Web Engineering Project

DayScript

Django Diary CRUD Application

A Bootstrap-styled, Django-based CRUD diary application demonstrating the MVT pattern, search/filter/sort, profile analytics, and admin management.

Team

Junaid Mohi Ud Din	Enrollment: 01-134222-071
Shayan Ikram Abbasi	Enrollment: 01-134222-140
Arif Hussain	Enrollment: 01-134222-029

December 2025

Contents

1 Project Overview

DayScript is a Django-based diary/journal web application. It implements full CRUD for diary entries, uses SQLite, follows Django's MVT architecture, and provides search, filter, sort, pagination, and profile statistics. The UI is styled with Bootstrap 5.

2 Objectives

- Demonstrate Django MVT with a text-only diary domain.
- Provide authentication (signup, login, logout).
- Enable CRUD with validation and forms.
- Offer search/filter/sort and a lightweight dashboard.
- Use SQLite as the default database.

3 Team

- Junaid Mohi Ud Din (01-134222-071)
- Shayan Ikram Abbasi (01-134222-140)
- Arif Hussain (01-134222-029)

4 Technology Stack

Python 3.12, Django 6.x, SQLite, Bootstrap 5, HTML/CSS.

5 System Architecture (MVT)

5.1 Overview

Django's Model–View–Template pattern separates concerns:

- **Model (Data Layer)**: Defines the data schema and relations; used for querying, validation, and persistence.
- **View (Logic Layer)**: Orchestrates requests, queries models, applies business logic, builds context, and returns rendered templates.
- **Template (Presentation Layer)**: Renders HTML using context from views; contains display-only logic.

URL routing maps HTTP requests to views. Middleware handles cross-cutting concerns (auth, sessions, CSRF).

5.2 Models

- **Tag**: name (unique), created_at.
- **DiaryEntry**: user (FK), title, content, mood (choices), tags (M2M), created_at, updated_at.

5.3 Views

Class-based views for list, detail, create, update, delete; function-based views for register, about, and profile. HomeView handles search, filter, sort, pagination, and stats.

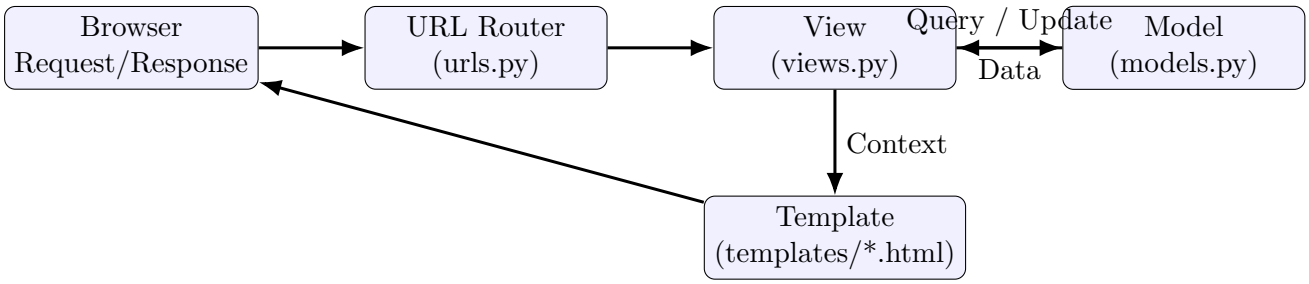
5.4 Templates

`base.html`, `home.html`, `entry_detail.html`, `entry_form.html`, `entry_delete.html`, `login.html`, `register.html`, `profile.html`, `about.html`.

5.5 URLs

Project `urls.py` includes `diary/urls.py`, mapping to home, auth, CRUD, profile, about.

5.6 MVT Flow Diagram



6 Features Implemented

Authentication	Signup, login, logout (Django auth)
CRUD Entries	Create, read, update, delete diary entries
Moods/Tags	Mood choices with emojis; tags with add-new in form
Search/Filter/Sort	Text search; filters: mood, date range, tag; sort by title/date/mood; pagination
Dashboard Stats	Total entries, tags count, entries last 30 days, recent count
Profile	Mood distribution, tags list, recent entries
Admin	Manage entries and tags via Django admin
UI/UX	Bootstrap 5 responsive layout, alerts, cards, buttons
Validation	Disallow future dates on entry creation/update

7 CRUD Operations (Diary Domain)

- **Create**: Entry form with mood, tags (existing + new), validation (no future date).
- **Read/List**: Home page shows entries (paginated) with search/filter/sort.
- **Detail**: Entry detail page with mood badge, tags, timestamps.
- **Update**: Pre-filled form to edit title/content/mood/tags/date.
- **Delete**: Confirmation page to remove entry (restricted to owner).

8 Setup and Execution

1. (Optional) `python -m venv .venv` and activate.
2. `pip install -r requirements.txt`
3. `python manage.py migrate`
4. `python manage.py createsuperuser`
5. `python manage.py runserver`
6. Open `http://127.0.0.1:8000/`

9 Testing Plan

- Auth: register, login, logout.
- CRUD: create/edit/delete entries; confirm detail view.
- Search/filter/sort/pagination on home.
- Profile stats and tags.
- Admin: view/edit entries and tags.

10 Screenshots (to include separately)

Home, create entry, detail, edit, delete confirmation, profile, about, (optional) admin.

11 Limitations and Future Work

- No rich text or file uploads (text-only by design).
- Delete confirmation uses a page (modal optional).
- Could add more stats (e.g., entries per tag, per month).

12 Conclusion

DayScript satisfies the core CRUD, MVT, and UI requirements for the course, with Bootstrap-based responsive design, search/filter/sort, and profile analytics on top of Django's auth and admin.