Report

Contact Manager with Flask and MongoDB

ON

Submitted in partial fulfillment of the
requirements of the degree of

Bachelor of Engineering (Information Technology)

By

Manav Punjabi - Roll No (44)

Under the guidance of

Dipti Karani

Department of Information Technology

VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY, Chembur, Mumbai 400074

(An Autonomous Institute, Affiliated to University of Mumbai)

Certificate

This is to certify that I have completed the project report on the topic **Contact Manager Web Application using Flask and MongoDB** satisfactorily in partial fulfilment of the requirements for the award of **Mini Project in WebX** Lab of Third Year (Semester-VI) in Information Technology under the guidance of **Ms. Dipti Karani** during the academic year 2023–2024, as prescribed by An Autonomous Institute Affiliated to University of Mumbai.

Supervisor/ Examiner

Table of Contents

1. Problem Statement

Create a contact management web application that allows users to securely add, view, search, update, and delete contacts. The system should also include a birthday reminder feature and enable users to share contacts using a QR code.

2. Introduction

The Contact Manager project is a web-based application developed using Flask (a Python web framework) and MongoDB (a NoSQL database). It provides functionality for managing personal or business contacts efficiently. Users can add, edit, delete, and search contacts. Unique features include a birthday reminder and QR code generation for sharing contact details.

3. Methodology :

1. Flask Setup for routing, template rendering, and request handling.
2. MongoDB used as backend NoSQL database using `pymongo`.
3. Authentication using hashed passwords and session-based login system.
4. Dashboard for users to manage their contacts.
5. QR Code generation using `qrcode` library with vCard formatting.
6. Birthday reminders based on current date.
7. Frontend with responsive HTML, CSS and Jinja2 templates.

4. Objective

The main goal of this project is to:

- Build a secure and user-friendly contact management system.
- Store contact data in MongoDB.
- Allow CRUD operations on contacts.
- Enable birthday tracking.
- Share contact via QR code using standard vCard format.

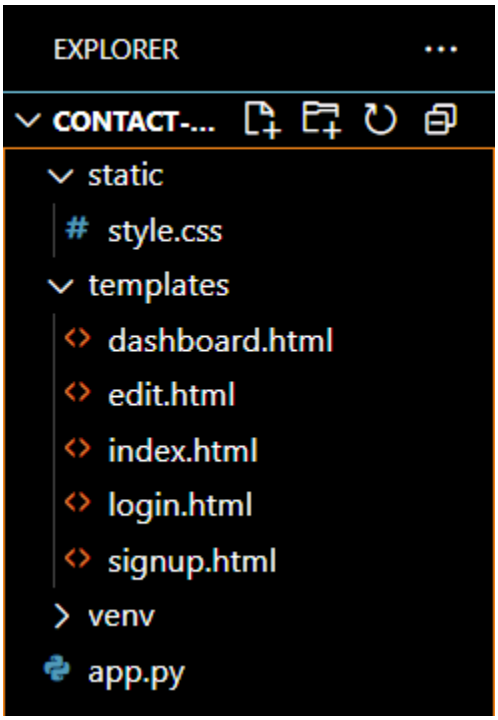
5. Tools and Technologies Used

Tools/Technology	Purpose
Python	Main Programming Language
Flask	Web framework to build the application
MongoDB	Database
HTML/CSS	For designing the user interface
VS Code	Code editor
Jinja2	Templating engine in Flask
qrcode	QR code generation

6. Features of the Project

Feature	Description
Signup/Login	Authenticated access for users
Add Contact	Add contact with name,phone,and DOB
Edit/Delete Contact	Modify or remove existing contacts
Birthday Reminder	Highlights today's birthdays.
QR Code Sharing	Generates vCard QR code for easy contact sharing
Responsive UI	Clean and simple design

7.Folder Structure of the Project



8. Working of the Project

8.1 UML Diagrams

To understand the system architecture and flow, the following UML diagrams are used:

Use Case Diagram

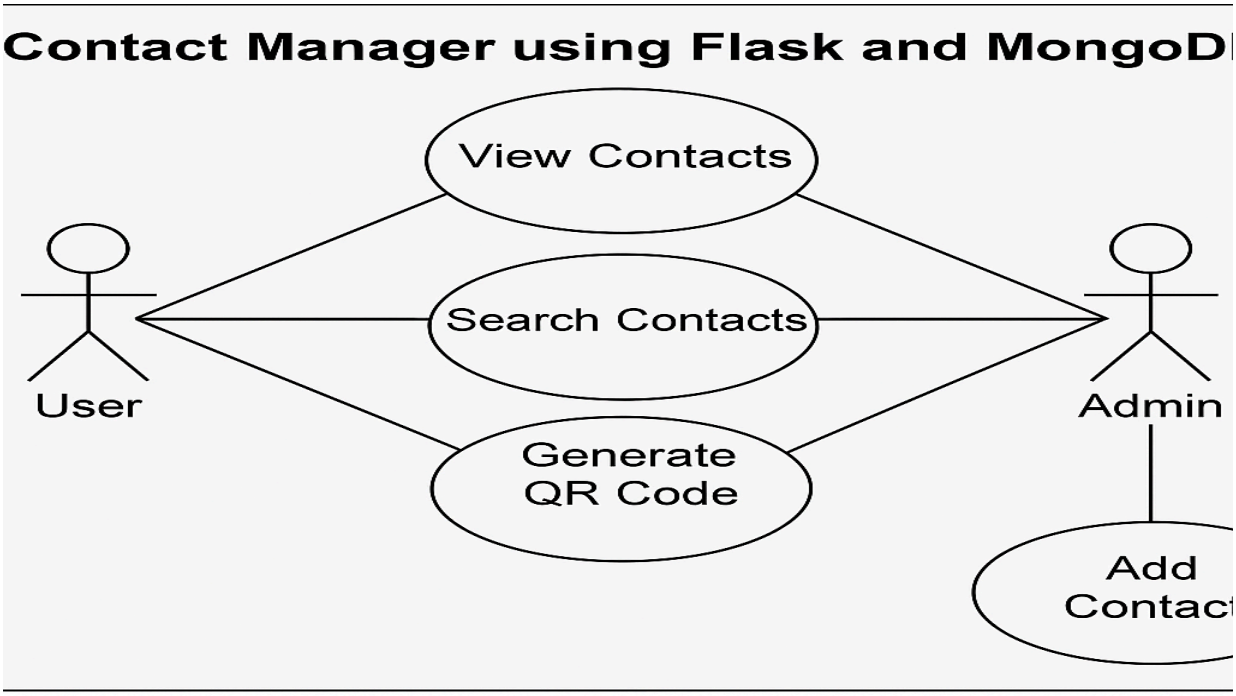


Figure 1 – Use Case Diagram of the Blog Web Application

7.2 Step-by-Step Working

1. User opens the application and logs in or signs up.
2. After login, user sees the dashboard with contact list and birthday reminders.
3. User can add new contact by entering name, phone, and date of birth.
4. Existing contacts can be edited or deleted.
5. Contacts can be searched using name/number.
6. A QR code button is shown beside each contact, generating a vCard QR code.
7. QR code can be scanned on any phone to import contact.
8. User can logout anytime.

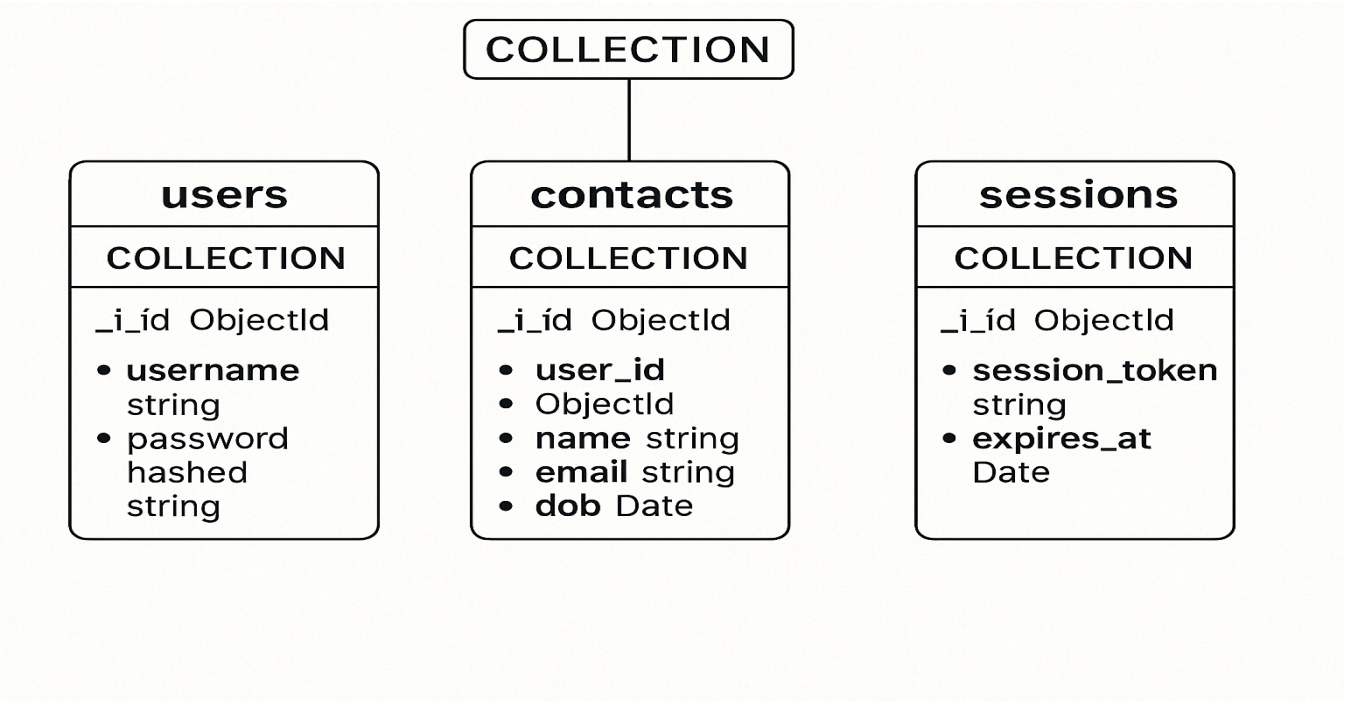
7.3 Database Representation Diagram

The below Entity-Relationship (ER) diagram represents the internal structure of the MongoDB database used in the **Contact Manager Web Application** project.
There are three collections:

- **users** – Stores user account information including username, email, and hashed password for authentication purposes.

- **contacts** – Stores contact details added by users such as name, phone number, email, and date of birth. Each contact is associated with a specific user via a foreign key reference (`user_id`).

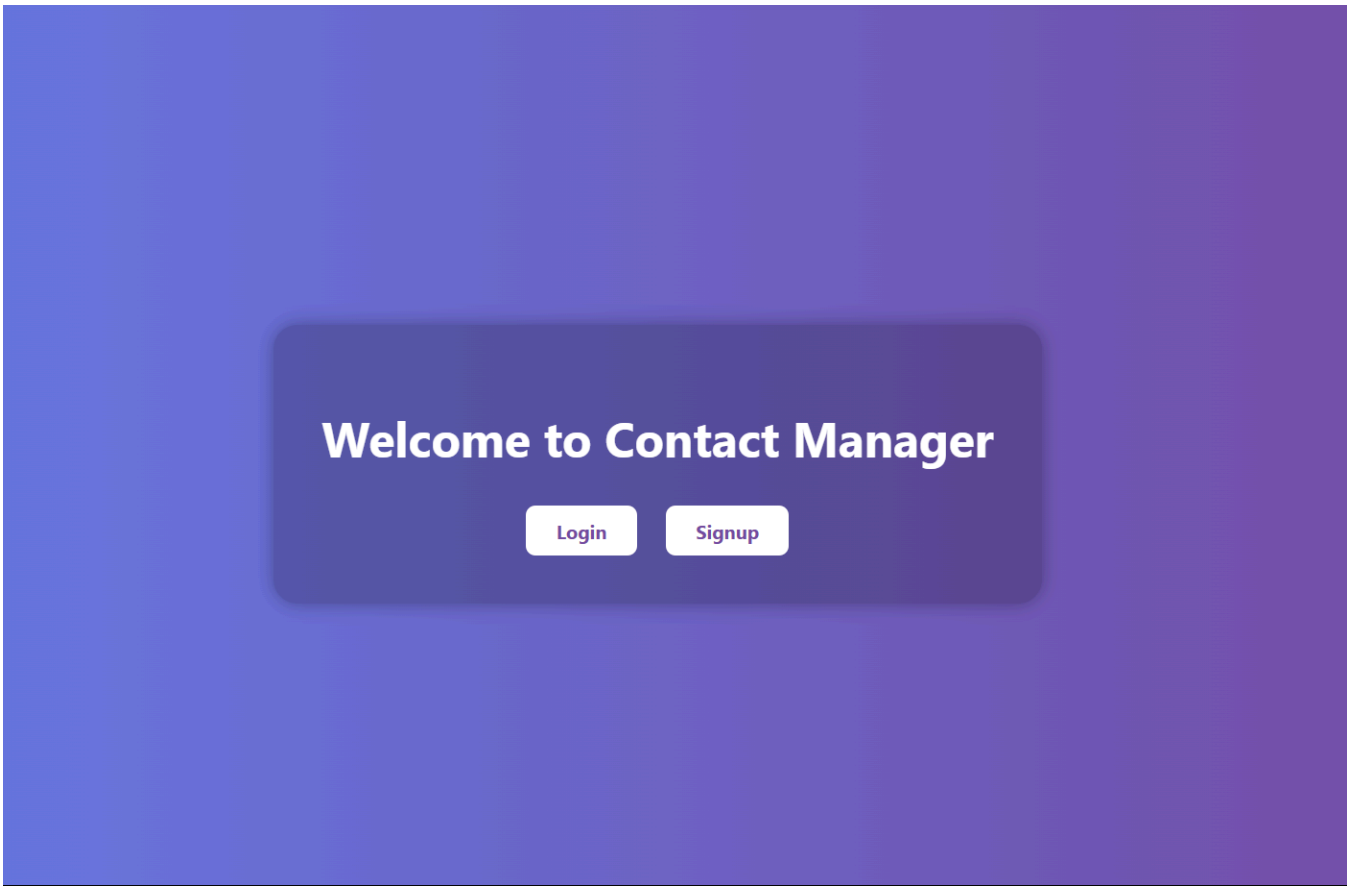
- **sessions** – Handles active user session tokens and session-related metadata to manage login sessions securely.



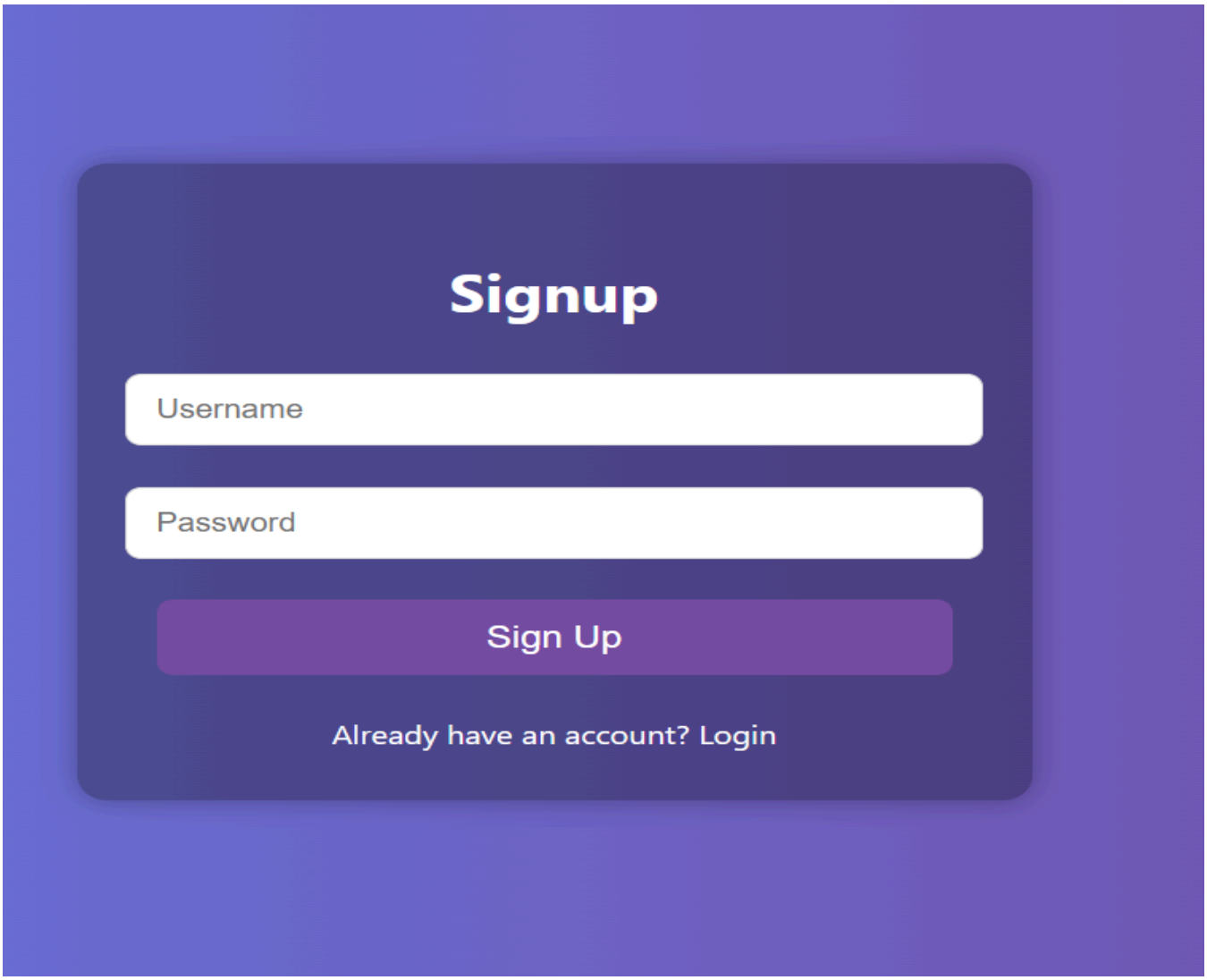
This diagram helps visualize how the data is organized and stored in MongoDB.

9.Screenshots:

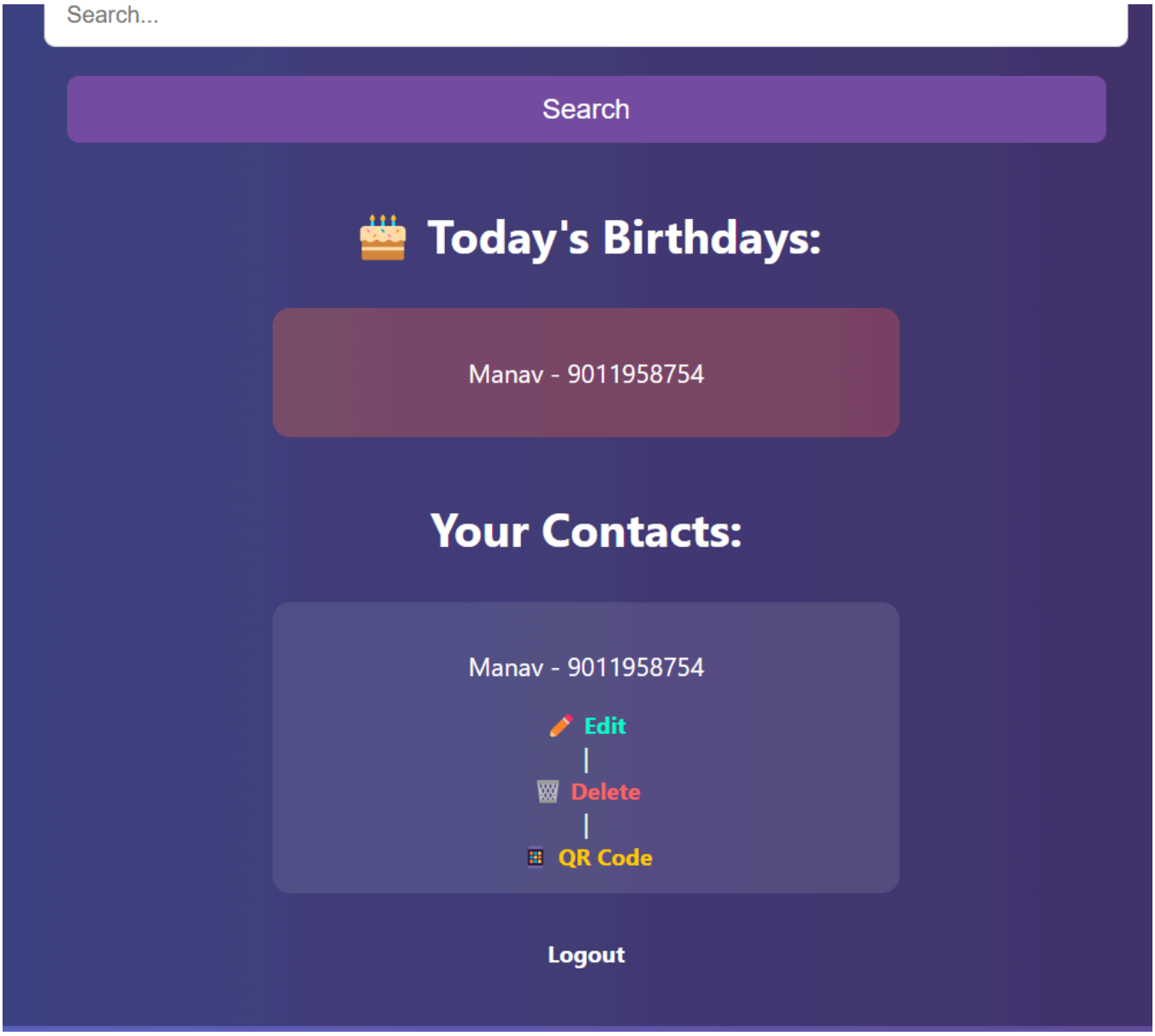
- **Login Page –Allows registered users to securely log into the contact manager using their credentials.**



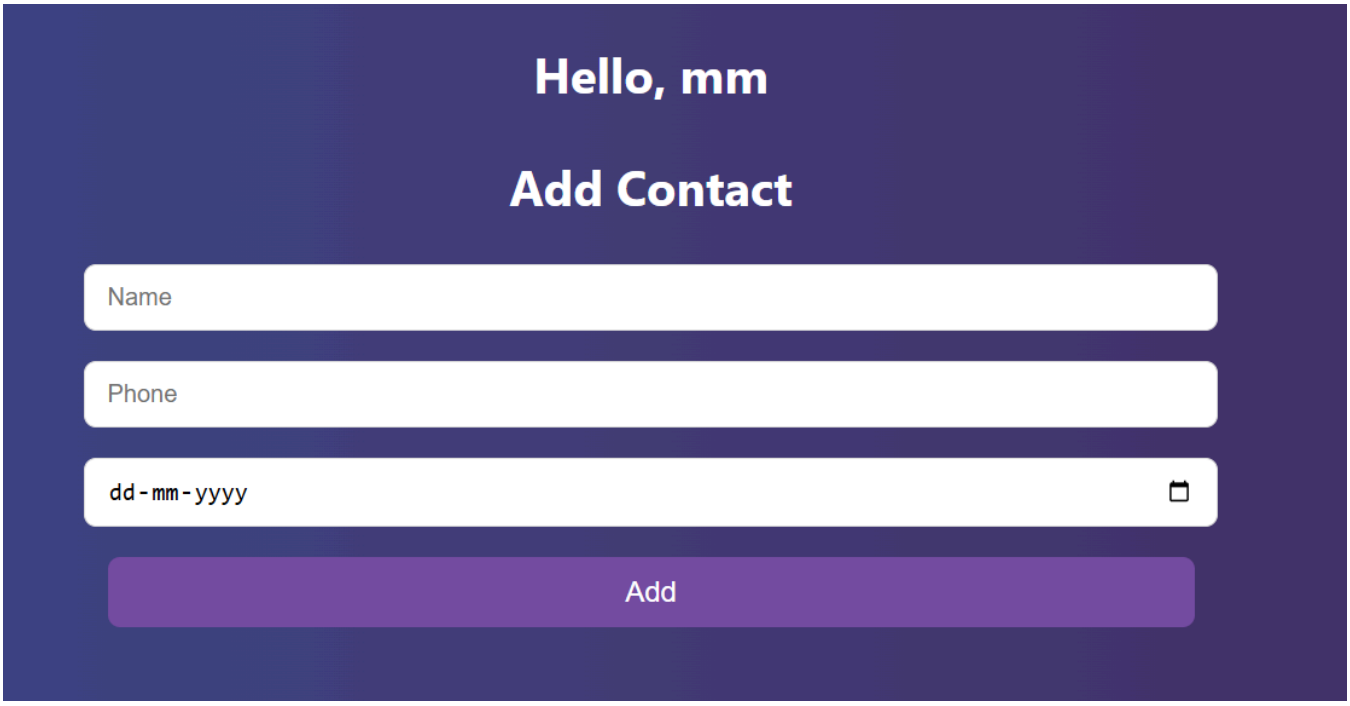
- **Signup Page –**
  Enables new users to create an account by providing username, email, and password.


Signup
Username
Password
Sign Up
Already have an account? Login

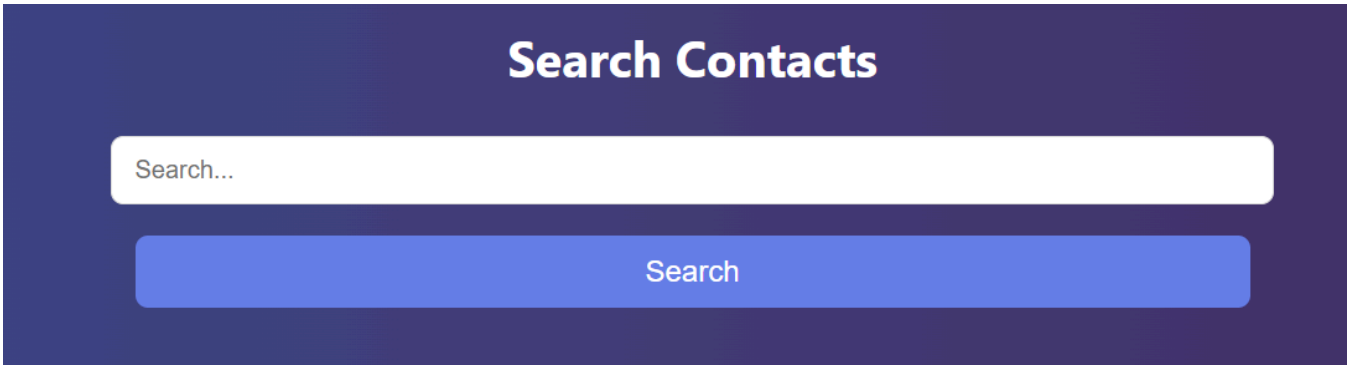
- **Dashboard Page –**
  Displays the list of saved contacts along with birthday reminders and action buttons.



- **Add Contact Page –**
  Lets users input new contact details such as name, phone number, and date of birth.



- **Search Page –**
  Provides a search bar to quickly find contacts by name or phone number.



- **QR Code Page –**
  Generates a QR code for each contact that can be scanned to share or save easily.

10. Sample Code Snippets

MongoDB Connection (credentials.py)

```
from pymongo import MongoClient
client = MongoClient("mongodb://localhost:27017/")
db = client['blog_app']
```

Flask Route for Creating Post (app.py)

```
@app.route('/create', methods=['GET', 'POST'])
def create_post():
    if request.method == 'POST':
        title = request.form['title']
        content = request.form['content']
        db.posts.insert_one({'title': title, 'content': content})
        return redirect(url_for('admin'))
    return render_template('create_post.html')
```

11. Challenges Faced

- Connecting Flask to MongoDB using Pymongo for the first time.
- Managing session-based login system securely.

- Displaying dynamic content on HTML pages using Jinja templates.
- Handling CRUD operations correctly with form validations.

12. Conclusion

This project helped build an end-to-end web application using Flask. I learned how to manage backend with MongoDB, create user sessions, and develop QR code-based contact sharing. It was a great experience applying full-stack development knowledge in a practical project.

13. Future Improvements

- Add contact group/tagging system.
- Upload contact profile pictures.
- Export contacts to Excel or CSV.
- Add 2FA for login.
- Use cloud MongoDB (Atlas) for remote hosting.

14. Reference:

- MongoDB Documentation
- W3Schools HTML/CSS Reference

15. Github Link