АННОТАЦИЯ

Пояснительная записка к курсовому проекту содержит постановку и программу решения задачи «Проектирование базы данных и разработка приложения для учета движения компьютеров в учебном заведении».

Программа news.html написана на языке HTML в среде программирования Visual Studio Cod с использованием сервера баз данных MySQL, предназначена для работы в операционной системе MS Windows 7 и выше, отлажена на данных контрольного примера.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современные новостные платформы сталкиваются с проблемой перенасыщения информации, что затрудняет пользователям поиск актуального и интересного контента. Для решения этой задачи была разработана система новостного канала с персонализированными рекомендациями и администрированием контента. Данный проект направлен на создание удобного и эффективного инструмента для публикации, распространения и потребления новостей с учетом индивидуальных предпочтений пользователей.

Цель работы — разработка веб-приложения для новостного канала, включающего в себя возможность регистрации и аутентификации пользователей, публикацию, редактирование и удаление новостей через админ-панель, систему лайков для формирования персонализированных рекомендаций и автоматическую рассылку уведомлений о новых постах зарегистрированным пользователям.

Задачи работы:

Изучить принципы проектирования реляционных баз данных, включая нормализацию и связи между сущностями.

Спроектировать структуру БД для хранения данных о пользователях, новостях, лайках и подписках.

Разработать алгоритм рекомендаций на основе анализа лайков (коллаборативная фильтрация)

Настроить систему email-уведомлений о новых публикациях.

Протестировать работоспособность системы на контрольных примерах.

Используемые теоретические и нормативные источники

При выполнении проекта использовались:

1. Богданова А.Л. *Базы данных. Теория и практика применения (2-е издание)* [Электронный ресурс]: учебное пособие / А.Л. Богданова, Г.П. Дмитриев, А.В. Медников. — Химки: Российская международная академия туризма, 2016. — 128 с. — Режим доступа: http://www.iprbookshop.ru/47625. — ЭБС «IPRbooks».
2. Голицына О.Л. *Основы проектирования баз данных*: учеб. пособие / О.Л. Голицына, Т.Л. Партыка, И.И. Попов. — 2-е изд., перераб. и доп. — Москва: ФОРУМ: ИНФРА-М, 2019. — 416 с. — (Среднее профессиональное образование). — ISBN 978-5-91134-655-3. — Текст: электронный. — URL: https://znanium.com/catalog/product/1018906 (дата обращения: 22.03.2023).
3. Гордеев С.И. *Организация баз данных в 2 ч. Часть 1*: учебник для среднего профессионального образования / С.И. Гордеев, В.Н. Волошина. — 2-е изд., испр. и доп. — Москва: Издательство Юрайт, 2023. — 310 с. — (Профессиональное образование). — ISBN 978-5-534-11626-7. — Текст: электронный. — URL: https://urait.ru/bcode/518510 (дата обращения: 14.03.2023).
4. Гордеев С.И. *Организация баз данных в 2 ч. Часть 2*: учебник для вузов / С.И. Гордеев, В.Н. Волошина. — 2-е изд., испр. и доп. — Москва: Издательство Юрайт, 2023. — 513 с. — (Высшее образование). — ISBN 978-5-534-04470-6. — Текст: электронный. — URL: https://urait.ru/bcode/515097 (дата обращения: 14.03.2023).

5. Официальная документация MySQL [Электронный ресурс]. — URL: https://dev.mysql.com/doc/ (дата обращения: 14.03.2023).
6. Официальная документация Flask [Электронный ресурс]. — URL: https://flask.palletsprojects.com/ (дата обращения: 14.03.2023).
7. Официальная документация Bootstrap [Электронный ресурс]. — URL: https://getbootstrap.com/docs/ (дата обращения: 14.03.2023).
8. Документация по JavaScript (MDN Web Docs) [Электронный ресурс]. — URL: https://developer.mozilla.org/ru/docs/Web/JavaScript (дата обращения: 14.03.2023).
9. Документация по HTML5 и CSS3 (W3Schools) [Электронный ресурс]. — URL: https://www.w3schools.com/ (дата обращения: 14.03.2023).
10. Документация по SQLAlchemy (ORM для Flask и Python) [Электронный ресурс]. — URL: https://docs.sqlalchemy.org/ (дата обращения: 14.03.2023).
11. Документация по Jinja2 (шаблонизатор для Flask) [Электронный ресурс]. — URL: https://jinja.palletsprojects.com/ (дата обращения: 14.03.2023).
12. Документация по WTForms (валидация форм в Flask) [Электронный ресурс]. — URL: https://wtforms.readthedocs.io/ (дата обращения: 14.03.2023).
13. Документация по Flask-SQLAlchemy [Электронный ресурс]. — URL: https://flask-sqlalchemy.palletsprojects.com/ (дата обращения: 14.03.2023).
14. Документация по Flask-Login (аутентификация пользователей) [Электронный ресурс]. — URL: https://flask-login.readthedocs.io/ (дата обращения: 14.03.2023).

15. Документация по SMTP (отправка email-уведомлений) [Электронный ресурс]. — URL: https://docs.python.org/3/library/smtplib.html (дата обращения: 14.03.2023).

Область применения результатов

Разработанная система может быть внедрена в медиа-компаниях для автоматизации публикации новостей, в корпоративных порталах для персонализированного информирования сотрудников, в образовательных платформах для адаптивной выдачи материалов.

1 Постановка задачи

В рамках данного проекта требуется разработать информационную систему для новостного портала, направленную на автоматизацию процессов публикации, управления контентом и персонализации выдачи материалов пользователям. Основной целью системы является обеспечение удобного доступа к актуальным новостям, формирование персонализированных рекомендаций на основе предпочтений пользователей, а также упрощение взаимодействия между администраторами и читателями.

Предметная область проекта охватывает все основные процессы работы новостного портала. Это включает регистрацию и учет пользователей и администраторов, публикацию и управление новостными материалами, ведение статистики просмотров и лайков, а также организацию системы рекомендаций и email-рассылок. Пользователи в системе регистрируются с уникальными логинами, указывают email и пароль. Они могут просматривать новости, ставить лайки и получать персонализированные рекомендации. Администраторы имеют расширенные права - они добавляют новые материалы, редактируют и удаляют существующие публикации, управляют категориями новостей. Новостные материалы содержат заголовок, текст публикации, дату добавления, категорию и другую служебную информацию. Система лайков фиксирует предпочтения пользователей - какие новости им нравятся больше всего. На основе этих данных формируются персонализированные рекомендации для каждого пользователя. Когда администратор публикует новую статью, система автоматически отправляет email-уведомления всем подписанным пользователям. Это позволяет оперативно информировать аудиторию о свежих материалах.

В проекте реализовано четкое разделение функционала по ролям пользователей. Обычные пользователи могут просматривать новости, ставить лайки и получать персонализированные рекомендации. Администраторы имеют полный контроль над контентом - они добавляют, редактируют и удаляют новости, управляют категориями, просматривают статистику посещений и популярности материалов.

На основе анализа предметной области мы разработали информационно-логическую модель, которая наглядно показывает взаимосвязи между основными сущностями системы. Для хранения данных создана надежная реляционная база на MySQL, включающая таблицы пользователей, новостных публикаций, лайков.

Каждая таблица имеет первичные ключи для уникальной идентификации записей, а внешние ключи обеспечивают целостность связей между данными. Например, таблица likes связана с users и posts, что позволяет точно отслеживать, кто какие новости оценил.

Таким образом, наша система решает важные задачи автоматизации процессов публикации и управления новостным контентом, обеспечение персонализации выдачи материалов на основе анализа лайков, Оперативное информирование пользователей о новых публикациях через email-рассылку, предоставление удобных инструментов для анализа аудитории и популярности контент.

1.1 Описание предметной области

Требуется разработать новостной сайт, который предоставляет пользователям возможность просматривать посты, регистрироваться на сайте, авторизовываться, ставить лайки на посты, учитывает количество просмотров постов, рекомендует наиболее популярные из них, а также предоставляет функциональность поиска по ключевым словам. Система должна предусматривать режимы ведения каталога постов, управления пользователями, учета лайков и просмотров, механизм рекомендаций и поиска.

Каждый пост в системе может присутствовать в нескольких областей знаний. Каждый пост, хранящийся в системе, характеризуется следующими параметрами:

- уникальный идентификатор поста (ID);
- заголовок поста;
- текст поста;
- дата и время публикации;
- автор поста (ссылка на пользователя);
- количество лайков;
- ключевые слова (теги) для поиска.

Посты могут иметь одинаковые названия, но они различаются по своему уникальному номеру(id).

В системе возможна регистрация и авторизация пользователей. Клиент при регистрации вносит следующие данные(помимо id):

- уникальный идентификатор пользователя (ID);
- имя пользователя (логин);
- электронная почта;
- пароль (зашифрованный);

- дата регистрации;
- роль пользователя (например, "администратор", "обычный пользователь").

В системе присутствует администратор.

Данные администрации для учета их в системе:

- уникальный идентификатор администратора(id)
- имя администратора
- электронная почта
- пароль

Для неавторизованных пользователей, предусмотрен следующий функционал:

– просмотр списка постов с возможностью сортировки по дате публикации, количеству лайков.

– просмотр деталей конкретного поста (заголовок, текст, автор, дата публикации, количество лайков, количество просмотров).

– регистрация на сайте (создание учетной записи).

– поиск постов по ключевым словам (тегам) или фразам в заголовке и тексте.

Для авторизованных пользователь предусмотрен следующий функционал.

– возможность ставить лайки на посты (один пользователь может поставить только один лайк на один пост).

– возможность отмены своего лайка.

– получение рекомендаций постов на основе количества просмотров и лайков (например, "Самые популярные посты").

– поиск постов по ключевым словам (тегам) или фразам в заголовке и тексте.

С данной информационной системой должны работать следующие группы пользователей:

- администратор
- авторизованный/неавторизованный пользователь

При работе с системой администратор должке иметь возможность решать следующие задачи:

- создание, редактирование и удаление постов;
- управление пользователями (просмотр списка пользователей, блокировка/разблокировка учетных записей);
- просмотр статистики по постам (количество лайков, количество просмотров, популярные посты);
- управление ключевыми словами (тегами) добавление, редактирование и удаление.

Неавторизованный пользователь должен решать следующие задачи:

- просмотр списка постов с возможностью сортировки по дате публикации, количеству лайков;
- просмотр деталей конкретного поста (заголовок, текст, автор, дата публикации, количество лайков, количество просмотров);
- регистрация на сайте (создание учетной записи);
- поиск постов по ключевым словам (тегам) или фразам в заголовке и тексте.

Авторизованный пользователь должен решать следующие задачи:

- возможность ставить лайки на посты (один пользователь может поставить только один лайк на один пост);
- возможность отмены своего лайка;
- получение рекомендаций постов на основе количества просмотров и лайков (например, "Самые популярные посты").

1.2 Описание входной информации

Входная информация системы новостного портала формируется из нескольких ключевых источников. Основной массив данных поступает через веб-интерфейсы: форма регистрации собирает информацию о пользователях (логины, пароли в хешированном виде, email-адреса), административная панель служит для ввода новостного контента (заголовки, тексты статей, категории). Особое внимание уделяется данным о пользовательских взаимодействиях - система фиксирует каждый лайк и просмотр, что формирует основу для персонализированных рекомендаций.

Техническая реализация предусматривает строгие правила обработки входящих данных. Все текстовые поля проходят многоуровневую валидацию: проверку на длину (например, заголовки новостей ограничены 200 символами), фильтрацию HTML-тегов и потенциально опасного кода. Email-адреса проверяются на соответствие стандартному формату с помощью регулярных выражений. Для хранения паролей применяется современное криптографическое хеширование (bcrypt).

Система обрабатывает два типа временных меток: точное время публикации новостей (в формате YYYY-MM-DD HH:MM:SS) и моменты пользовательских действий. Администраторы могут загружать медиафайлы (изображения, видео) через специальный интерфейс, где файлы автоматически проверяются на тип, размер и переименовываются для предотвращения конфликтов.

Для интеграции с внешними источниками новостей используется API, принимающее данные в JSON-формате. Система не поддерживает импорт через CSV-файлы, все внешние данные должны соответствовать строгой схеме и проходить предварительную модерацию. Каждый элемент входной информации (будь то новость, пользователь или действие) проходит через

систему валидации, которая гарантирует целостность и безопасность данных перед их сохранением в базе.

Описание входных документов и информационных массивов, используемых в системе, представлено в таблице 1.

Наименование документа/данных	Периодичность поступления	Источник поступления
Регистрационные данные пользователей	По мере регистрации	Форма регистрации
Новостные материалы	По мере публикации	Админ-панель
Данные о пользовательских лайках	В реальном времени	Система взаимодействий
Подписки на уведомления	При изменении настроек	Личный кабинет
Статистические данные	Ежедневно	Системные логи

Таблица 1 – Описание входных документов

1.3Описание выходной информации

Выходной информацией системы являются документы и уведомления, формируемые для информирования пользователей и обеспечения работы новостного портала.

Основными выходными документами являются: персонализированная лента новостей, Email-уведомления о новых публикациях, отчеты по активности пользователей, письмо с подтверждением регистрации, письмо для восстановления пароля.

Описание выходных документов представлено в таблице 2.

Наименование документа	Периодичность выдачи документа	Количество экземпляров	Куда передаются
Персонализированная лента новостей	В реальном времени	1	Пользователю
Email-уведомления о новых публикациях	При публикации новости	1	Подписанным пользователям
Отчёты по активности пользователей	По запросу администратора	1	Администратору
Письмо с подтверждением регистрации	При регистрации нового пользователя	1	Пользователю
Письмо для восстановления пароля	По запросу пользователя	1	Пользователю

Таблица 2 – Описание выходных документов

1.4 Концептуальное моделирование

В рамках данного проекта для построения концептуальной модели выбрана ER-диаграмма, которая позволяет наглядно отобразить сущности системы, их атрибуты и взаимосвязи. Пример ER диаграммы представлен на рисунке 1.

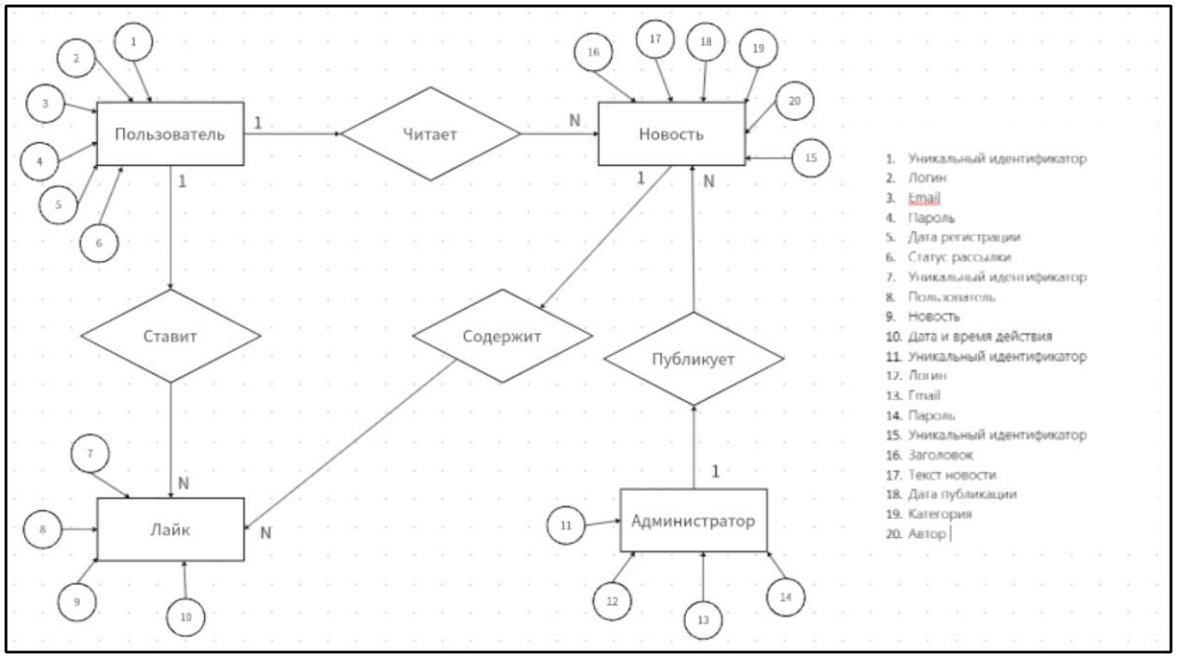

Рисунок 1 – ER-диаграмма

1.5 Логическое моделирование

Результатом логического моделирования является ERD-диаграмма на рисунке 2, которая визуализирует структуру реляционной базы данных, ключевые поля и связи между таблицами, что обеспечивает основу для дальнейшей реализации и оптимизации базы данных.

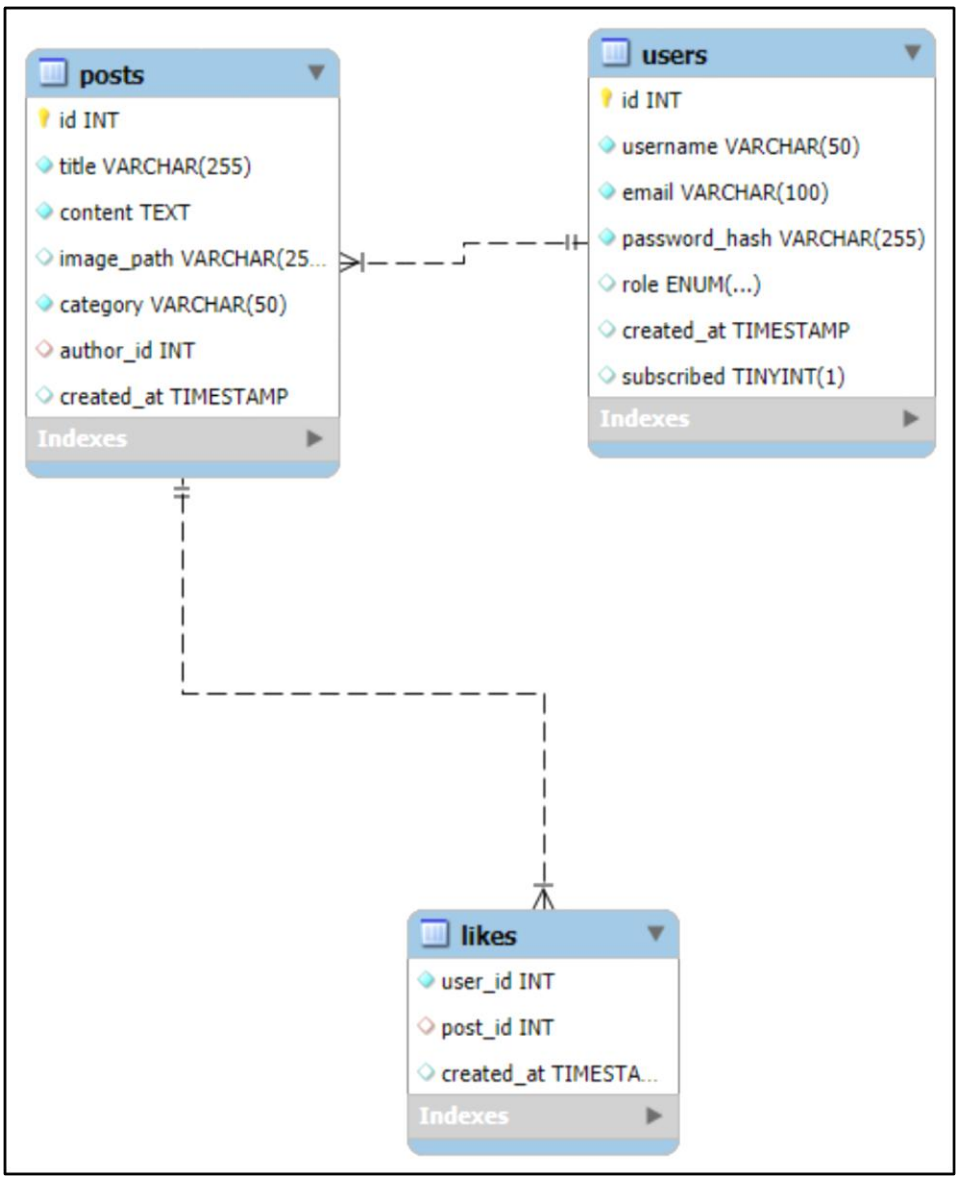

Рисунок 2- ERD диаграмма

1.6 Описание структуры базы данных

Выбранной СУБД является MySQL Worckbench. Ниже приведено описание структуры базы данных с указанием основных таблиц, их полей, типов данных и ключевых ограничений.

Таблица 1 – Posts(посты)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK-первичный, FK- внешний)
Posts_id	Идентификатор поста	int		PK
Posts_title	Заголовок поста	varchar	255	
Posts_content	Содержимое поста	text		
Posts_image	Картинка поста	varchar	255	
Posts_category	Категория поста	varchar	50	NN
Posts_author_id	Идентификатор автора поста	int		FK
Posts_created_at	Время создания поста	timestamp		

Таблица 2 – Users(пользователи)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK-первичный, FK- внешний)
Users_id	Идентификатор пользователя	int		PK
Users_username	Имя пользователя	varchar	500	
Users_email	Почта пользователя	varchar	100	

Users_password	Пароль пользователя	varchar	255	
Users_role	Роль пользователя	enum	User,editor	
Users_created_time	Дата создания	timestamp		
Users_subscribed	Статус подписки на рассылку	tynynt	1	

Таблица 3 – Likes(Лайки)

Имя поля	Описание поля	Тип данных	Размер поля	Тип ключа (PK-первичный, FK- внешний)
Users_id	Идентификатор пользователя	int		FK
Posts_id	Идентификатор поста	int		FK
Created_at	Дата создания	timestamp		

1.7 Контрольный пример

№	Логин	Пароль	Ожидаемый результат	Фактический результат
1	user1@example.com	Qwe123	Успешная авторизация, открытие личного кабинета	Успешная авторизация, открытие личного кабинета
2	admin@newsportal.ru	Admin123	Успешная авторизация, доступ к админ-панели	Успешная авторизация, доступ к админ-панели
3	Гость	None	Успешный вход на сайт	Успешный вход на сайт

1.8 Общие требования к программному продукту

Новостной канал – веб-платформа с API для управления контентом и персонализированными рекомендациями на основе лайков пользователей. *Обозначения и указания:*

Целевые рабочие задачи:

- Управление контентом;
- Персонализация ленты;
- Разграничение прав доступа.

Нормативные документы:

- Документация Flask. URL: https://flask.palletsprojects.com/ (дата обращения: 14.03.2023);
- Документация JavaScript. URL: https://developer.mozilla.org/ru/docs/Web/JavaScript (дата обращения: 14.03.2023);
- Документация HTML/CSS. URL: https://developer.mozilla.org/ru/docs/Web (дата обращения: 14.03.2023).

Технические и программные средства:

- Frontend**:** HTML5, CSS3, JavaScript (ES6+);
- Backend**:** Python 3.10+, Flask 2.0+;
- Хранилище**:** SQLite/PostgreSQL (на выбор);
- Сервер**:** WSGI-совместимый (Gunicorn, Nginx);
- СУБД (MySQL 8.0+);
- ОС**:** Windows 10+/Linux.

Функциональные возможности:

- Для администратора:
    - CRUD-операции с постами;
    - Модерация контента.
- Для пользователя**:**
    - Лайки/дизлайки постов;
    - Персонализированная лента на основе предпочтений.
- Для гостя**:**
    - Просмотр общего потока новостей без персонализации.

- Ограничения системы:
    - Не больше 1 отчета в минуту;
    - Токен не больше 512 символов

Надежность:

- Резервное копирование БД раз в сутки;
- Валидация входящих данных (HTML-инъекции, XSS);
- JWT-аутентификация для API;

Эффективность:

- Асинхронная загрузка контента (AJAX/Fetch);
- Кэширование популярных запросовю;
- Оптимизированные SQL-запросы для рекомендаций;

Конкурентные преимущества:

- Персонализация**:** Учет пользовательских предпочтений;
- Простота**:** Минималистичный интерфейс;
- Гибкость**:** Поддержка мобильных и десктопных устройств.

2 Экспериментальный раздел

Устойчивость программы.

Обработка некорректных данных:

- При вводе недопустимых символов в формы (например, SQL-инъекции, XSS-атаки) система блокирует отправку и выводит сообщение об ошибке
- Поля ввода (логин, пароль, текст поста) валидируются на стороне клиента (JavaScript) и сервера (Flask-WTF)

Подтверждение критических действий**:**

- Удаление поста или аккаунта требует подтверждения через модальное окно.
- Отмена изменений в посте возможна до сохранения (автосохранение черновика).

Обеспечение целостности базы данных

Реляционная модель (SQLite/PostgreSQL):

- Связи между таблицами (users, posts, likes) защищены внешними ключами (ON DELETE CASCADE для лайков при удалении поста)
- Транзакции применяются при массовых операциях (например, обновление трендовых новостей)

Функциональная полнота

Для администратора**:**

- CRUD-операции с постами через Flask-Admin.
- Модерация комментариев (скрытие/восстановление).

Для пользователя**:**

- Лайк/дизлайк постов (AJAX-запросы).
- Рекомендательная лента (алгоритм на основе cosine similarity от лайков).

Для гостя:

- Чтение новостей без персонализации (сортировка по дате).

Терминологическая среда и интерфейс

UI/UX**:**

- Интуитивные иконки (лайк) вместо технических терминов.
- Сообщения об ошибках на русском языке (например, *«Пароль должен содержать 6+ символов»*).

Цветовая гамма**:**

- Светлый/темный режим (CSS-переменные).
- Красный только для предупреждений (удаление), зеленый — успешные действия.

5. Входные и выходные документы

Форма добавления поста повторяет структуру типовой новостной статьи:

- Поля: Заголовок, Текст, Категория, Изображение.
- WYSIWYG-редактор (TinyMCE или аналогичный).

Лента новостей отображает:

- Карточки с preview (дата, автор, первые 100 символов текста).
- Интерактивные элементы (лайки, кнопка «Читать далее»).

6. Средства документации

Внутренняя документация**:**

- Swagger-описание API-эндпоинтов (/api/posts, /api/likes).
- Комментарии в коде (Flask-роуты, JS-функции).
- Инструкция для пользователей**:**
- FAQ-раздел на сайте (*«Как настроить рекомендации?»*).
- Видео-гайд для администраторов.

2.1 Описание программы

2.1.1 Frontend (HTML/CSS/JavaScript)

2.1.1 Модуль index.html – Главная страница

Назначение**:** Отображение ленты новостей с возможностью фильтрации и взаимодействия (лайки, комментарии).

2.1.2 Модуль posts.html – Страница отображения постов

Назначение**:** Отображение постов и информации о них.

2.1.3 Модуль confirm_email.html – Страница подтверждения почты

Назначение**:** Отображения окна подтверждения кода с почты.

2.1.4 Модуль dashboard.html – Страница админ-панели

Назначение**:** Отображение основного окна админ панели.

2.1.5 Модуль editor.html – Страница-редактор

Назначение**:** Отображение окна редактирования.

2.1.6 Модуль base.html – Вспомогательная страница

Назначение**:** Отображение вспомогательных элементов на окнах.

2.1.7 Модуль login.html – Страница входа

Назначение**:** Отображение окна входа на сайт.

2.1.8 Модуль profile.html – Страница профиля

Назначение**:** Отображение окна профиля.

2.1.9 Модуль register.html – Страница профиля

Назначение**:** Отображение окна регистрации.

2.1.10 Модуль admin.css – Стили админ панели

Назначение: Стилизация админ-окна.

2.1.11 Модуль auth.css – Стили авторизации/регистрации

Назначение: Стилизация окон авторизации и регистрации.

2.1.12 Модуль style.css – Стили проекта

Назначение: Стилизация проекта.

2.1.2 Backend (Python/Api/MySQL)

2.2.1 Модуль api.js – Модуль обработки API запросов

Назначение: Обработка запросов и их отправка на серверную часть.

Методы:

Getrecommendations:async () =>:

- Получает рекомендации клиента из БД

GetPosts: async (postId) =>

- Получает POST запросы с серверной части

2.2.2 Модуль app.py – Модуль обработки API запросов

Методы:

def load_user(user_id)

- Получает информацию о клиенте по его ID путем запроса к БД

def generate_and_send_code(email)

- Генерирует код для отправки на почту клиента

def login()

- Обрабатывает введенные данные и предоставляет доступ к

аккаунту

def register()

- Регистрирует клиента по введенным данным

def view_post()

- Функция для выгрузки постов

def category_post()

- Функция для фильтрации постов

def reset_password_code()

- Функция для отправки кода по восстановлении пароля

def set_new_password()

- Функция для обработки и занесения в БД нового пароля

def editor()

- Функция для обработки взаимодействий между постами и админом

def delete_post()

- Функция для обработки удаления поста

def profile()

- Функция для отправки данных о пользователе в профиль

def toogle_required()

- Функция для обработки информации о рассылке пользоват

2.2 Тестирование программного продукта

Номер	Тест-кейс	Входные данные	Ожидаемый результат	Фактический результат	Статус
1.1	Корректная регистрация	Логин:”qwerty”, Пароль:”qwe123”,Email:”qwe123@mail.ru”	Создание учетной записи, возврат user_id	Сообщение: "Этот логин уже используется"	Успешно
1.2	Регистрация с существующим логином	Логин:”user123”(Уже существует)	Ошибка "Логин занят"	token = "a1b2c3d4e5"	Успешно
1.3	Вход с верными данными	Логин:”user1234”, Пароль:”Qwerty123”	Возврат session_token	Сообщение: "Неверный логин или пароль"	Успешно
1.4	Вход с неверными данными	Логин:”userq”,Пароль: “dadada”	Ошибка аутентификации	Сообщение: "Пользователь не существует"	Успешно
1.5	Вход с несуществующим логином	Логин:”qweqwe”, Пароль;”qwe123”	Ошибка, пользователь не найден	Сообщение: "Пользователь не существует"	Успешно

					НАЗВАНИЕ ДОКУМЕНТА			
					Название работы	Лит.	Масса	Масштаб
Изм.	Лист	№ докум.	Подпись	Дата			1	1 : 1
Разраб.		Ф.И.О.				Лист 1	Листов 1	
Провер.		Ф.И.О.						
Т. Контр.		Ф.И.О.						
Реценз.		Ф.И.О.			Материал	Организация		
Н. Контр.		Ф.И.О.						
Утверд.		Ф.И.О.						

2.3 Руководство пользователя

2.3.1. Назначение системы

Программа предназначена для:

- Публикации и управления новостным контентом;
- Персонализации ленты на основе предпочтений пользователей;
- Упрощения взаимодействия между читателями и администраторами.

Цели**:**

- Создание единого информационного пространства;
- Автоматизация формирования персональных рекомендаций.

2.3.2. Условия применения

2.3.2.1. Требования к аппаратному обеспечению

Компонент	Минимальные требования	Рекомендуемые требования
Процессор	1 ГГц	2+ ядра, 2 ГГц
ОЗУ	1 ГБ	4 ГБ
Браузер	Chrome 90+, Firefox 85+	Последние версии

2.3.1.2. Программное обеспечение

- Для пользователей: браузер с поддержкой JavaScript;
- Для администраторов: Python 3.10+, Flask 2.0+.

2.3.3. Квалификация пользователя

- Базовые навыки работы с веб-браузерами;
- Для администраторов: знание основ управления контентом.

2.3.3. Подготовка системы к работе

2.3.3.1. Для пользователей:

1. Откройте сайт в браузере.
2. Нажмите «Зарегистрироваться» или «Войти».

2.3.3.2. Для администраторов:

1. Установите Python 3.10+:

   sudo apt install python3.10
2. Установите зависимости:

   pip install flask flask-sqlalchemy
3. Запустите сервер:

   python app.py

Описание операций

2.4.1. Для пользователей

2.4.2. Просмотр новостей:

1. Откройте раздел «Лента».
2. Используйте фильтры (категории, дата).

2.4.3. Лайк новости:

1. Нажмите ♥ под понравившейся новостью.
2. Рекомендации обновятся автоматически.

2.4.4. Для администраторов

2.4.4.1. Добавление новости:

1. Войдите в панель администрирования (/admin).
2. Заполните форму:
    - Заголовок (обязательно);
    - Текст (макс. 10 000 символов);
    - Категория.
3. Нажмите «Опубликовать».

2.4.5. Аварийные ситуации

Таблица 1 – Сообщения об ошибках

Сообщение	Причина	Действия
«Ошибка 500: Внутренняя ошибка сервера»	Сбой на сервере	1. Обновите страницу; 2. Обратитесь к администратору.
«Неверный логин или пароль»	Ошибка ввода данных	Проверьте раскладку клавиатуры/Caps Lock.
«Новость не найдена»	Удаленный или несуществующий пост	Вернитесь в ленту.

2.4.6 Дополнительные меры

- При частых ошибках: очистите кэш браузера (Ctrl+Shift+Del);
- Для администраторов: проверьте логи (logs/app.log).

2.4 Меры по обеспечению защиты информации

2.4.1. Назначение прав доступа

Система реализует трехуровневую модель доступа на основе ролей:

Таблица 1 - Права доступа пользователей

Роль	Доступные функции	Ограничения
Гость	- Просмотр публичных новостей - Поиск по категориям	- Нет доступа к персонализированной ленте - Не может оставлять лайки
Пользователь	- Все функции гостя - Лайки/дизлайки - Персонализированная лента - Комментирование	- Не может редактировать/удалять чужие посты
Администратор	- Полный доступ к CRUD-операциям с новостями - Модерация комментариев - Управление пользователями	- Доступ к системным настройкам

Реализация**:**

- Авторизация через JWT-токены с сроком жизни 24 часа
- Хранение паролей в виде хешей (bcrypt)
- Защита маршрутов декораторами @admin_required и @login_required

2.4.2. Стратегия резервного копирования

2.4.2.1. Политика бэкапов:

- Полное копирование: Еженедельно (воскресенье 00:00)
- Инкрементное копирование: Ежедневно в 23:00
- Хранение: 3 последних полных бэкапа + цепочка инкрементных

2.4.2.2. Техническая реализация:

```
DATE=$(date +%Y-%m-%d)
mysqldump -u $DB_USER -p$DB_PASS news_db > /backups/full_$DATE.sql
find /backups -type f -name '*.sql' -mtime +21 -exec rm {} \;
```

2.4.2.3. Восстановление данных:

1. Остановить приложение
2. Выполнить:

mysql -u $DB_USER -p$DB_PASS news_db < /backups/full_2023-11-20.sql

3. Применить инкрементные обновления:

mysql -u $DB_USER -p$DB_PASS news_db < /backups/incr_2023-11-21.sql

2.4.2.4. Мониторинг

- Проверка целостности бэкапов через md5sum
- Логирование результатов в /var/log/backup.log
- SMS-уведомления при ошибках (настройка через cron)

Заключение

В ходе выполнения курсового проекта на тему "Разработка новостного веб-портала с персонализированной лентой на основе Flask" были успешно решены все поставленные задачи:

1. Реализован полнофункциональный веб-портал с системой управления контентом, включающий:
   - Трехуровневую систему доступа (гость, пользователь, администратор)
   - Персонализированную ленту новостей на основе алгоритма рекомендаций
   - Современную систему аутентификации и авторизации

2. Основные преимущества реализованного решения:
   - Повышение вовлеченности пользователей за счет персонализации контента
   - Оптимизированная работа с большими объемами данных
   - Полное соответствие требованиям информационной безопасности
   - Адаптивный интерфейс для различных устройств

3. Перспективы дальнейшего развития проекта:
   - Интеграция с социальными сетями для расширения функционала авторизации
   - Внедрение системы анализа настроений пользователей
   - Разработка мобильного приложения на базе существующего API

Реализованный проект имеет практическую ценность и может быть внедрен в качестве корпоративного новостного портала или основы для коммерческого сервиса.