OBITUARY MEMORIAL WEBSITE - DOCUMENTATION

Version 1.0

Last Updated: 3/28/2025

1. Introduction

This documentation covers the setup, development, and usage of the Obituary Memorial Website, a Flask-based web application that allows users to create, view, and manage memorial pages for loved ones.

Key Features

1. User-Friendly Memorial Creation – Submit obituaries with names, dates, and biographies.
2. Responsive Design – Works on mobile, tablet, and desktop.
3. SEO Optimized – Dynamic sitemap for better search engine visibility.
4. Secure Database Integration – MySQL-backed storage for obituaries.
5. Modern UI – Clean, respectful design with easy navigation.

2. Project Structure

|  |  |
| --- | --- |
| File | Purpose |
| app.py | Main Flask application (routes, database logic). |
| config.py | Flask app configuration (database, security). |
| .env | Environment variables (DB credentials, secret key). |
| base.html | Base template (header, footer, navigation). |
| index.html | Homepage with hero section and features. |
| form.html | Form for submitting new obituaries. |
| obituary\_detail.html | Detailed view of a single memorial. |
| sitemap\_template.xml | Dynamic sitemap generator for SEO. |
| style.css | Stylesheet for responsive design. |

3. Setup & Installation

Prerequisites

* Python 3.8+
* MySQL Server
* Flask (pip install flask)
* Flask-MySQLdb (pip install flask-mysqldb)
* python-dotenv (pip install python-dotenv)

Step 1: Clone the Repository

*git clone [your-repo-url]*

*cd obituary-memorial-website*

Step 2: Configure Environment

1. Rename .env.example to .env and fill in:

env

*MYSQL\_HOST=localhost*

*MYSQL\_USER=your\_username*

*MYSQL\_PASSWORD=your\_password*

*MYSQL\_DB=obituary\_db*

*SECRET\_KEY=your\_secret\_key*

1. Update config.py if needed (e.g., debug mode).

Step 3: Database Setup

1.Create a MySQL database:

sql

*CREATE DATABASE obituary\_db;*

2.Run the Flask app to auto-generate tables (if using SQLAlchemy):

*python app.py*

Step 4: Run the Application

*python app.py*

* Access at: http://localhost:5000

4. Development Process

Key Routes (app.py)

|  |  |
| --- | --- |
| Route | Description |
| / | Homepage (index.html) |
| /create | Form for new obituaries (form.html) |
| /obituary/<id> | Detailed obituary view (obituary\_detail.html) |
| /sitemap.xml | Dynamic sitemap for SEO |

Database Schema

1. Obituaries table:

* id (INT, PK)
* name (VARCHAR)
* dob (DATE)
* dod (DATE)
* content (TEXT)
* author (VARCHAR)
* created\_at (TIMESTAMP)

Adding New Features

1.User Authentication

* Extend app.py with Flask-Login.

2.Image Uploads

* Use Flask-Uploads for memorial photos.

3.Comments/Tributes

* Add a tributes table linked to obituaries.

5. Usage Guide

1.For Admins

* Adding Obituaries

Visit /create and fill out the form.

* Editing/Deleting

(Future feature) Add admin dashboard.

2.For Visitors

* Viewing Memorials

Browse / and click on any obituary.

* Sharing

Copy the URL (e.g., /obituary/1) to share.

6. Deployment (Production)

Using Gunicorn & Nginx

1.Install Gunicorn:

*pip install gunicorn*

2.Run with:

*gunicorn -w 4 app:app*

3.Configure Nginx:

*server {*

*listen 80;*

*server\_name yourdomain.com;*

*location / {*

*proxy\_pass http://127.0.0.1:8000;*

*}*

*}*

Security Best Practices

* Use HTTPS (Let’s Encrypt).
* Restrict MySQL to localhost.
* Rotate SECRET\_KEY periodically.

7. Troubleshooting

|  |  |
| --- | --- |
| Issue | Solution |
| Database connection fails | Check .env credentials. |
| CSS not loading | Ensure static/style.css path is correct. |
| Form submission error | Verify required fields in form.html. |

8. Future Improvements

* User accounts (Flask-Login)
* Image uploads (Flask-Uploads)
* Search functionality (SQL LIKE queries)
* PDF memorial generation (ReportLab)

9. Conclusion

This Flask-based Obituary Memorial Website provides a respectful, easy-to-use platform for remembering loved ones.

Next Steps:

* Deploy to a cloud provider (AWS, DigitalOcean).
* Add user authentication.