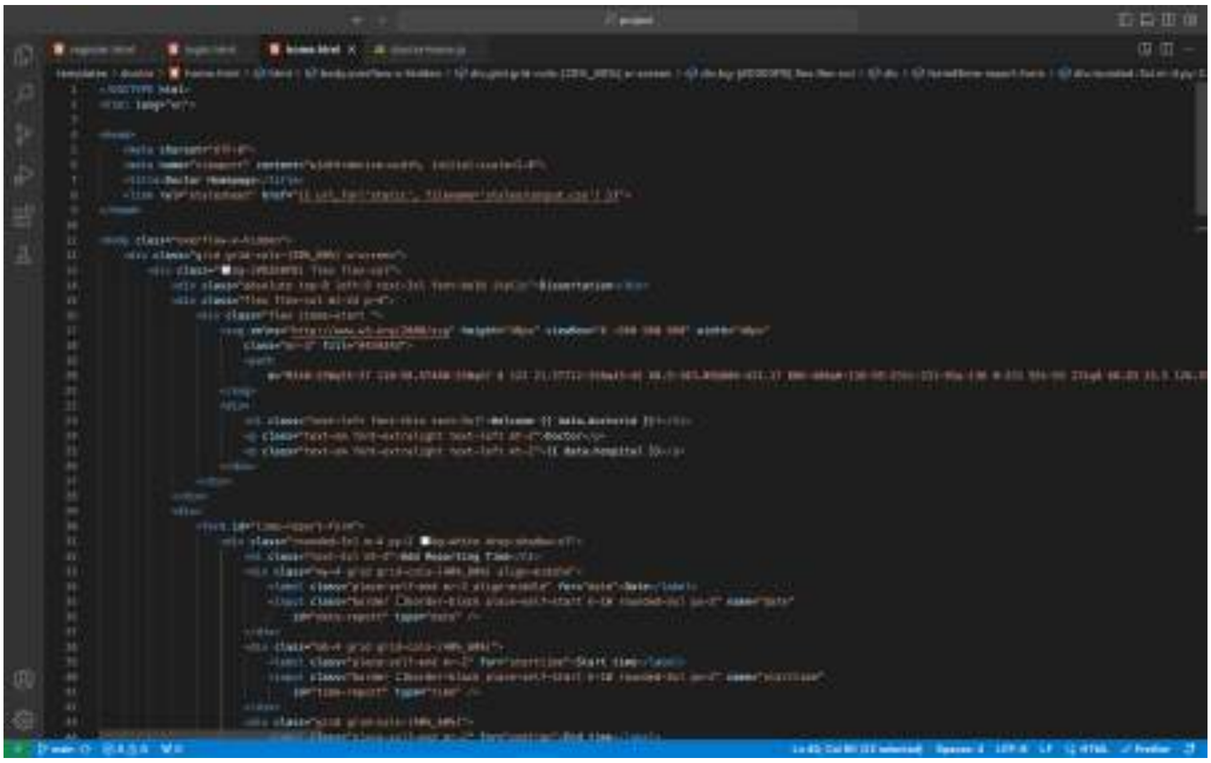
Register



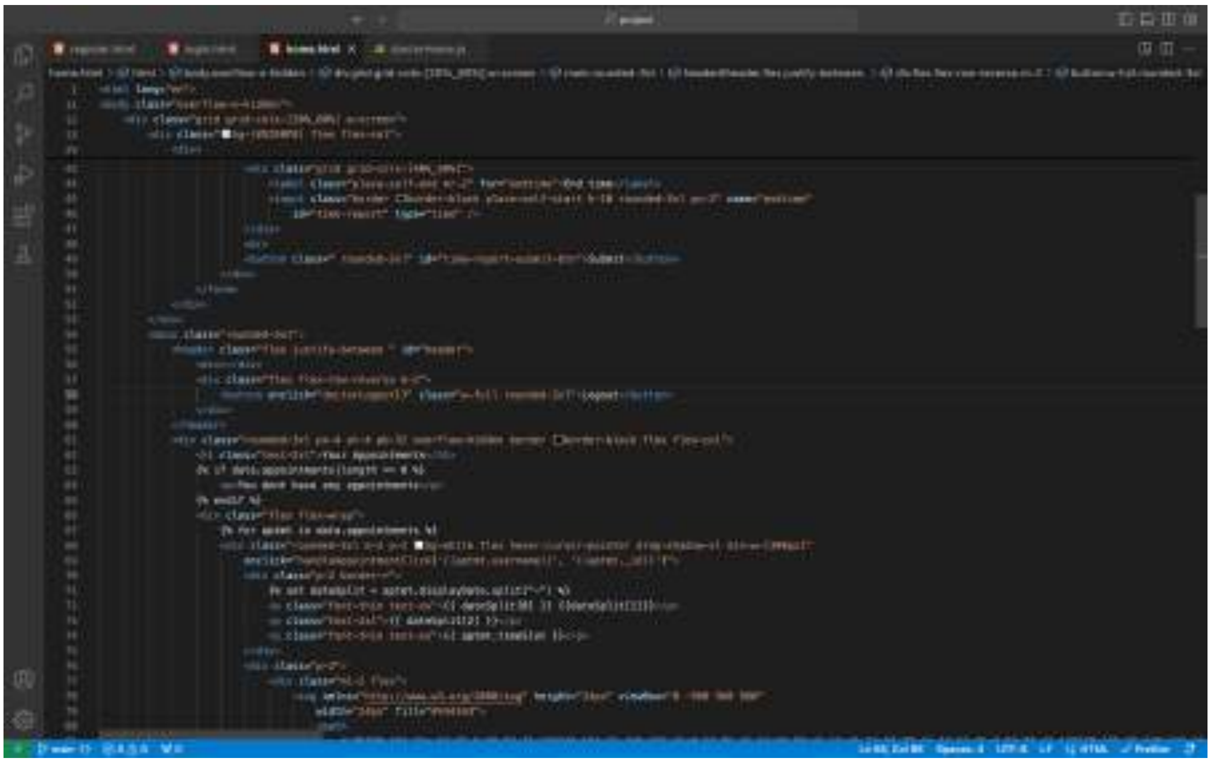
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Doctor login</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='[illegible]') }}">
</head>
<body>
    <header id="header">
        <h1 class="text-4xl">Doctor Login screen</h1>
    </header>
    <hr>
    <main class="px-5 py-5">
        <h2 class="text-5xl">Enter your credentials</h2>
        <br>
        <form method="post" action="/doc/login">
            <div class="py-2">
                <label for="doctorid">Doctor ID</label>
                <input class="input-box ml-5" name="doctor_id"><br>
            </div>
            <div class="py-2">
                <label for="password">Password</label>
                <input class="input-box" name="password"><br>
            </div>
            <br>
            <button class="rounded-btn" type="submit">Login</button>
        </form>
        <br>
        <br>
        <p class="py-2">Don't have an account?</p>
        <button class="rounded-btn"><a href="/doc/register">Register</a></button>
    </main>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
</html>
```

7.3 Doctor dashboard screen

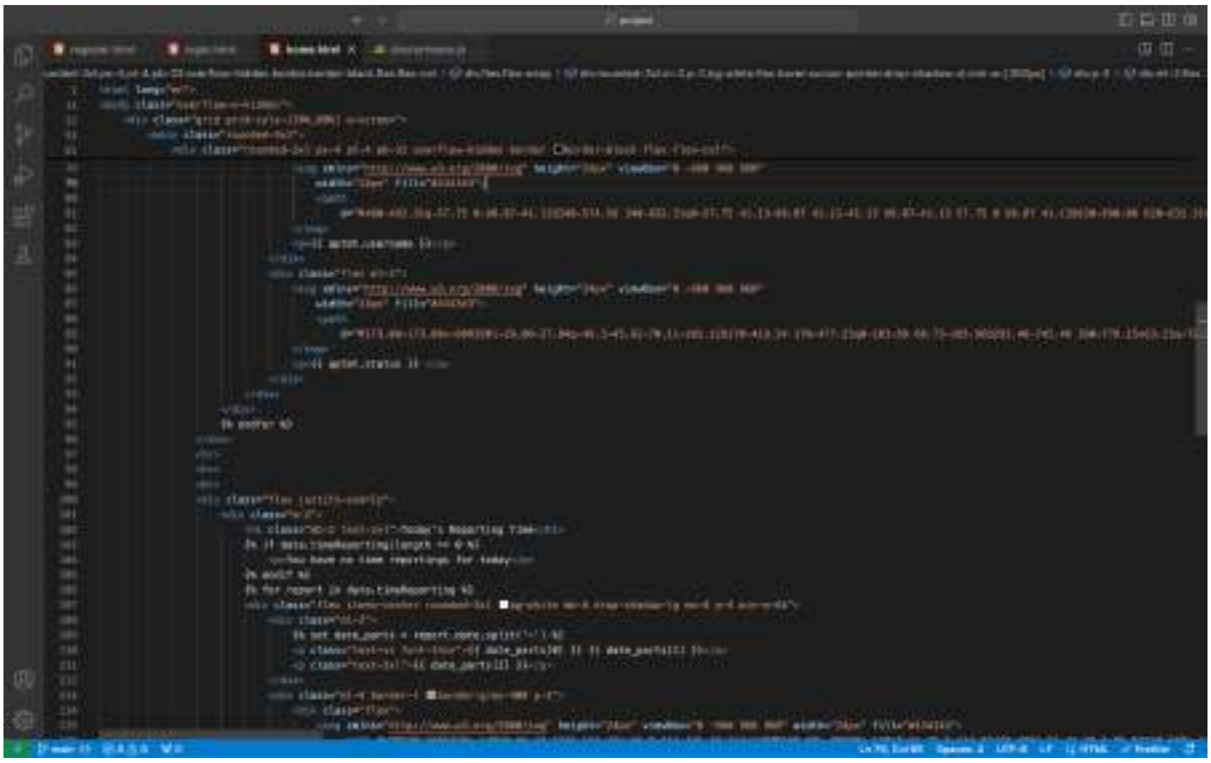




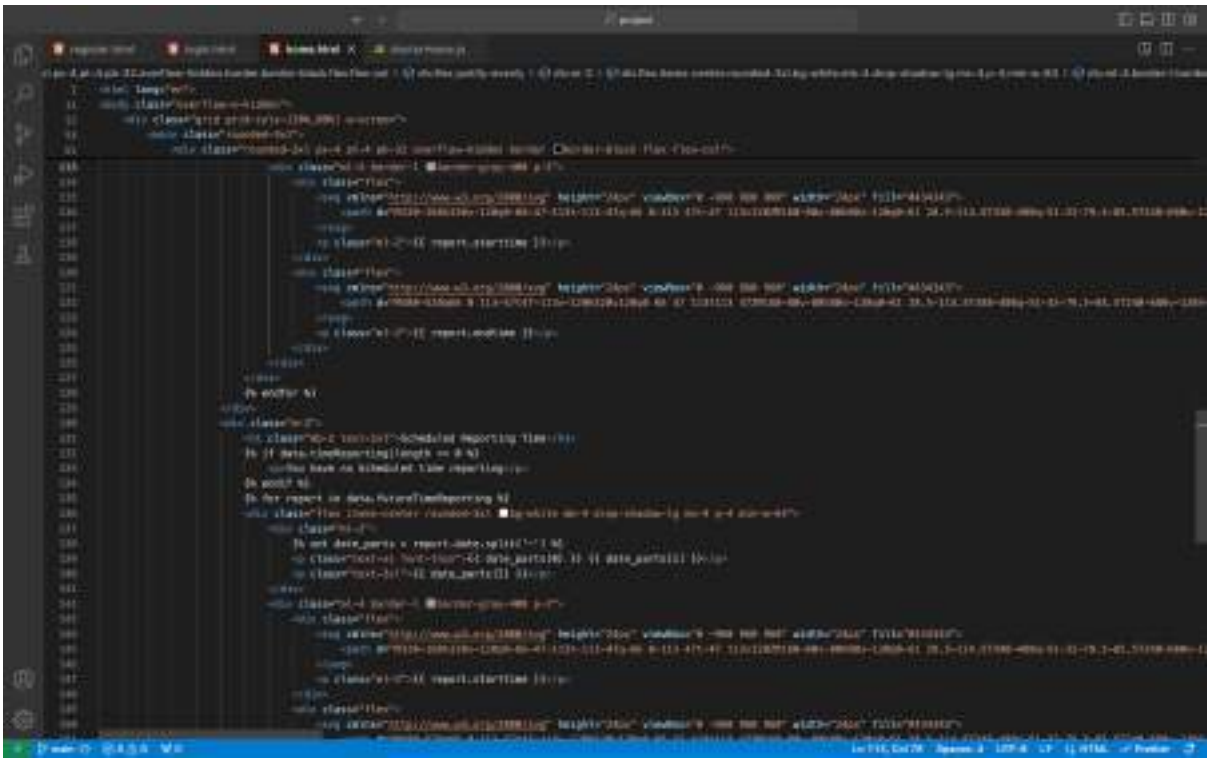
```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Doctor Homepage</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='[illegible].css') }}">
</head>

<body class="[illegible]">
    <div class="grid grid-cols-[illegible] w-screen">
        <div class="■bg-[illegible] flex flex-col">
            <div class="absolute top-0 left-0 text-3xl [illegible]">Dissertation</div>
            <div class="flex flex-col [illegible]">
                <div class="flex [illegible]">
                    <svg xmlns="http://www.w3.org/2000/svg" height="[illegible]" viewBox="0 -960 960 960" width="[illegible]"
                        class="[illegible]" fill="[illegible]">
                        <path
                            d="[illegible]
                    </svg>
                    <div>
                        <h1 class="text-left [illegible]">Welcome {{ data.doctorid }}</h1>
                        <p class="text-sm font-extralight text-left [illegible]">Doctor</p>
                        <p class="text-sm font-extralight text-left [illegible]">{{ data.hospital }}</p>
                    </div>
                </div>
            </div>
            <div>
                <form id="time-report-form">
                    <div class="rounded-[illegible] ■bg-white drop-shadow-[illegible]">
                        <h1 class="text-[illegible]">Add Reporting Time</h1>
                        <div class="[illegible] grid grid-cols-[illegible] align-middle">
                            <label class="place-self-end [illegible] align-middle" for="date">Date</label>
                            <input class="border □border-black place-self-start [illegible] rounded-[illegible]" name="date"
                                id="date-report" type="date" />
                        </div>
                        <div class="[illegible] grid grid-cols-[illegible]">
                            <label class="place-self-end [illegible]" for="starttime">Start time</label>
                            <input class="border □border-black place-self-start [illegible] rounded-[illegible]" name="starttime"
                                id="time-report" type="time" />
                        </div>
                        <div class="grid grid-cols-[illegible]">
                            <label class="place-self-end [illegible]" for="endtime">End time</label>
```



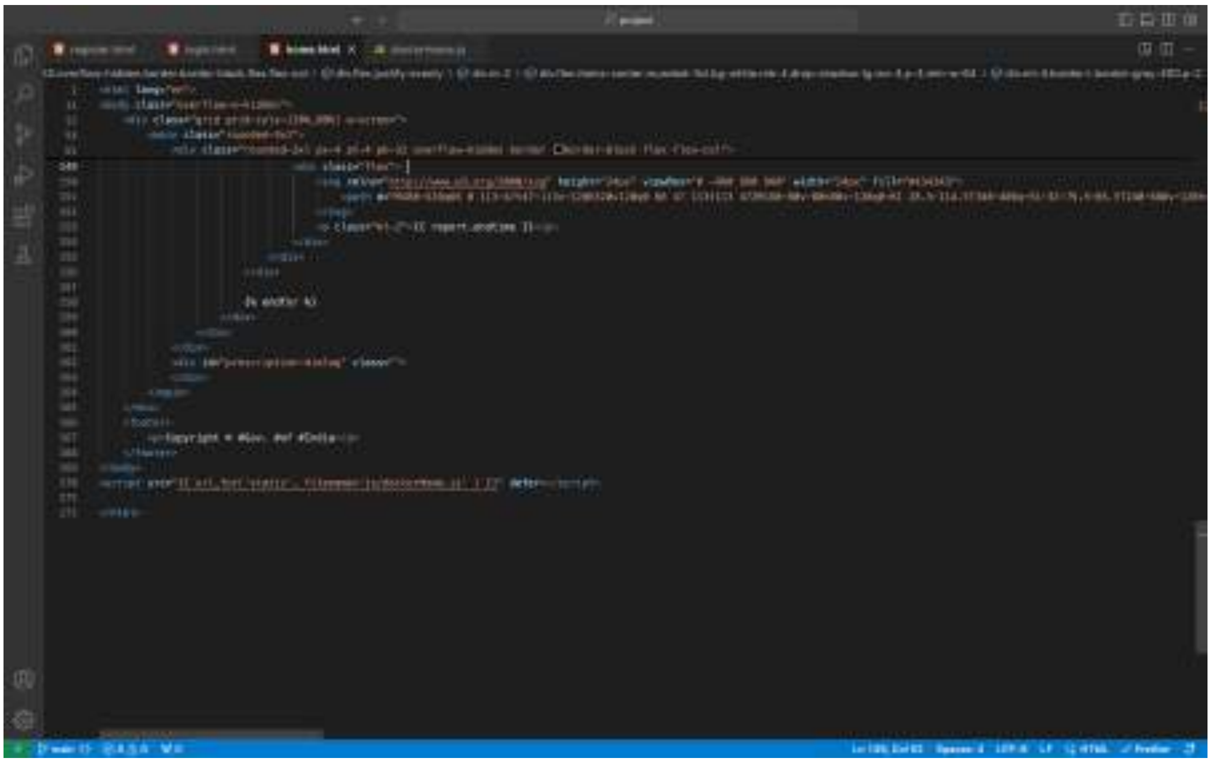
```
<html lang="en">
<body class="[illegible]">
    <div class="grid grid-cols-[illegible] w-screen">
        <div class="■bg-[illegible] flex flex-col">
            <div>
                        <div class="grid grid-cols-[illegible]">
                            <label class="place-self-end [illegible]" for="endtime">End time</label>
                            <input class="border □border-black place-self-start [illegible] rounded-[illegible]" name="endtime"
                                id="time-report" type="time" />
                        </div>
                        <div>
                        <button class=" rounded-[illegible]" id="time-report-submit-[illegible]">Submit</button>
                    </div>
                </form>
            </div>
        </div>
        <main class="rounded-[illegible]">
            <header class="flex justify-between " id="header">
                <div></div>
                <div class="flex [illegible]">
                    <button onclick="doctorLogout()" class="[illegible] rounded-[illegible]">Logout</button>
                </div>
            </header>
            <div class="rounded-[illegible] overflow-hidden border □border-black flex flex-col">
                <h1 class="text-[illegible]">Your Appointments</h1>
                {% if data.appointments|length == 0 %}
                    <p>You dont have any appointments</p>
                {% endif %}
                <div class="flex flex-wrap">
                    {% for apmt in data.appointments %}
                    <div class="rounded-[illegible] ■bg-white flex hover:cursor-pointer drop-shadow-[illegible]"
                        onclick="[illegible]('{{apmt.username}}', '{{apmt._id}}')">
                        <div class="[illegible]">
                            {% set dateSplit = apmt.displayDate.split("-") %}
                            <p class="text-[illegible]">{{ dateSplit[0] }} {{dateSplit[1]}}</p>
                            <p class="text-[illegible]">{{ dateSplit[2] }}</p>
                            <p class="text-[illegible]">{{ apmt.timeSlot }}</p>
                        </div>
                        <div class="[illegible]">
                            <div class="[illegible] flex">
                                <svg xmlns="http://www.w3.org/2000/svg" height="[illegible]" viewBox="0 -960 960 960"
                                    width="[illegible]" fill="[illegible]">
                                    <path
```



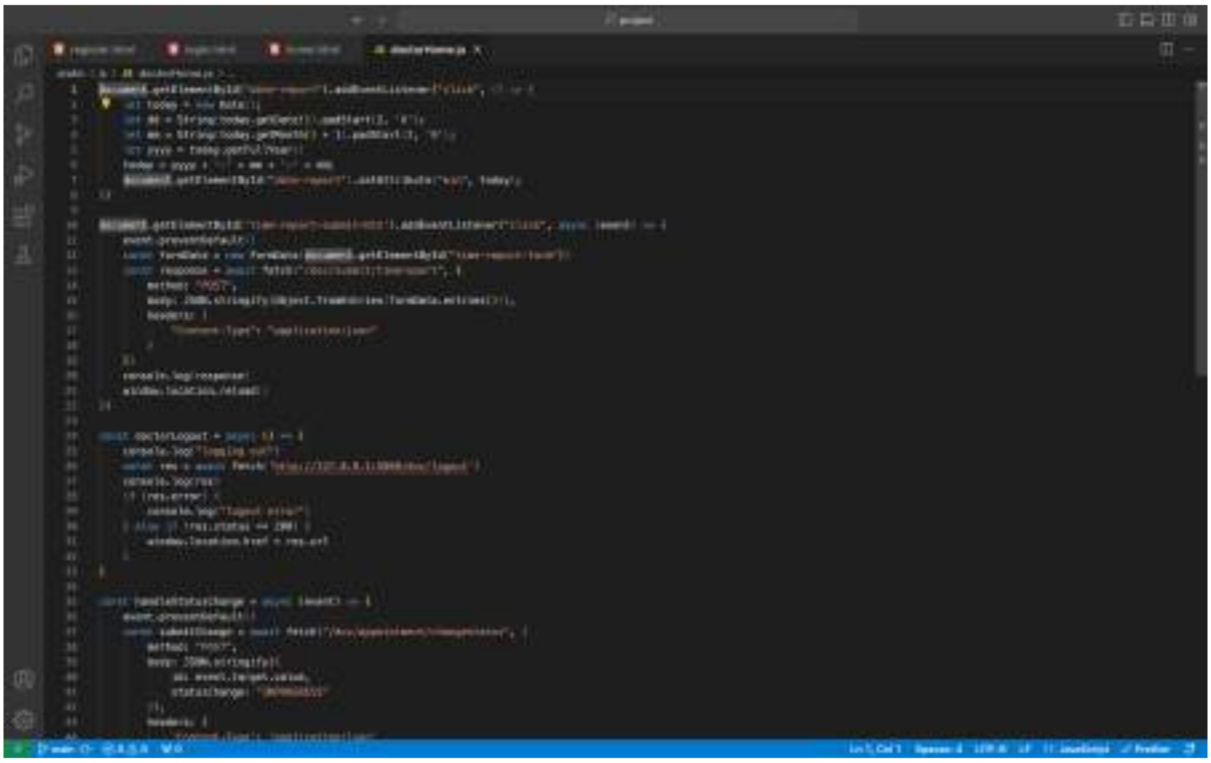
```
<html lang="en">
<body class="overflow-x-hidden">
  <div class="grid grid-cols-[196,896] w-screen">
    <main class="[illegible]">
      <div class="[illegible] border border-black flex flex-col">
                  <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                    width="24px" fill="[illegible]">
                    <path
                      d="[illegible]"
                  </svg>
                  <p>{{ aptmt.username }}</p>
                </div>
                <div class="flex [illegible]">
                  <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                    width="24px" fill="[illegible]">
                    <path
                      d="[illegible]"
                  </svg>
                  <p>{{ aptmt.status }}</p>
                </div>
              </div>
            </div>
            {% endfor %}
          </div>
        </div>
      </div>
      </div>
      <div class="flex justify-evenly">
        <div class="w-2/5">
          <h1 class="[illegible] text-2xl">Today's Reporting Time</h1>
          {% if data.timeReporting|length == 0 %}
            <p>You have no time reportings for today</p>
          {% endif %}
          {% for report in data.timeReporting %}
          <div class="flex items-center rounded-3xl bg-white [illegible]">
            <div class="[illegible]">
              {% set date_parts = report.date.split('-') %}
              <p class="text-xl font-[illegible]">{{ date_parts[0] }} {{ date_parts[1] }}</p>
              <p class="text-3xl">{{ date_parts[2] }}</p>
            </div>
            <div class="[illegible] border-l border-gray-400 p-4">
              <div class="flex">
                <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="[illegible]">
```



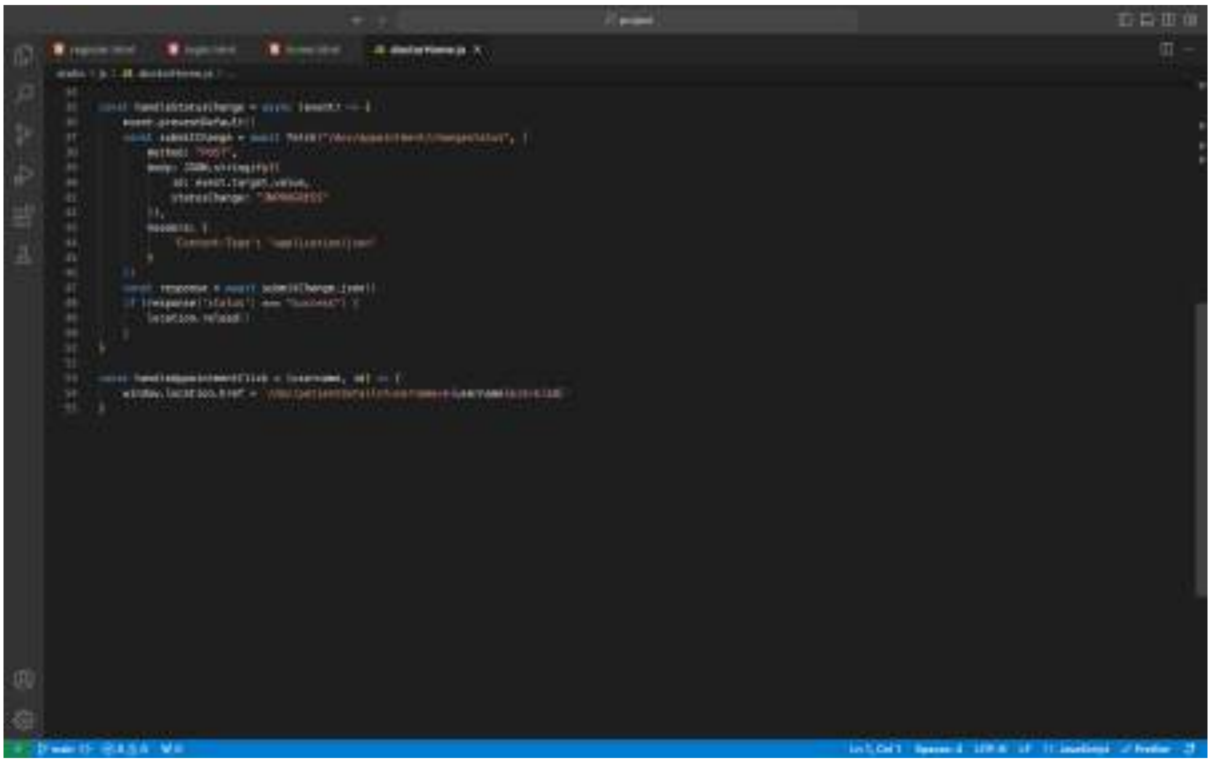
```
<html lang="en">
<body class="overflow-x-hidden">
  <div class="grid grid-cols-[196,896] w-screen">
    <main class="[illegible]">
      <div class="[illegible] border border-black flex flex-col">
            <div class="[illegible] border-l border-gray-400 p-4">
              <div class="flex">
                <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="[illegible]">
                  <path d="[illegible]"
                </svg>
                <p class="ml-2">{{ report.startTime }}</p>
              </div>
              <div class="flex">
                <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="[illegible]">
                  <path d="[illegible]"
                </svg>
                <p class="ml-2">{{ report.endTime }}</p>
              </div>
            </div>
          </div>
          {% endfor %}
        </div>
        <div class="w-2/5">
          <h1 class="[illegible] text-2xl">Scheduled Reporting Time</h1>
          {% if data.timeReporting|length == 0 %}
            <p>You have no Scheduled time reporting</p>
          {% endif %}
          {% for report in data.futureTimeReporting %}
          <div class="flex items-center rounded-3xl bg-white [illegible]">
            <div class="[illegible]">
              {% set date_parts = report.date.split('-') %}
              <p class="text-xl font-[illegible]">{{ date_parts[0] }} {{ date_parts[1] }}</p>
              <p class="text-3xl">{{ date_parts[2] }}</p>
            </div>
            <div class="[illegible] border-l border-gray-400 p-4">
              <div class="flex">
                <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="[illegible]">
                  <path d="[illegible]"
                </svg>
                <p class="ml-2">{{ report.startTime }}</p>
              </div>
              <div class="flex">
                <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="[illegible]">
```



```
<html lang="en">
<body class="overflow-x-hidden">
    <div class="grid grid-cols-[200,800] [illegible]">
        <main class="[illegible]">
            <div class="[illegible] overflow-hidden border [illegible] flex flex-col">
                            <div class="flex">
                                <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="[illegible]">
                                    <path d="[illegible]"/>
                                </svg>
                                <p class="[illegible]">{{ report.onetime }}</p>
                            </div>
                        </div>
                    </div>

                    {% endfor %}
                </div>
            </div>
        </div>
        <div id="prescription-dialog" class=""></div>
        </div>
    </main>
    </div>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
<script src="{{ url_for('static', filename='js/doctorHome.js') }}" defer></script>

</html>
```

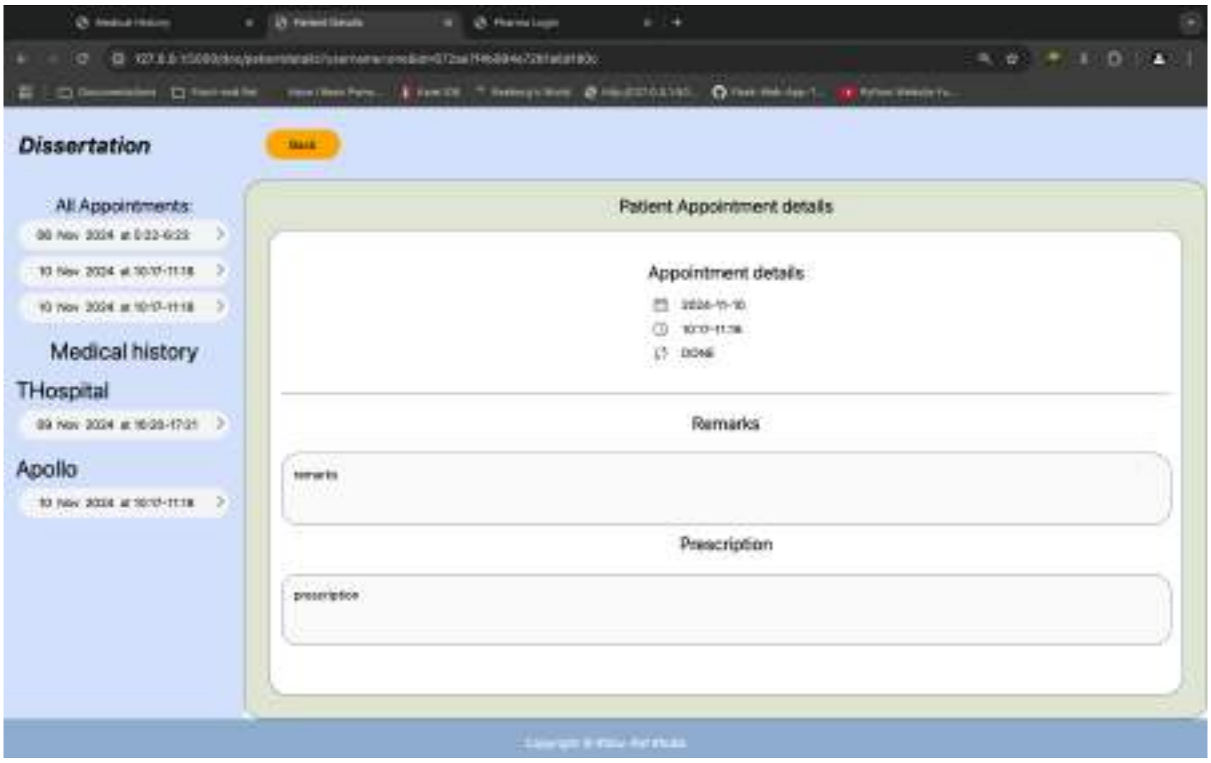


```
document.getElementById("[illegible]").addEventListener("click", () => {
    let today = new Date();
    let dd = String(today.getDate()).padStart(2, '0');
    let mm = String(today.getMonth() + 1).padStart(2, '0');
    let yyyy = today.getFullYear();
    today = yyyy + '-' + mm + '-' + dd;
    document.getElementById("[illegible]").setAttribute("min", today);
});

document.getElementById("[illegible]").addEventListener("click", async (event) => {
    event.preventDefault();
    const formData = new FormData(document.getElementById("[illegible]"));
    const response = await fetch("[illegible]", {
        method: "POST",
        body: JSON.stringify(Object.fromEntries(formData.entries())),
        headers: {
            "Content-Type": "application/json"
        }
    });
    console.log(response);
    window.location.reload();
});

const doctorLogout = async () => {
    console.log("logging out");
    const res = await fetch("http://127.0.0.1:5000/doc/logout");
    console.log(res);
    if (res.error) {
        console.log("logout error");
    } else if (res.status == 200) {
        window.location.href = res.url;
    }
}

const handleStatusChange = async (event) => {
    event.preventDefault();
    const submitChange = await fetch("/doc/appointment/changeStatus", {
        method: "POST",
        body: JSON.stringify({
            id: event.target.value,
            statusChange: "[illegible]"
        }),
        headers: {
            "Content-Type": "application/json"
```



```
const handleStatusChange = async (event) => {
    event.preventDefault()
    const submitChange = await fetch("/doc/appointment/changeStatus", {
        method: "POST",
        body: JSON.stringify({
            id: event.target.value,
            statusChange: "[illegible]"
        }),
        headers: {
            'Content-Type': 'application/json'
        }
    })
    const response = await submitChange.json()
    if (response['status'] == "Success") {
        location.reload()
    }
}

const handleAppointmentClick = (username, id) => {
    window.location.href = `[illegible]`
}
```

7.4 User’s appointments and Medical History



Dissertation

Back

All Appointments:

08 Nov 2024 at 5:22-6:22

10 Nov 2024 at 10:17-11:18

10 Nov 2024 at 10:17-11:18

Medical history

THospital

09 Nov 2024 at 16:20-17:21

Apollo

10 Nov 2024 at 10:17-11:18

Patient Appointment details

Appointment details

2024-11-10

10:17-11:18

DONE

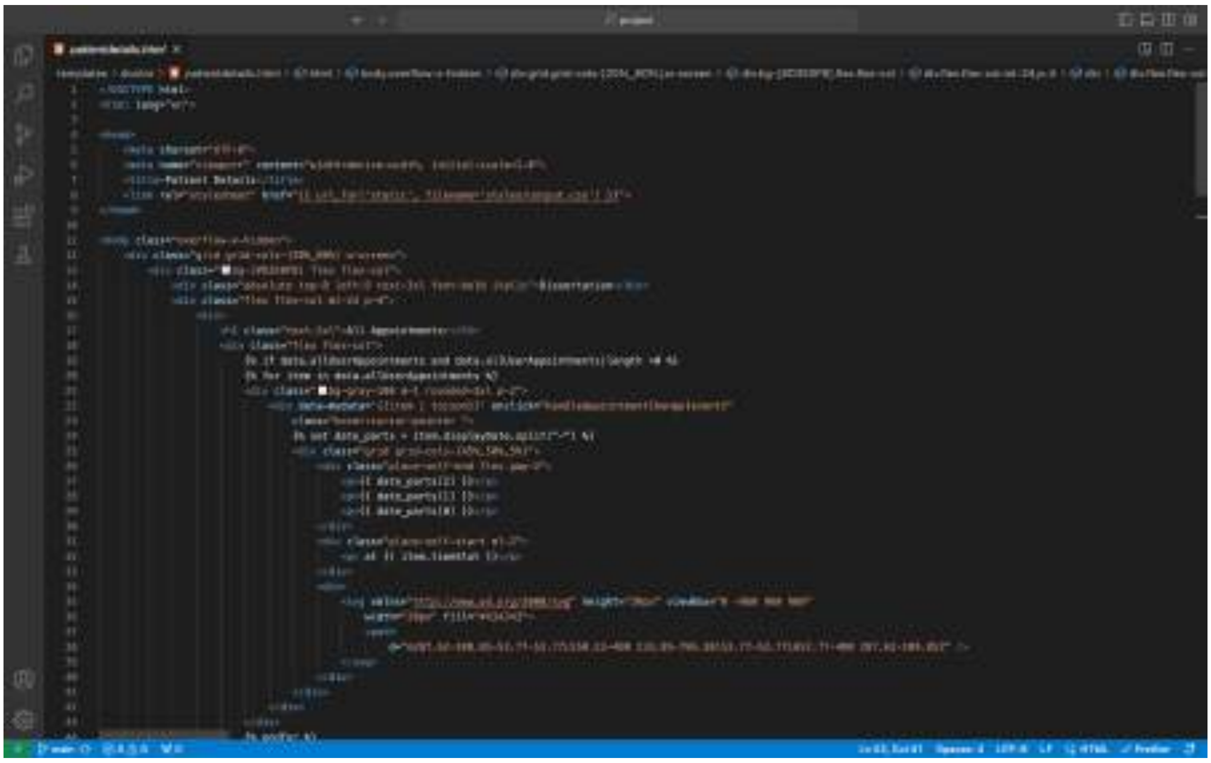
Remarks

remarks

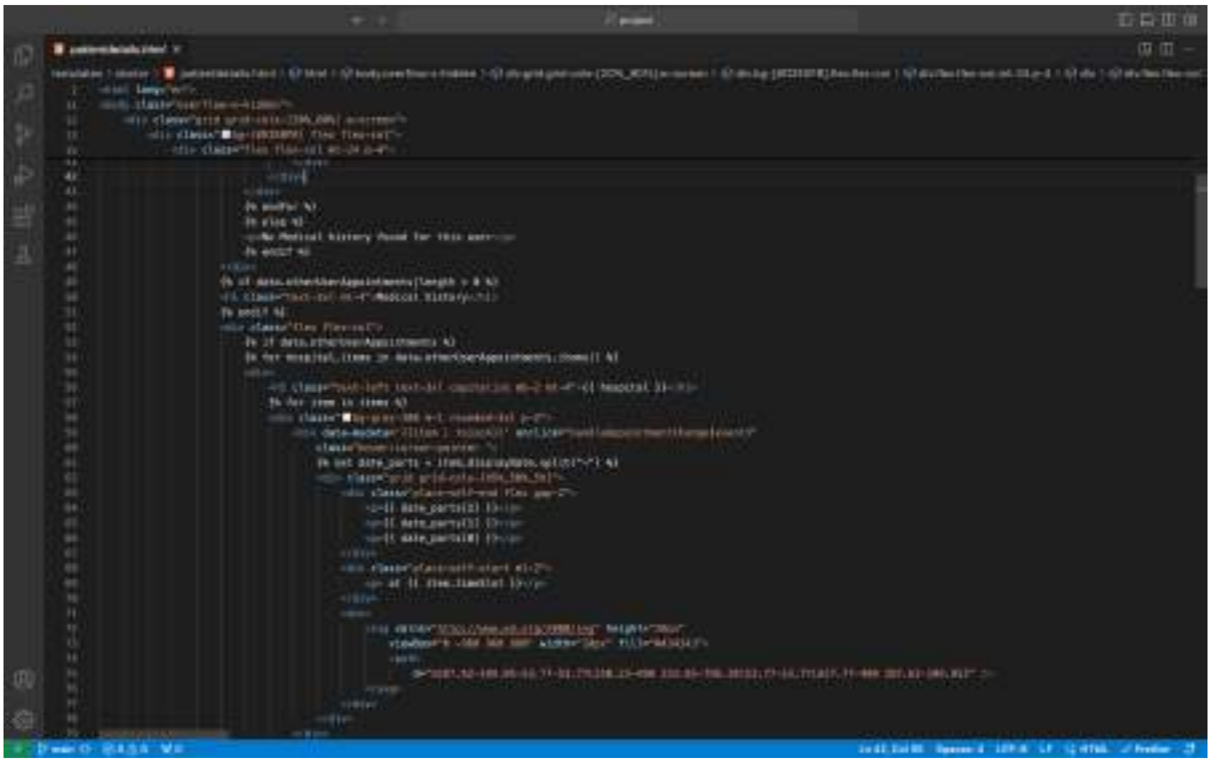
Prescription

prescription

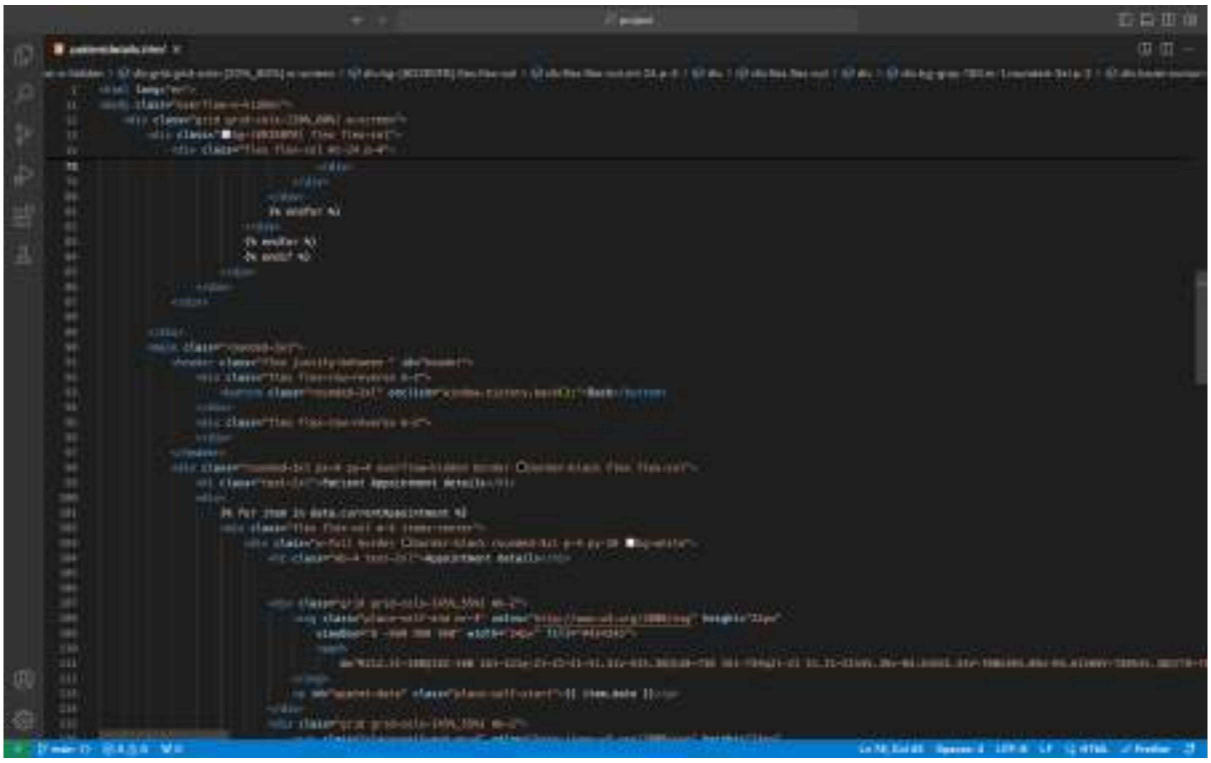
Copyright © [illegible]



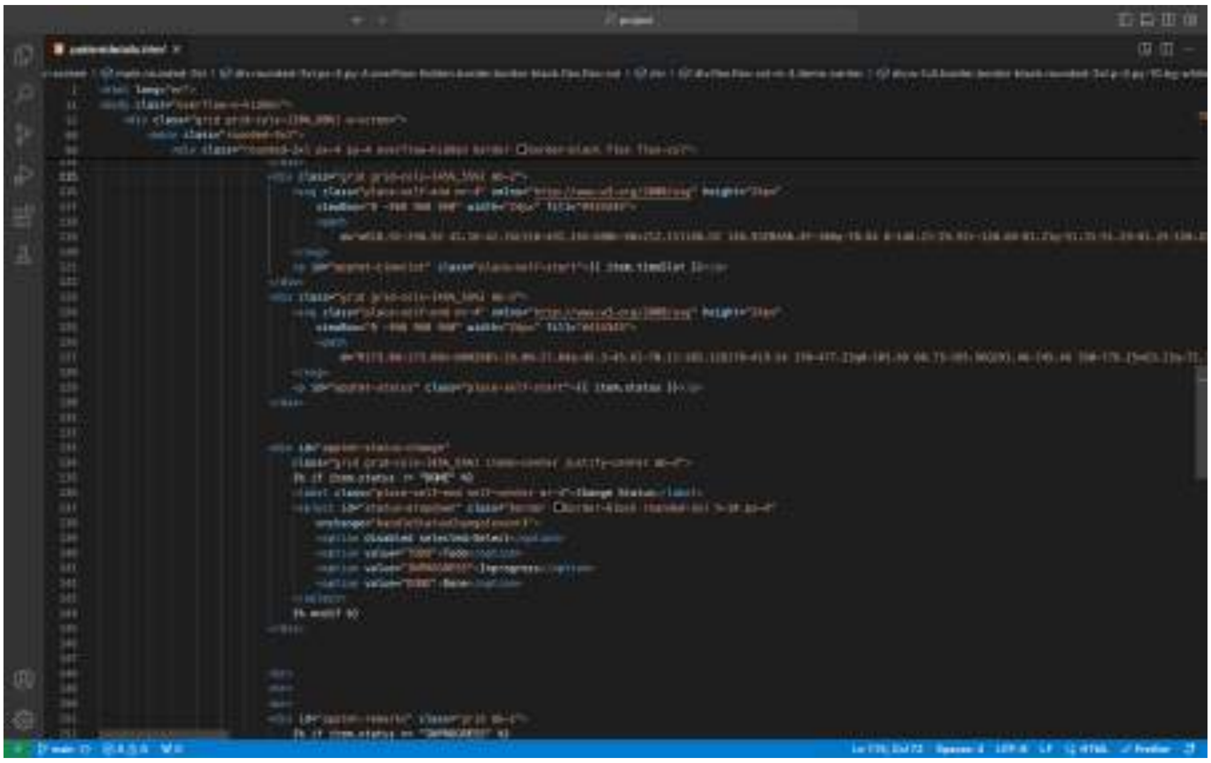
```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Patient Details</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='[illegible].css') }}">
</head>

<body class="overflow-x-hidden">
    <div class="grid grid-cols-[20%_80%] w-screen">
        <div class="bg-[#D2E0FB] flex flex-col">
            <div class="absolute top-0 left-0 text-2xl font-bold italic">Dissertation</div>
            <div class="flex flex-col mt-24 p-4">
                <div>
                    <h1 class="text-2xl">All Appointments</h1>
                    <div class="flex flex-col">
                        {% if data.allUserAppointments and data.allUserAppointments|length >0 %}
                        {% for item in data.allUserAppointments %}
                        <div class="bg-gray-300 m-1 rounded-2xl p-2">
                            <div data-[illegible]="{{item | tojson}}" onclick="[illegible]"
                                class="[illegible]" />
                                {% set date_parts = item.displayDate.split("-") %}
                                <div class="grid grid-cols-[45%_50%_5%]">
                                    <div class="place-self-end flex gap-2">
                                        <p>{{ date_parts[2] }}</p>
                                        <p>{{ date_parts[1] }}</p>
                                        <p>{{ date_parts[0] }}</p>
                                    </div>
                                    <div class="place-self-start ml-2">
                                        <p> at {{ item.timeSlot }}</p>
                                    </div>
                                    <div>
                                        <svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960"
                                            width="24px" fill="#[illegible]">
                                            <path
                                                d="[illegible]" />
                                        </svg>
                                    </div>
                                </div>
                            </div>
                        </div>
                        {% endfor %}
```



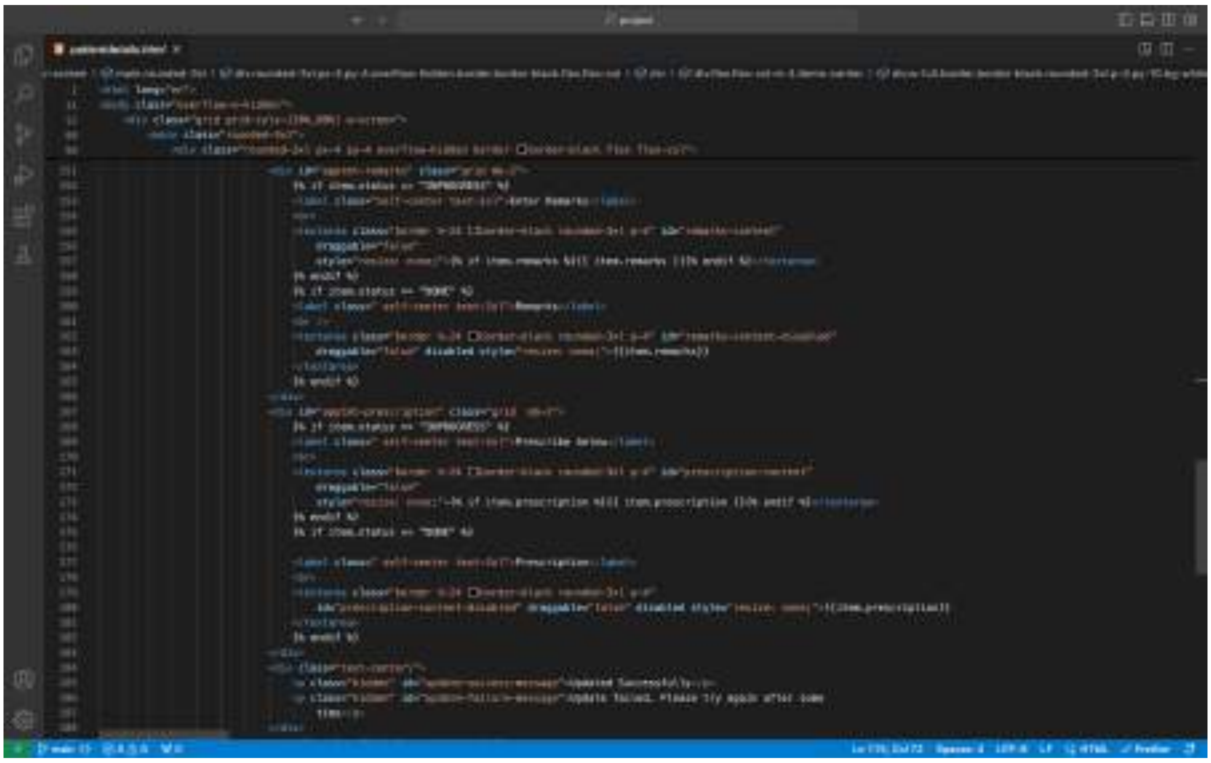
```
<html lang="en">
<body class="[illegible]">
    <div class="grid grid-cols-[illegible] [illegible]">
        <div class="[illegible] flex flex-col">
            <div class="flex flex-col [illegible]">
                        </div>
                    </div>
                </div>
                {% endfor %}
                {% else %}
                <p>No Medical history found for this user</p>
                {% endif %}
            </div>
            {% if data.otherUserAppointments|length > 0 %}
            <h1 class="[illegible]">Medical history</h1>
            {% endif %}
            <div class="flex flex-col">
                {% if data.otherUserAppointments %}
                {% for hospital, items in data.otherUserAppointments.items() %}
                <div>
                    <h3 class="text-left [illegible]">{{ hospital }}</h3>
                    {% for item in items %}
                    <div class="[illegible]">
                        <div data-[illegible] [illegible]
                            class="[illegible]">
                            {% set date_parts = item.displayDate.split("-") %}
                            <div class="grid grid-cols-[illegible]">
                                <div class="place-self-end flex gap-2">
                                    <p>{{ date_parts[2] }}</p>
                                    <p>{{ date_parts[1] }}</p>
                                    <p>{{ date_parts[0] }}</p>
                                </div>
                                <div class="place-self-start [illegible]">
                                    <p> at {{ item.timeSlot }}</p>
                                </div>
                                <div>
                                    <svg xmlns="http://www.w3.org/2000/svg" height="[illegible]"
                                        viewBox="[illegible]" width="[illegible]" fill="[illegible]">
                                        <path
                                            d="[illegible]" />
                                    </svg>
                                </div>
                            </div>
                        </div>
```



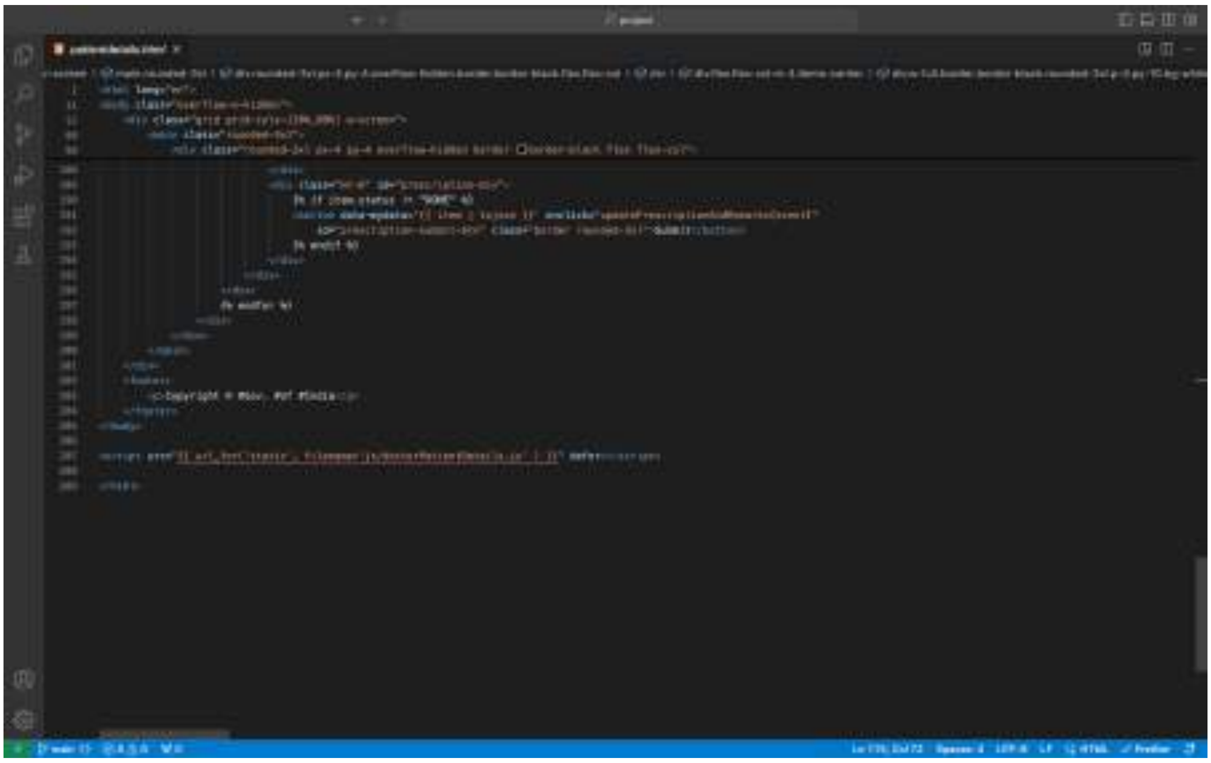
```
<html lang="en">
<body class="[illegible]">
    <div class="grid grid-cols-[illegible] [illegible]">
        <div class="[illegible] flex flex-col">
            <div class="flex flex-col [illegible]">
                                </div>
                            </div>
                        </div>
                        {% endfor %}
                    </div>
                    {% endfor %}
                    {% endif %}
                </div>
            </div>
        </div>

    </div>
    <main class="[illegible]">
        <header class="flex justify-between [illegible]" id="header">
            <div class="flex flex-row-reverse [illegible]">
                <button class="[illegible]" onclick="window.history.back()">Back</button>
            </div>
            <div class="flex flex-row-reverse [illegible]">
            </div>
        </header>
        <div class="[illegible] border-black flex flex-col">
            <h1 class="[illegible]">Patient Appointment details</h1>
            <div>
                {% for item in data.currentAppointment %}
                <div class="flex flex-col [illegible] items-center">
                    <div class="[illegible] border-black [illegible] bg-white">
                        <h2 class="[illegible]">Appointment details</h2>


                        <div class="grid grid-cols-[illegible]">
                            <svg class="place-self-end [illegible]" xmlns="http://www.w3.org/2000/svg" height="[illegible]"
                                viewBox="[illegible]" width="[illegible]" fill="[illegible]">
                                <path
                                    d="[illegible]" />
                            </svg>
                            <p id="[illegible]" class="place-self-start">{{ item.date }}</p>
                        </div>
                        <div class="grid grid-cols-[illegible]">
```



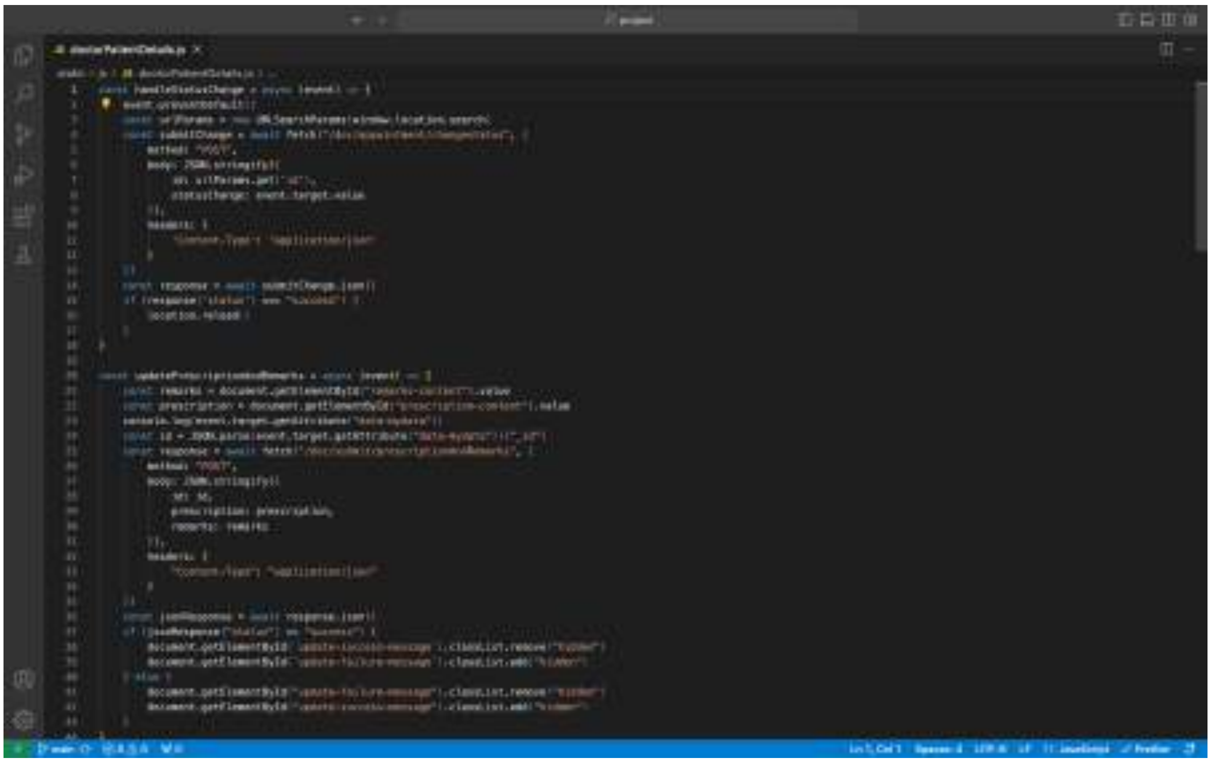
```
<html lang="en">
<body class="[illegible]">
    <div class="grid grid-rows-[[illegible]] [illegible]">
        <main class="[illegible]">
            <div class="rounded-2xl px-4 py-4 overflow-hidden border border-black flex flex-col">
                        </div>
                        <div class="grid grid-cols-[[illegible]] [illegible]">
                            <svg class="place-self-end [illegible]" xmlns="http://www.w3.org/2000/svg" height="[illegible]"
                                viewBox="0 -960 960 960" width="[illegible]" fill="[illegible]">
                                <path
                                    d="[illegible]" />
                            </svg>
                            <p id="[illegible]" class="place-self-start">{{ item.timeSlot }}</p>
                        </div>
                        <div class="grid grid-cols-[[illegible]] [illegible]">
                            <svg class="place-self-end [illegible]" xmlns="http://www.w3.org/2000/svg" height="[illegible]"
                                viewBox="0 -960 960 960" width="[illegible]" fill="[illegible]">
                                <path
                                    d="[illegible]" />
                            </svg>
                            <p id="[illegible]-status" class="place-self-start">{{ item.status }}</p>
                        </div>


                        <div id="[illegible]-status-change"
                            class="grid grid-cols-[[illegible]] items-center justify-center [illegible]">
                            {% if item.status != "DONE" %}
                            <label class="place-self-end self-center [illegible]">Change Status</label>
                            <select id="status-dropdown" class="border border-black rounded-[illegible] [illegible]"
                                onchange="[illegible]">
                                <option disabled selected>Select</option>
                                <option value="TODO">Todo</option>
                                <option value="INPROGRESS">Inprogress</option>
                                <option value="DONE">Done</option>
                            </select>
                            {% endif %}
                        </div>


                        <br>
                        <br>
                        <br>
                        <div id="[illegible]-remarks" class="grid [illegible]">
                            {% if item.status == "INPROGRESS" %}
```



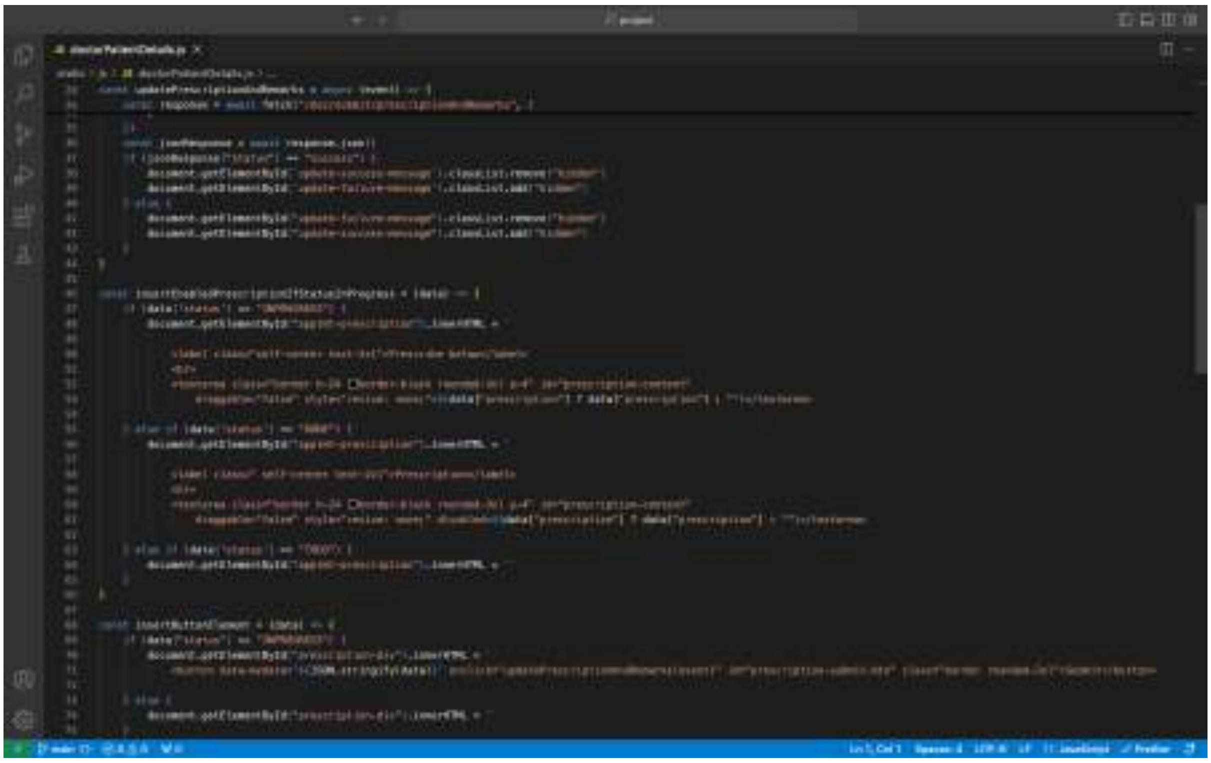
```
<html lang="en">
<body class="[illegible]">
    <div class="grid grid-rows-[[illegible]] [illegible]">
        <main class="[illegible]">
            <div class="rounded-2xl px-4 py-4 overflow-hidden border border-black flex flex-col">
                        <div id="[illegible]-remarks" class="grid [illegible]">
                            {% if item.status == "INPROGRESS" %}
                            <label class="self-center [illegible]">Enter Remarks</label>
                            <br>
                            <textarea class="border [illegible] border-black rounded-[illegible] [illegible]" id="remarks-content"
                                draggable="false"
                                style="resize: none;">{% if item.remarks %}{{ item.remarks }}{% endif %}</textarea>
                            {% endif %}
                            {% if item.status == "DONE" %}
                            <label class="[illegible] self-center [illegible]">Remarks</label>
                            <br />
                            <textarea class="border [illegible] border-black rounded-[illegible] [illegible]" id="remarks-content-disabled"
                                draggable="false" disabled style="resize: none;">{{item.remarks}}
                            </textarea>
                            {% endif %}
                        </div>
                        <div id="[illegible]-prescription" class="grid [illegible]">
                            {% if item.status == "INPROGRESS" %}
                            <label class="[illegible] self-center [illegible]">Prescribe below</label>
                            <br>
                            <textarea class="border [illegible] border-black rounded-[illegible] [illegible]" id="prescription-content"
                                draggable="false"
                                style="resize: none;">{% if item.prescription %}{{ item.prescription }}{% endif %}</textarea>
                            {% endif %}
                            {% if item.status == "DONE" %}

                            <label class="[illegible] self-center [illegible]">Prescription</label>
                            <br>
                            <textarea class="border [illegible] border-black rounded-[illegible] [illegible]"
                                id="prescription-content-disabled" draggable="false" disabled style="resize: none;">{{item.prescription}}
                            </textarea>
                            {% endif %}
                        </div>
                        <div class="text-center">
                            <p class="hidden" id="update-success-message">Updated Successfully</p>
                            <p class="hidden" id="update-failure-message">Update failed. Please try again after some
                                time</p>
                        </div>
```



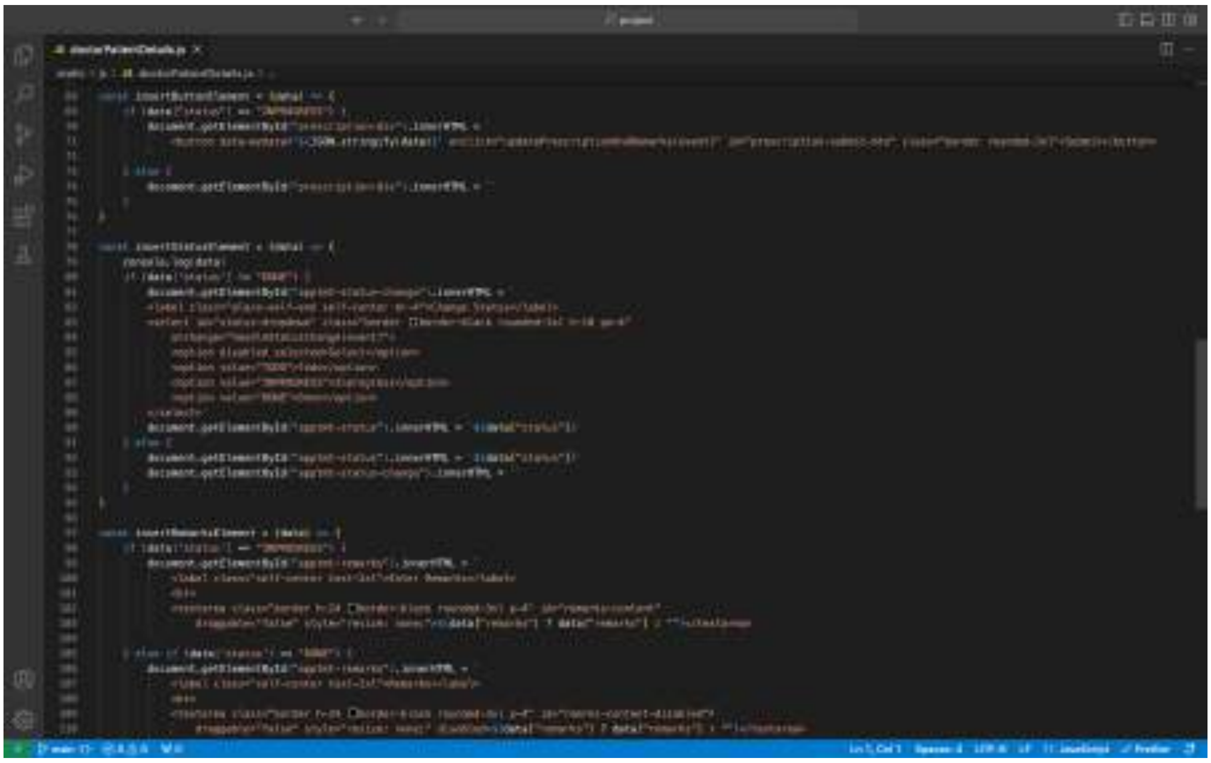
```html
<html lang="en">
<body class="overflow-x-hidden">
    <div class="grid grid-rows-[100,200] h-screen">
        <main class="rounded-3xl">
            <div class="rounded-2xl px-4 py-4 overflow-hidden border border-black flex flex-col">
                            </div>
                            <div class="mt-4" id="prescription-div">
                                {% if item.status != "DONE" %}
                                <button data-mydata='{{ item | tojson }}' onclick="updatePrescriptionAndRemarks(event)"
                                    id="prescription-submit-btn" class="border rounded-xl">Submit</button>
                                {% endif %}
                            </div>
                        </div>
                    </div>
                {% endfor %}
            </div>
        </div>
    </main>
    </div>
    <footer>
        <p>Copyright © Misc. Ref #India</p>
    </footer>
</body>

<script src="{{ url_for('static', filename='js/doctorPatientDetails.js') }}" defer></script>

</html>
```



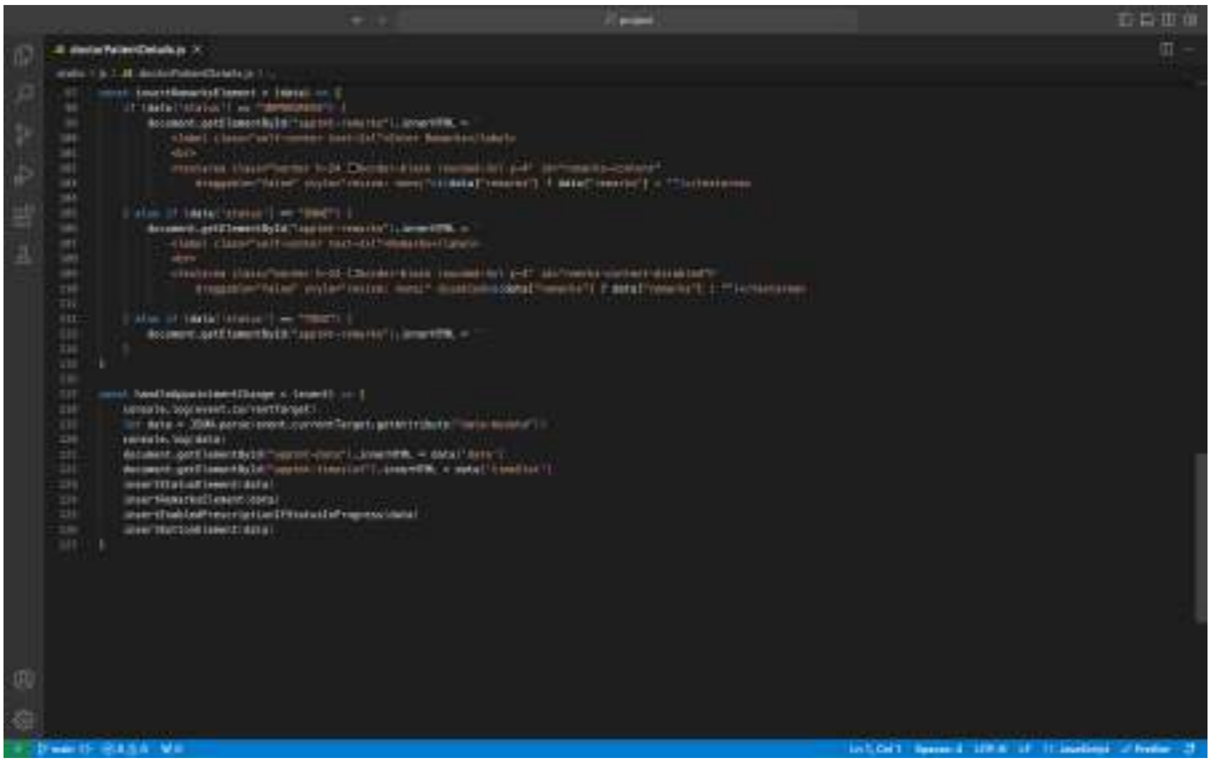
```javascript
const handleStatusChange = async (event) => {
    event.preventDefault()
    const urlParams = new URLSearchParams(window.location.search);
    const submitChange = await fetch("/doc/appointment/changestatus", {
        method: "POST",
        body: JSON.stringify({
            id: urlParams.get("id"),
            statusChange: event.target.value
        }),
        headers: {
            "Content-Type": "application/json"
        }
    })
    const response = await submitChange.json()
    if (response["status"] === "success") {
        location.reload()
    }
}

const updatePrescriptionAndRemarks = async (event) => {
    const remarks = document.getElementById("remarks-content").value
    const prescription = document.getElementById("prescription-content").value
    console.log(event.target.getAttribute("data-mydata"))
    const id = JSON.parse(event.target.getAttribute("data-mydata"))["_id"]
    const response = await fetch("/doc/submit/prescriptionandremarks", {
        method: "POST",
        body: JSON.stringify({
            id: id,
            prescription: prescription,
            remarks: remarks
        }),
        headers: {
            "Content-Type": "application/json"
        }
    })
    const jsonResponse = await response.json()
    if (jsonResponse["status"] == "success") {
        document.getElementById("update-success-message").classList.remove("hidden")
        document.getElementById("update-failure-message").classList.add("hidden")
    } else {
        document.getElementById("update-failure-message").classList.remove("hidden")
        document.getElementById("update-success-message").classList.add("hidden")
    }
}
```



```
const updatePrescriptionAndRemarks = async (event) => {
    const response = await fetch("[illegible]", {
        [illegible]
    });
    const jsonResponse = await response.json();
    if (jsonResponse["status"] == "success") {
        document.getElementById("update-success-message").classList.remove("hidden");
        document.getElementById("update-failure-message").classList.add("hidden");
    } else {
        document.getElementById("update-failure-message").classList.remove("hidden");
        document.getElementById("update-success-message").classList.add("hidden");
    }
}

const insertEnabledPrescriptionIfStatusInProgress = (data) => {
    if (data["status"] == "INPROGRESS") {
        document.getElementById("append-prescription").innerHTML = `

            <label class="self-center text-2xl">Prescribe below</label>
            <div>
            <textarea class="border h-24 border-black rounded-3xl p-4" id="prescription-content"
                draggable="false" style="resize: none;">${data["prescription"] ? data["prescription"] : ""}</textarea>

    } else if (data["status"] == "DONE") {
        document.getElementById("append-prescription").innerHTML = `

            <label class="self-center text-2xl">Prescription</label>
            <div>
            <textarea class="border h-24 border-black rounded-3xl p-4" id="prescription-content"
                draggable="false" style="resize: none;" disabled>${data["prescription"] ? data["prescription"] : ""}</textarea>

    } else if (data["status"] == "TODO") {
        document.getElementById("append-prescription").innerHTML = `

    }
}

const insertButtonElement = (data) => {
    if (data["status"] == "INPROGRESS") {
        document.getElementById("prescription-div").innerHTML = `
            <button data-[illegible]='${JSON.stringify(data)}' onclick="updatePrescriptionAndRemarks(event)" id="prescription-submit-btn" class="border rounded-2xl">Submit</button>

    } else {
        document.getElementById("prescription-div").innerHTML = `
    }
```



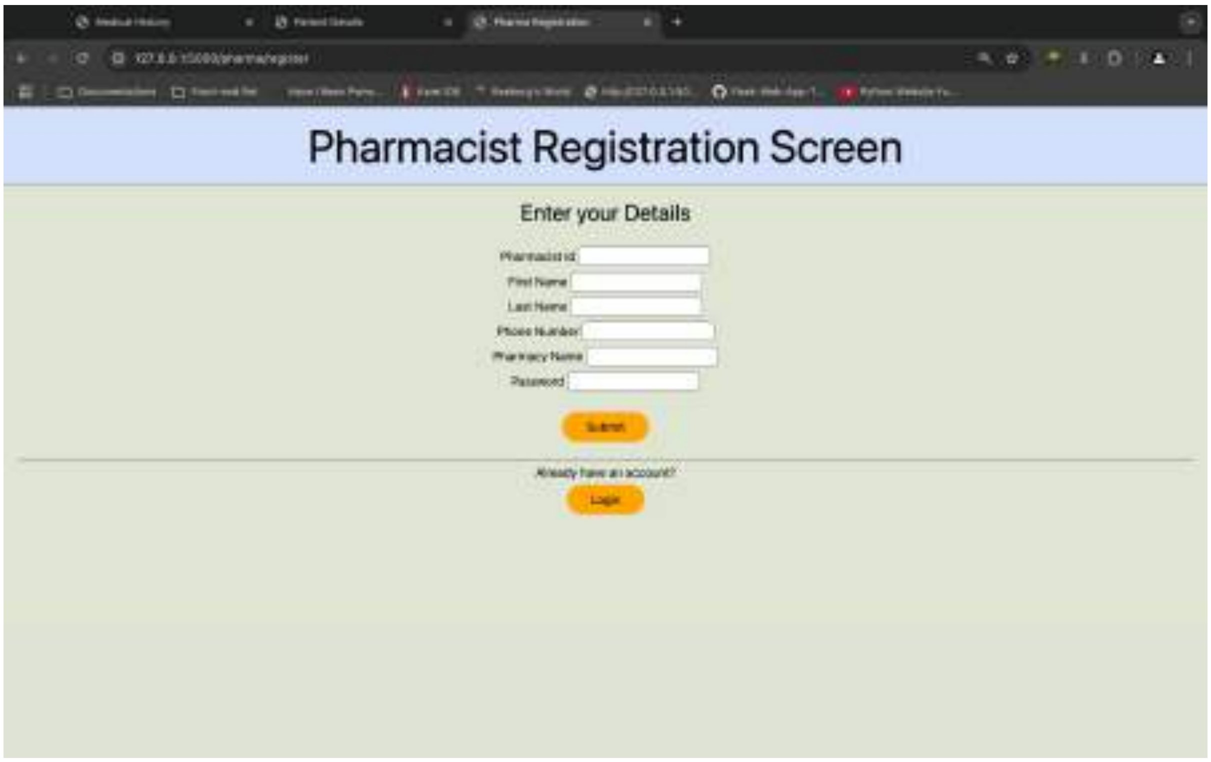
```
const insertButtonElement = (data) => {
    if (data["status"] == "INPROGRESS") {
        document.getElementById("prescription-div").innerHTML = `
            <button data-[illegible]='${JSON.stringify(data)}' onclick="updatePrescriptionAndRemarks(event)" id="prescription-submit-btn" class="border rounded-2xl">Submit</button>

    } else {
        document.getElementById("prescription-div").innerHTML = `
    }
}

const insertStatusElement = (data) => {
    console.log(data)
    if (data["status"] != "DONE") {
        document.getElementById("append-status-change").innerHTML = `
        <label class="place-self-end self-center mr-4">Change Status</label>
        <select id="status-dropdown" class="border border-black rounded-3xl h-10 px-4"
            onchange="[illegible](event)">
            <option disabled selected>Select</option>
            <option value="TODO">Todo</option>
            <option value="INPROGRESS">In progress</option>
            <option value="DONE">Done</option>
        </select>`
        document.getElementById("append-status").innerHTML = `${data["status"]}`
    } else {
        document.getElementById("append-status").innerHTML = `${data["status"]}`
        document.getElementById("append-status-change").innerHTML = ``
    }
}

const insertRemarksElement = (data) => {
    if (data["status"] == "INPROGRESS") {
        document.getElementById("append-remarks").innerHTML = `
            <label class="self-center text-2xl">Enter Remarks</label>
            <div>
            <textarea class="border h-24 border-black rounded-3xl p-4" id="remarks-content"
                draggable="false" style="resize: none;">${data["remarks"] ? data["remarks"] : ""}</textarea>

    } else if (data["status"] == "DONE") {
        document.getElementById("append-remarks").innerHTML = `
            <label class="self-center text-2xl">Remarks</label>
            <div>
            <textarea class="border h-24 border-black rounded-3xl p-4" id="remarks-content-disabled"
                draggable="false" style="resize: none;" disabled>${data["remarks"] ? data["remarks"] : ""}</textarea>
```



```
const insertRemarksElement = (data) => {
    if (data['status'] == "INPROGRESS") {
        document.getElementById("appt-remarks").innerHTML = `
            <label class="self-center text-[illegible]">Enter Remarks</label>
            <div>
            <textarea class="border h-24 [illegible]border-black rounded-[illegible] p-4" id="remarks-content"
                draggable="false" style="resize: none;">${data['remarks'] ? data['remarks'] : ""}</textarea>

    } else if (data['status'] == "DONE") {
        document.getElementById("appt-remarks").innerHTML = `
            <label class="self-center text-[illegible]">Remarks</label>
            <div>
            <textarea class="border h-[illegible] [illegible]border-black rounded-[illegible] p-4" id="remarks-content-disabled"
                draggable="false" style="resize: none;" disabled>${data['remarks'] ? data['remarks'] : ""}</textarea>

    } else if (data['status'] == "TODO") {
        document.getElementById("appt-remarks").innerHTML = ``
    }
}

const handleAppointmentChange = (event) => {
    console.log(event.currentTarget)
    let data = JSON.parse(event.currentTarget.getAttribute("data-details"))
    console.log(data)
    document.getElementById("appt-date").innerHTML = data['date']
    document.getElementById("appt-timeslot").innerHTML = data['timeslot']
    insertStatusElement(data)
    insertRemarksElement(data)
    insertEnabledPrescriptionIfStatusInProgress(data)
    insertButtonElement(data)
}
```

8 Chapter 8

8.1 Pharma Registration screen



127.0.0.1:5000/pharmaregister

Pharmacist Registration Screen

Enter your Details

Pharmacist id

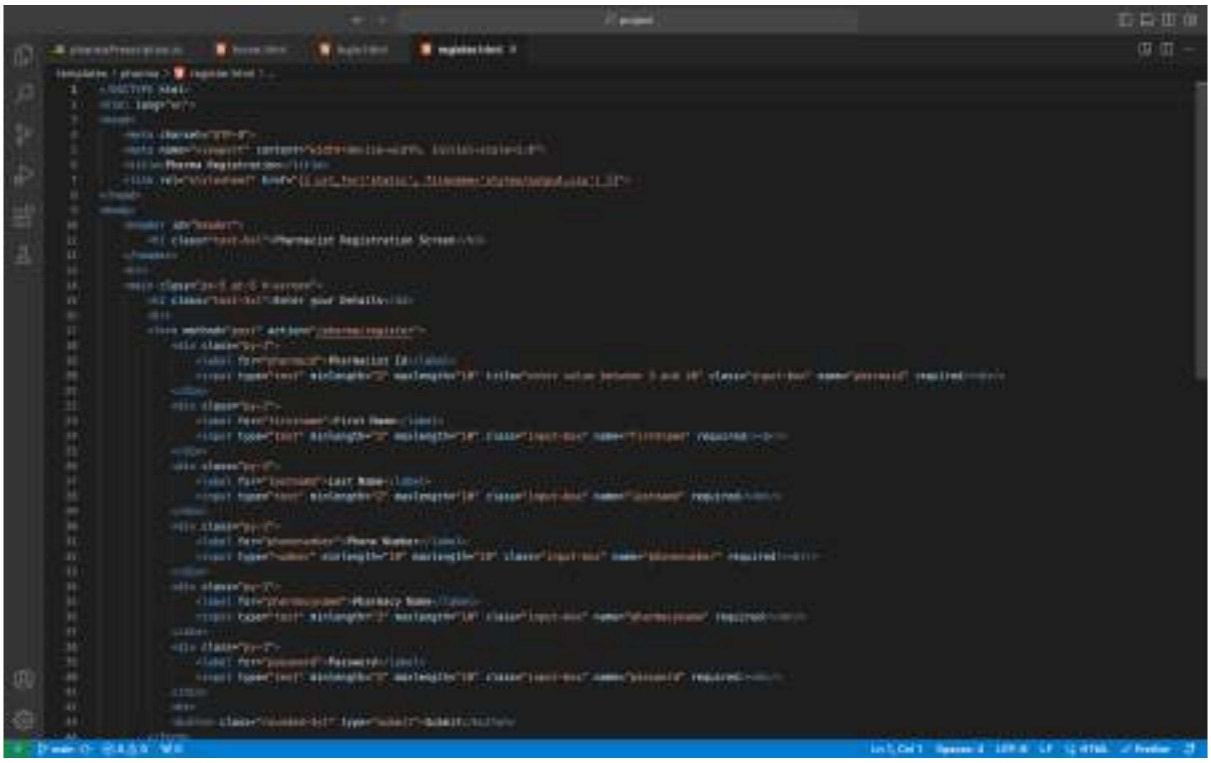
First Name

Last Name

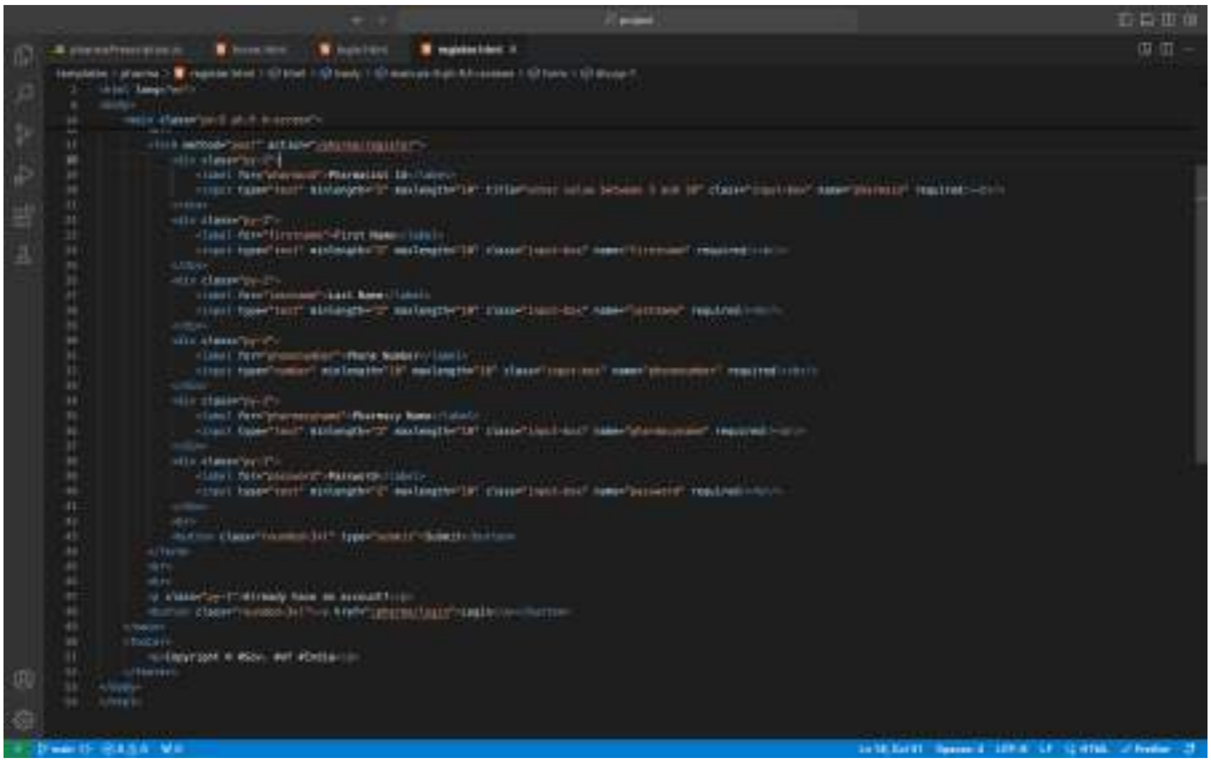
Phone Number

Pharmacy Name

Password

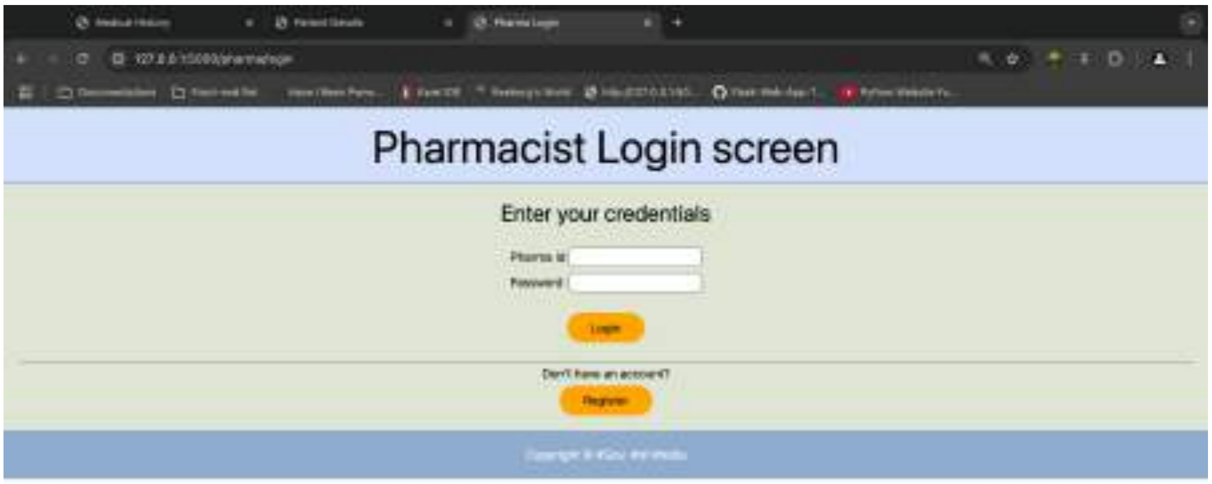


```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Pharma Registration</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='styles/output.css') }}">
</head>
<body>
    <header id="header">
        <h1 class="text-4xl">Pharmacist Registration Screen</h1>
    </header>
    <hr>
    <main class="px-5 mt-5 h-screen">
        <h2 class="text-3xl">Enter your Details</h2>
        <hr>
        <form method="post" action="/pharma/register">
            <div class="py-2">
                <label for="pharmaid">Pharmacist Id</label>
                <input type="text" minlength="3" maxlength="10" title="enter value between 3 and 10" class="input-box" name="pharmaid" required><br>
            </div>
            <div class="py-2">
                <label for="firstname">First Name</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="firstname" required><br>
            </div>
            <div class="py-2">
                <label for="lastname">Last Name</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="lastname" required><br>
            </div>
            <div class="py-2">
                <label for="phonenumber">Phone Number</label>
                <input type="number" minlength="10" maxlength="10" class="input-box" name="phonenumber" required><br>
            </div>
            <div class="py-2">
                <label for="pharmacyname">Pharmacy Name</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="pharmacyname" required><br>
            </div>
            <div class="py-2">
                <label for="password">Password</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="password" required><br>
            </div>
            <hr>
            <button class="rounded-btn" type="submit">Submit</button>
```



```html
<html lang="en">
<body>
    <main class="px-5 mt-5 h-screen">
        <form method="post" action="/pharma/register">
            <div class="py-2">
                <label for="pharmaid">Pharmacist Id</label>
                <input type="text" minlength="3" maxlength="10" title="enter value between 3 and 10" class="input-box" name="pharmaid" required><br>
            </div>
            <div class="py-2">
                <label for="firstname">First Name</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="firstname" required><br>
            </div>
            <div class="py-2">
                <label for="lastname">Last Name</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="lastname" required><br>
            </div>
            <div class="py-2">
                <label for="phonenumber">Phone Number</label>
                <input type="number" minlength="10" maxlength="10" class="input-box" name="phonenumber" required><br>
            </div>
            <div class="py-2">
                <label for="pharmacyname">Pharmacy Name</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="pharmacyname" required><br>
            </div>
            <div class="py-2">
                <label for="password">Password</label>
                <input type="text" minlength="3" maxlength="10" class="input-box" name="password" required><br>
            </div>
            <hr>
            <button class="rounded-btn" type="submit">Submit</button>
        </form>
        <hr>
        <hr>
        <p class="py-2">Already have an account?</p>
        <button class="rounded-btn"><a href="/pharma/login">Login</a></button>
    </main>
    <footer>
        <p>Copyright © #Gov. #of #India</p>
    </footer>
</body>
</html>
```

8.2 Pharma Login screen



127.0.0.1:5000/pharma/login

Pharmacist Login screen

Enter your credentials

Pharma Id

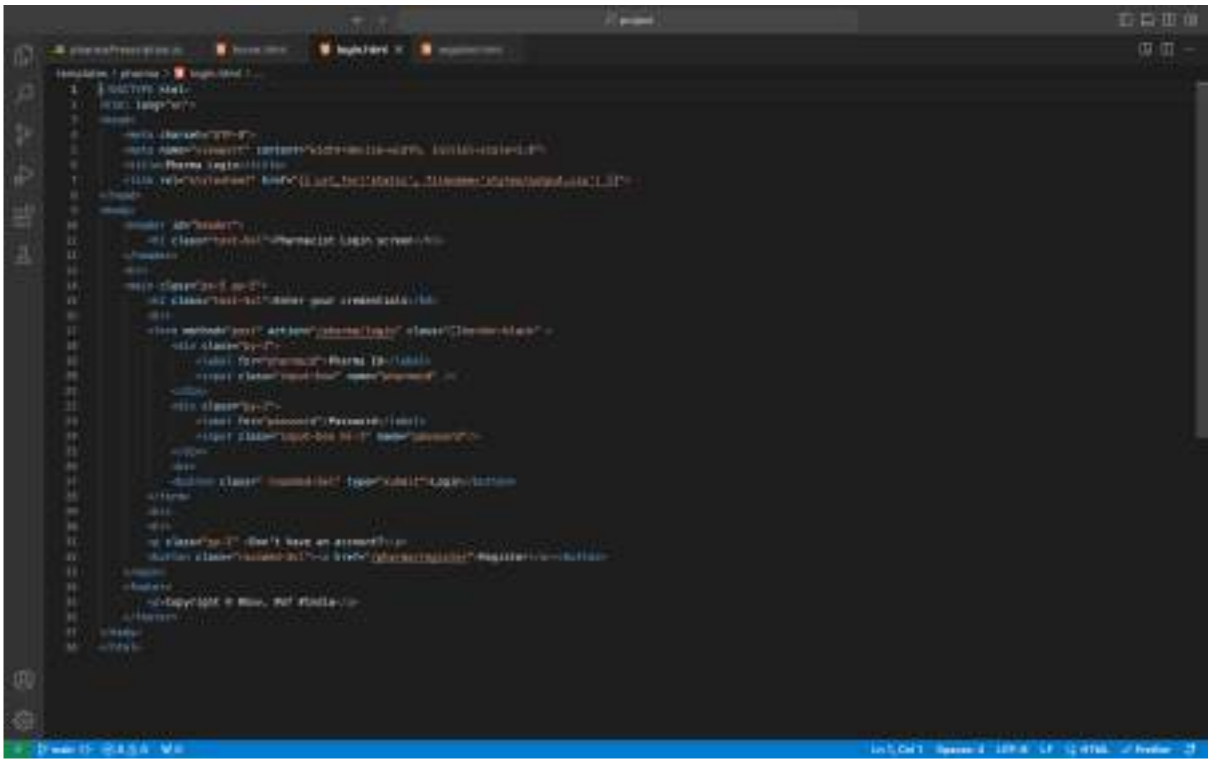
Password

Login

Don't have an account?

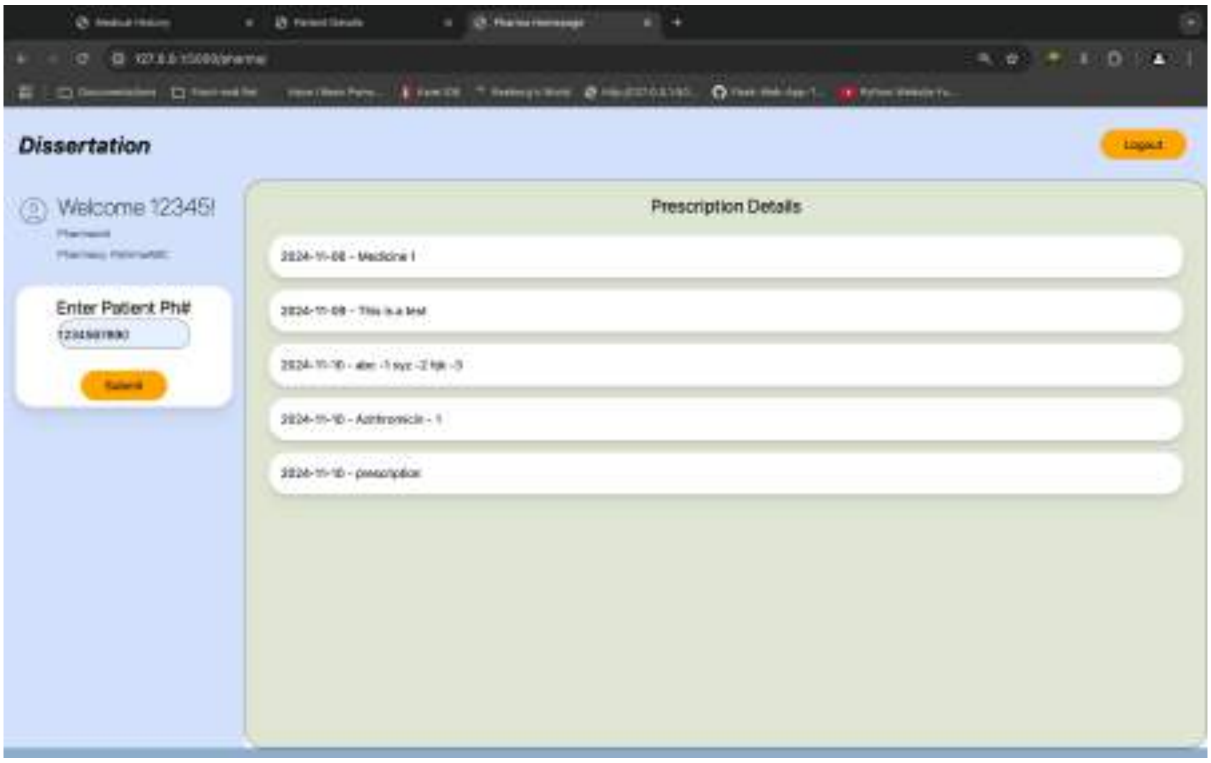
Register

Copyright © [illegible]



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Pharma Login</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='[illegible]') }}">
</head>
<body>
    <header id="header">
        <h1 class="[illegible]">Pharmacist Login screen</h1>
    </header>
    <hr>
    <main class="[illegible]">
        <h2 class="[illegible]">Enter your credentials</h2>
        <br>
        <form method="post" action="/pharma/login" class="[illegible]">
            <div class="[illegible]">
                <label for="pharmaid">Pharma ID</label>
                <input class="input-box" name="pharmaid" />
            </div>
            <div class="[illegible]">
                <label for="password">Password</label>
                <input class="input-box [illegible]" name="password"/>
            </div>
            <br>
            <button class="[illegible]" type="submit">Login</button>
        </form>
        <hr>
        <br>
        <p class="[illegible]">Don't have an account?</p>
        <button class="[illegible]"><a href="/pharma/register">Register</a></button>
    </main>
    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
</html>
```

8.3 Pharma Home Page



Dissertation

Logout

Welcome 12345!

Enter Patient Ph#

1234567890

Submit

Prescription Details

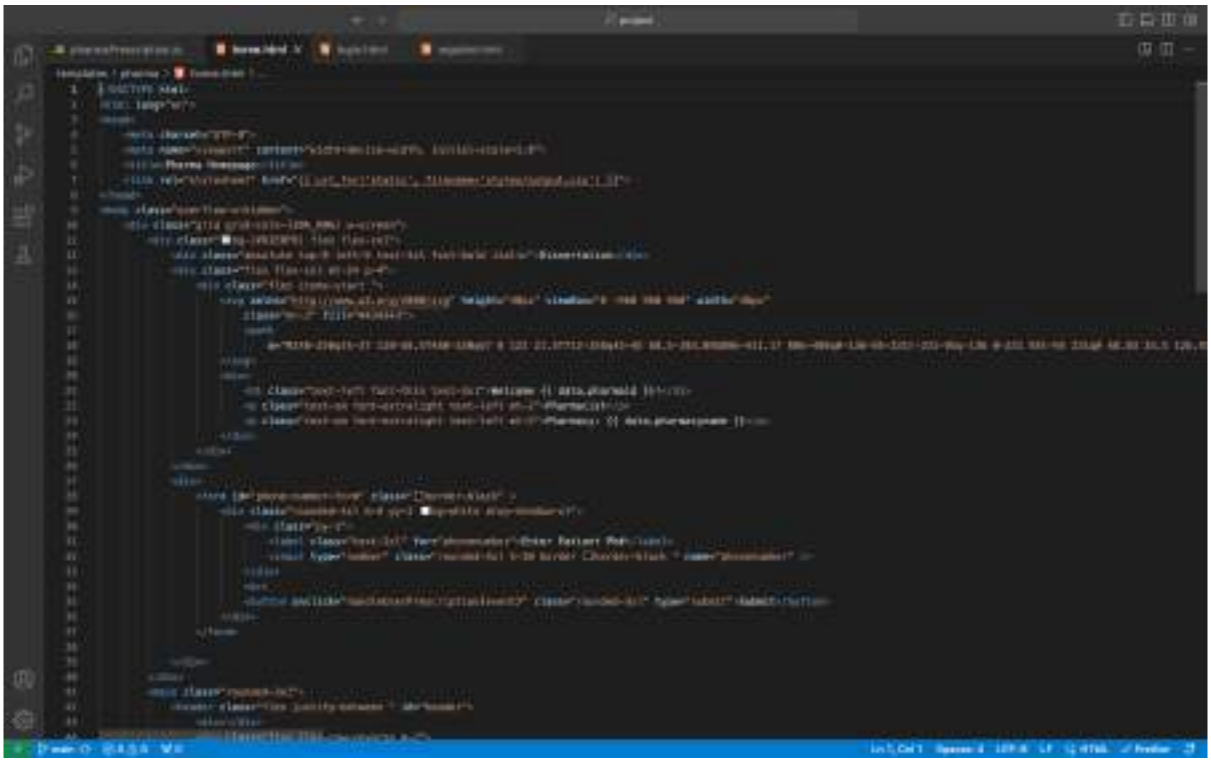
2024-11-08 - Medicine I

2024-11-08 - This is a test

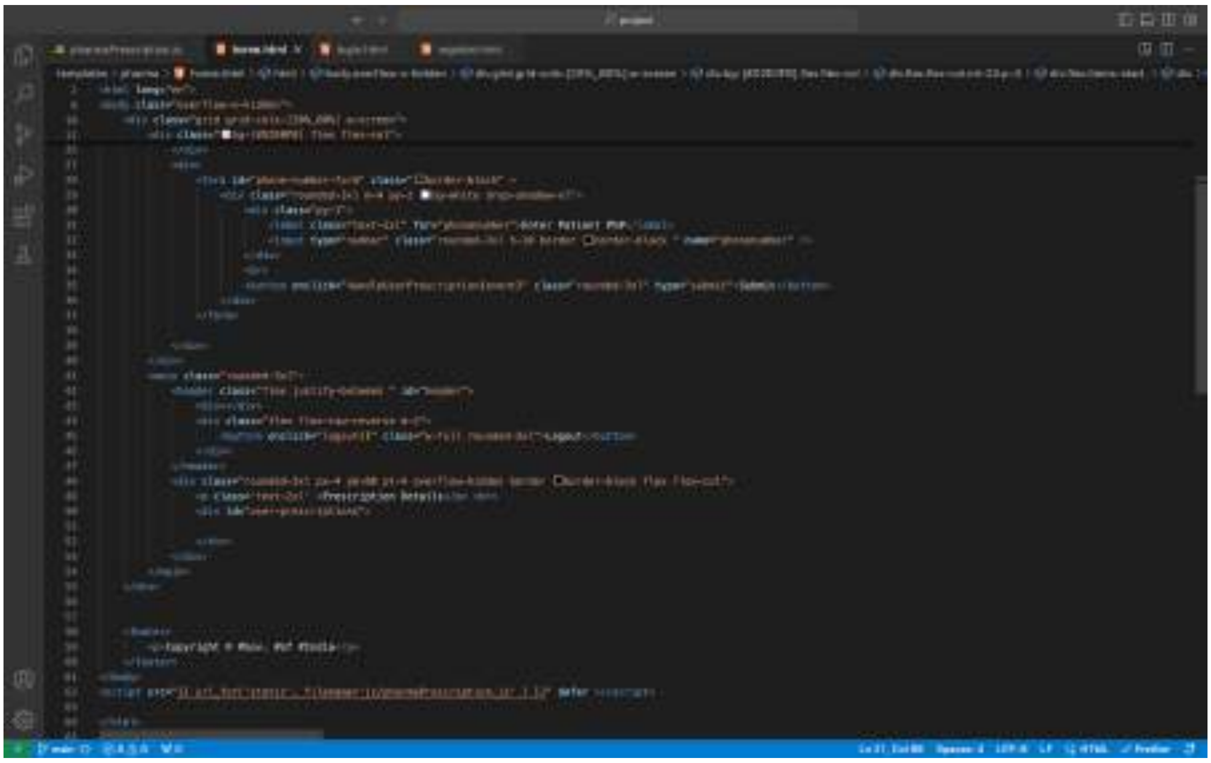
2024-11-10 - abc -1 xyz -2 hjk -3

2024-11-10 - Azithromicin - 1

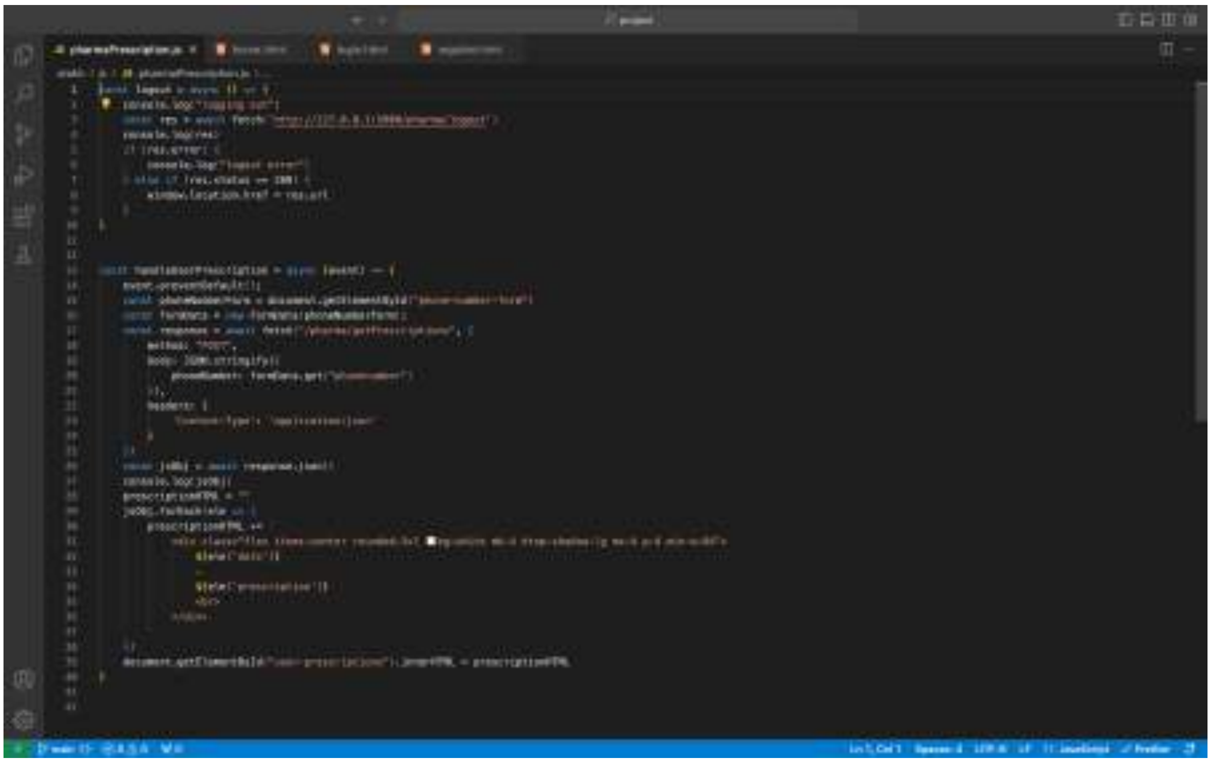
2024-11-10 - prescription



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Pharma Homepage</title>
    <link rel="stylesheet" href="{{ url_for('static', filename=[illegible]) }}">
</head>
<body class=[illegible]>
    <div class=[illegible]>
        <div class=[illegible]>
            <div class=[illegible]>Dissertation</div>
            <div class=[illegible]>
                <div class=[illegible]>
                    <svg xmlns=[illegible] height=[illegible] viewBox=[illegible] width=[illegible]
                        class=[illegible] fill=[illegible]>
                        <path
                            d=[illegible]
                    </svg>
                    <div>
                        <h1 class=[illegible]>Welcome {{ data.pharmaid }}!</h1>
                        <p class=[illegible]>Pharmacist</p>
                        <p class=[illegible]>Pharmacy: {{ data.pharmacyname }}</p>
                    </div>
                </div>
            </div>
            <div>
                <form id="phone-number-form" class=[illegible]>
                    <div class=[illegible]>
                        <div class=[illegible]>
                            <label class=[illegible] for="phonenumber">Enter Patient Ph#</label>
                            <input type="number" class=[illegible] name="phonenumber" />
                        </div>
                        <br>
                        <button onclick=[illegible] class=[illegible] type="submit">Submit</button>
                    </div>
                </form>

            </div>
        </div>
        <main class=[illegible]>
            <header class=[illegible] id="header">
                [illegible]
```



```html
<html lang="en">
<body class="overflow-x-hidden">
    <div class="grid grid-cols-[20%_80%] w-screen">
        <div class="bg-[#02DB95] flex flex-col">
            </div>
            <div>
                <form id="phone-number-form" class="border-black" >
                    <div class="rounded-3xl m-4 px-2 bg-white drop-shadow-xl">
                        <div class="py-2">
                            <label class="text-2xl" for="phonenumber">Enter Patient PH#</label>
                            <input type="number" class="rounded-3xl h-10 border border-black " name="phonenumber" />
                        </div>
                        <br>
                        <button onclick="handleUserPrescription(event)" class="rounded-3xl" type="submit">Submit</button>
                    </div>
                </form>

            </div>
        </div>
        <main class="rounded-3xl">
            <header class="flex justify-between " id="header">
                <div></div>
                <div class="flex flex-row-reverse m-2">
                    <button onclick="logout()" class="w-full rounded-3xl">Logout</button>
                </div>
            </header>
            <div class="rounded-3xl px-4 pb-80 pt-4 overflow-hidden border border-black flex flex-col">
                <p class="text-2xl">Prescription Details</p>
                <div id="user-prescriptions">

                </div>
            </div>
        </main>
    </div>


    <footer>
        <p>Copyright © [illegible]</p>
    </footer>
</body>
<script src="{{ url_for('static', filename='js/pharmaPrescription.js') }}" defer></script>

</html>
```



```javascript
const logout = async () => {
    console.log("logging out")
    const res = await fetch("http://127.0.0.1:8000/pharma/logout")
    console.log(res)
    if (res.error) {
        console.log("logout error")
    } else if (res.status == 200) {
        window.location.href = res.url
    }
}


const handleUserPrescription = async (event) => {
    event.preventDefault();
    const phoneNumberForm = document.getElementById("phone-number-form")
    const formData = new FormData(phoneNumberForm);
    const response = await fetch("/pharma/getPrescriptions/", {
        method: "POST",
        body: JSON.stringify({
            phoneNumber: formData.get("phonenumber")
        }),
        headers: {
            "Content-Type": "application/json"
        }
    })
    const jsObj = await response.json()
    console.log(jsObj)
    prescriptionHTML = ""
    jsObj.forEach(ele => {
        prescriptionHTML += `
        <div class="flex items-center rounded-3xl bg-white m-4 drop-shadow-lg m-4 p-4 min-w-80">
            ${ele['date']}
            -
            ${ele['prescription']}
            <br>
        </div>

    `})
    document.getElementById("user-prescriptions").innerHTML = prescriptionHTML
}
```

9 Chapter 9

9.1 Bibliography

- PyMongo documentation - https://pymongo.readthedocs.io/en/stable/tutorial.html
- Flask framework installation - https://flask.palletsprojects.com/en/stable/quickstart/
- Jinja template documentation - https://jinja.palletsprojects.com/en/stable/templates/
- Tailwindcss - https://tailwindcss.com/docs/installation

9.2 Project Status

Serial Number of Task	Tasks or subtasks to be done (be precise and specific)	Planned duration in weeks	Specific Deliverables in terms of the project	Status
1	Requirement Gathering	Week 1	Sprint 1	Completed
2	Documentation of design for the website	Week 2 - 3	Sprint 2	Completed
3	Development of Dashboard of the website	Week 4	Sprint 3	Completed
4	Development of Doctor's module	Week 5 - 6	Sprint 4	Completed
5	Development of Pharmacist's module	Week 7 - 8	Sprint 5	Completed
6	Development of Lab's module	Week 9 - 10	Sprint 6	Future Scope
7	Development of Patient's module	Week 11 - 12	Sprint 7	Completed
8	Customizations	Week 13	Sprint 8	Completed

9	Integration of Dashboard, Doctor’s module, Pharmacist’s module and Patient’s module	Week 14 - 15	Sprint 9	Completed
10	Integration of Database at the back end	Week 16	Sprint 10	Completed

9.2 Conclusion

- The Lab module is moved to future scope.
- The Integration of the database and the website is done.
- The website is fully functional with one module moved to future scope.