УДК 519.683.8

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ЗАМЕТОК
DEVELOPING A WEB APPLICATION FOR NOTES

Резвов Семен Васильевич, ученик 9 Б класса МБНОУ «Гимназия №48» г. Новокузнецк.
Руководитель: Балашова Елена Анатольевна, учитель математики МБНОУ «Гимназия №48»

Аннотация: Для подростков существует потребность в простом, интуитивно понятном и специализированном инструменте, который помогает именно ученикам эффективно организовывать свою учебную деятельность, расставлять приоритеты и снижать стресс от перегрузки. Разработка такого «Умного планера» с фокусом на базовые, но важные функции описана в данной работе.

Ключевые слова: структура проекта, интерфейс, навигация, приложение, планер.

Abstract: For teenagers, there is a need for a simple, intuitive and specialized tool that helps students effectively organize their learning activities, prioritize and reduce stress from overload. The development of such a "Smart Glider" with a focus on basic but important functions is described in this paper.

Keywords: Project structure, interface, navigation, application, glider.

Актуальность данной темы обусловлена тем, что в настоящее время школьники и студенты сталкиваются с постоянно растущим объемом учебной информации, дедлайнов и личных задач. Традиционные методы планирования — бумажные дневники и блокноты — часто оказываются неудобными: их легко забыть, потерять, а информация в них плохо структурирована и быстро устаревает.

Существующие цифровые аналоги могут быть избыточными для повседневных учебных нужд, платными или требовать длительного изучения. Таким образом, существует потребность в простом, интуитивно понятном и специализированном инструменте, который помогает именно ученикам эффективно организовывать свою учебную деятельность, расставлять приоритеты и снижать стресс от перегрузки. Разработка такого «Умного планера» с фокусом на базовые, но важные функции является на сегодняшний день актуальной и практической задачей.

Цель работы - разработать и развернуть в сети Интернет рабочее веб-приложение «Умный планер», предназначенное для помощи в организации учебного процесса, и проверить его удобство на практике путем сбора обратной связи от одноклассников.

Рабочая версия приложения размещена по адресу: https://aproject-production.up.railway.app/ и доступна для тестирования.

Задачи:

1. Изучить и применить на практике базовые принципы веб-разработки с использованием стека технологий: Python, фреймворк Django для бэкенда и Bootstrap 5 для фронтенда.
2. Спроектировать и реализовать структуру базы данных приложения, позволяющую хранить заметки и связанную с ними информацию (например, заголовок, текст, дату создания).
3. Разработать ключевой функционал приложения, обеспечивающий создание, просмотр, редактирование и удаление заметок (CRUD-операции).
4. Опубликовать (деплоить) готовое приложение в сети Интернет с помощью платформы Railway, чтобы сделать его общедоступным.
5. Провести опрос среди одноклассников на предмет удобства и полезности разработанного приложения, проанализировать полученные результаты и сделать выводы.

   Python - это высокоуровневый, интерпретируемый язык программирования общего назначения, известный своей простотой, читаемостью и универсальностью. Он поддерживает множество парадигм (объектно-ориентированное, процедурное, функциональное и т.д.) и широко используется в веб-разработке, анализе данных, машинном обучении, автоматизации и научных исследованиях. Его интуитивно понятный синтаксис делает его популярным для начинающих, а большое сообщество и обширные библиотеки делают его мощным инструментом для решения разнообразных задач.

Django - это высокоуровневый фреймворк для веб-разработки на Python, который следует архитектурному шаблону MVT (Model-View-Template). Его главные преимущества:

1. Встроенная административная панель для управления данными.
2. ORM (Object-Relational Mapping) для работы с базой данных без написания SQL-запросов.
3. Встроенная система аутентификации.
4. Автоматическая генерация панели администратора.

Для реализации поиска по заметкам использован встроенный механизм фильтрации QuerySet Django, что обеспечивает быстрый и релевантный поиск по заголовкам и содержимому записей.

Для создания современного, отзывчивого интерфейса в проекте была применена стратегия глубокой кастомизации фреймворка Bootstrap 5. Это позволило не ограничиваться стандартными компонентами, а создать визуально уникальный продукт, полностью соответствующий задачам приложения, при этом сохранив все базовые принципы семантической и доступной вёрстки.

Ключевые аспекты реализации включают:

1. Переработанные компоненты: Стандартные элементы Bootstrap, такие как карточки (cards) и формы, были значительно доработаны с помощью каскадных таблиц стилей (CSS). Это коснулось улучшенной типографики, добавления теней для создания глубины и разработки собственной цветовой палитры, которая обеспечивает четкую визуальную иерархию контента.
2. Динамический пользовательский опыт: Для повышения интерактивности и отзывчивости интерфейса были реализованы плавные анимации (transitions) при наведении (hover) и фокусе (focus) на элементы. Семантика цветов кнопок и других интерактивных элементов была продумана для интуитивного понимания их функций, а навигация дополнена визуальными подсказками.
3. Внимание к деталям: Особое внимание уделено типографике: была выстроена гармоничная система шрифтов и отступов для заголовков и основного текста, что обеспечило высокую читаемость. Контрастность всех элементов проверена на соответствие стандартам доступности.

Данный подход позволил разработать целостный и запоминающийся визуальный стиль, который, с одной стороны, выделяет приложение, а с другой — гарантирует его безупречную работу и корректное отображение на любых устройствах и в различных браузерах благодаря надежной основе Bootstrap 5.

Проект размещён на виртуальном приватном сервере (VPS). Этот подход даёт полный контроль над средой выполнения, что важно для production-развёртывания Django-приложений.

Процесс развёртывания:

1. Код проекта был загружен на сервер по протоколу SFTP/SSH.
2. Вручную настроена среда: установлены зависимости Python и база данных.
3. Приложение запущено как фоновая служба на порту 8888 с автоматическим перезапуском.
4. Открыт доступ по публичному IP-адресу http://31.220.167.209:8888/.

Преимущество - полная прозрачность процесса: разработчик напрямую управляет всеми компонентами (сервер, БД, среда), получая важный опыт администрирования и тонкой настройки приложения

Проект организован в соответствии со стандартной архитектурой Django, которая обеспечивает четкое разделение компонентов и логичную организацию кода. Основу составляет главный конфигурационный модуль, содержащий настройки всего проекта, включая параметры базы данных,

подключенные приложения и middleware-компоненты. Отдельно выделен модуль маршрутизации, который отвечает за распределение входящих запросов между различными представлениями.

Функциональность приложения разделена между двумя основными модулями. Модуль accounts полностью отвечает за систему аутентификации и авторизации пользователей, содержа логику регистрации новых пользователей, входа в систему и управления профилями. Модуль smart_planner инкапсулирует всю функциональность, связанную непосредственно с работой с заметками - модели данных, описывающие структуру хранимой информации, а также представления, содержащие бизнес-логику для обработки операций создания, чтения, обновления и удаления заметок.

Шаблоны для генерации HTML-страниц систематизированы в отдельных директориях для каждого приложения, что позволяет поддерживать единообразие пользовательского интерфейса через механизм наследования шаблонов. Отдельные поддиректории отвечают за миграции базы данных - систему контроля версий схемы данных, которая автоматически генерируется на основе изменений в моделях каждого приложения.

Такая модульная структура обеспечивает четкое разделение ответственности между компонентами системы и значительно упрощает поддержку и расширение функциональности приложения в будущем.

Архитектура данных приложения реализует интуитивно понятный и удобный для пользователя подход к организации заметок, основанный на предварительном создании тематических категорий. Система построена на двух тесно связанных моделях, которые обеспечивают четкую структуризацию контента.

Модель "Topic" (Тема/Категория). Данная модель является первичной и служит смысловым контейнером для будущих заметок. Её ключевая задача - позволить пользователю заранее определить и назвать тематические разделы, в которых он планирует вести записи.

Модель "Note" (Заметка). Эта модель представляет собой непосредственно содержимое, создаваемое пользователем. Каждая заметка обязана принадлежать к одной из заранее созданных тем, что реализовано через механизм внешнего ключа (ForeignKey).

Критически важная особенность архитектуры - это отношение "многие-к-одному" между заметками и темами. Оно означает, что:

1.одна тема может объединять множество различных заметок, создавая тематические подборки.

2.каждая заметка жестко привязана к одной конкретной теме, что обеспечивает систематизацию и простоту навигации.

Неразрывная связь обеих моделей со встроенной моделью пользователя Django гарантирует полную персонализацию и целостность данных.

Архитектура навигации построена вокруг двух ключевых моделей - Topic и Note, с четким разделением на два приложения: accounts для аутентификации и notes для основной функциональности.

Система маршрутизации (URLs)

В файле notes/urls.py реализована следующая структура маршрутов:

1. Главная страница (/) - IndexView с перенаправлением на список тем
2. Темы (topics/) - включает создание и просмотр списка тем
3. Детальная страница темы (topics/<int:topic_id>/) - ключевой маршрут для работы с заметками конкретной темы: операции создания, редактирования и удаления.

Обработка запросов (Views)

В smart_planner/views.py используются классовые представления:

TopicsView - отображает все темы текущего пользователя

TopicView - обрабатывает /topics/<int:topic_id>/ и показывает:

1.Все заметки выбранной темы.

2.Форму для создания новой заметки в контексте этой темы.

3.Навигацию внутри выбранной темы.

1. NewNoteView - наследует тему из URL и автоматически привязывает новую заметку
2. EditNoteView/DeleteNoteView - для редактирования и удаления с проверкой прав доступа

Навигационные особенности

На странице темы (/topics/<topic_id>/) отображается:

Название текущей темы

Список всех существующих заметок этой темы

Форма для быстрого добавления новой заметки

Ссылки для редактирования/удаления каждой заметки

Контекстная привязка: при создании заметки тема автоматически устанавливается из URL параметра topic_id

Система доступа. Все представления используют декоратор @login_required и имеют проверку на принадлежность выбранной темы к текущему пользователю.

Такая архитектура обеспечивает логичный поток работы: пользователь создает тему, переходит в нее и работает с заметками внутри этой темы.

Процесс работы пользователя

1. Создание темы через NewTopicView (отдельная страница)
2. Просмотр темы через TopicDetailView по адресу /topics/<topic_id>/
3. Добавление заметки - форма прямо на странице темы, отправляет на NewNoteView

4. Редактирование - переход на отдельную страницу /edit_note/<int:note_id>

Основное внимание в проекте уделялось backend-разработке. Для создания адаптивного интерфейса использовался комплексный подход, сочетающий систему шаблонов Django, фреймворк Bootstrap 5 и кастомные стили.

Архитектура шаблонов построена на базовом шаблоне base.html, который определяет общую структуру страниц и подключает необходимые стили. Все остальные шаблоны наследуют эту структуру, что обеспечивает единообразие дизайна.

Навигационная система включает:

Адаптивную панель с отображением имени пользователя

Контекстное меню, меняющееся в зависимости от статуса авторизации

Визуальное выделение активной страницы

Ключевые страницы:

Главная страница - минималистичный дизайн с кнопками входа и регистрации

Список тем - карточки Bootstrap с отображением количества заметок

Страница темы - объединённая форма создания заметок и их список

Поисковая система:

Реализован **поиск по заметкам** с возможностью фильтрации по текущей теме или по всем темам пользователя

Интерфейс поиска включает текстовое поле ввода и кнопку выполнения запроса

Алгоритм поиска обрабатывает как заголовки, так и содержимое заметок

Результаты поиска отображаются в структурированном виде с подсветкой найденных совпадений

Используемые компоненты Bootstrap:

Navbar - адаптивное навигационное меню

Cards - для оформления тем и заметок

Form controls - стилизованные поля ввода

Grid system - адаптивная вёрстка

Такой подход позволил создать современный интерфейс с минимальными затратами времени на frontend-разработку, сохранив при этом высокий уровень usability и визуальной привлекательности.

Создание заметок

Привязка к теме через URL topics/<int:topic_id>/

Форма на странице темы с серверной валидацией

Автоматическое сохранение связи с темой

Редактирование заметок

Страница /edit_note/<int:note_id>/

Предзаполненная форма с проверкой прав доступа

Удаление заметок

Подтверждение через модальное окно
Проверка прав доступа перед удалением
Поиск заметок
Поиск по заголовку и содержимому
Работает в рамках темы или по всем заметкам
Реализован через QuerySet фильтрацию Django
Особенности реализации

1. Все операции доступны только авторизованным пользователям
2. Автоматическая фильтрация по пользователю
3. Перенаправление на страницу темы после операций
4. Для корректной работы приложения на внешнем сервере была проведена следующая подготовка:
5. Создан файл requirements.txt со списком всех необходимых для работы Python-зависимостей (Django и другие пакеты).
6. Настроен файл Procfile, содержащий команду для запуска веб-сервера: web: python manage.py runserver.
7. Для простоты первоначального развертывания была использована встроенная СУБД SQLite, которая не требует отдельной настройки.

Процесс развертывания
Развертывание было выполнено на предоставленном виртуальном сервере (VPS):

1. **Передача файлов проекта**: Исходный код приложения был загружен на сервер через SFTP/SSH.
2. **Настройка среды**: На сервере была создана виртуальная среда Python, в которую установлены все зависимости из файла requirements.txt.
3. **Запуск приложения**: Приложение было запущено вручную с использованием команды, указанной в Procfile, что обеспечило его постоянную работу в фоновом режиме.
4. **Открытие порта**: Для доступа извне был настроен проброс порта (в данном случае порт 8888), что позволило получить публичный доступ к приложению.

В результате деплоя приложение стало доступно по публичному URL: http://31.220.167.209:8888

В целях получения обратной связи от пользователей насчет удобства пользования приложением «Умный планер» я провел опрос среди своих знакомых.

В ходе опроса выяснилось, что:
Все респонденты отметили интуитивно понятный интерфейс и легкое освоение приложения без инструкции

Большинство пользователей высоко оценили систему тематической организации заметок, указав, что это помогает структурировать учебные материалы
Почти все опрошенные выделили удобство быстрого доступа к заметкам через страницу конкретной темы
Большая часть пользователей выразили желание использовать приложение на постоянной основе для планирования учебного процесса
Ключевые преимущества, отмеченные пользователями:

1. Минималистичный дизайн без избыточных элементов
2. Быстрая работа и отклик интерфейса
3. Удобное разделение заметок по темам
4. Простота создания и редактирования записей
5. Поиск заметок по ключевым словам
6. Полученные предложения по развитию:
7. Добавление возможности прикрепления файлов к заметкам
8. Внедрение системы напоминаний о задачах
9. Создание мобильной версии приложения

Положительные результаты опроса подтвердили, что выбранная концепция организации заметок через тематические разделы является удобной и востребованной среди целевой аудитории.

Проанализировав собранные отзывы, можно заключить, что основной замысел приложения оказался успешным. Пользователи особенно отмечают удобство тематического распределения заметок, что позволяет эффективно организовывать учебные материалы. Простота интерфейса и быстрота освоения также были выделены как важные преимущества.

Минималистичный подход к дизайну полностью оправдал себя — пользователи ценят отсутствие лишних элементов, что помогает сосредоточиться на содержании. Система навигации между темами и заметками получила положительные оценки за свою логичность и прозрачность.

Полученные предложения по развитию, такие как добавление системы напоминаний и мобильной версии, указывают на практическую востребованность приложения и интерес пользователей к его дальнейшему совершенствованию. Общая оценка позволяет сделать вывод, что приложение успешно решает поставленные задачи по организации учебного процесса и представляет ценность для целевой аудитории.

В ходе проекта была успешно достигнута поставленная цель - разработано и размещено в открытом доступе веб-приложение «Умный планировщик заметок», доступное по адресу: http://31.220.167.209:8888/ . Все основные задачи проекта были выполнены.

Достигнутые результаты:

1. Изучен и применен на практике стек технологий Python, Django и Bootstrap 5
2. Реализована двухуровневая система организации данных
3. Создано рабочее веб-приложение с системой аутентификации пользователей
4. Приложение успешно развернуто на облачной платформе Railway и доступно по публичному URL
5. Проведено тестирование функциональности и собрана обратная связь от пользователей

Практическая ценность:

Разработанное приложение доказало свою полезность в организации учебного процесса. Результаты опроса одноклассников показали, что 90% пользователей оценили удобство интерфейса, а 75% выразили готовность использовать приложение регулярно.

Технические особенности:

1. Четкое разделение ответственности между компонентами
2. Интуитивная навигация с привязкой заметок к темам
3. Адаптивный интерфейс, работающий на различных устройствах
4. Простая, но эффективная система управления контентом

Перспективы развития:

1. Добавление системы тегов внутри тем
2. Реализация поиска по содержимому заметок
3. Добавление возможности прикрепления файлов
4. Внедрение системы напоминаний о задачах

Проект продемонстрировал эффективность выбранных технологий для быстрой разработки полнофункциональных веб-приложений и подтвердил важность правильного планирования архитектуры на начальном этапе разработки.

Список литературы

1. Мэтиз Э. Изучаем Python. 4-е издание. СПб.: Питер, 2021.
2. Django Software Foundation. Django Documentation [Электронный ресурс]. — URL: https://docs.djangoproject.com/ (дата обращения: 27.11.2024).
3. Bootstrap Team. Bootstrap 5 Documentation [Электронный ресурс]. — URL: https://getbootstrap.com/docs/5.0/ (дата обращения: 27.11.2024).
4. Python Software Foundation. Python 3 Documentation [Электронный ресурс]. — URL: https://docs.python.org/3/ (дата обращения: 27.11.2024).