DEVREV - DOCUMENTATION FOR COVID SLOT BOOKING APP

Ananya Khare - BT19GCS100

Tech Stack :Python - Flask backend, Tailwind CSS for basic UI, HTML

To run files locally

1. First, we need to install all the dependencies. One must have Python to run this app
2. Next run pip install -r requirements. One may want to create a virtual environment so that these dependencies do not get installed on your system (python -m venv .\venv)
3. Inside your folder execute .\venv\Scripts\activate to activate the virtual environment.
4. One must also instantiate the database. Go into the REPL(Read-Eval-Print Loop) by typing “python” in your terminal. Next, from app import db, db.create_all(). This creates a database
5. Finally to run the app. In the terminal, type, **python app.py**

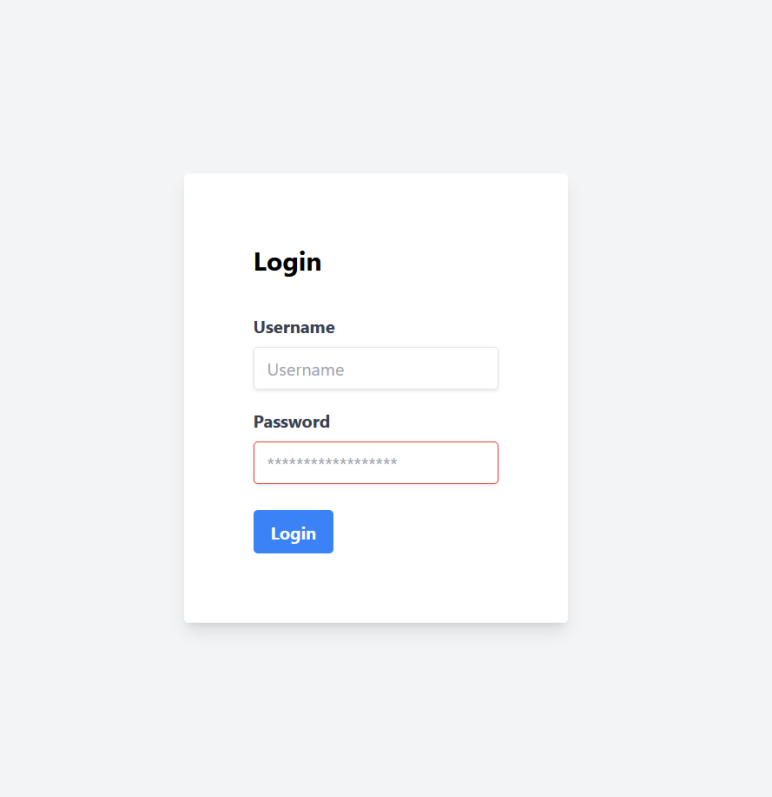
Application Walk through and End Points

We have 3 models : ->User with an is_admin attribute to define if a user is an admin or not
->Vaccination_Centre: A model of all vaccination centers, their opening and closing time, locations and a JSON to keep track of days vs remaining slots in the day

-> Bookings: A model for keeping track of all booking and connects to vaccination centre and user by their unique id

USER ENDPOINTS

1. **/login, /**
   Login Screen for user role, takes username and password.
2. /register



3. **/signup:**
   This is a form to register a new user, once the user has been registered it shows a message indicating user is registered
4. **/register**
   This is to test the user input and validate it against a few constraints and push the data into sqlite db
5. **/loginvalidate**
   To check the password hash and valid user

```python
# Validating Login
@app.route('/loginvalidate', methods=['POST','GET'])
def login():
    if request.method == "POST":
        username = request.form.get('username')
        password = request.form.get('password')
        if not username or not password:
            return jsonify({'error': 'Username and password are required'}), 400
        user = User.query.filter_by(username=username).first()
        if not user or not bcrypt.check_password_hash(user.password, password):
            return jsonify({'error': 'Invalid username or password'}), 401
        login_user(user) # Logs in the user
        return render_template("slot.html")
    else:
        return redirect(url_for('userloginform'))
```

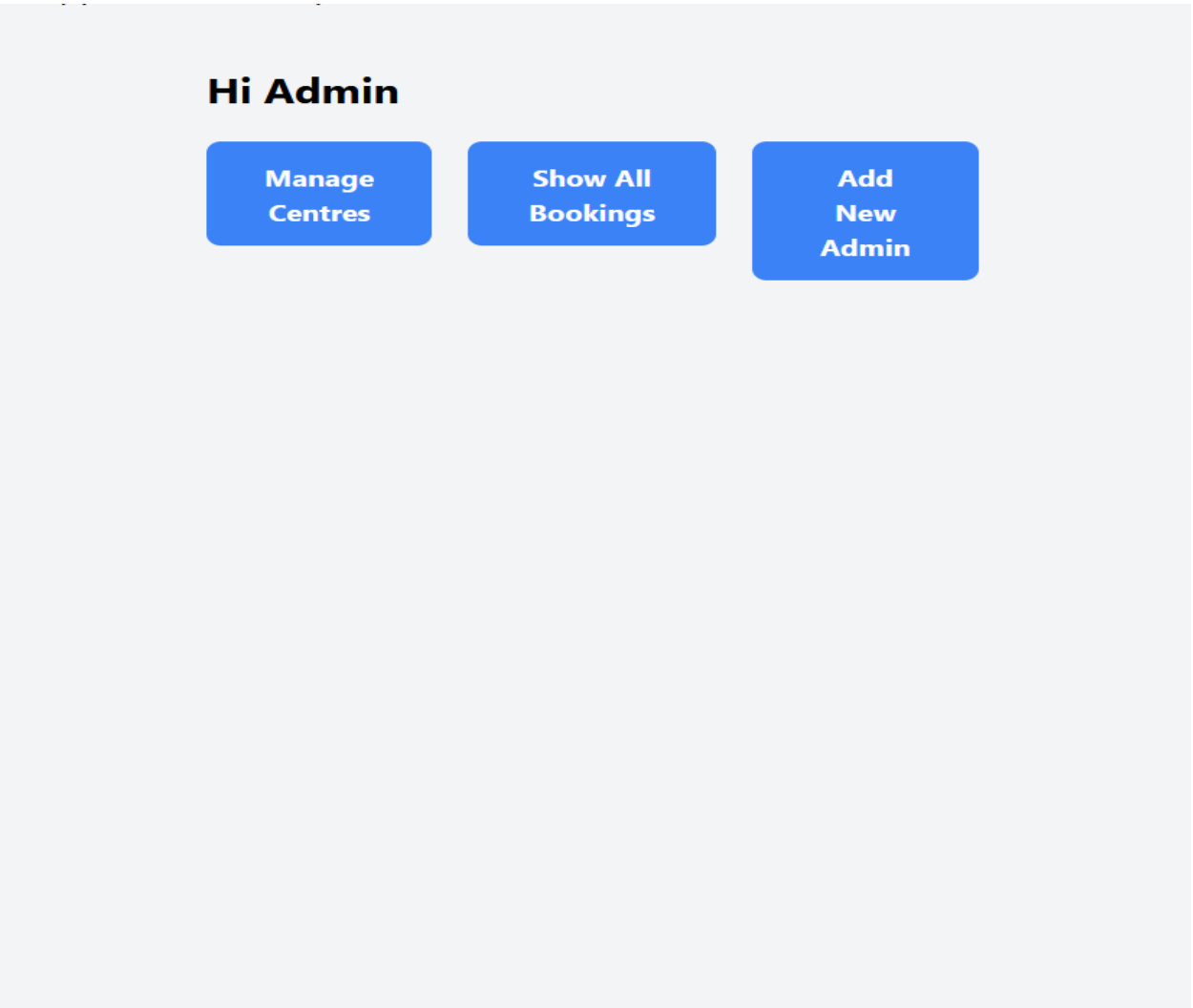
6. **/bookslot**
   A page with hourly time intervals, the data from here is sourced in the availcentres with the help of session[], to filterout all the centres that are not available in that duration

```python
#User apply for slot route
@app.route('/bookslot', methods=['POST','GET']) #button
@login_required
def selectslot():
    session['slot'] = request.form.get('slot')
    return redirect(url_for('availvc'))
```

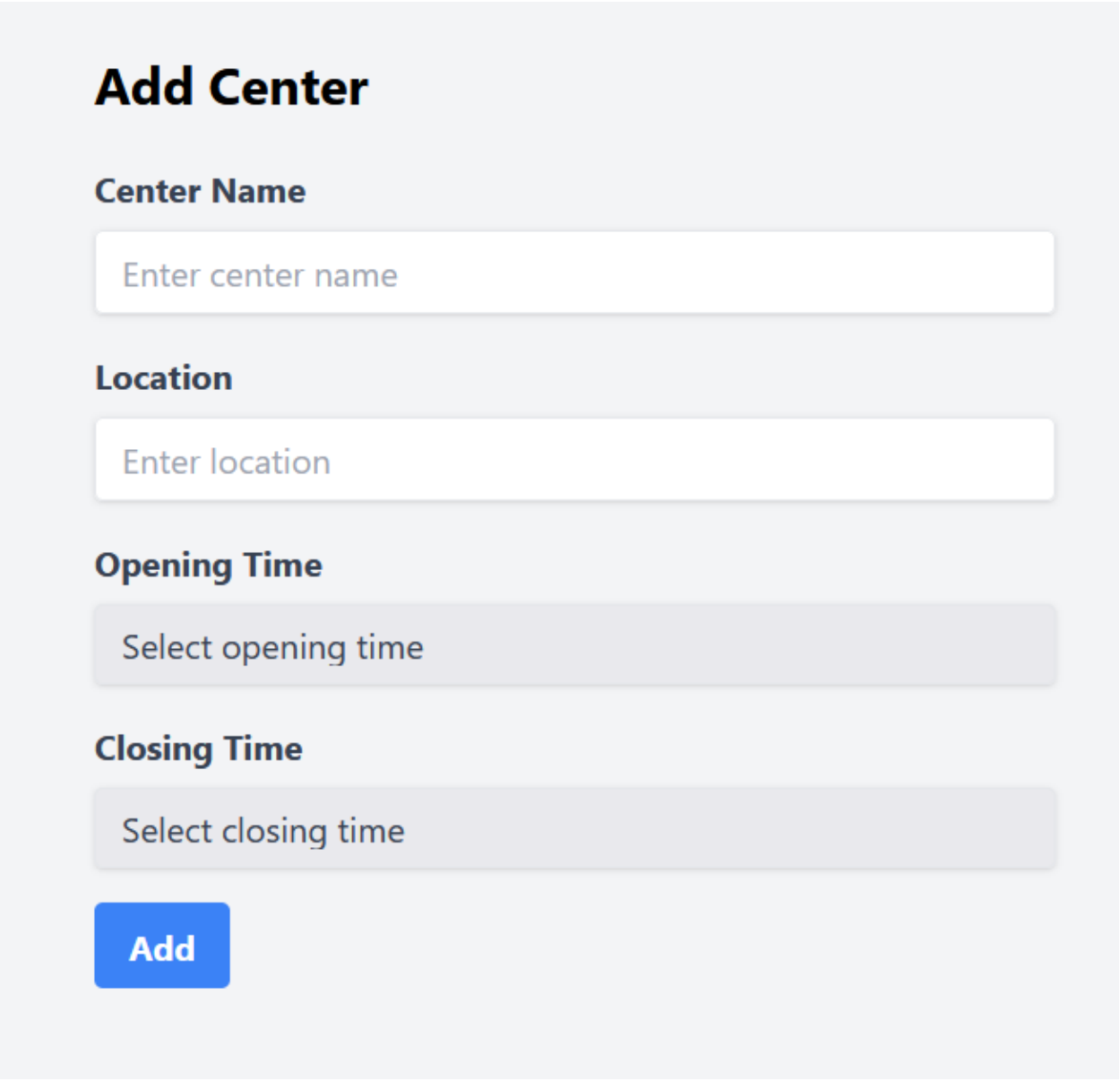
7. **/availcentres:**
   Displays all available centres for the chosen slot
8. **/selectdate:**
   Finally one needs to select date, this date is stored in a JSON in which date takes the key position and value is number of slots
9. **/date:**
   Push date to session
10. **/book:**
    Checks whether the centre is available for the chosen date
11. **/logout:**
    Logsout, basically sets the is_autheticated attribute to False

ADMIN ENDPOINTS

1. **admin/loginform**
   Login form for admin, there is a default admin username : admin and password: secretkey. Any admin registration can only be done by an already existing admin thus a default case was need to initialise
2. **admin/login, To login registered admins**
3. **admin/signup, Posts the data to the server**
4. **admin/register, Validation checks for new admin registeration**



5. **admin/allbooking, Displays all bookings to the admin**
6.
7. **admin/DisplayCentres, the manage centre button takes you to a list of all the centres**
8. **admin/add_vaccination_centre: there is an add button of top of this list to add a new centre through the following form**



9. **admin/dosage_details**
   This was a little vague for me, so assuming that this is the admin asking for how many doses are required at a center, we can join the booking and vaccination centre and then order by vaccination centre to return the required amount of dosage at a centre

```python
#Admin get dosage details route
@app.route('/admin/dosage_details', methods=['GET'])
@login_required
def get_dosage_details():
    if not current_user.is_admin:
        return jsonify({'error': 'Only admin can perform this action'}), 401
    dosage_details = db.session.query(VaccinationCentre.name, db.func.count(Booking.id).lab
    response = []
    for dd in dosage_details:
        response.append({
        'vaccination_centre': dd[0],
        'dosage_count': dd[1]
        })
        return jsonify(response), 200
```

10. **admin/remove_vaccination_centre/<id>**
    Finally you can delete any of the vaccination centre using the delete button