**Государственное профессиональное образовательное учреждение**

**ЯО Ярославский градостроительный колледж**

**Курсовой проект**

По МДК 09.02.

**«Оптимизация веб-приложений»**

На тему:

**«**Разработка Web-сайта ветеринарной клиники**»**

Проверил преподаватель Выполнил студент

ФИО Гр. ИС 1-31 \_\_\_\_\_\_

Оценка Косолапов Кирилл Андреевич

Ярославль 2025

Оглавление

**Элементы оглавления не найдены.**

Определение целей и требований

Введение

В современном мире наличие собственного веб-сайта является обязательным элементом успешного продвижения услуг и взаимодействия с клиентами. Ветеринарная клиника не является исключением: сайт позволяет оперативно предоставлять информацию о работе клиники, услугах, акциях, специалистах и режиме работы.

Сайт ветеринарной клиники нужен для повышения узнаваемости бренда, увеличения клиентской базы и удобства пользователей. Он дает возможность быстро найти контактные данные, записаться на прием, ознакомиться с профессиональными компетенциями врачей, а также получить полезные советы по уходу за питомцами. Наличие отзывов, фотогалереи и раздела новостей способствует укреплению доверия к клинике.

В условиях высокой конкуренции на рынке ветеринарных услуг важно иметь удобный, функциональный и привлекательный сайт, который будет соответствовать требованиям SEO-оптимизации и современным ожиданиям пользователей.

Цели и задачи:

1. Обеспечить клиентскую базу ветеринарной клинике
2. Увеличить поток клиентов
3. Обеспечить клиентам удобный доступ к контактным данным клиники

Целевая аудитория

Люди, которые заботятся о своих питомцах и озабоченны их здоровьем

Пол: Любой

Возраст: 20-50 (средний возраст)

Средний доход: 50000 руб в месяц и больше

Анализ рынка и конкурентов:

Ветеринарный центр “ПРИРОДА”

Ссылка - <https://vet-76.ru/>

Зооцентр "ЮВЕНТА"

Ссылка - <https://uventa76.ru/>

Ветеринарная клиника Магистр-Вет

Ссылка - <https://vet-mataeva.ru/>

**Что я предпочту использовать в своем сайте?**

1. Картинка на заднем фоне сайта
2. Вкладке в "шапке" сайта
3. Контактные данные на главной странице

**Чего бы я предпочел избежать?**

1. Нарушение цветовой гаммы и шрифтов сайта в элементах (особенно в контактных данных)
2. Неадаптивные элементы на сайте

Функциональные требования

1. Введение блога поликлиники (новости, фотогалерея)
2. Обеспечение доступа к контактным данным ветеринаров и адресу поликлиники

Сбор семантического ядра

Определение ключевых слов

1. ветеринарная клиника
2. ветеринарная помощь
3. запись к ветеринару
4. лечение кошек Ярославль
5. вакцинация собак
6. стерилизация животных
7. услуги ветеринара
8. консультация ветеринара онлайн
9. ветеринар рядом
10. клиника для животных Ярославль
11. помощь животным круглосуточно
12. хороший ветеринар отзывы
13. ветклиника Ярославль
14. стоматология для животных
15. анализы для животных

Формирование задания на внутреннюю оптимизацию

1. Правильные заголовки на страницах сайта, например:

<h1>Ветеринарная клиника в Ярославле — забота о ваших питомцах</h1>

1. Метатеги (<title>, <meta name="description">) через vue js
2. Оптимизация контента страниц
   * В текстах стараться естественно вписывать ключевые слова
   * Не более 3-5% ключей от общего объема текста
   * Делать разбивку на короткие абзацы, списки, подзаголовки
3. ЧПУ (человекопонятные URL)

Разработка структуры сайта

с учетом SEO

Физическая структура сайта

**Frontend**

Корневая папка: **frontend/**

* **public/** — папка публичных ресурсов (например, favicon).
* **src/** — папка исходного кода проекта:
  + **assets/** — статичные ресурсы (изображения, стили):
    - **photogallery/** — изображения для фотогалереи:
      * background.png
    - Файлы:
      * base.css — базовые стили проекта.
      * main.css — основные стили проекта.
      * logo.png — логотип клиники.
      * news\_placeholder.png — заглушка для новостей без изображения.
  + **components/** — отдельные компоненты проекта:
    - **icons/** — иконки и списки для отображения данных:
      * employeeList.vue — компонент списка сотрудников.
      * feedbacksList.vue — компонент списка отзывов.
      * feedbackTextArea.vue — форма добавления отзыва.
      * newsList.vue — компонент списка новостей.
  + **router/** — маршрутизация сайта:
    - index.js — все пути сайта.
  + **stores/** — глобальное хранилище состояния:
    - index.js — настройки состояния приложения (например, данные пользователей, авторизация).
  + **utils/** — утилиты (пока пусто или для вспомогательных функций).
  + **views/** — страницы сайта (основные представления):
    - AboutUs.vue — страница «О нас».
    - AuthToAdmin.vue — страница авторизации администратора.
    - ContactsUs.vue — страница «Контакты».
    - FeedBack.vue — страница отзывов.
    - HomeView.vue — главная страница.
    - PhotoGallery.vue — страница фотогалереи.
  + **App.vue** — главный компонент приложения (обёртка всех страниц).
  + **main.js** — точка входа приложения Vue.js.

**Файлы конфигурации:**

* .editorconfig — настройки редактора кода.
* .prettierrc.json — настройки форматирования кода.
* Dockerfile — для контейнеризации приложения.
* eslint.config.js — настройки линтинга кода.
* index.html — основной HTML-шаблон.
* jsconfig.json — конфигурация для поддержки импорта модулей.
* package.json — файл зависимостей проекта.
* package-lock.json — зафиксированные версии пакетов.
* README.md — документация проекта.

**Корневая папка: backend/**

* **src/** — исходный код сервера:
  + **auth/** — модуль авторизации (JWT, логика входа в админ-панель).
  + **common/** — общие утилиты, декораторы, константы.
  + **employee/** — модуль управления данными сотрудников (врачи клиники).
  + **feedbacks/** — модуль отзывов (создание, удаление, просмотр отзывов).
  + **file/** — модуль загрузки и управления файлами (например, фото сотрудников, галерея).
  + **news/** — модуль управления новостями (добавление, редактирование, удаление новостей).
  + app.module.ts — основной модуль приложения NestJS, где собираются все остальные модули.
  + main.ts — точка входа в серверное приложение (запуск сервера NestJS).
* **test/** — папка для тестов проекта.
* **uploads/** — папка для хранения загруженных файлов (фото и другие медиа-файлы).

**Конфигурационные файлы:**

* .eslintrc.js — настройки линтинга кода (проверка качества кода).
* .prettierrc — настройки автоформатирования кода.
* backend-0.0.1.tgz — собранный архив проекта (например, для публикации в npm или локального разворачивания).
* Dockerfile — инструкция для контейнеризации backend-приложения (развёртывание через Docker).
* nest-cli.json — конфигурация NestJS CLI для сборки и управления проектом.
* package.json — описание зависимостей и скриптов проекта.
* package-lock.json — фиксированные версии всех пакетов.
* tsconfig.json — базовые настройки компилятора TypeScript.
* tsconfig.build.json — отдельная конфигурация для сборки проекта.

Логическая структура сайта и навигация

1. Главная страница
   1. Конткатные данные
   2. Врачи-сотрудники
   3. Новости
2. Фотогалерея
   1. Фото из поликлинники
3. О нас
   1. О нашей ветеринарной клинике
   2. Наша миссия
   3. Наша команда
   4. Наш подход
4. Контакты
   1. Контактные данные клиники (расширенные)
5. Отзывы
   1. Отзывы клиники
   2. Форма для написания отзыва
6. Авторизация в админ панель
   1. Форма для авторизации в админ панеле (Логин и пароль – admin)
   2. После авторизации все элементы сайта становятся редактируемыми для администратора

Разработка дизайна

Цветовая схема

Типографика

**Комментарии по коду**

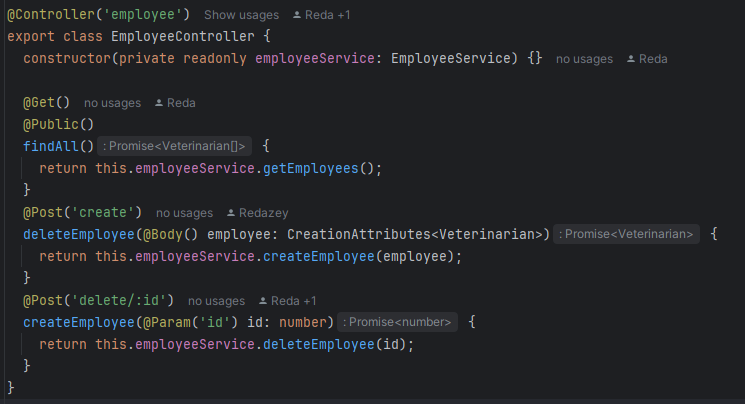
**Backend**

Контроллер, отвечающий за CRUD операции с новостями  


Сервисы, которые использует контроллер



Контроллер, отвечающий за CRUD операции с ветеринарами (сотрудниками)



Сервисы, которые использует контроллер



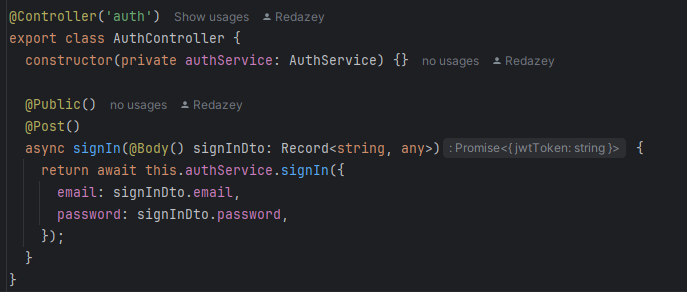
Контроллер, отвечающий за CRUD операции с отзывами



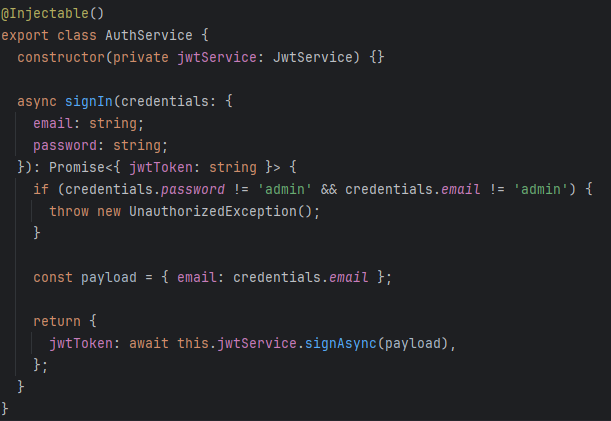
Сервисы, которые использует контроллер



Контроллер, отвечающий за получение JWT – токена для авторизации в админ панеле

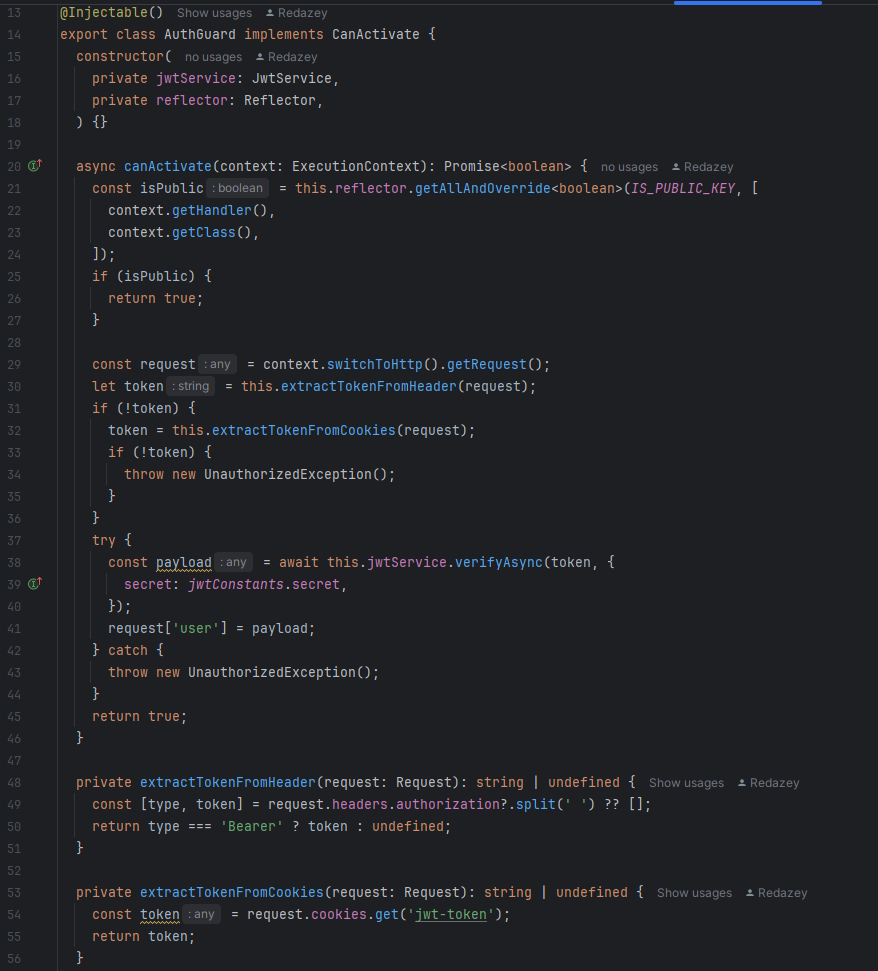


Сервис, который использует контроллер



Метод защиты непубличных (админ) роутов путем проверки JWT токена

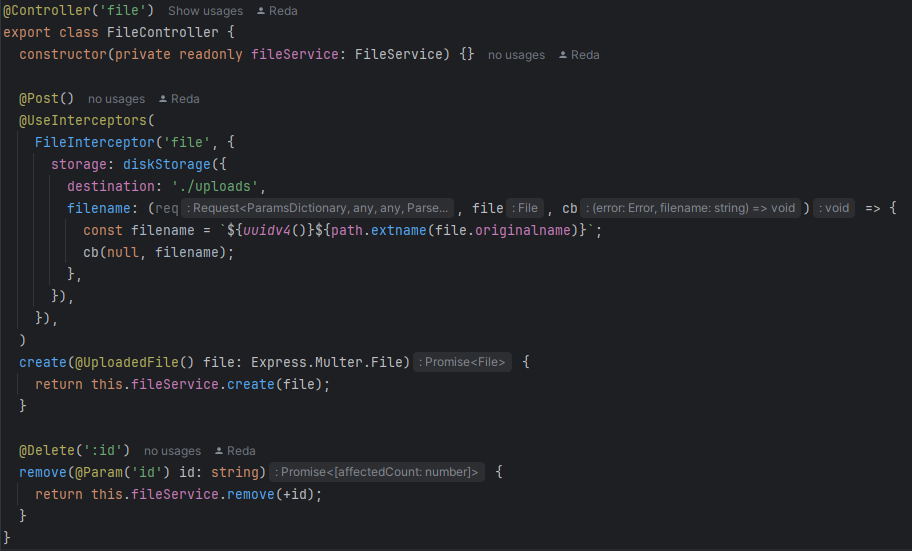
Он встраивается как middleware на все роуты, которые не обозначены как @Public() и проверяет заголовки запроса на наличие Authorization: Bearer [токен], затем валидирует токен и если он корректный, то пускает запрос на роут, в ином случае возвращает статус 403 (Неавторизован)



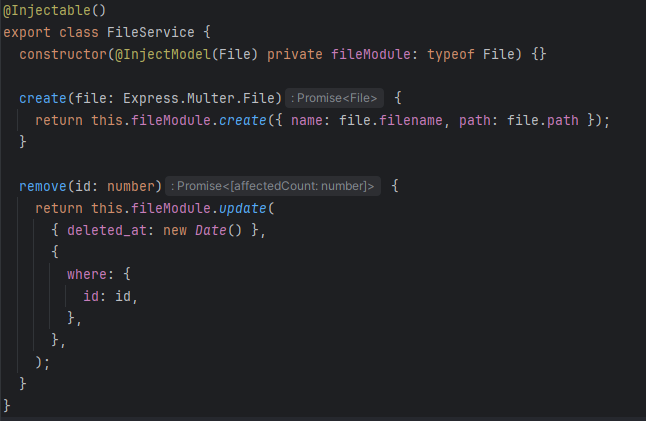
Все картинки, которые загружаются через формы хранятся в backend/uploads/ и подгружаются по адресу <http://localhost:3001/uploads/имя_файла>

Это возможно благодаря папке, которая обозначена как доступная для загрузки с нее файлов

  
Контроллер, отвечающий за загрузку файлов на сервер backend

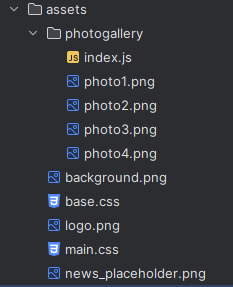


Сервисы, которые использует контроллер

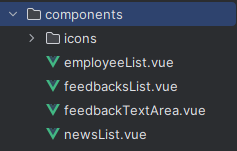


**Frontend**

Стили, статичные картинки (логотип, фотогалерея) и т.д. хранятся в папке assets



**Компоненты, из которых состоят страницы**

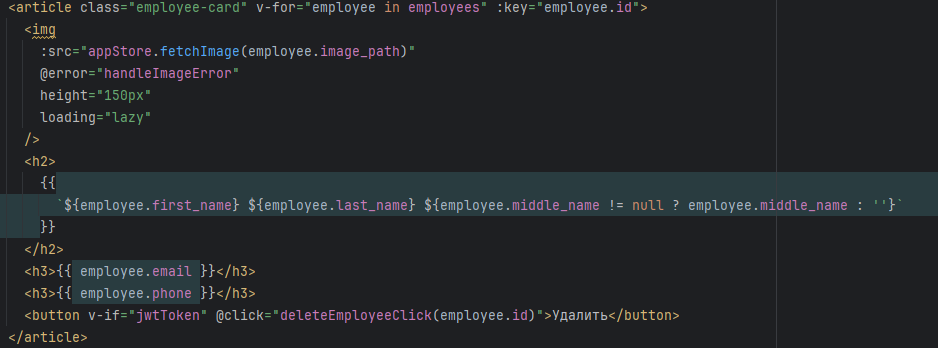


Компонент «сотрудники» (employeeList.vue)

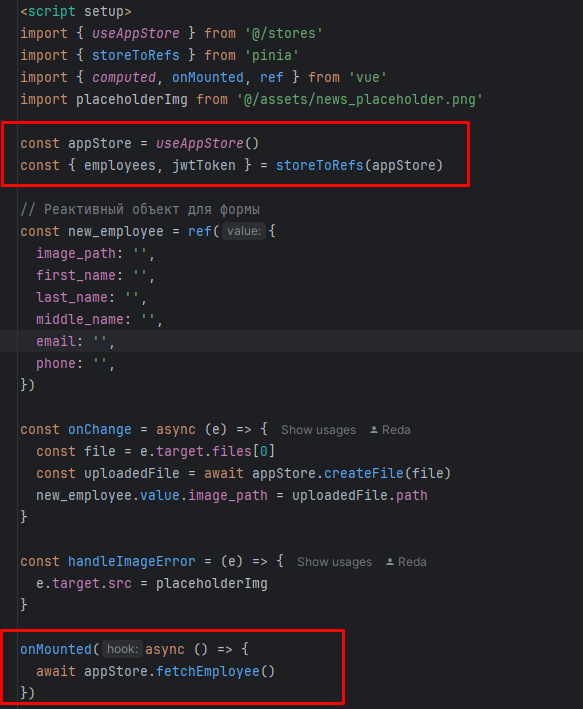
Он очень обьемный, так что обозначу основные моменты

«Карточки» сотрудников, которые генерируются из списка сотрудников, получаемого через API

Под каждой есть кнопка удалить, в случае если пользователь авторизован

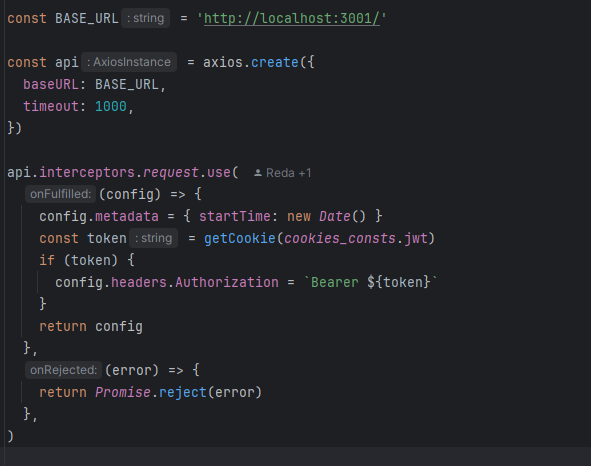


Получение сотрудников по API



Методы, использованные для этого

Создание Axios обьекта, для упрощенного обращения к API и автоматического использования Cookie



CRUD операции с сотрудниками



Компонент «Новости» (newsList.vue)



«Карточки» новостей, которые генерируются из списка новостей, получаемого через API (по тому же принципу, что и сотрудники, отличаются только роуты и названия переменных)

Под каждой есть кнопка удалить, в случае если пользователь авторизован

Форма админ панели, через которую добавляются новые новости



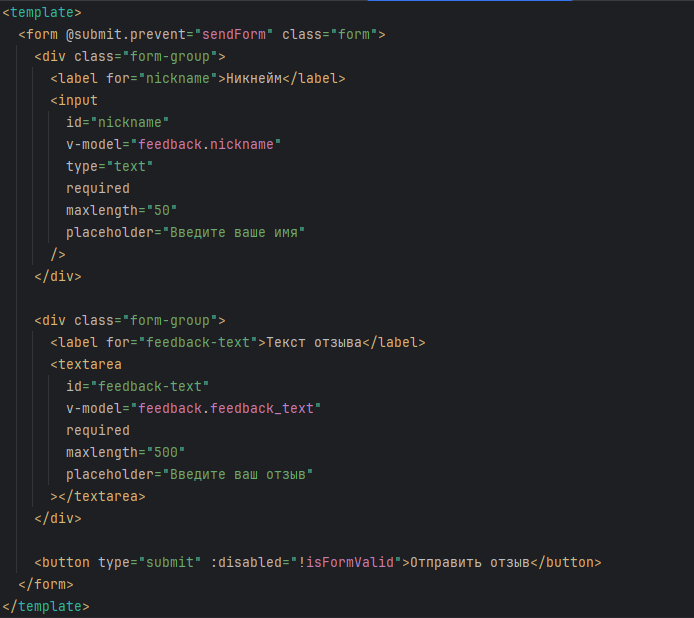
Компонент «Отзывы» (feedbacksList.vue)

Отзывы, которые генерируются из списка, получаемого через API

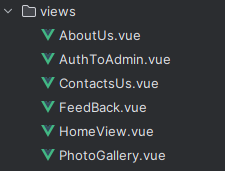
Под каждым есть кнопка удалить, в случае если пользователь авторизован



Форма добавления отзыва



Далее идут Views (фактически страницы, собранные из компонентов)



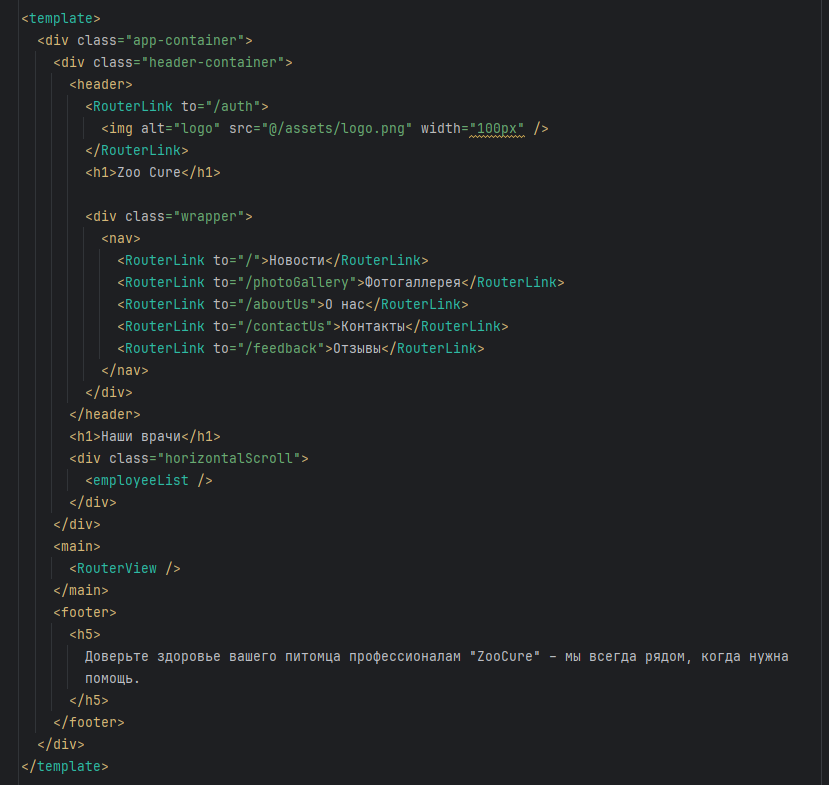
Форма авторизации в админ панель



Фотогалерея, реализованная в виде карусели, использует статичные данные



Главная страница (страница, в которой меняется контент, в ней же находится панель навигации)



Роутер, отвечающий за привязку views к определенным путям



LocalStorage сайта, или же PiniaStore, в котором определяются переменные, хранящие в себе данные, полученные через API, а так же методы получения этих данных

