Список погибших ВОВ

выполнена студентами

Использованные технологии:

- Фреймворк: Django 5.1.7 на Python
- База данных: PostgreSQL
- Фронтенд: «голый» HTML и CSS без фреймворков
- Управление зависимостями: `requirements.txt`
- Переменные окружения: библиотека `python-dotenv`
- Дизайн: макет в Figma

По заданию преподавателя требовалось реализовать сайт с:

- Главной страницей для вывода списка записей из базы данных
- Отдельной страницей добавления новых записей
- Встроенной админ-панелью Django для управления данными
- Адаптивным дизайном, корректно отображающимся на мобильных устройствах

Тематика проекта — память о погибших в Великой Отечественной войне

Архитектура проекта

Проект построен по стандартной структуре Django с двумя приложениями:

- `memory` — основное приложение:
  - Модель `Fallen` для хранения данных о погибших
  - Представления для главной страницы и формы добавления
  - Шаблоны в папке `templates/memory/`
  - Стили в `static/memory/style.css`
  - Админ-панель для управления записями
- `users` — приложение аутентификации:
  - Кастомная модель пользователя `CustomUser`, расширяющая `AbstractUser`
  - Страницы входа, регистрации и профиля
  - Отдельные шаблоны и маршруты

Главная конфигурация проекта (`settings.py`, `urls.py`) расположена в папке `myproject/myproject/`.

Работа с базой данных

- СУБД: PostgreSQL, подключённая через переменные окружения (`.env` файл)
- Модель данных: таблица `memory_fallen` с полями:
  - ФИО погибшего
  - Год рождения (опционально)
  - Год гибели
  - Описание (место захоронения, дополнительная информация)
- Заполнение базы данных:
  - Изначально данные загружались из CSV-файла с помощью отдельного Python-скрипта
  - Скрипт парсил файл и создавал записи через Django ORM
  - Дополнительно записи можно добавлять вручную через админ-панель или форму на сайте
- Миграции: стандартный механизм миграций Django (`makemigrations` / `migrate`)

Фронтенд и дизайн

- Дизайн: макет создан в Figma до начала разработки — использовался как основа для вёрстки
- Вёрстка:
  - Минималистичный HTML без использования библиотек (Bootstrap и др.)
  - Стили написаны с нуля в одном CSS-файле
  - Базовая адаптивность реализована через media-запросы
  - На мобильных устройствах (до 320px) меню переходит в вертикальный режим, карточки занимают всю ширину экрана
- Шаблонизация:
  - Базовый шаблон `base.html` с единой структурой навигации
  - Дочерние шаблоны расширяют базовый через наследование
  - Единая цветовая схема и типографика по всем страницам

Функционал сайта

Для обычных пользователей:

- Главная страница — просмотр всех записей в виде карточек
- Поиск и фильтрация реализованы на уровне базы данных (будет добавлено в будущем)

Для администраторов:

- Встроенная админ-панель Django (`/admin/`) с полным CRUD-функционалом
- Страница добавления записи (`/add/`) с формой и валидацией полей
- Возможность редактировать и удалять записи

Особенности реализации

- Безопасность:
  - Все формы защищены CSRF-токенами
  - Пароли хранятся в хэшированном виде через механизм Django
  - Доступ к админ-панели ограничен авторизацией
- Развертывание:
  - Проект запускается локально через `python manage.py runserver`
  - Для продакшена предусмотрен WSGI-сервер (настройки в `wsgi.py`)
  - Переменные окружения вынесены в `.env` для гибкости конфигурации
- Тестирование:
  - Базовые unit-тесты для моделей и представлений (файл `tests.py` в каждом приложении)

Итоги проекта

✅ Реализован полноценный веб-сайт на Django с двумя приложениями

✅ Настроено подключение к PostgreSQL через переменные окружения

✅ Создана кастомная модель пользователя для гибкости аутентификации

✅ Реализован базовый адаптивный дизайн без использования фреймворков

✅ База данных заполнена через автоматизированный скрипт (CSV → Django ORM)

✅ Дизайн разработан заранее в Figma и перенесён в код

✅ Проект готов к локальному запуску и дальнейшему развитию