Geekbrains

Специальность: Frontend-программист “Цифровые профессии”

Дипломным проект: “Создание веб-приложения – специализированной гостиницы для кошечек, позволяющего подобрать гостиничный номер по своим параметрам и забронировать его”

Ладатко Тимофей Викторович

Воронеж, 2024 год.

**Дипломным проект: “** Создание веб-приложения – специализированной гостиницы для кошечек, позволяющего подобрать гостиничный номер по своим параметрам и забронировать его**”**

**Содержание**

|  |  |
| --- | --- |
| 1. Введение | 4 |
| 1. Веб-приложение и способы обработки и сохранения данных. | 7 |
| * 1. Веб-приложение: основы | 7 |
| * 1. Frontend веб-приложения, его особенности | 9 |
| * 1. Backend веб-приложения, его особенности | 12 |
| * 1. Управление базой данных | 13 |
| 1. Программные продукты и инструменты web-разработки | 14 |
| 1. Программные продукты разработки серверной части | 17 |
| * 1. Backend | 17 |
| * 1. Базы данных | 18 |
| * 1. Docker | 20 |
| 1. Инструменты Frontend части. | 20 |
| * 1. HTML | 20 |
| * 1. CSS | 24 |
| * 1. Bootstrap | 25 |
| * 1. JavaScript и Vue.js | 27 |
| * 1. Axios | 29 |
| 1. Инструменты Backend части. | 31 |
| * 1. Spring Boot | 31 |
| * 1. Flyway | 32 |
| 1. Инструменты базы данных. | 33 |
| * 1. СУБД PostgreSQL | 33 |
| 1. Практическая разработка веб-приложения | 35 |
| * 1. Установка и запуск базы данных | 35 |
| * 1. Установка и настройка Backend | 37 |
| * 1. Установка и настройка Frontend | 40 |
| * Установка Vue CLI | 40 |
| * Установка Vuex | 41 |
| * Установка Vue-Router | 42 |
| * Навигация между маршрутами | 43 |
| * Корневой компонент Frontend | 45 |
| * Дочерние компоненты Frontend | 48 |
| 1. Сборка и развертывание проекта на сервер | 49 |
| 1. Тестирование работоспособности - UI/UX | 51 |
| 1. Подведение итогов, оценка результатов и заключение | 52 |
| 1. Список использованной литературы | 54 |

**Введение**

Суть дипломного проекта: создание веб-приложения – специализированной гостиницы для кошечек, позволяющего подобрать гостиничный номер по своим параметрам и забронировать его.

Основная цель проекта — предоставить платформу, в которой будет удобно найти и выбрать лучшие условия содержания, забронировать для своих любимых питомцев(кошечек) гостиничный номер и оставить их на передержку, в случаях, когда необходимо уехать из дома.

Проект будет включать в себя несколько ключевых компонентов, в том числе:

* Пользовательский интерфейс: пользовательский интерфейс веб-сайта должен быть разработан таким образом, чтобы обеспечить удобство для посетителей. Он будет включать функции поиска, отправки заявки на бронирование номера и другие функции, облегчающие пользователям поиск необходимой им информации;
* Серверная часть: обработка запросов от пользовательского интерфейса и сохранение данных в базу данных.
* База данных: собранные данные будут храниться в базе данных. База данных планируется к проектированию таким образом, чтобы обеспечивать эффективный поиск и извлечение информации на основе пользовательского ввода;

План разработки веб-приложения с использованием HTML, CSS, Bootstrap, JS, Vue.js, Axios, Java 17, Spring Boot, PostgreSQL, будет включать в себя:

* Разработку пользовательского интерфейса (UI) и взаимодействие с пользователем (UX) приложения с помощью HTML, CSS. Пользовательский интерфейс должен быть визуально привлекательным и простым в навигации. Bootstrap планируется использовать для ускорения процесса разработки и создания адаптивного дизайна;
* Использование Vue.js для добавления интерактивности и динамического контента в пользовательский интерфейс. Vue.js также можно использовать для создания компонентов и обработки пользовательского ввода, что сделает приложение более отзывчивым и удобным для пользователя;
* Использование Axios для отправки и получения данных на backend. Полученные данные также будут сохраняться в базу данных PostgreSQL;
* Использование Java 17, Spring Boot для создания масштабируемого и современного сервера, который обрабатывает HTTP-запросы, обрабатывает поступающие данные и сохраняет их в базу данных;
* Использование PostgreSQL, для создания, хранения базы данных и работать с данными с помощью запросов на языке SQL.
* Использование Docker, для разворачивания базы данных, а в дальнейшем всех частей приложения на сервере

Проект будет разрабатываться собственными силами. Однако, с точки зрения создания и запуска веб-приложения, в подобных проектах могут участвовать следующие разработчики:

* Веб-разработчики, обладающие знаниями в области HTML, CSS и JavaScript, могут помочь создать внешний пользовательский интерфейс и добавить интерактивность в приложение с помощью таких фреймворков, как Vue.js;
* Backend разработчики с опытом работы с Java 17, Spring Boot, PostgreSQL и управлением базами данных могут помочь создать инфраструктуру на стороне сервера, конечные точки API и подключения к базе данных;
* Инженеры данных могут помочь со сбором, очисткой и хранением данных;
* QA Инженеры могут помочь с тестированием приложения на наличие ошибок и проблем. Они могут создавать тестовые случаи и сценарии, чтобы убедиться, что приложение работает должным образом;
* UX-дизайнеры могут помочь с разработкой удобного и визуально привлекательного интерфейса для приложения. Они также могут предоставить рекомендации по передовым методам использования и доступности;
* Инженеры DevOps могут помочь с развертыванием приложения в производственной среде.

Для создания успешного проекта важно, чтобы команда разработчиков обладала разнообразным набором навыков и опыта. Сотрудничество и общение между членами команды имеют решающее значение для обеспечения того, чтобы проект соответствовал требованиям и был выполнен вовремя.

Дипломный проект по созданию веб-приложения потребует сочетания навыков программирования, знаний в области веб-разработки, backend-разработки и опыта работы с базами данных и программированием на Java. Конечный продукт станет полезным инструментом для владельцев кошек, которые хотят на некоторое время оставить своих «пушистиков», на время отсутствия.

**Веб-приложение и способы обработки и сохранения данных.**

**Веб-приложение: основы.**

Веб-приложение — это тип программного приложения, доступ к которому и использование которого осуществляется через веб-браузер. В отличие от традиционных настольных приложений, веб-приложения не нужно устанавливать на компьютер или устройство пользователя, и они предназначены для работы на любом устройстве, подключенным к Интернету.

Содержание веб-приложения может варьироваться в зависимости от его назначения и функциональности. Веб-приложение может представлять собой как простой калькулятор или список дел, так и сложное корпоративное приложение, управляющее рабочими и бизнес-процессами. Как правило, веб-приложение состоит из внешнего пользовательского интерфейса, внутреннего сервера и базы данных, в которой хранятся данные и информация.

Принципы разработки веб-приложений включают использование веб-технологий, таких как HTML, CSS и JavaScript, для создания динамического и интерактивного пользовательского интерфейса, а также использование серверных технологий, таких как Node.js, PHP или Ruby on Rails, для обработки запросов от пользователей. Веб-приложения также используют веб-протоколы, такие как HTTP и HTTPS, для передачи данных между клиентом и сервером.

Цели веб-приложений могут различаться в зависимости от потребностей пользователя. Некоторые общие цели веб-приложений включают в себя:

* Электронная коммерция: веб-приложения можно использовать для продажи товаров и услуг в Интернете, позволяя клиентам просматривать, сравнивать и покупать товары из любого места, где есть подключение к Интернету;
* Социальные сети: веб-приложения могут предоставлять службы социальных сетей, которые позволяют пользователям создавать профили, обмениваться контентом и взаимодействовать с другими пользователями;
* Продуктивность. Веб-приложения могут предоставлять инструменты для совместной работы, управления задачами и планирования проектов, позволяя командам эффективно и продуктивно работать вместе;
* Управление информацией. Веб-приложения можно использовать для сбора, хранения и анализа данных, что позволяет организациям принимать обоснованные решения и оптимизировать свою деятельность.

Применительно к дипломному проекту, приложение для поиска и бронирования специализированного гостиничного номера для кошек — это тип веб-приложения, которое позволяет пользователям найти по своим параметрам уютное место для своего питомца на случай поездки, узнать о наличии доступных номеров и уточнить правила проживания с животными и забронировать номер. Суть веб-приложения будет включать в себя функцию поиска гостиничного номера, а также базу данных с информацией о номерах, условиях нахождения в них питомцев и бронирования.

С точки зрения принципов разработки, веб-приложение должно быть разработано с учетом масштабируемости и удобства обслуживания, учитывая, что оно будет извлекать данные из нескольких внешних источников.

Цель веб-приложения состоит в том, чтобы предоставить пользователям удобный способ поиска и бронирования гостиничных номеров для своих кошек.

Гостиница для домашних животных — это место, где их хозяева могут на некоторое время оставить своих «пушистиков», не переживая на предмет того, что они будут испытывать дискомфорт во время разлуки. Услуга приобрела большую популярность. Люди настоящего не желают сидеть дома, но при этом далеко не всегда присутствует возможность исключить расставание с «четвероногим другом».

**Frontend веб-приложения, его особенности.**

Внешний (frontend) интерфейс веб-приложения будет состоять из пользовательского интерфейса, отображающего информацию о сервисе, информацию о номерах, условиях нахождения в них домашних питомцев и возможности бронирования на определенный срок.

Внешний интерфейс веб-приложения относится к той части приложения, которая видна пользователям и взаимодействует с ними напрямую. Обычно он состоит из пользовательского интерфейса, который представлен в веб-браузере и позволяет пользователям взаимодействовать с приложением.

В случае веб-приложения для бронирования гостиничных номеров для кошек внешний интерфейс может состоять из следующих компонентов:

* Пользовательский интерфейс: пользовательский интерфейс будет визуальным представлением веб-приложения и будет включать такие элементы, как главную страницу с общей информацией, страницу с номерами, страницу отзывов и страницу с более детальной информацией о конкретном номере;
* HTML — это язык разметки, используемый для структурирования содержимого веб-страниц. HTML используется для определения структуры пользовательского интерфейса, включая заголовки, абзацы, списки и таблицы;
* CSS — это язык стилей, который используется для управления внешним видом веб-страниц. CSS используется для определения визуальных свойств пользовательского интерфейса, таких как цвет и шрифт текста, расположение элементов и общий стиль веб-приложения;
* Bootstrap — это популярный интерфейсный фреймворк, который предоставляет предварительно разработанные компоненты и стили для создания адаптивных мобильных веб-приложений. Bootstrap используется для ускорения разработки и обеспечения единообразного внешнего вида веб-приложения;
* Vue.js — это среда JavaScript для создания пользовательских интерфейсов и одностраничных (SPA) приложений. Vue используется для создания динамических и отзывчивых пользовательских интерфейсов, которые обновляются в режиме реального времени на основе пользовательского ввода;
* Axios — библиотека JavaScript для выполнения HTTP-запросов со стороны клиента веб-приложения. Используется для получения данных с сервера и отображения их пользователю в режиме реального времени.

Frontend должен иметь понятный и простой в использовании интерфейс, который позволяет пользователям просматривать информацию о доступных номерах, информацию об предоставляемых дополнительных услугах и бронировать номер на определенный срок. Пользовательский интерфейс должен быть разработан таким образом, чтобы он реагировал и адаптировался к различным размерам экрана и устройствам.

Интерфейс должен отображать обновления в реальном времени для пользователей, когда данные из Backend изменяются, например, когда добавляется информация о новоv номере или услуге, или когда меняются цены на книги.

Пользовательский интерфейс должен быть разработан с учетом высокой производительности и быстрой загрузкой страниц, а также минимальной задержкой.

Внешний интерфейс должен быть совместим с различными веб-браузерами и операционными системами, плюс предназначен для работы на мобильных устройствах, а также на настольных компьютерах.

Пользовательский интерфейс должен быть разработан с учетом безопасности, чтобы защитить личную информацию пользователей за счет использования безопасных протоколов, таких как HTTPS.

**Backend веб-приложения, его особенности.**

Серверная часть веб-приложения будет состоять из сервера отдельного приложения, написанного на языке Java с использованием Spring Framework, который обрабатывает запросы пользователей и сохраняет информацию о бронировании в базу данных.

Веб-приложению также потребуется база данных для хранения информации о бронировании номеров, а в дальнейшем для любой информации, например информации о пользователях, а также отзывов пользователей.

Серверная часть будет состоять из следующих компонентов:

* Java — это многоплатформенный, объектно-ориентированный язык программирования, работающий на миллиардах устройств по всему миру. На нем работают приложения, операционные системы смартфонов, корпоративное программное обеспечение и многие известные программы. Несмотря на то, что Java был изобретен более 20 лет назад, в настоящее время он является самым популярным языком программирования среди разработчиков приложений.
* Spring Boot — это популярный фреймворк для создания веб-приложений с использованием Java. Это часть фреймворка Spring, которая представляет собой набор инструментов и библиотек для создания приложений корпоративного уровня. Spring Boot упрощает создание автономных приложений производственного уровня, которые можно легко развернуть и запустить с минимальной конфигурацией.
* Flyway — это фреймворк, который обновляет версии баз данных с помощью миграций. Миграции можно писать на SQL (с синтаксисом, специфичным для конкретной СУБД) или на Java. Миграции могут быть версионными или повторяющимися. Первые имеют уникальную версию и применяются ровно один раз. У вторых номера версии нет, и они применяются, когда у них изменяется контрольная сумма. Повторяющиеся миграции в рамках одного запуска всегда применяются после выполнения версионных миграций.

Серверной части необходимо будет обрабатывать данные, полученные с frontend части веб-приложения.

Backend должен управлять базой данных для хранения собранных данных. База данных должна быть спроектирована с учетом высокой производительности и масштабируемости для обработки больших объемов данных.

Backend должен предоставить API, который может использоваться внешним интерфейсом для доступа к собранным данным.

Серверная часть в идеале должна быть спроектирована с учетом масштабируемости, с возможностью обработки большого количества запросов от внешнего интерфейса и обработки больших объемов данных.

**Управление базой данных**

Для управления базой данных будет использоваться PostgreSQL.

PostgreSQL — это объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире. Имеет открытый исходный код и является альтернативой коммерческим базам данных.

**Программные продукты и инструменты web-разработки**

Любая разработка начинается с выбора среды разработки. Интегрированные среды разработки (IDE) — это программные приложения, которые обеспечивают комплексную среду для разработки, тестирования и развертывания программного обеспечения. IDE обычно включают в себя редактор кода, компилятор или интерпретатор и инструменты отладки, а также другие функции, такие как интеграция контроля версий и инструменты управления проектами.

Разработчики часто предпочитают IDE, потому что они обеспечивают универсальное решение для всех аспектов процесса разработки. Разработчики могут писать и редактировать код, компилировать и запускать код, а также устранять ошибки в одной и той же среде. IDE также часто предоставляют дополнительные функции, такие как завершение кода, подсветка синтаксиса и инструменты отладки, которые могут ускорить и повысить эффективность разработки.

Однако некоторые разработчики предпочитают использовать текстовые редакторы и инструменты командной строки, поскольку они обеспечивают большую гибкость и могут быть настроены в соответствии с конкретными потребностями разработки. В конечном счете, выбор между использованием IDE или текстового редактора зависит от личных предпочтений и требований проекта разработки.

Наиболее популярные примеры IDE приведены далее.

Visual Studio — это интегрированная среда разработки, разработанная Microsoft, поддерживает широкий спектр языков программирования.

Eclipse — IDE с открытым исходным кодом, поддерживающая широкий спектр языков программирования, включая Java, C++ и Python.

Xcode — разработанная Apple, эта IDE используется для разработки приложений для macOS, iOS, watchOS и tvOS.

IntelliJ IDEA — разработанная JetBrains, эта IDE обеспечивает поддержку Java, Kotlin и других языков.

PyCharm — также разработанная JetBrains, эта IDE ориентирована на разработку Python.

Frontend часть дипломного проекта будет разрабатываться в VS Code.

Visual Studio Code (VS Code) — это бесплатный редактор кода с открытым исходным кодом, разработанный Microsoft. Он широко используется разработчиками для различных задач программирования, включая веб-разработку и машинное обучение. Он доступен для операционных систем Windows, macOS и Linux.

Одной из главных особенностей VS Code является обширный набор расширений и плагинов, которые позволяют разработчикам настраивать редактор под свои конкретные нужды. VS Code поддерживает широкий спектр языков программирования, и доступны расширения для всего: от подсветки синтаксиса и форматирования кода до отладки и интеграции с Git.

Некоторые из ключевых особенностей VS Code включают в себя:

* IntelliSense: эта функция предоставляет интеллектуальные предложения по завершению кода по мере ввода в зависимости от языка, с которым вы работаете.
* Встроенная поддержка Git: VS Code включает встроенную поддержку Git, что позволяет легко управлять репозиториями кода, не выходя из редактора.
* Поддержка отладки: VS Code предоставляет встроенный отладчик для ряда языков программирования, позволяющий легко выполнять код, выявлять и исправлять ошибки.
* Запуск задач: VS Code поддерживает создание пользовательских задач, которые могут автоматизировать общие рабочие процессы, такие как создание и тестирование кода.
* Встроенный терминал: VS Code включает встроенное окно терминала, позволяющее выполнять команды оболочки непосредственно в редакторе.
* Настройка темы: VS Code включает в себя ряд встроенных цветовых тем, и вы также можете загружать и устанавливать пользовательские темы, чтобы настроить внешний вид редактора.
* Поддержка нескольких языков: VS Code поддерживает широкий спектр языков программирования, включая JavaScript, Python, Ruby и многие другие.

VS Code — очень гибкий и настраиваемый редактор кода, предоставляющий ряд мощных функций для разработчиков. Он имеет большое и активное сообщество пользователей и разработчиков, а в Интернете доступно множество ресурсов, которые помогут начать работу и научиться эффективно его использовать.

**Программные продукты разработки серверной части**

**Backend**

Backtend часть дипломного проекта будет разрабатываться в IntelliJ IDEA.

IntelliJ IDEA — это IDE, интегрированная среда разработки (комплекс программных средств, который используется для написания, исполнения, отладки и оптимизации кода) для Java, JavaScript, Python и других языков программирования от компании JetBrains. Отличается обширным набором инструментов для рефакторинга (перепроектирования) и оптимизации кода.

IDE предоставляет интеллектуальную помощь во время написания кода:

* выполняет глубокий анализ и создает виртуальную карту проекта;
* обнаруживает ошибки и предлагает варианты исправления;
* автоматически дополняет код, учитывая контекст;
* проводит валидацию (проверку на соответствие стандартам) кода;
* выполняет рефакторинг кода — делает его проще и понятнее;
* поддерживает работу со вставками, написанными на других языках программирования;
* позволяет использовать шаблоны для вставки повторяющихся фрагментов кода;
* предлагает оптимизацию с помощью профилировщика — инструмента, который анализирует производительность кода и оценивает нагрузку на процессор и оперативную память.

Для компиляции готовых проектов в IntelliJ IDEA реализована интеграция со всеми популярными системами автоматической сборки приложений — Maven, Gradle, Ant и Gant.

**Базы данных**

Для взаимодействия с базой данных можно также использовать IntelliJ IDEA, в комплект поставки которой входит плагин SQL.

Существуют также различные GUI PostgreSQL, приведу самые распространенные:

* **pgAdmin** – кроссплатформенный графический инструмент с открытым исходным кодом.

**Плюсы:**

* + совместим с Linux, Windows, macOS;
  + позволяет работать с несколькими серверами одновременно;
  + экспорт в CSV;
  + планирование запросов;
  + возможность отслеживать ваши сеансы, блокировки БД с помощью панели мониторинга;
  + ярлыки в редакторе SQL для более удобной работы;
  + встроенный отладчик процедурного языка;
  + тщательная документация и активное сообщество.

**Минусы:**

* + медленный и не всегда интуитивно понятный пользовательский интерфейс по сравнению с платными конкурентами;
  + тяжелый;
  + высокий порог вхождения;
  + для работы с несколькими базами данных одновременно потребуются продвинутые навыки.
* **DBeaver** – инструмент управления PostgreSQL с открытым исходным кодом, поддерживающий коннект к нескольким базам данных.

**Плюсы:**

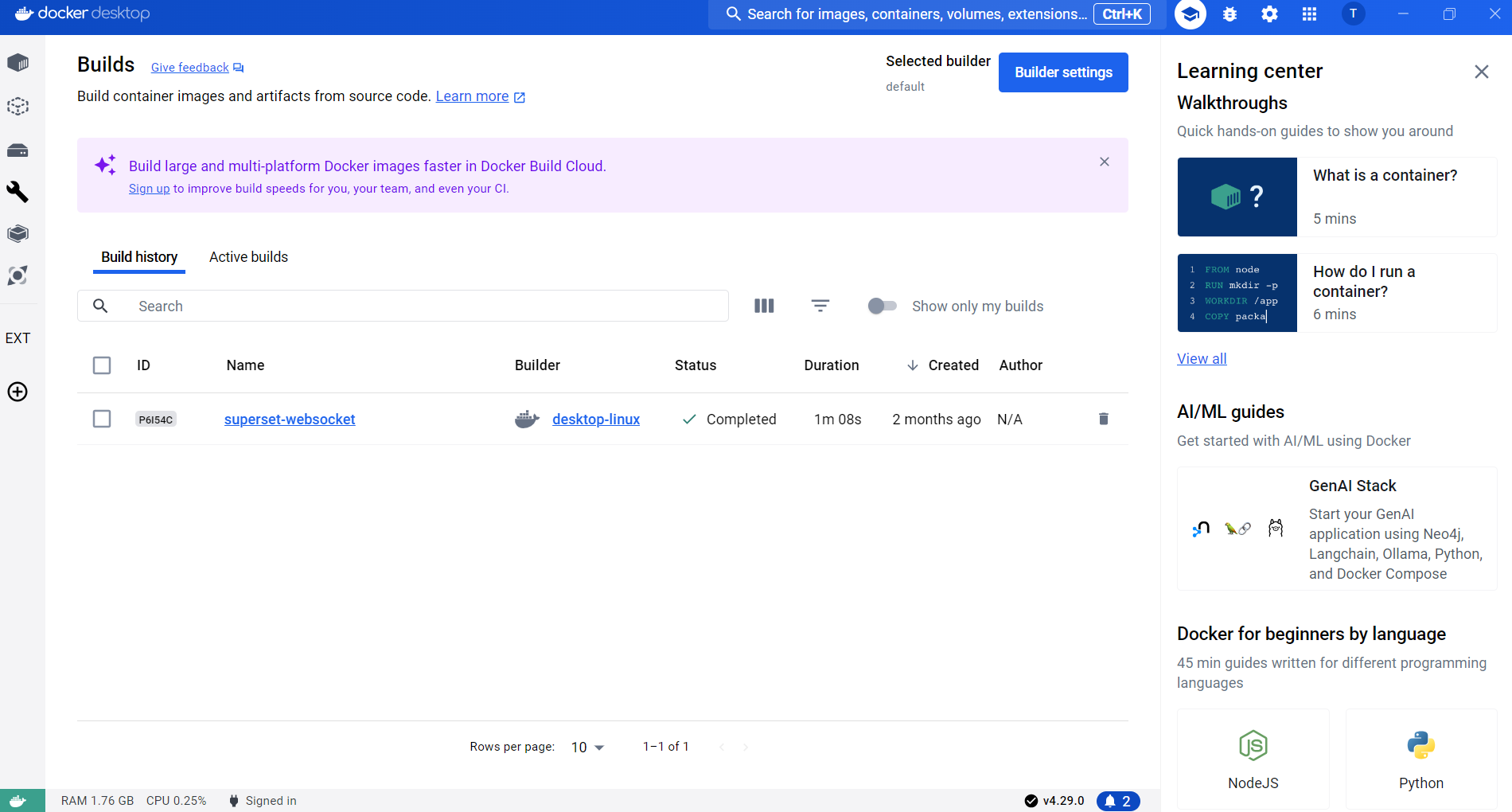
* + кроссплатформенность;
  + поддержка более 80 баз данных;
  + визуальный конструктор, позволяющий добавлять запросы без навыков работы с SQL;
  + несколько представлений данных;
  + импорт/экспорт данных в CSV, HTML, XML, JSON, XLS, XLSX;
  + повышенная безопасность данных;
  + полнотекстовый поиск данных и возможность отображения результатов в виде таблиц/представлений;
  + доступен бесплатный тарифный план.

**Минусы:**

* + низкая производительность по сравнению с конкурентами;
  + слишком частые обновления, что раздражает;
  + после некоторого времени бездействия DBeaver отключается от базы данных

**Docker**

Для взаимодействия с Docker используется Docker Desktop для Windows.



**Инструменты frontend части.**

**HTML**

HTML (HyperText Markup Language) является основным языком разметки веб-страниц. Он используется для создания содержимого веб-страниц, которое отображается в браузере.

HTML является стандартизированным языком, который определяет структуру веб-страницы с помощью различных элементов и атрибутов. Каждый элемент представляет определенный тип содержимого, такой как заголовки, параграфы, списки, изображения и т.д.

HTML также используется для создания ссылок на другие веб-страницы, внутренние и внешние. Он позволяет создавать формы, которые пользователи могут заполнять и отправлять, а также добавлять мультимедийный контент, такой как видео и аудио.

HTML является важным инструментом для создания веб-страниц и веб-приложений. Он обеспечивает структуру и содержимое веб-страниц, которые пользователи видят в браузере. Без HTML веб-страницы были бы просто набором несвязанных текстов и изображений.

HTML использует теги (tags) для определения элементов страницы. Теги создаются с использованием угловых скобок <>, и обычно имеют начальный и конечный тег, где конечный тег отличается от начального символом "/".

Внутри тегов могут находиться атрибуты (attributes), которые определяют дополнительные характеристики элементов, такие как их цвет, размер, ссылки на другие страницы и т.д.

Некоторые наиболее часто используемые теги в HTML:

* <html>: Начальный тег для HTML-документа.
* <head>: Содержит метаданные документа, такие как заголовки страницы, ссылки на стили CSS, и другие данные, которые не отображаются на странице.
* <title>: Определяет заголовок документа.
* <body>: Определяет содержимое страницы, которое отображается в браузере.
* <div>: Определяет блок элементов на странице.
* <p>: Определяет абзац текста.
* <img>: Определяет изображение на странице.
* <a>: Определяет ссылку на другую страницу или ресурс.
* <form>: Определяет форму на странице, которую пользователь может заполнить и отправить.

В современной верстке для распределения элементов на странице чаще всего используются теги <div> и <section>.

<div> - это тег в HTML, который используется для определения контейнеров на веб-страницах. Он создает блоки, в которые можно помещать другие элементы, такие как текст, изображения, видео, таблицы и т.д. <div> не предназначен для определения семантики содержимого, а скорее служит для оформления и стилизации. Он не влияет на структуру документа, но может использоваться для группировки и организации элементов страницы.

Пример использования тега <div>:

|  |
| --- |
| <div>  <h1>Заголовок</h1>  <p>Это простой текст на странице.</p> </div> |

Здесь создается блок, внутри которого находится заголовок первого уровня и абзац текста. Контейнер <div> позволяет объединить эти элементы, чтобы упростить их оформление и стилизацию. <div> можно также использовать для разделения элементов страницы на группы, например для создания секций или блоков с футером и хедером. В сочетании с CSS стилями, <div> может быть мощным инструментом для создания разнообразных макетов и оформления веб-страниц.

<section> — это тег в HTML, который используется для определения разделов на веб-страницах. Он обычно содержит группы связанных элементов, таких как заголовки, параграфы, изображения и другие элементы. <section> обычно используется для структурирования документа и улучшения доступности и семантической структуры.

Пример использования тега <section>:

|  |
| --- |
| <section>  <h2>Название раздела</h2>  <p>Это текст в разделе.</p>  <img src="image.jpg" alt="Изображение"> </section> |

Здесь создается раздел страницы, содержащий заголовок второго уровня, абзац текста и изображение. Контейнер <section> позволяет группировать эти элементы, чтобы указать, что они относятся к одному разделу страницы.

<section> обычно используется в сочетании с другими семантическими тегами, такими как <article>, <header>, <footer> и <nav>, чтобы создать структурированную и понятную семантическую разметку веб-страницы. <section> также может использоваться для стилизации страницы с помощью CSS. Он позволяет определять общие стили для групп связанных элементов и упрощает оформление веб-страницы.

Понимание тегов является необходимым для создания и правильного размещения элементов на странице. Например, правильное использование тегов позволяет браузеру корректно интерпретировать содержимое страницы и отображать его правильно. Также это может повлиять на поисковую оптимизацию (SEO), что важно для любого веб-сайта.

**CSS**

CSS (Cascading Style Sheets) - это язык стилей, который используется для оформления веб-страниц. Он определяет, как должны выглядеть элементы HTML на странице, включая их цвет, размер, расположение и другие характеристики.

CSS позволяет разработчикам создавать стильные и привлекательные веб-страницы, которые могут быть адаптированы под различные устройства и размеры экранов.

Основные принципы CSS:

* Каскадность - стили могут быть определены на разных уровнях, и значения, определенные на более высоком уровне, могут быть переопределены на более низком уровне.
* Наследование - стили, определенные для родительского элемента, могут быть унаследованы его дочерними элементами.
* Приоритетность - стили могут иметь разную приоритетность, и значение, определенное для элемента с более высокой приоритетностью, будет использоваться вместо значения элемента с более низкой приоритетностью.

Пример CSS:

|  |
| --- |
| body {  background-color: #f0f0f0;  font-family: Arial, sans-serif; } h1 {  color: #333;  font-size: 28px;  text-align: center; } p {  color: #666;  font-size: 16px;  line-height: 1.5; } |

В примере выше определены стили для элементов <body>, <h1> и <p>. <body> задает фоновый цвет и шрифт для всей страницы, <h1> задает цвет, размер и выравнивание для заголовков первого уровня, а <p> задает цвет, размер и высоту строки для параграфов.

CSS может быть определен внутри HTML-документа с использованием тега <style>, или может быть вынесен в отдельный файл .css для повторного использования на других страницах. С помощью CSS можно создавать сложные макеты страниц, анимации и интерактивные элементы, чтобы улучшить пользовательский опыт на веб-сайте.

**Bootstrap**

Bootstrap — это бесплатный фреймворк для разработки веб-сайтов, который содержит набор готовых компонентов, шаблонов и стилей на основе CSS и JavaScript. Он был разработан компанией Twitter и впервые был выпущен в 2011 году.

Основные преимущества Bootstrap:

* Гибкость и масштабируемость - Bootstrap предлагает широкий выбор компонентов и стилей, которые можно использовать для быстрой разработки веб-сайтов. Он также поддерживает адаптивный дизайн, что позволяет создавать сайты, которые хорошо отображаются на любом устройстве, включая смартфоны и планшеты.
* Кросс-браузерность - Bootstrap обеспечивает одинаковый внешний вид веб-сайта в различных браузерах, что значительно упрощает разработку и тестирование сайта.
* Отзывчивый дизайн - Bootstrap обеспечивает адаптивный дизайн, что позволяет сайту подстраиваться под различные размеры экранов. Это делает веб-сайт более доступным для пользователей на мобильных устройствах и улучшает пользовательский опыт.
* Широкое сообщество - Bootstrap имеет большое сообщество пользователей, которые создают и обмениваются компонентами, шаблонами и советами, что позволяет быстро решать проблемы и находить решения для разных задач.

Bootstrap включает в себя готовые компоненты, такие как меню навигации, кнопки, формы, карусели, модальные окна и многие другие, которые можно использовать для создания веб-сайтов. Он также содержит набор классов, которые можно использовать для быстрого и простого применения стилей к элементам на странице.

Bootstrap можно использовать как самостоятельный фреймворк или в сочетании с другими технологиями, такими как HTML, CSS, JavaScript, jQuery и другими. Он также поддерживает различные плагины и инструменты, такие как Sass и Less, которые облегчают разработку и обслуживание веб-сайтов.

**JavaScript и Vue.js**

JavaScript — это язык программирования, обычно используемый для создания динамических веб-страниц и веб-приложений. Это язык высокого уровня, поддерживающий объектно-ориентированный, императивный и функциональный стили программирования. JavaScript можно использовать для различных задач, таких как добавление интерактивных функций на веб-страницы, создание браузерных игр, создание веб-приложений и мобильных приложений, а также разработка серверных приложений.

Некоторые из ключевых особенностей JavaScript включают в себя:

* Программирование, управляемое событиями: JavaScript позволяет разработчикам создавать интерактивные веб-страницы, которые реагируют на действия пользователя, такие как нажатие кнопки, прокрутка страницы или ввод данных в форму.
* Асинхронное программирование: JavaScript поддерживает асинхронное программирование, что позволяет разработчикам писать код, который не блокирует выполнение другого кода.
* Кроссплатформенная совместимость: JavaScript поддерживается всеми основными веб-браузерами, что делает его популярным выбором для веб-разработки.
* Расширяемость: JavaScript можно расширить с помощью библиотек и фреймворков, таких как jQuery, React и Vue.js, которые упрощают разработку сложных веб-приложений.

Vue.js — это прогрессивный фреймворк JavaScript, который используется для создания пользовательских интерфейсов и одностраничных приложений (SPA). Он представляет собой модульный фреймворк, который может использоваться вместе с другими библиотеками и инструментами, такими как Vuex, Vue Router, Vuetify и другие.

Основные преимущества Vue.js:

* Легковесность - Vue.js имеет небольшой размер и быстро загружается на страницу, что позволяет создавать быстрые и отзывчивые пользовательские интерфейсы.
* Гибкость - Vue.js можно использовать как для маленьких, простых приложений, так и для крупных и сложных проектов. Он также предоставляет широкий выбор опций и настроек, которые позволяют настроить его под различные потребности.
* Простота в использовании - Vue.js легко устанавливается и используется даже начинающими разработчиками. Он также предоставляет простые и интуитивно понятные API и документацию.
* Однофайловые компоненты - Vue.js поддерживает однофайловые компоненты, которые содержат в себе все необходимые HTML, CSS и JavaScript коды, что облегчает разработку и обслуживание приложений.

Vue.js предоставляет набор компонентов и директив, которые можно использовать для создания пользовательского интерфейса, а также API для управления состоянием приложения. Он также поддерживает использование шаблонов, фильтров и многих других функций, которые упрощают разработку.

Vue.js можно использовать как самостоятельный фреймворк, так и в сочетании с другими технологиями, такими как HTML, CSS, JavaScript, Bootstrap и другие. Он также может быть использован с различными инструментами и плагинами, такими как Vuex, Vue Router, Vuetify и другие, что позволяет создавать более сложные приложения и улучшать производительность.

**Axios**

Axios — это популярная библиотека JavaScript, используемая для выполнения HTTP-запросов из веб-приложения. Он основан на интерфейсе XMLHttpRequest, встроенном в веб-браузеры, и предоставляет простой и удобный в использовании API для отправки и получения данных с сервера. Axios широко используется в современных веб-приложениях, особенно в приложениях, созданных с помощью Vue.js, React и других интерфейсных фреймворков.

Axios предоставляет ряд функций, которые делают его популярным выбором для HTTP-запросов, в том числе:

* API на основе промисов: Axios построен на основе промисов JavaScript, что позволяет разработчикам писать асинхронный код, который легко читать и понимать.
* Простой API: Axios предоставляет простой и интуитивно понятный API для выполнения HTTP-запросов с методами обработки запросов GET, POST, PUT и DELETE.
* Перехватчики (Interceptors): Axios предоставляет функцию, называемую перехватчиками, которая позволяет разработчикам перехватывать и изменять HTTP-запросы и ответы до их отправки или получения.
* Обработка ошибок (Error handling): Axios предоставляет мощную систему обработки ошибок, которая упрощает обработку сетевых ошибок и ошибок на стороне сервера.
* Кроссплатформенная совместимость: Axios совместим как со средой браузера, так и со средой Node.js, что делает его популярным выбором для создания клиентских и серверных веб-приложений.

Axios можно использовать для различных задач, таких как получение данных с сервера, загрузка файлов и обработка аутентификации пользователей.

**Инструменты backend части.**

**Spring Boot**

Spring Boot — это фреймворк на базе Java, который предоставляет более простой и быстрый способ создания приложения Spring Boot производственного уровня.

Он предлагает широкий спектр готовых функций, таких как автоконфигурация, что упрощает настройку вашего приложения Spring Boot.

Spring Boot также предоставляет надежные функции безопасности, что делает его популярным выбором для корпоративных приложений.

Spring Boot — это фреймворк на базе Java, который используется для создания приложений корпоративного уровня. Он обеспечивает более быстрый и простой способ разработки приложений производственного уровня.

Он предлагает широкий спектр готовых функций, таких как автоконфигурация, что упрощает настройку вашего приложения.

Другие широко известные варианты использования Spring Boot включают:

СОЗДАНИЕ МИКРОСЕРВИСОВ

Spring Boot обычно используется для создания микросервисов и облачных приложений, поскольку поддерживает контейнеризацию и облачные платформы.

СОЗДАНИЕ ВЕБ-ПРИЛОЖЕНИЙ

Его также можно использовать для создания пользовательских веб-приложений и RESTful API, поскольку он предоставляет широкий спектр готовых библиотек и инструментов для работы с базами данных, обеспечения безопасности и других распространенных задач веб-разработки.

СОЗДАНИЕ ПАКЕТНЫХ ПРИЛОЖЕНИЙ

Spring Boot хорошо подходит для создания приложений пакетной обработки, таких как задания ETL и конвейеры обработки данных.

**Flyway**

Flyway - популярная платформа миграции баз данных с открытым исходным кодом для Java. Она привносит структуру и уверенность в эволюцию вашей схемы базы данных. Она действительно проста в использовании, но при этом мощна и удобна как для разработчиков, так и для администраторов баз данных. Flyway поддерживает наиболее распространенные в отрасли базы данных, включая Oracle, SQL Server, DB2, MySQL, PostgreSQL, HsqlDB, H2 и Derby. Миграции могут быть написаны на простом старом SQL или Java и выполняться через API, плагин Maven, Ant-задачи или инструмент командной строки.

Flyway разработан и построен на идее, что добавление миграций баз данных в ваш проект должно быть максимально простым. Его размер намеренно невелик. Она полностью поддерживает соглашение о настройке, а ее API состоит всего из шести основных методов: clean, init, migrate, validate, info и repair.

Использование инструмента для обработки изменений в вашей базе данных требует доверия. Flyway прилагает все усилия для достижения и поддержания этого.

Все миграции выполняются в рамках транзакций. В базах данных с поддержкой транзакций DDL (PostgreSQL, DB2, SQL Server и Derby) Flyway автоматически откатывает всю миграцию при возникновении сбоя и сообщает об ошибке. Во всех других базах данных миграция будет помечена как неудачная в таблице метаданных и будет выдано сообщение об ошибке. Затем вам нужно будет вручную отменить изменения и запустить команду восстановления Flyway, чтобы исправить таблицу метаданных. Это соответствует правилу быстрого сбоя. Ошибки выявляются как можно быстрее, чтобы избежать их незамеченного проникновения в вашу систему.

**Инструменты базы данных.**

**СУБД PostgreSQL**

**PostgreSQL** распространяется на условиях лицензии PostgreSQL, которая является бесплатной и открытой. Это означает, что пользователи могут использовать PostgreSQL для любых целей, без необходимости платить за использование. В отличие от других СУБД, например, Oracle или Microsoft SQL Server, PostgreSQL предоставляет все функции бесплатно.

**PostgreSQL** имеет простой и интуитивно понятный интерфейс, который позволяет пользователям управлять базой данных без особых сложностей. Более того, PostgreSQL является гибкой СУБД, что означает, что она может быть легко настроена для любых нужд пользователя.

**PostgreSQL** является одной из самых надежных СУБД. Это достигается за счет использования транзакций, которые гарантируют целостность данных даже в случае сбоев в системе.

Кроме того, PostgreSQL имеет множество встроенных механизмов безопасности, включая поддержку SSL/TLS, аутентификацию Kerberos и GSSAPI, а также механизмы защиты от SQL-инъекций. PostgreSQL также позволяет настраивать права доступа к базам данных и таблицам, что обеспечивает еще большую безопасность данных.

**PostgreSQL** обладает высокой производительностью даже при обработке больших объёмов данных. Благодаря оптимизированной архитектуре, PostgreSQL может обрабатывать тысячи запросов в секунду, а также поддерживает широкий спектр функциональности, включая поддержку транзакций, многоверсионность и расширяемость. Это делает PostgreSQL отличным выбором для крупных и сложных проектов, где высокая производительность является ключевым фактором успеха.

**PostgreSQL** имеет множество дополнительных модулей и расширений, которые позволяют пользователям увеличить её функциональность. Например, с помощью PostgreSQL можно создавать полнотекстовый поиск, геопространственный поиск и многое другое.

**PostgreSQL** — это СУБД с открытым исходным кодом. Это означает, что любой желающий может просмотреть исходный код PostgreSQL и внести свои изменения.

**Практическая разработка веб-приложения**

**Установка и запуск базы данных**

PostgreSQL подходит для работы в любой операционной системе: Linux, macOS, Windows. Пользователь получает систему «из коробки» — чтобы установить и использовать программу, не нужны дополнительные инструменты.

Установим PostgreSQL с помощью Docker. Обычно мы запускаем контейнер Docker, используя публичный образ Docker, или берём предварительно настроенные Docker-образы сервера баз данных PostgreSQL из Docker Hub.

Сначала запустим контейнер Docker с базой данных PostgreSQL, используя публичный образ PostgreSQL. Позже мы создадим пользовательский Dockerfile для установки сервера PostgreSQL в контейнер Docker. Также мы научимся создавать резервные копии и восстанавливать базу данных с помощью контейнера Docker.

Для создания контейнера Docker необходимо чтобы был установлен докер на сервере. Все сервера работают на операционной системе Linux и на них установлен docker и docker-compose.

Для различных операционных систем установка отличается:

Для Windows необходимо установить Docker Desktop для Windows.

Для Linux-Ubuntu необходимо выполнить следующие команды:

* sudo apt update
* sudo apt install docker-ce -y
* sudo apt-get install docker-compose

База данных устанавливается в Docker с помощью файла docker-compose.yml, который лежит в корне Backend проекта и имеет структуру:

postgres:  
 container\_name: db-cat-hotel  
 image: postgres:15-alpine  
 environment:  
 POSTGRES\_DB: "cat-hotel"  
 POSTGRES\_USER: "postgres"  
 POSTGRES\_PASSWORD: "1q2w3e4r"  
 ports:  
 - "5432:5432"

Для разворачивания базы данных и запуска в контейнере Docker необходимо запустить команду:

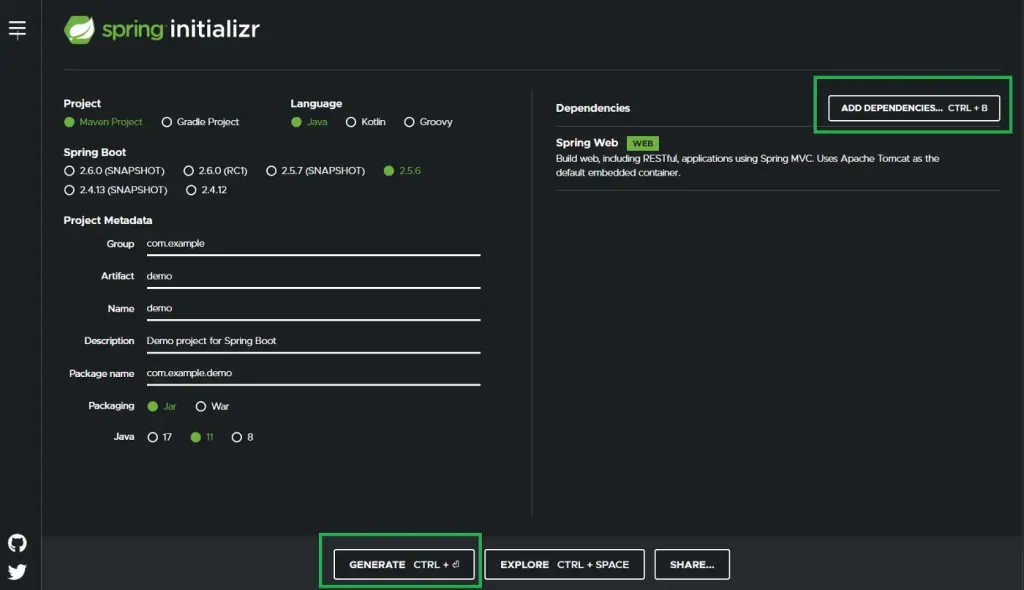
docker-compose up -d

**Установка и настройка backend:**

Создать проект на Spring Boot:

Существует несколько способов создания проекта на Spring Boot, но наиболее распространённым методом является использование веб-сайта Spring Initializer (<https://start.spring.io/>) или инструмент Spring CLI (интерфейс командной строки).

Использование веб-сайта Spring Initializer:



* Перейдите на веб-сайт Spring Initializer ([https://start.spring.io /)](https://start.spring.io/).
* Выберите параметры, которые для своего проекта, такие как тип проекта, язык и версия Spring Boot.
* Выберите зависимости, которые вы хотите включить в свой проект.
* Нажмите на кнопку Generate, чтобы загрузить ZIP-файл, содержащий файлы проекта.
* Извлеките содержимое ZIP-файла в каталог на вашем компьютере.
* Импортируйте проект в предпочитаемую вами IDE (интегрированную среду разработки) и приступайте к разработке своего приложения.

Для работы с установленной базой данных необходимо в файле application.yaml, который лежит в папке resources, внести следующие настройки:

spring:  
 datasource:  
 url: ${DB\_URL}  
 username: ${DB\_USERNAME}  
 password: ${DB\_PASSWORD}  
 driver-class-name: org.postgresql.Driver  
 hikari:  
 pool-name: HikariPool  
 maximum-pool-size: ${DB\_POOL\_SIZE:10}  
 max-lifetime: ${DB\_CONNECTION\_MAX\_LIFETIME:580000}  
 connection-timeout: ${DB\_CONNECTION\_TIMEOUT:20000}  
 jpa:  
 show-sql: ${JPA\_SHOW\_SQL:false}  
 generate-ddl: true  
 database: *postgresql* properties:  
 javax:  
 persistence:  
 sharedCache:  
 mode: ENABLE\_SELECTIVE  
 hibernate:  
 ddl-auto: update  
 jdbc:  
 batch\_size: 10  
 order\_inserts: true  
 order\_updates: true  
 batch\_fetch\_style: PADDED  
 query:  
 in\_clause\_parameter\_padding: true  
 fail\_on\_pagination\_over\_collection\_fetch: true  
  
 format\_sql: ${HIBERNATE\_FORMAT\_SQL:false}  
 generate\_statistics: ${HIBERNATE\_GENERATE\_STATISTICS:false}  
 open-in-view: false

а в файле application-local.yaml, который предназначен для настроек локального профиля, внести запись:

DB\_URL: jdbc:postgresql://localhost:5432/cat-hotel  
DB\_USERNAME: postgres  
DB\_PASSWORD: 1q2w3e4r

Данные файлы используются для конфигурации Spring Boot приложения.

Для запуска приложения в Windows, в консоли IDEA из корневой папки необходимо запустить команду:

mvn spring-boot:run -D"spring-boot.run.profiles"=local

**Установка и настройка frontend**

Vue.js был выбран за его компонентный подход к созданию пользовательского интерфейса, использование директив и реактивных свойств для управления динамическим контентом, а также удобной интеграции с другими библиотеками и фреймворками.

Веб-приложение с использованием Vue.js состоит из компонентов, которые можно сравнить с независимыми блоками кода, отвечающими за отображение, логику и данные. Компоненты могут быть переиспользованы в разных частях приложения, что делает код более модульным и поддерживаемым.

Для создания интерфейса веб-приложения на Vue.js были использованы HTML-шаблоны, содержащие директивы Vue.js и JavaScript-код, отвечающий за логику и управление состоянием приложения.

**Установка Vue CLI**

|  |
| --- |
| npm install -g @vue/cli |

Данный тип установки позволяет использовать компонентный подход в разработке веб приложения.

После установки создается папка, в которой преднастроены все необходимые модули.

Запуск разработки производится командой:

npm run serve

**Установка Vuex**

Vuex — паттерн управления состоянием + библиотека для приложений на Vue.js. Он служит централизованным хранилищем данных для всех компонентов приложения с правилами, гарантирующими, что состояние может быть изменено только предсказуемым образом. Vuex интегрируется с официальным расширением

Директивы, из которых состоит хранилище Vuex:

• В центре любого Vuex приложения находится специальный контейнер, в котором хранится состояние нашего приложения. Описывается этот контейнер в специальной директиве state.

• Данные, которые хранятся в нашем хранилище нельзя изменять напрямую из любой точки нашего приложения. Это одна из концепций архитектуры, которая лежит в основе Vuex. Чтобы дать нам возможность управлять нашим состоянием, Vuex предоставляет специальные функции, которые это могут делать. Такие функции описываются в директиве mutations.

• Помимо хранения и изменения данных, нам необходимо уметь получать наше состояние в любом месте (в любом компоненте) нашего приложения. Учитывая эту необходимость, Vuex предоставляет нам специальные функции (можно провести аналогию с computed свойствами компонентов), которые описываются в директиве getters.

• Помимо перечисленных директив, существует еще одна группа функций, которая описывается в директиве actions. Функции, содержащиеся в данной директиве, могут содержать бизнес-логику, относящуюся управлению хранилищем (например, получение данных).

Установка производиться командой:

npm install vuex@3

Дальше необходимо непосредственно подключить ее к нам в приложение. Создаем файл store/index.js с таким контентом:

store/index.js

**Установка Vue-Router**

Чтобы установить Vue-Router к нам в приложении, достаточно воспользоваться командой npm install:

npm install -S vue-router@4

Добавим в точку входа в приложение main.js следующий код:

import { createApp } from "vue";

import App from "./App.vue";

import store from "./store";

import router from "./router/router";

var cors = require('cors')

createApp(App).use(store).use(router).mount("#app");

Маршруты — это массив объектов, в котором мы описываем правила для отображения компонентов в зависимости от маршрута.

Чтобы посмотреть на работу Vue Router, добавим компонент <router-view>, который отвечает за отображение. Он является своего рода контейнером, в котором отрисовывается компонент соответствующего маршрута.

**Навигация между маршрутами**

Для привычной навигации по ссылкам необходимо использовать компонент <router-link>. Он принимает проп :to, который указывает, куда будет вести ссылка. Можно прописать полный путь или параметры маршрута — например, имя и query-парметры, которые мы рассмотрим далее.

Создадим файл router.js и в нем пропишем все маршруты.

import MainPage from "@/pages/MainPage.vue";

import RoomDetailsPage from '@/pages/RoomDetailsPage.vue'

import CatalogPage from '@/pages/CatalogPage.vue'

import Error404Page from "@/pages/Error404Page.vue";

import ReviewsPage from '@/pages/ReviewsPage.vue'

import { createRouter, createWebHistory } from "vue-router";

const routes = [

{

path: '/',

component: MainPage,

},

{

path: '/catalog',

component: CatalogPage,

},

{

path: '/reviews',

component: ReviewsPage,

},

{

path: '/catalog/details/:id',

name: 'details',

component: RoomDetailsPage,

props: true,

},

{

path: '/:CatchAll(.\*)',

name: 404,

component: Error404Page,

},

]

const router = createRouter({

history: createWebHistory(),

routes,

scrollBehavior(to, from, savePosition) {

return {

top: 0,

};

},

});

// app.use(router);

export default router;

**Корневой компонент frontend**

App.vue — корневой компонент приложения Vue.js. Это начальная точка приложения, где визуализируются все остальные компоненты. Это однофайловый компонент, который служит контейнером для других компонентов и определяет базовую структуру приложения.

App.vue обычно состоит из трех основных частей: шаблона, скрипта и стиля. Раздел шаблона содержит HTML-структуру приложения, раздел сценария содержит код JavaScript, управляющий поведением приложения, а раздел стиля содержит код CSS, определяющий стиль приложения.

App.vue может также содержать другие компоненты и импортировать библиотеки по мере необходимости. Это центральная точка приложения, позволяющая легко управлять компонентами и данными приложения.

App.vue созданного приложения включает в себя два компонента HeaderComp.vue и FooterComp.vue, которые отображаются на всех страницах, и главную страницу MainPage.vue, а также роутинг по другим страницам.

App.vue

<template>

<div class="header-wrap">

<div class="wrap">

<HeaderComp />

</div>

</div>

<router-view></router-view>

<div class="footer" id="footer">

<FooterComp />

</div>

<div class="header-wrap">

<div class="wrap">

<HeaderComp />

</div>

</div>

</template>

<script>

import HeaderComp from "./components/HeaderComp.vue";

import FooterComp from "./components/FooterComp.vue";

import MainPage from "./pages/MainPage.vue";

export default {

name: "App",

components: {

HeaderComp,

MainPage,

FooterComp,

},

};

Главная страница MainPage.vue:

<template>

<div class="main-wrap">

<img class="main-wrap\_\_bkg" src="@/assets/img/cat-main.jpg" alt="#" />

<div class="wrap">

<div class="main\_picture">

<p class="main\_picture\_\_sity">Воронеж</p>

<h1 class="main\_picture\_\_site">Котейка</h1>

<p class="main\_picture\_\_text">Уютная гостиница для котов и кошек</p>

<ButtonBook></ButtonBook>

</div>

<div class="flex-center">

<div class="projects-wrap">

<div class="projects\_\_text-box">

<h1>Почему мы?</h1>

</div>

<div class="projects\_\_box">

<FirstCard

v-for="(card, index) in firstCards"

:key="index"

:card="card"

/>

</div>

</div>

</div>

<div class="room-wrap">

<div class="wrap">

<div class="room-wrap\_\_title">

<h1>Номера</h1>

</div>

<SwipperZoom></SwipperZoom>

</div>

</div>

</div>

</div>

<div id="footer"></div>

</template>

Содержит баннер с котиком, блок информации для пользователя, выделяющей лучшие стороны данной гостиницы, скролл из нескольких номеров для примера с возможностью перейти на страницу с номерами.

**Дочерние компоненты Frontend**:

* ButtonBook.vue - компонент с кнопкой «Забронировать»;
* ContactModal.vue - компонент с модальным окном - формой ввода информации о бронировании номера;
* FirstCard.vue - компонент с выводом рекламной информации о преимуществах данной гостиницы;
* FooterComp.vue - компонент с «подвалом» сайта;
* HeaderComp.vue - компонент с «шапкой» сайта.
* ResponseModal.vue - компонент с модальным окном, появляющимся в случае успешного сохранения данных о бронировании, ответа 200 от серверной части.
* ReviewsComp.vue – компонент с отзывами, пока захардкожен, в дальнейшем планируется создать форму для оставления пользователем отзыва и сохранения его через сервер в БД.
* Room.vue – компонент с информационными карточками номеров
* RoomCard.vue – компонент со всеми номерами
* RoomDetails.vue - компонент с карточкой номера, имеющей полную информацию по номеру и кнопкой для бронирования

**Сборка и развертывание проекта на сервер:**

Для того чтобы собрать все части приложения и развернуть на сервере, папка cat-hotel-frontend с Frontend-приложением была помещена в корневую папку Backend-приложения.

Для создания образа Frontend-приложения в его корневую папку добавлен файл Dockerfile со следующим содержанием:

FROM node:latest  
  
WORKDIR /app/frontend  
  
EXPOSE 8080  
  
COPY ["package.json", "package-lock.json\*", "./"]  
  
RUN npm install  
  
COPY . .  
  
CMD ["npm", "run", "serve"]

Для создания образа Backtend-приложения в его корневую папку добавлен файл Dockerfile со следующим содержанием:

FROM maven:3.8.6-amazoncorretto-17 AS *builder*WORKDIR /app  
COPY . /app/.  
RUN mvn -f /app/pom.xml clean package -Dmaven.test.skip=true  
  
FROM eclipse-temurin:17-jre-alpine  
WORKDIR /app  
COPY --from=*builder* /app/target/\*.jar /app/\*.jar  
ENTRYPOINT ["java", "-jar", "/app/\*.jar"]

Для запуска сборки контейнеров всех частей в Docker добавлен файл

docker-compose.yml со следующим содержанием:

version: '3.8'  
  
services:  
  
 backend-cat-hotel:  
 image: 'spring-boot:latest'  
 build:  
 context: .  
 dockerfile: Dockerfile  
 depends\_on:  
 - db-cat-hotel  
 environment:  
 - SPRING\_DATASOURCE\_URL=jdbc:postgresql://db-cat-hotel:5432/cat-hotel  
 - SPRING\_DATASOURCE\_USERNAME=postgres  
 - SPRING\_DATASOURCE\_PASSWORD=1q2w3e4r  
 - SPRING\_JPA\_HIBERNATE\_DDL\_AUTO=update  
 ports:  
 - "8085:8085"  
  
 db-cat-hotel:  
 image: 'postgres:15-alpine'  
 environment:  
 POSTGRES\_DB: "cat-hotel"  
 POSTGRES\_USER: "postgres"  
 POSTGRES\_PASSWORD: "1q2w3e4r"  
 ports:  
 - "5432:5432"  
  
 frontend-cat-hotel:  
 image: frontend:0.0.1  
 build: cat-hotel-frontend  
 restart: always  
 ports:  
 - '8080:8080'  
 volumes:  
 - /app/frontend/node\_modules  
 - ./cat-hotel-frontend:/app/frontend

теперь достаточно запустить одну команду из корневой папки:

**docker-compose up -d**

и все части нашего приложения запустятся в разных контейнерах, соответствующих своим названиям:

**frontend-cat-hotel -** Frontend-приложения

**backend-cat-hotel -** Backtend-приложения

**db-cat-hotel –** База данных

**и будут взаимодействовать между собой!**

**Сайт будет доступен по адресу: http://localhost:8080/**

**Тестирование работоспособности - UI/UX**

UI (пользовательский интерфейс) и UX (пользовательский опыт) — это два термина, которые часто используются вместе для описания дизайна и разработки цифровых продуктов, таких как веб-сайты, мобильные приложения и программное обеспечение.

Пользовательский интерфейс относится к визуальным и интерактивным элементам продукта, с которыми взаимодействует пользователь. Сюда входят такие вещи, как кнопки, меню, значки, текст, цвета и анимация. Цель дизайна пользовательского интерфейса — создать интерфейс, который будет визуально привлекательным, простым в навигации и интуитивно понятным в использовании.

UX, с другой стороны, относится к общему опыту пользователя при взаимодействии с продуктом. Это включает в себя все, от простоты использования до эмоционального отклика пользователя при использовании продукта. Цель UX-дизайна — создать продукт, удобный для пользователя, привлекательный и отвечающий потребностям и ожиданиям пользователя.

**Подведение итогов, оценка результатов и заключение**

В результате успешно выполненного дипломного проекта было создано полноценное веб-приложение, позволяющее получать полную информацию как в целом об услугах специализированной гостиницы для животных, так и о каждом номере конкретно. Также в данном приложении реализована функция бронирования номеров, для этого было создано еще одно приложение – серверная часть (Backend), обрабатывающая информацию о бронировании номеров и сохраняющая её в базу данных.

В данном дипломном проекте было разработано веб-приложение, включающая себя Frontend-часть, написанную с использованием технологий HTML, CSS, JavaScript, Vue.js, Axios и Backend-часть, написанную с использованием технологий Java 17, Spring Framework, Flyway, а также развернутую в контейнере Docker СУБД Postgres.

Одним из главных достижений проекта является умение правильно использовать различные инструменты и технологии для достижения поставленных целей. Кроме того, были применены современные подходы и методологии в разработке, что позволило получить высокое качество и удобство использования приложения.

Frontend был разработан с использованием Vue.js и Bootstrap, а backend был реализован на Java 17 с использованием Spring Boot.

В процессе разработки проекта были рассмотрены основные принципы и подходы к созданию веб-приложений, включая UI/UX дизайн, frontend и backend разработку, а также использование различных технологий для достижения поставленных целей.

Для успешной разработки веб-приложений необходимы хорошие знания и практические навыки работы с технологиями HTML, CSS, JavaScript, Vue.js, Axios, Java 17, Spring Framework, Flyway, PostgreSQL, Docker. Для начинающих разработчиков рекомендуется изучение соответствующих руководств, видеоуроков и книг по каждой из этих технологий.

В целом, проект был успешно завершен и достиг своей цели – создания веб-приложения приложения – специализированной гостиницы для кошечек, позволяющего подобрать гостиничный номер по своим параметрам и забронировать его, с использованием современных технологий и инструментов разработки.

**Список использованной литературы**

* **HTML, CSS, JS**

1. W3Schools - <https://www.w3schools.com/>

2. Codecademy - <https://www.codecademy.com/learn/learn-html> и по JS - <https://www.codecademy.com/learn/introduction-to-javascript>

3. FreeCodeCamp - <https://www.freecodecamp.org/>

4. MDN Web Docs - <https://developer.mozilla.org/en-US/docs/Learn>

5. CSS-Tricks - <https://css-tricks.com/>

6. HTML Dog - <https://htmldog.com/>

7. Learn to Code HTML & CSS - <https://learn.shayhowe.com/html-css/>

8. TutorialsPoint - <https://www.tutorialspoint.com/index.htm>

9. SoloLearn - <https://www.sololearn.com/>

10. Документация на русском языке JS - <https://learn.javascript.ru/>

11. <https://skillbox.ru/media/code/chto_takoe_html/>

12. <https://gb.ru/posts/chto-takoe-css-obyasnyaem-prostymi-slovami>

13. <https://developer.mozilla.org/ru/docs/Learn/JavaScript/First_steps/What_is_JavaScript>

14.

* **Bootstrap, Axios**

1. Официальная документация Bootstrap - <https://getbootstrap.com/docs/5.2/getting-started/introduction/>

2. Bootstrap обучение W3Schools - <https://www.w3schools.com/bootstrap/>

3. Bootstrap обучение Tutorialspoint - <https://www.tutorialspoint.com/bootstrap/index.htm>

4. FreeCodeCamp - <https://www.freecodecamp.org/news/tag/bootstrap/>

5. Официальная документация Axios - <https://axios-http.com/docs/intro>

6. Axios обучение Vue Mastery - <https://www.vuemastery.com/courses/real-world-vue-js/axios/>

7. <https://itchief.ru/bootstrap/introduction>

8. <https://blog.skillfactory.ru/glossary/bootstrap/>

9. <https://guides.hexlet.io/ru/bootstrap/>

10. <https://habr.com/ru/companies/ruvds/articles/477286/>

11. <https://www.8host.com/blog/kak-ispolzovat-axios-v-prilozhenii-javascript/>

12. Официальная документация Java - <https://docs.oracle.com/en/java/>

13. Официальная документация Spring - <https://docs.spring.io/spring-framework/reference/index.html>

14. Официальная документация Docker - <https://docs.docker.com/>

15. Статья на Хабр по развертыванию приложений в Docker - <https://habr.com/ru/articles/467161/>

* **Youtube каналы по направлениям обучения:**

<https://www.youtube.com/@freecodecamp> - html, css, node.js, express

<https://www.youtube.com/@VladilenMinin> - html, css, js

<https://www.youtube.com/@dmitrylavrik> - vue

<https://www.youtube.com/@TraversyMedia> - html, css, js, vue, puppeteer, cheerio

<https://www.youtube.com/@MichaelKitas> - puppeteer

<https://www.youtube.com/@itgid> - js, node.js

<https://www.youtube.com/@ArchakovBlog> - js, node.js, express