Министерство образования и науки Хабаровского края

Краевое государственное бюджетное

Профессиональное образовательное учреждение

«Хабаровский колледж отраслевых технологий и сферы обслуживания»

Специальность 09.02.07 Информационные системы и программирование

ПРЕДДИПЛОМНАЯ ПРАКТИКА

ОТЧЕТ

ПДП.09.02.07.696.2024

|  |  |
| --- | --- |
| Проверил преподаватель: | Выполнил студент группы ИС-41 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_/Малышев А.С./ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Наугольнов А.И./ |
| «\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024г. | «\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024г. |

2024

СОДЕРЖАНИЕ

[Введение 3](#_Toc165224129)

[1 Анализ предметной области 4](#_Toc165224130)

[2 Анализ аналогичных сайтов 5](#_Toc165224131)

[3 Техническое задание 6](#_Toc165224132)

[4 Обоснование выбранных средств разработки 8](#_Toc165224133)

[5 Информационная система сайта 12](#_Toc165224134)

[6 Описание реализации основной функции сайта 14](#_Toc165224135)

[6.1 Создать удобную навигацию по сайту (фильтр и поиск) 14](#_Toc165224136)

[6.2 Регистрация 18](#_Toc165224137)

[6.3 Создание почты 21](#_Toc165224138)

[6.4 Извлечение геоданных 24](#_Toc165224139)

[Заключение 28](#_Toc165224140)

[Список использованных источников 29](#_Toc165224141)

## ВВЕДЕНИЕ

Отчет представляет собой описание преддипломной практики, проведенной в рамках специальности 09.02.07 «Информационные системы и программирование». В ходе выполнения данной работы был разработан и создан веб-сайт в соответствии с требованиями научного руководителя проекта.

Целью преддипломной практики ставится демонстрация практических навыков в разработке веб-приложений, а также применение полученных знаний и умений в реальной рабочей среде. В рамках специальности 09.02.07 «Информационные системы и программирование» рассматривается разработка веб-сайтов и веб-приложений. В ходе преддипломной практики были рассмотрены такие вопросы, как проектирование интерфейса веб-сайта, разработка дизайна пользовательского интерфейса, создание графических элементов, разработка технического задания, проектирование веб-приложений, разработка и описание базы данных, административной части и др., проектирование информационной системы, оптимизация веб-приложения, сбор информации.

В связи с учащением случаев экстремальных погодных условий, резких перемен погодных условий все чаще среди населения возникает потребность обращения к сервисам, предоставляющим информационные услуги в области метеорологии. В связи с ограничением доступа российским пользователям к наиболее популярным сервисам (часто предустановленным на устройства пользователей), возникает потребность в поиске таких же удобных альтернатив. Разрабатываемый проект представляет собой такую альтернативу.

В данной работе представлены результаты проделанной работы, описание разработанного веб-сайта, анализ достигнутых результатов и примененных методов.

## 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Предметная область данного проекта - метеорологические прогнозы и их распространение через веб-сайт. Основная цель состоит в предоставлении пользователям актуальной и точной информации о текущих и будущих погодных условиях в различных регионах.

* Ключевые компоненты предметной области включают:
* Визуализация прогнозов в понятном для пользователей формате, включая графические изображения, таблицы и текстовые описания.
* Предоставление прогнозов на различные временные периоды, от ближайших часов до нескольких дней вперед.
* Интеграция с геопространственными данными для привязки прогнозов к конкретным местоположениям.
* Пользовательский интерфейс веб-сайта, обеспечивающий удобный доступ к прогнозам и возможность настройки отображаемой информации.

Для успешной реализации проекта необходимо решить ряд задач, связанных с получением, обработкой и визуализацией метеорологических данных, а также созданием удобного и информативного веб-интерфейса.

Анализ потенциальных пользователей сайта прогноза погоды позволяет выделить следующие целевые группы:

* Жители городов и населенных пунктов. Для этой группы пользователей прогноз погоды имеет практическое значение в планировании повседневной деятельности, выборе одежды, организации мероприятий на открытом воздухе и т.д.
* Туристы и путешественники. Данной категории важно знать погодные условия в местах назначения для эффективного планирования маршрутов, выбора экипировки и обеспечения безопасности.
* Представители экстремальных видов спорта и активного отдыха. Для данной группы погодные условия определяют безопасность и возможность проведения мероприятий на открытом воздухе.
* Строители и ремонтные бригады. Прогноз погоды важен для планирования и организации работ на открытых строительных площадках.

Понимание потребностей и особенностей каждой из этих групп позволит адаптировать структуру, функционал и интерфейс сайта для максимального удовлетворения запросов различных категорий пользователей.

2 АНАЛИЗ АНАЛОГИЧНЫХ САЙТОВ

OpenWeatherMap предоставляет услуги доступа к метеостанциям и метеорологическим данным для разрабатываемого проекта, но так же и предоставляет собственные информационные услуги в области метеорологии.

Через сервисы OpenWeatherMap можно получить доступ к идентичным данным, однако, OpenWeatherMap предоставляет интерфейс, который может показаться недостаточно удобным конечному пользователю.

Сервис не запрашивает геоданные пользователя, что значит, что пользователю придется вписывать название своего города вручную в строку поиска, которая требует полного совпадения с официальным названием города. Разрабатываемый проект может получать данные из API браузера пользователя и пытаться угадать текущее местоположение пользователя. Особенно эффективно это получается на мобильных устройствах, у которых есть встроенный GPS.

Разрабатываемый проект так же позволяет отслеживать погоду в нескольких городах одновременно, такого функционала в OpenWeatherMap нет совсем.

3 ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Техническое задание проектируемой информационной системы

Общие сведения

Разрабатываемая информационная система является информационным сайтом, предоставляющем услуги прогнозирования погоды. Разрабатываемый сайт позволяет пользователям получить прогноз погоды в их городе, использовать поиск, чтобы получить прогноз погоды в другом городе, а так же добавлять города в список отслеживаемых и получать доступ к текущей погоде и прогнозе на 15 часов вперед в любое время.

Формулировка задания

Реализовать существующую и предоставленную информационную систему в учебных целях.

Цели, достигаемые разработкой

* Обеспечение доступа к информации о погоде;
* Создание информационной системы, позволяющей работать с прогнозами погоды;

Категории пользователей

Результат разработки ориентирован на следующих пользователе: пользователи всех возрастов, проживающие в городах

Наименование организации-заказчика

Нет

Основание для проведения работ

Индивидуальное задание, выданное в целях проведения производственной практики.

Описание предметной области

Сфера деятельности: прогнозирование погодных условий в населенных пунктах на основе метеорологических данных.

Состав данных и алгоритмов обработки информации

В разрабатываемом проекте хранятся данные о пользователе: хэшированный пароль, электронная почта, используемая для активации (а следовательно, в базе данных так же хранятся данные об активации), сохраненный город пользователя и отслеживаемые города (хранятся названия городов).

Так же разрабатываемый проект обращается к серверам OpenWeatherMap, откуда получает данные о: долготе и широте города, название города, состояние погоды (айди иконки, тип и локализированное описание типа), температура (минимальная, максимальная, текущая, температура по ощущениям), влажность, ветер (скорость и направление), давление и многое другое. Эти данные отображаются на странице пользователя.

Недостатки существующих проектных решений

OpenWeatherMap предоставляет услуги доступа к метеостанциям и метеорологическим данным для разрабатываемого проекта, но так же и предоставляет собственные информационные услуги в области метеорологии.

Через сервисы OpenWeatherMap можно получить доступ к идентичным данным, однако, OpenWeatherMap предоставляет интерфейс, который может показаться недостаточно удобным конечному пользователю.

Сервис не запрашивает геоданные пользователя, что значит, что пользователю придется вписывать название своего города вручную в строку поиска, которая требует полного совпадения с официальным названием города. Разрабатываемый проект может получать данные из API браузера пользователя и пытаться угадать текущее местоположение пользователя. Особенно эффективно это получается на мобильных устройствах, у которых есть встроенный GPS.

Разрабатываемый проект так же позволяет отслеживать погоду в нескольких городах одновременно, такого функционала в OpenWeatherMap нет совсем.

4 ОБОСНОВАНИЕ ВЫБРАННЫХ СРЕДСТВ РАЗРАБОТКИ

Для разрабатываемого веб-сайта не было использовано никакого бэкенд фреймворка, серверная часть разработана средствами стандартной библиотеки Golang. Пользовательский интерфейс был разработан с использованием библиотеки React.js и TailwindCSS

библиотека стандартных пакетов языка программирования Golang, особенно пакеты net и net/http, обеспечивают ряд преимуществ перед полноценными фреймворками, такими как Remix и Next.js. Эти преимущества вытекают из идеологии и философии, лежащей в основе разработки Go, которая подчеркивает простоту, производительность и масштабируемость. В отличие от подхода «все в одном», принятого во многих фреймворках, Go предлагает более гибкий и модульный подход, позволяющий разработчикам выбирать только те компоненты, которые необходимы для конкретного проекта. Пакеты net и net/http предоставляют мощный и эффективный инструментарий для создания веб-приложений и работы с сетевыми протоколами, при этом оставляя место для гибкости и индивидуальной настройки.

Одним из главных преимуществ использования стандартных пакетов Go является их производительность. Будучи написанными на низкоуровневом языке, они обеспечивают превосходную производительность и эффективность использования ресурсов, что имеет решающее значение для высоконагруженных веб-приложений и масштабируемых систем. Кроме того, стандартные пакеты Go являются кроссплатформенными, что обеспечивает переносимость кода между различными операционными системами и аппаратными платформами, снижая затраты на техническое обслуживание и упрощая развертывание приложений.

В отличие от фреймворков, таких как Remix и Next.js, которые могут иметь сложную структуру и требовать значительных ресурсов, стандартные пакеты Go отличаются легковесностью и простотой в использовании. Это позволяет разработчикам сосредоточиться на основной логике приложения, не отвлекаясь на излишнюю сложность и избыточные абстракции. Кроме того, Go имеет строгую систему типов и мощные средства для параллельного программирования, что способствует созданию надежных и масштабируемых приложений.

Помимо этого, сообщество Go постоянно развивается, и с каждым новым выпуском вносятся улучшения и новые возможности.

React.js – это популярная JavaScript-библиотека с открытым исходным кодом для создания пользовательских интерфейсов.

Она была разработана и поддерживается Facebook. React.js используется для построения интерактивных веб-приложений и одностраничных приложений, обеспечивая высокую производительность и эффективность. Ядром React.js является декларативный подход к созданию пользовательских интерфейсов, где разработчики описывают желаемое состояние приложения, а библиотека обновляет и отображает соответствующие компоненты пользовательского интерфейса.

React.js упрощает разработку сложных пользовательских интерфейсов, разбивая их на повторно используемые компоненты. Это способствует модульности и облегчает поддержку кода.

Благодаря виртуальному DOM, React.js оптимизирован для эффективного рендеринга, обновляя только те части пользовательского интерфейса, которые действительно нуждаются в изменении.

React.js интегрируется с широким спектром библиотек и инструментов, что делает его чрезвычайно гибким и расширяемым.

При разработке сайта React.js может использоваться для создания динамических пользовательских интерфейсов, интерактивных виджетов, анимаций и других элементов, требующих высокой производительности и реактивности.

Библиотека TailwindCSS предлагает ряд преимуществ по сравнению со своими конкурентами в области CSS-фреймворков и методологий, таких как Bootstrap, Foundation или методология БЭМ. Философия TailwindCSS основана на использовании утилитарных классов, которые обеспечивают более прямой и понятный способ стилизации элементов интерфейса, способствуя повышению производительности и удобства разработки.

Одним из главных преимуществ TailwindCSS является ее гибкость и легко настраиваемый подход к стилизации. В отличие от традиционных CSS-фреймворков, которые предлагают готовые компоненты и стили, TailwindCSS позволяет разработчикам создавать уникальные дизайны, комбинируя низкоуровневые утилитарные классы. Это означает, что вместо того, чтобы полагаться на предопределенные стили, разработчики могут точно настроить внешний вид своих компонентов, используя классы для управления отступами, цветами, размерами шрифтов и другими аспектами визуального оформления. Такой подход обеспечивает большую гибкость и контроль над стилизацией, что особенно важно для проектов с уникальными дизайнерскими требованиями.

Кроме того, TailwindCSS способствует повышению производительности и сокращению времени разработки. Благодаря использованию утилитарных классов, разработчикам не нужно создавать и поддерживать сложные каскадные стили, что может привести к конфликтам и трудностям в управлении кодом. Вместо этого TailwindCSS предлагает более прямой и понятный подход, где каждый элемент стилизуется отдельно, без необходимости вложенности и сложных селекторов. Это упрощает процесс разработки, повышает производительность и облегчает сотрудничество в команде.

Стоит также отметить, что TailwindCSS обладает хорошей производительностью и оптимизацией для веб-разработки. Библиотека использует подход «утилитарных классов», что позволяет включать в проект только необходимые стили, избегая излишней нагрузки на производительность. Кроме того, TailwindCSS обеспечивает удобную интеграцию с популярными фреймворками и библиотеками, такими как React, Vue и Angular, что позволяет легко использовать ее в различных проектах и стеках технологий.

Благодаря своей гибкости, производительности и удобству использования, TailwindCSS завоевала популярность среди разработчиков, предлагая альтернативный подход к стилизации веб-приложений. Хотя этот подход может потребовать некоторой адаптации и изменения привычек, преимущества TailwindCSS, такие как легкая настройка, повышенная производительность и удобство совместной разработки, делают ее привлекательным выбором для современных веб-проектов.

Интерфейс разрабатываемого приложения выполнен в минималистичном, сеточном стиле.

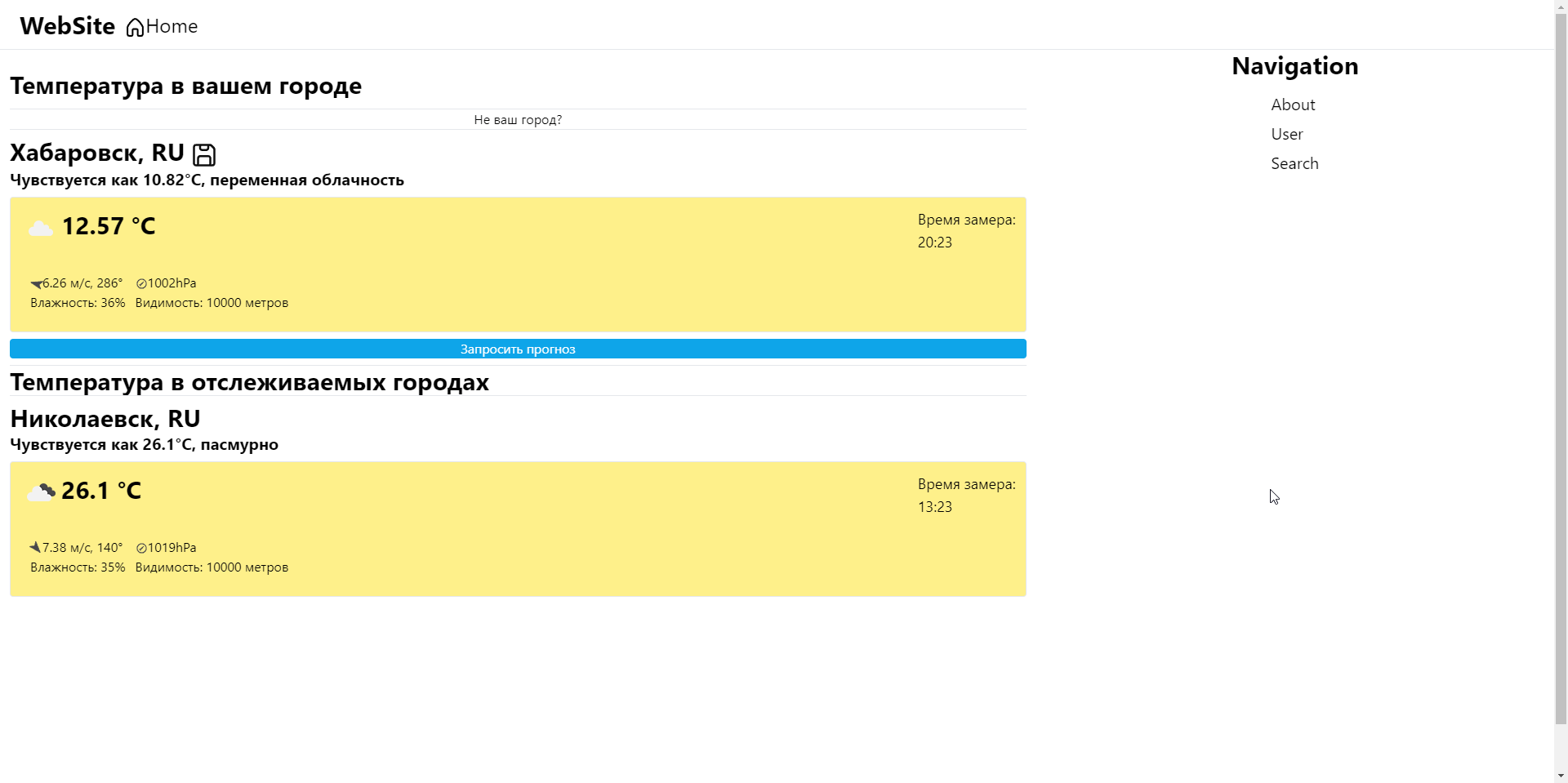


Рисунок 2 – Макет разрабатываемого сайта

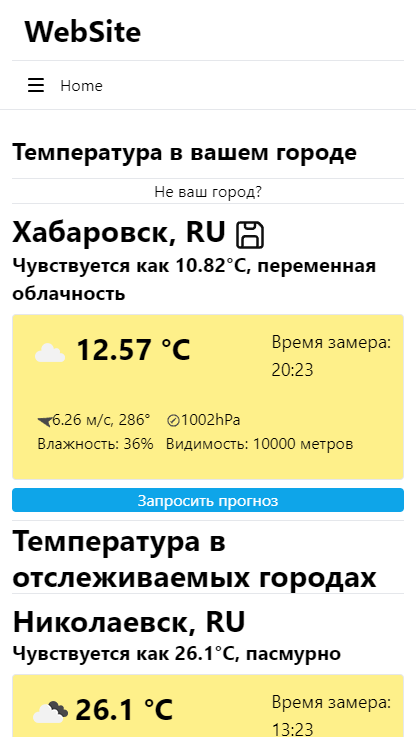


Рисунок 3 – Мобильная версия разрабатываемого проекта

Структура разрабатываемого сайта определена в главе 2.2.

Описание разрабатываемой базы данных и соответствующей административной части веб-приложения определено в главе 2.3.

* ИНФОРМАЦИОННАЯ СИСТЕМА САЙТА

Информационная модель

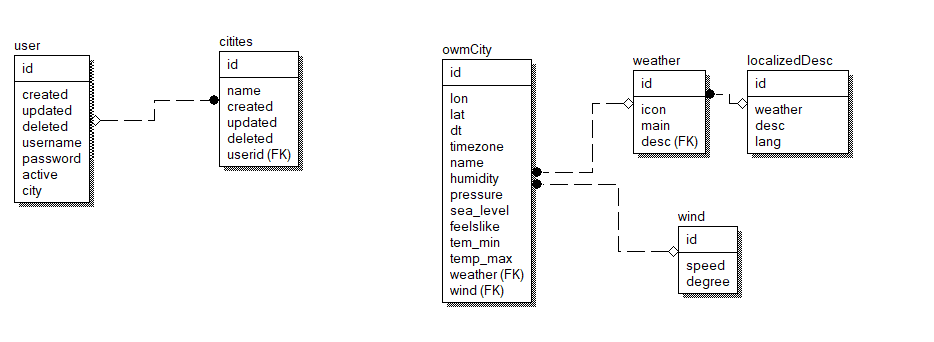


Рисунок 1 – Схема информационной модели сайта

База данных информационной системы состоит из нескольких кластеров, не связанных друг с другом. Все сущности в БД хранят динамически изменяемые по запросу пользователя данные для отображения на сайте предприятия.

Сущность «User» содержит информацию о пользователе.

Сущности «Cities» содержит информацию об отслеживаемых городах пользователя.

Сущности «owmCity», «weather», «localizedDesc», «wind» представляют собой информацию полученную с серверов OpenWeatherMap.

Сущность «owmCity» хранит данные погодных условиях в выбранном городе: температуру (мин и макс, ощущаемую), погоду, время замера с часовым поясом, название города, влажность, давление, уровень моря, долготу и ширину и ветер (скорость и направление).

Сущность «weather», хранит информацию о виде погоды: иконку, которую можно запросить с CDN OpenWeatherMap для использования на сайте, вид погоды и локализованное описание. Описание берется из сущности «localizedDesc», который хранит информацию о переводах описания погоды.

Сущность «wind» содержит информацию о ветре.

Структура сайта

Далее представлена структура разрабатываемого веб-сайта

* Главная
  + О нас
  + Панель пользователя (войти/зарегистрироваться в случае если пользователь не вошел в аккаунт)
  + Поиск

Требования к информационному обеспечению

При регистрации, пользователю придется ввести электронную почту регистрируемого пользователя, на которую затем придет письмо с ссылкой на активацию аккаунта. Так же потребуется ввести имя пользователя и пароль, который позже будет захеширован перед записью в базу данных.

При работе с сохраненными городами, пользователь может сохранить город из вкладки поиска, сохраненный город добавится в таблицу на странице панели пользователя. Там пользователь сможет удалить этот город из отслеживаемых нажатием кнопки.

6 ОПИСАНИЕ РЕАЛИЗАЦИИ ОСНОВНОЙ ФУНКЦИИ САЙТА

6.1 Создать удобную навигацию по сайту (фильтр и поиск)

Функционал поиска определен в файле Search.tsx. Это компонент React осуществляющий рендеринг результатов поиска на страницу, для использования пользователей. Функционал поиска можно разделить на следующие блоки кода по функционалу.

1. Импорт необходимых компонентов и библиотек:

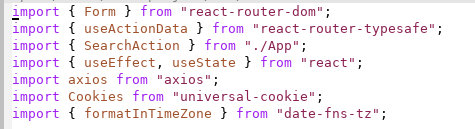
****

Рисунок 4 – Блок импортов

Этот блок импортирует различные компоненты и библиотеки, необходимые для работы данного модуля, такие как, useActionData для получения данных с сервера, axios для осуществления http запросов, formatInTimeZone для работы с датами и временем.

2. Функция action:

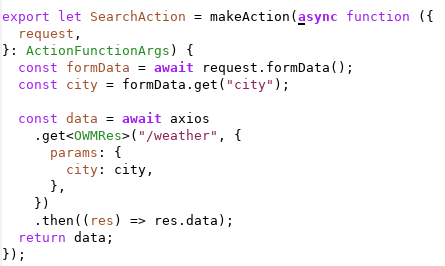


Рисунок 5 – Функция обработчика

Эта функция является частью сторонней библиотеки React Router, обеспечивающей маршрутизацию в React приложениях и используется для обработки запросов из форм. Она извлекает параметры поиска, отправляет данные на сервер, который затем обрабатывает информацию.

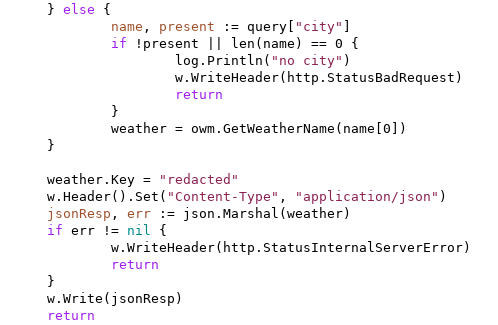


Рисунок 6 – Функция-обработчик на сервере

На сервере извлекается название города, с использованием которого затем происходит поиск и получение прогноза погоды. Далее данные с сервера возвращаются обратно клиенту.

3. Компонент Search:

Это основной компонент React, который отображает интерфейс поиска и результаты. Он использует данные, загруженные функцией action, и отображает различные элементы пользовательского интерфейса, такие как навигационный компонент, хлебные крошки, форму поиска и результаты.

Листинг 1 – Определение состояний

const data = useActionData<typeof SearchAction>();

const [weather, setWeather] = useState<OWMRes>();

const [hiddenAnimation, setHiddenAnimation] = useState<boolean>(true);

const [isError, setIsError] = useState<boolean>(false);

const [success, setSuccess] = useState<{ success: boolean; message?: string }>();

Этот блок использует хук useActionData для получения данных из формы поиска и определяет различные состояния для управления отображением компонента.

4. Форма поиска:

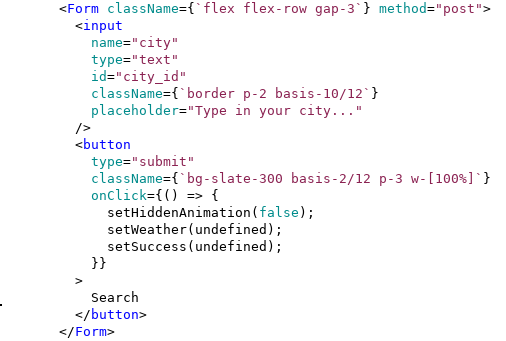


Рисунок 7 – Форма поиска

Этот блок кода отображает форму поиска с текстовым полем для ввода поискового запроса и кнопкой, которая осуществляет поиск, а так же сбрасывает состояния, очищая их и подготавливая к ответу от сервера.

5. Отображение результатов поиска:

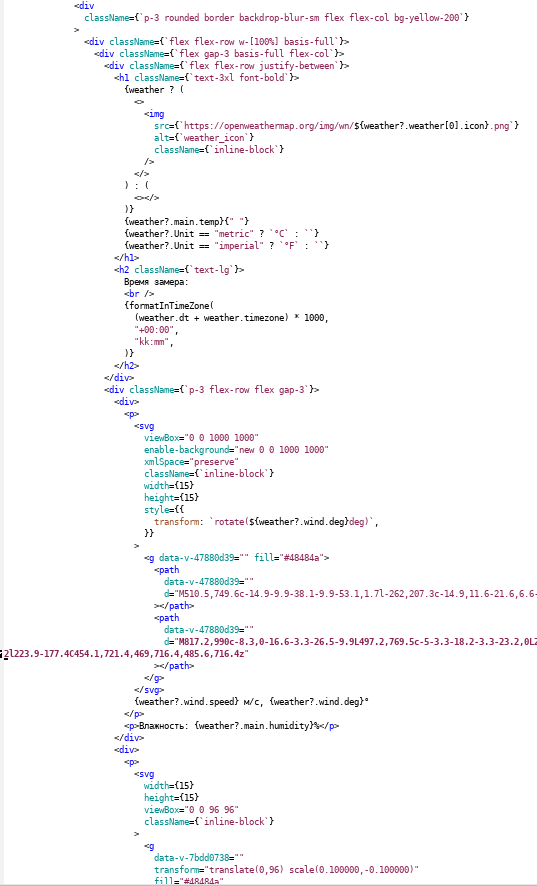


Рисунок 8 – Блок отображения результатов, часть 1



Рисунок 9 – Блок отображения результатов, часть 2

Этот блок кода отображает результаты поиска в виде карточек с данными о текущей погоде: название города, страна, описание погоды, температура, ветер, влажность, ветер и так далее. Так после карточки рендерится кнопка, которая осуществляет запрос на сервер и добавляет город в избранные пользователя.

Таким образом, этот код реализует функциональность поиска на сайте, загружая данные на сервере, фильтруя их в соответствии с запросом поиска и отображая результаты в пользовательском интерфейсе.

6.2 Регистрация

Регистрация и авторизация почти идентичны. Далее будет описана только авторизация, поскольку единственная разница между авторизацией и регистрацией – четкое разделение полей имени пользователя и почты, а так же запись данных в БД вместо поиска.

1. Импорт необходимых модулей и функций:



Рисунок 10 – Блок импортов

Этот блок импортирует типы данных и функции.

2. Функции валидации:

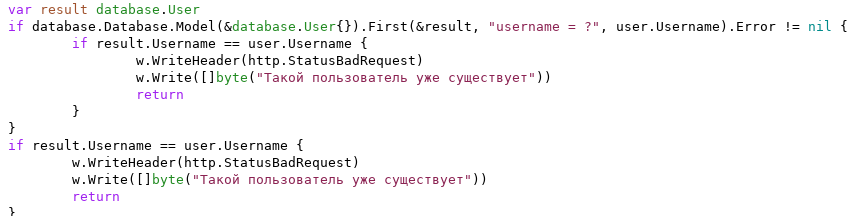


Рисунок 11 – Функции валидации

Эти функции используются для проверки корректности введенных данных (имя пользователя, пароль).

3. Функция действия (action):



Рисунок 12 – Функция action

Эта функция Remix обрабатывает отправленные формы. Она извлекает данные из формы, отправляя их на сервер.

4. Серверная часть:

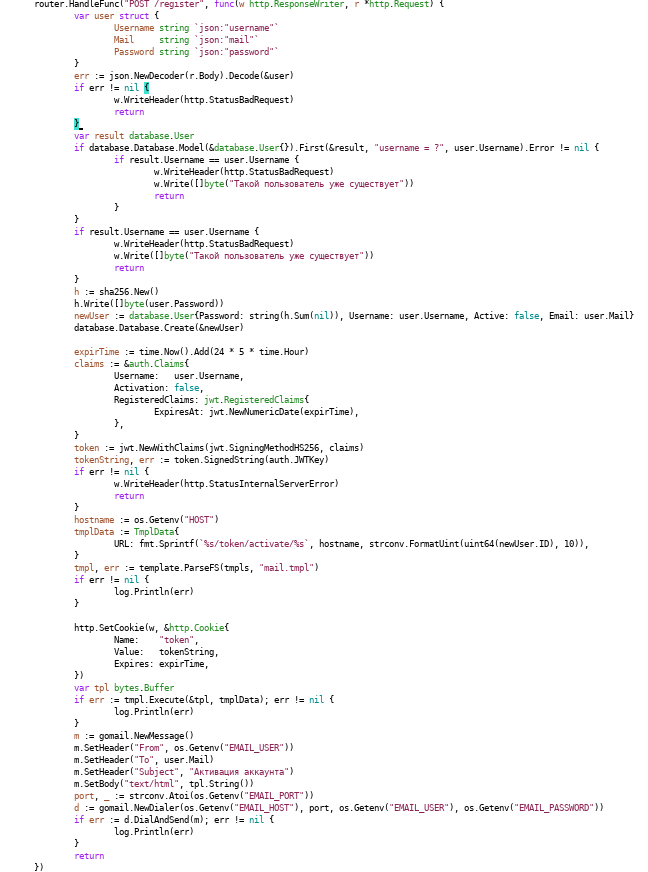


Рисунок 13 – Серверная часть

Это функция обработчик на сервере, который занимается извлечением данных из тела запроса, валидацией данных, созданием JWT токена, который будет использоваться после регистрации и отправкой письма на почту для активации аккаунта.

5. Форма регистрации:



Рисунок 14 – Форма регистрации

Этот блок кода отображает форму входа с полями для ввода имени пользователя, пароля и почты.

6. Отображение сообщений об ошибках:

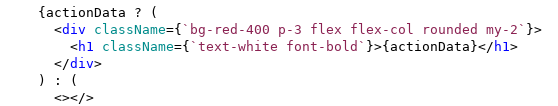


Рисунок 15 – Функция отображения сообщений об ошибках

Этот блок кода получает ответ с сервера и отображает его пользователю.

6.3 Создание почты

Во время регистрации и после изменения данных пользователя на почту будет отправлено письмо с ссылкой на активацию. Далее представлен код, генерирующий HTML письма и отправляющий его на почту.

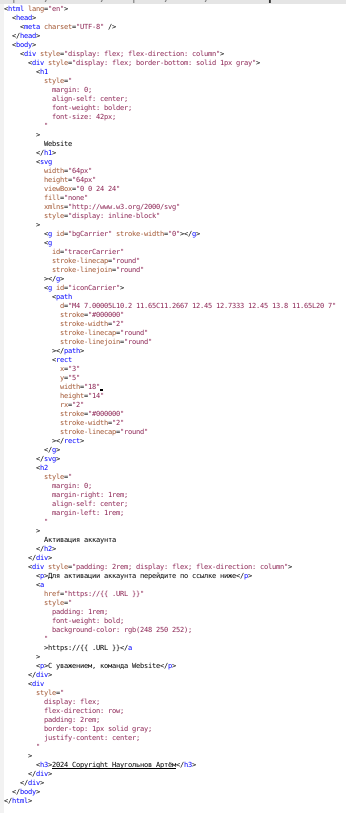


Рисунок 16 – Генерация письма



Рисунок 17 – Отправление почты

В конечном итоге на почту пользователя придет следующее письмо.

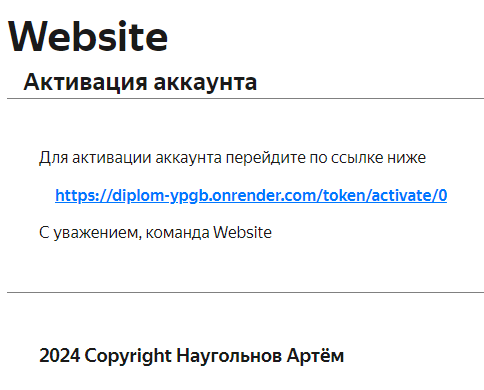


Рисунок 18 – Письмо с активацией аккаунта

6.4 Извлечение геоданных

В разрабатываемом веб-приложении, в случае если у пользователя нет аккаунта, пользователь еще не вошел в аккаунт или в базе данных не установлен город по умолчанию для данного пользователя, данные о текущем местоположении извлекаются из API браузера под названием navigation.



Рисунок 19 – Функция получения геоданных

Это асинхронная функция getWeather, которая используется для получения текущей погоды на основе геолокации пользователя.

1. Проверка разрешения на доступ к геолокации:



Рисунок 20 – Проверка разрешения

Эта строка использует API «Permissions» для запроса разрешения на доступ к геолокации устройства пользователя. Метод «query» возвращает объект «PermissionStatus», который содержит текущее состояние разрешения.

1. Обработка различных состояний разрешения:

Листинг 2 – Блок then

.then(function (result) {

if (result.state === "granted") {

// код для случая, когда разрешение уже предоставлено

} else if (result.state === "prompt") {

// код для случая, когда пользователю будет показан запрос на разрешение

} else if (result.state === "denied") {

// код для случая, когда разрешение было отклонено

}

});

Этот блок «then» обрабатывает различные состояния разрешения на доступ к геолокации:

1. «granted»: Если разрешение уже предоставлено, функция получает текущие координаты устройства с помощью «navigator.geolocation.getCurrentPosition».
2. «prompt»: Если пользователю будет показан запрос на разрешение, функция также получает текущие координаты устройства после предоставления разрешения.
3. «denied»: Если разрешение было отклонено, функция выводит сообщение в консоль.
4. Получение координат устройства:

Листинг 3 – Получение координат

navigator.geolocation.getCurrentPosition(async (pos) => {

// код для получения погоды с использованием координат

});

Этот блок кода использует метод «getCurrentPosition» из API «Geolocation» для получения текущих координат устройства (долгота и широта). Полученные координаты передаются в анонимную функцию в качестве параметра «pos».

1. Получение погоды с использованием координат:

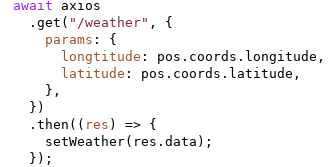


Рисунок 21 – Получение погоды с использованием координат

Этот блок кода отправляет GET-запрос на сервер по пути «/weather», передавая долготу и широту в качестве параметров запроса. Ответ от сервера обрабатывается в блоке «then», где данные погоды устанавливаются в состояние «weather» с помощью функции «setWeather».

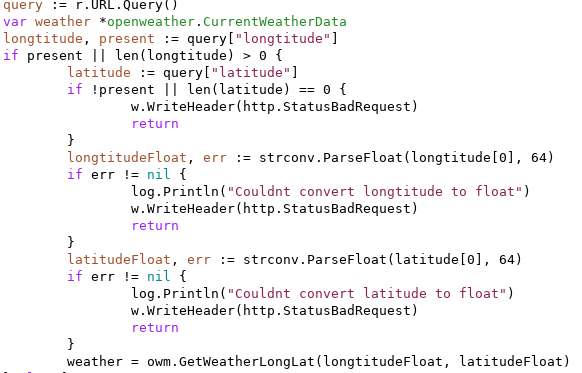


Рисунок 22 – Обработка запроса с координатами на сервере

Таким образом, эта функция «getWeather» запрашивает разрешение на доступ к геолокации устройства, получает текущие координаты и использует их для получения данных о погоде с сервера.

ЗАКЛЮЧЕНИЕ

Цель выполнения преддипломной практики на предприятии филиал ППК «Роскадастр» «Дальневосточное АГП» состояла в демонстрации практических навыков в проектировании и разработке веб-сайтов. Одной из основных задач практики было создание веб-сайта, обрабатывающего запросы с метеостанций, предоставляемых сервисом OpenWeatherMap.

В ходе прохождения практики на предприятии филиал ППК «Роскадастр» «Дальневосточное АГП» был получен ценный опыт работы в профессиональной среде, расширен кругозор в области информационных технологий и углублены знания в веб-разработке. В рамках практики были выполнены следующие задачи:

Проектирование и разработка метеорологического веб-сайта;

Изучение основных языков программирования и технологий, применяемых при создании веб-приложений;

Расширение практических навыков в проектировании и разработке веб-приложений;

Получение опыта работы в профессиональной среде.

Результатом прохождения практики стало разработка новых функций для дипломного проекта – метеорологического вебсайта, отвечающий всем современным требованиям и стандартам веб-разработки. Прохождение производственной на предприятии филиал ППК «Роскадастр» «Дальневосточное АГП» в области веб-разработки позволило получить практический опыт работы с процессом создания сайта от идеи до реализации, а также ознакомиться с основными инструментами и технологиями, применяемыми при разработке веб-приложений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абрамян, Михаил Технология LINQ на примерах. Практикум с использованием электронного задачника Programming Taskbook for LINQ / Михаил Абрамян. - М.: ДМК Пресс, 2021. - 222 c.

2. Агуров, Павел C#. Разработка компонентов в MS Visual Studio 2005/2008 / Павел Агуров. - М.: БХВ-Петербург, 2023. - 469 c.

3. Дунаев, В. Сценарии для Web-сайта. PHP и JavaScript / В. Дунаев. - М.: БХВ-Петербург, 2021. - 576 c.

4. Журнал Открытые системы. СУБД №2. - М.: Открытые Системы, 2020. - 651 c.

5. Каслдайн, Э. Изучаем jQuery / Э. Каслдайн. - М.: Питер, 2021. - 768 c.

6. Литвин, Евгений Прибыльный блог. Создай, раскрути и заработай / Евгений Литвин. - М.: Питер, 2021. - 272 c.

7. Майкл, С. Миковски Разработка одностраничных веб-приложений / Майкл С. Миковски, Джош К. Пауэлл. - М.: ДМК Пресс, 2022. - 512 c.

8. Рассел, Джесси Закон Ципфа / Джесси Рассел. - М.: VSD, 2022. - 971 c.

9. Уайт, Т. Hadoop. Подробное руководство / Т. Уайт. - М.: Питер, 2021. - 941 c.

10. Шапошников, И. Web-сайт своими руками / И. Шапошников. - М.: Книга по Требованию, 2022. - 224 c.