ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

Кафедра вычислительных систем

КУРСОВАЯ РАБОТА

по дисциплине «Web-разработка»

на тему

Разработка web-приложения «Личный дневник»

Выполнил студент Кулик Павел Евгеньевич

Ф.И.О.

Группы ИС-241

Работу принял Старший преподаватель кафедры ВС Курзин А.С.

Оценка Подпись

Новосибирск – 2025

СОДЕРЖАНИЕ

Постановка задачи

Целью данной работы является реализация web-приложения «Личный дневник». Приложение должно состоять из 2 частей: фронтенд и бэкенд.

Требования к технологиям при разработке бэкенда:

- .NET Web API;
- СУБД для хранения данных;
- ORM-фреймворк.

Фронтенд должен быть реализован с использованием React.

Требования к приложению:

- Может быть использовано множеством пользователей;
- Аутентификация и авторизация;
- Безопасное хранение паролей;
- Разделение ролей пользователей;
- Использование JWT для защиты данных.

Процесс выполнения

1. Бэкенд

Реализация бэкенда началась с определения моделей, которые будут использоваться в приложении. Были определены следующие модели:

- Модель Note для хранения пользовательских заметок. Хранит в себе следующее:
  - Id;
  - Title;
  - Content;
  - Mood;
  - CreatedAt;
  - UpdatedAt;
  - UserId;
- Модель User для хранения информации о пользователе. Хранит в себе следующее:
  - Id;
  - Email;
  - PasswordHash;
  - Role;
  - Notes.

Далее модели были связаны с СУБД sqlite с использованием EntityFrameworkCore. Для полей моделей были описаны технические требования, такие как обязательность значений в полях, отношения (у одного пользователя много заметок и т.д.), максимальная длина текстовых полей, поведение при удалении и другие.

Затем были реализованы сервисы, содержащие CRUD-операции и валидацию для защиты бизнес-логики и консистентности данных. Были реализованы 3 сервиса: AuthService, NoteService, UserService. Для сервисов

дополнительно были реализованы соответствующие DTO, упрощающие взаимодействие с ними. Помимо этого, был реализован специальный шаблонный класс Result, который хранит статус выполнения операции, данные и сообщения об ошибках. Каждый метод любого сервиса возвращает данные и сообщения об ошибках через этот класс.

Затем были реализованы контроллеры, формирующие интерфейс для взаимодействия фронтенда с бэкендом. Контроллеры AuthController, NoteController, UserController соответствуют описанным выше сервисам. Через них проходят запросы для управления заметками, регистрации и входу пользователей, а также управление пользователями для админов приложения. Контроллеры используют авторизацию для управления правами пользователей и защиты данных от несанкционированного доступа. Контроллеры включают в себя следующие эндпоинты:

- (post) /api/v1/Auth/register;
- (post) /api/v1/Auth/login;
- (post) /api/v1/Note;
- (get) /api/v1/Note;
- (get) /api/v1/Note/{id};
- (put) /api/v1/Note/{id};
- (delete) /api/v1/Note/{id};
- (get) /api/v1/User/{id};
- (put) /api/v1/User/{id};
- (get) /api/v1/User;

Далее была написана программа, которая объединяет вместе всё, что было реализовано, настраивает подключение к базе данных, создаёт базу данных при её отсутствии, сразу создаёт пользователя с ролью админа и выводит в терминал его JWT-токен, настраивает использование swagger для тестирования эндпоинтов (в swagger используется токен первого пользователя для

авторизированного доступа) и настраивает CORS-политику, разрешающую фронтенду взаимодействовать с бэкендом.

2. Фронтенд

Фронтенд был реализован с использованием vite и React. В приложения были реализованы следующие страницы:

- Вход/регистрация;
- Все заметки;
- Создание новой заметки;
- Просмотр заметки;
- Админ-панель;
- Список пользователей (только для админов).

На странице со всеми заметками выводятся все заметки пользователя, есть возможность совершить поиск по названиям, просмотреть заметки нажатием на них, есть кнопка для создания новой заметки.

На странице создания новой заметки есть возможность выбрать настроение заметки, заголовок (обязательно), ввести содержание (обязательно), посмотреть makdown.

На странице просмотра заметки текст отображается в формате markdown.

На админ-панели есть возможность перейти к просмотру собственных заметок, либо перейти к списку пользователей.

При открывании списка пользователей доступен список всех пользователей и поиск по почте. Админ не может смотреть заметки пользователей на клиентской стороне, но может удалить пользователя.

Помимо этого, реализовано переключение между светлой и тёмной темой.

Демонстрация работы приложения

На рис. 1 представлена страница входа.

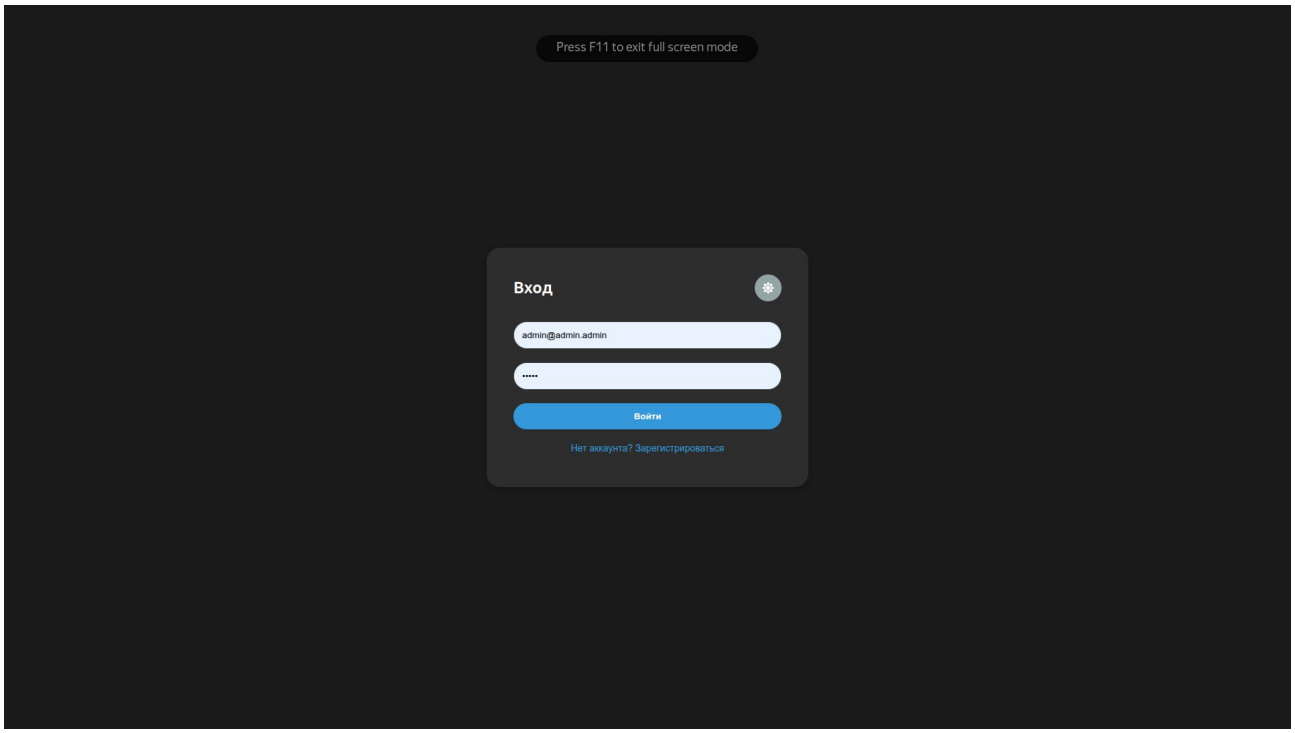

Рисунок 1. Страница входа

Если в приложение входит админ, то после входа он видит админ-панель, как это изображено на рис. 2. Админ может добавлять собственные заметки, для этого необходимо нажать кнопку «Мои заметки». После нажатия на эту кнопку откроется страница, которая аналогично выглядит у обычного пользователя, только для пользователя она будет отображена сразу после входа. Страница со списком всех заметок изображена на рис. 3. Как видно на рисунке, на этой странице отображаются названия и настроения каждой заметки. Есть возможность совершить поиск по заметкам, либо открыть заметку. Если открыть заметку, то её содержимое будет отображено в markdown стиле, как это видно на рис. 4. Можно нажать кнопку, чтобы вернуться назад и создать новую заметку, в таком случае пользователь попадёт на страницу создания заметку, аналогичную странице редактирования, в которую можно попасть из открытой заметки. Страница создания/редактирования изображена на рис. 5.

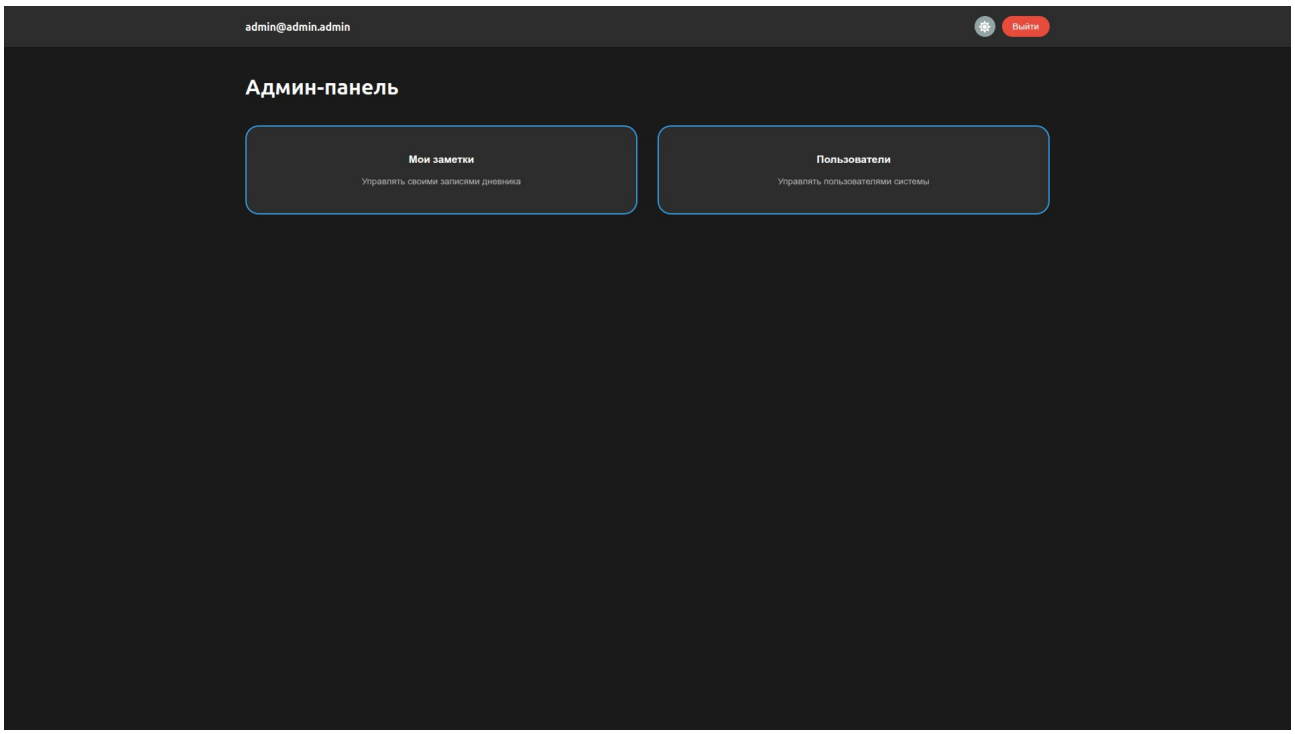

Рисунок 2. Админ-панель

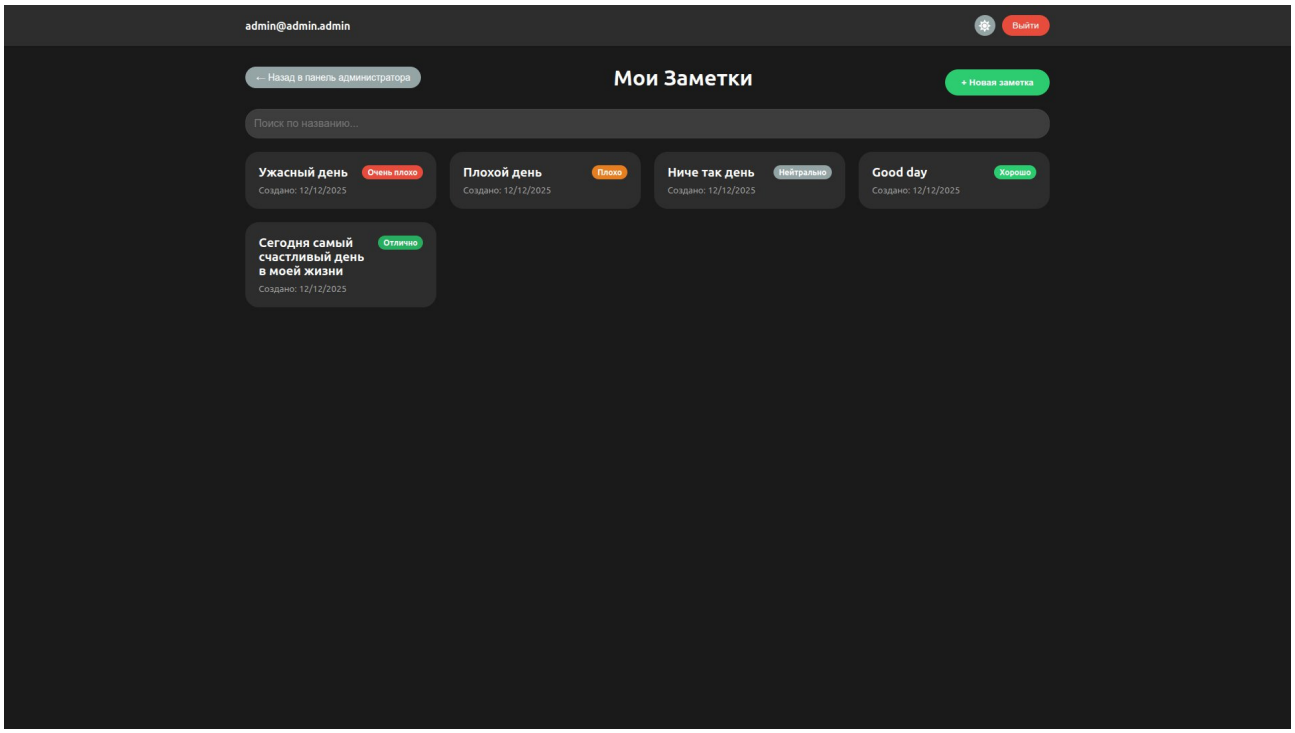

Рисунок 3. Список всех заметок

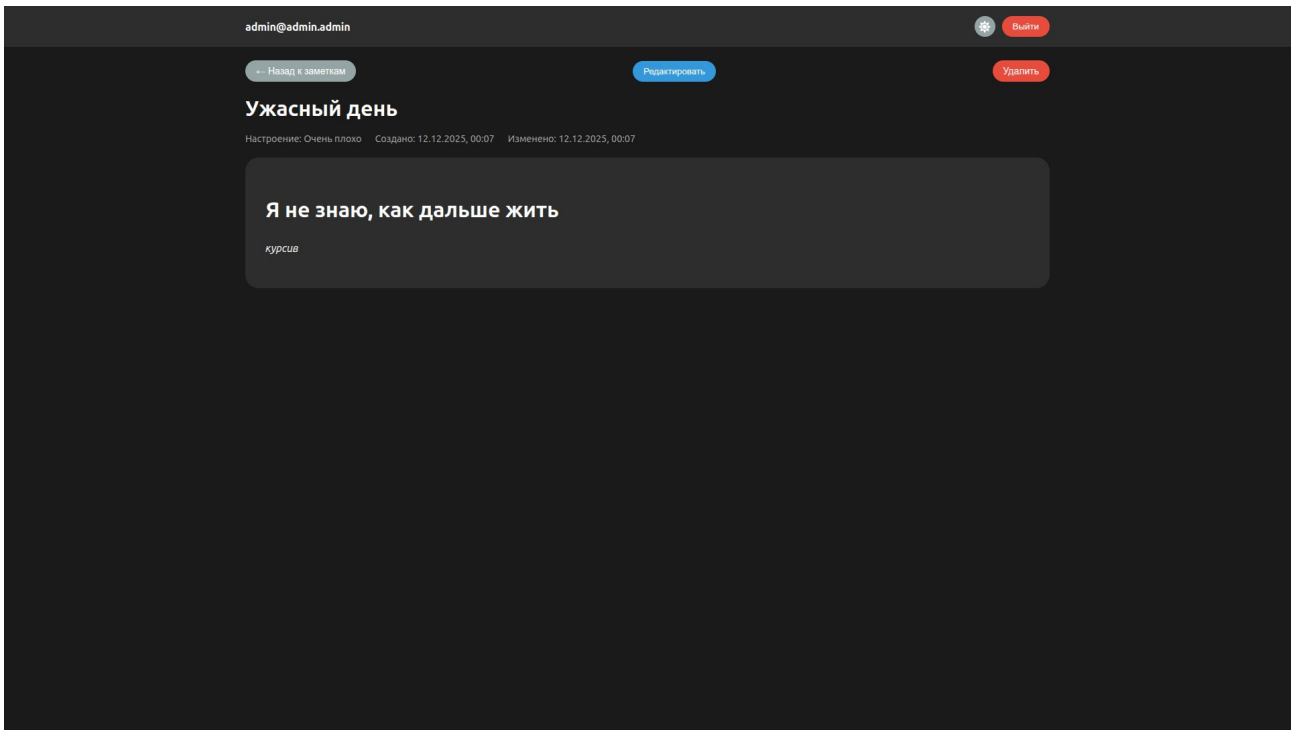

Рисунок 4. Просмотр заметки

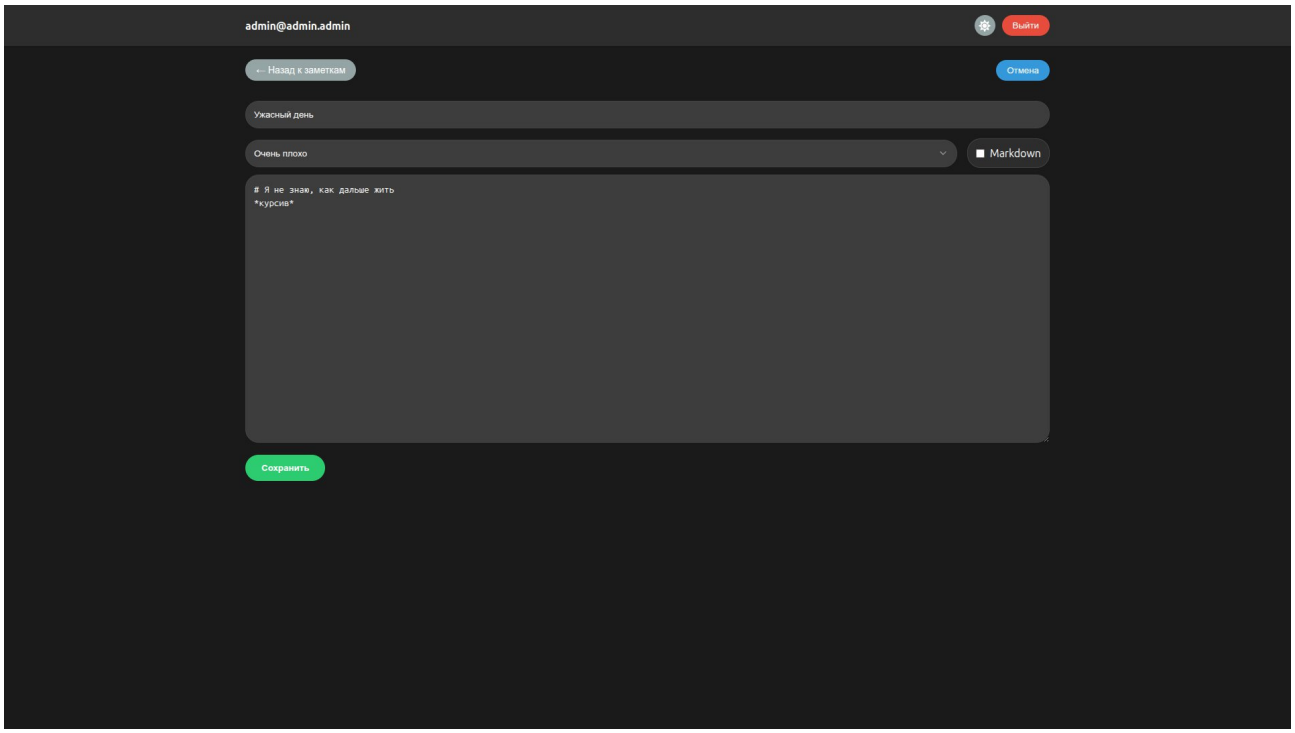

Рисунок 5. Редактирование заметки

При редактировании заметки можно включить предпросмотр markdown, либо сохранить заметку. Заметка не может быть сохранена, если у неё нет названия и содержания.

Также, имеется страница со списком пользователей, доступная только администрам. На этой странице администратор может удалить любого пользователя. Страница со списком пользователей изображена на рис. 6. Для примера включена светлая тема.

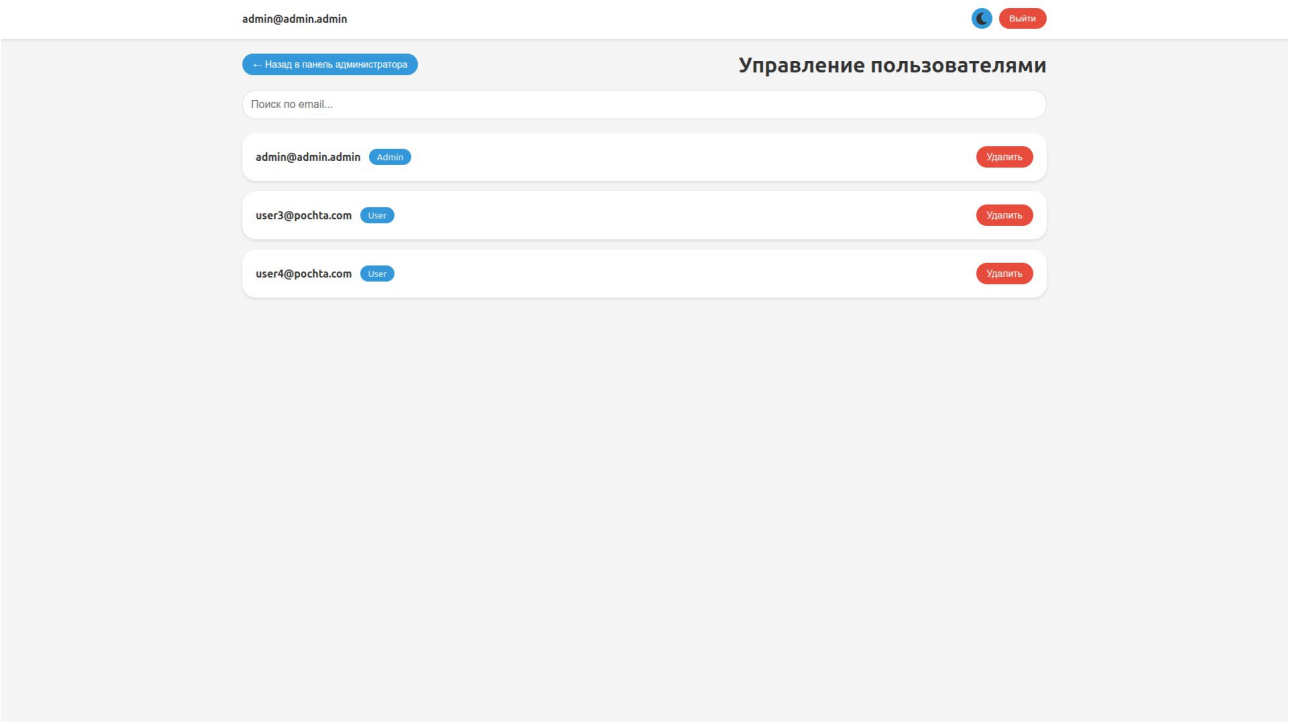

Рисунок 6. Список пользователей

Заключение

В ходе работы было реализовано web-приложение «Личный дневник». В ходе разработки были использованы все технологии в соответствии с изначальными требованиями. Таким образом, задание было выполнено.