МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Факультет компьютерных наук

Кафедра программирования и информационных технологий

Курсовая работа по дисциплине «Технологии программирования»

Сайт с прогнозом погоды

Направление 09.03.02 Информационные системы и технологии

Преподаватель _____В.С. Тарасов, ст. преподаватель __.__20__

Обучающийся _____А.А. Маслов, 3 курс, д/о

Обучающийся _____В.И. Лысенко, 3 курс, д/о

Обучающийся _____Д.Е. Карпов, 3 курс, д/о

Воронеж 2023

Содержание

Введение

В эру цифровизации, где информационные технологии стали неотъемлемой частью нашей жизни, применение веб-сервисов и приложений, предлагающих прогноз погоды, выросло в геометрической прогрессии. Это связано не только с общим увеличением мобильных устройств и доступа к Интернету, но и с необходимостью получения точных и своевременных данных о погодных условиях в различных частях мира. Однако многие из существующих сервисов зачастую обеспечивают не точную информацию, и им не хватает индивидуального подхода к представлению информации.

Курсовая работа, посвященная созданию сайта с прогнозом погоды с реализацией собственного API, имеет важное значение для глубокого понимания процесса разработки и управления веб-сервисами. Это включает различные аспекты, такие как дизайн пользовательского интерфейса, обработка и визуализация данных, а также применение веб-технологий и программирования.

Настоящая работа представляет собой концепцию создания сайта, предоставляющего информацию о погоде, с использованием собственного API для доступа к прогнозам погоды. Основная цель - разработать функциональный и удобный в использовании веб-сервис, который будет предоставлять точные и актуальные прогнозы погоды в удобном для пользователей формате.

Собственное API позволит обеспечить высокую гибкость и возможность интеграции с другими веб-сервисами и приложениями. Этот подход будет способствовать улучшению качества предоставляемой информации и позволит создать уникальный продукт, отличающийся от существующих аналогов на рынке.

Работа структурирована таким образом, чтобы охватить все этапы процесса разработки, начиная от предварительного анализа и концепции, и

заканчивая тестированием и внедрением продукта. Все эти этапы и компоненты представлены в соответствующих главах настоящей работы.

Основная часть

1 Постановка задачи

В эпоху цифровизации все больше услуг переходят в онлайн-сферу, стирая географические границы и предлагая пользователям более удобные и доступные решения. Один из важных и актуальных примеров - сервисы прогноза погоды, которые выходят за рамки предоставления информации только о температуре, влажности и осадках, и включают дополнительные функции, такие как гороскопы и рекомендации по одежде.

Основной задачей этой курсовой работы является разработка сайта, обеспечивающего информацию о погоде, гороскопы и рекомендации по одежде с использованием собственного API. Это задание предполагает выполнение следующих подзадач:

- Сначала необходимо провести детальный анализ потребностей пользователей в области прогноза погоды, включая определение наиболее важных характеристик и функций.
- Следующим шагом будет разработка собственного API для обеспечения доступа к данным о погоде, гороскопах и рекомендациях по одежде. Это включает в себя проектирование архитектуры API, выбор технологий и разработку конкретных эндпоинтов для обеспечения нужной функциональности.
- Затем необходимо разработать сам веб-сайт. Это включает в себя разработку пользовательского интерфейса и пользовательского опыта (UI/UX), а также внедрение API, которое было разработано на предыдущем этапе.
- После разработки сайта и API, важным этапом является тестирование и отладка всех функций и функциональных возможностей, чтобы убедиться, что они работают корректно и без ошибок.

— После успешного тестирования и отладки, сайт и API могут быть развернуты для общего доступа. Однако это не конец процесса. Важно также предусмотреть надлежащую поддержку и обслуживание продукта после его внедрения, чтобы обеспечить его надежную и эффективную работу на протяжении всего периода эксплуатации.

Все эти этапы должны быть осуществлены с учетом потребностей пользователя, чтобы обеспечить высокую степень удовлетворенности и удобства использования сайта.

2 Анализ предметной области

2.1 Терминология (глоссарий) предметной области

Веб-приложение — клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера.

Backend — логика работы сайта, внутренняя часть продукта, которая находится на сервере и скрыта от пользователя.

Frontend — презентационная часть информационной или программной системы, ее пользовательский интерфейс и связанные с ним компоненты.

REST (Representational state transfer) API – это архитектурный стиль, который определяет правила обмена данными между клиентом и сервером, основан на протоколе HTTP. API представляет собой набор определений и протоколов для построения и интеграции программного обеспечения. API определяет, как должны взаимодействовать компоненты программного обеспечения между собой. Он позволяет различным программным продуктам и сервисам взаимодействовать друг с другом без необходимости знать, как, например, работает та или иная часть программного кода.

2.2 Диаграммы, иллюстрирующие работу системы

Диаграмма прецедентов (Рисунок 1) представляет собой графическое представление того, кто может взаимодействовать с системой и какие действия они могут выполнять. Она позволяет четко и наглядно определить область ответственности системы, видеть, какие функции она может выполнять, и кто будет с ней взаимодействовать. Это помогает в процессе проектирования и анализа системы, позволяет лучше понять потребности пользователей и требования к функциональности.

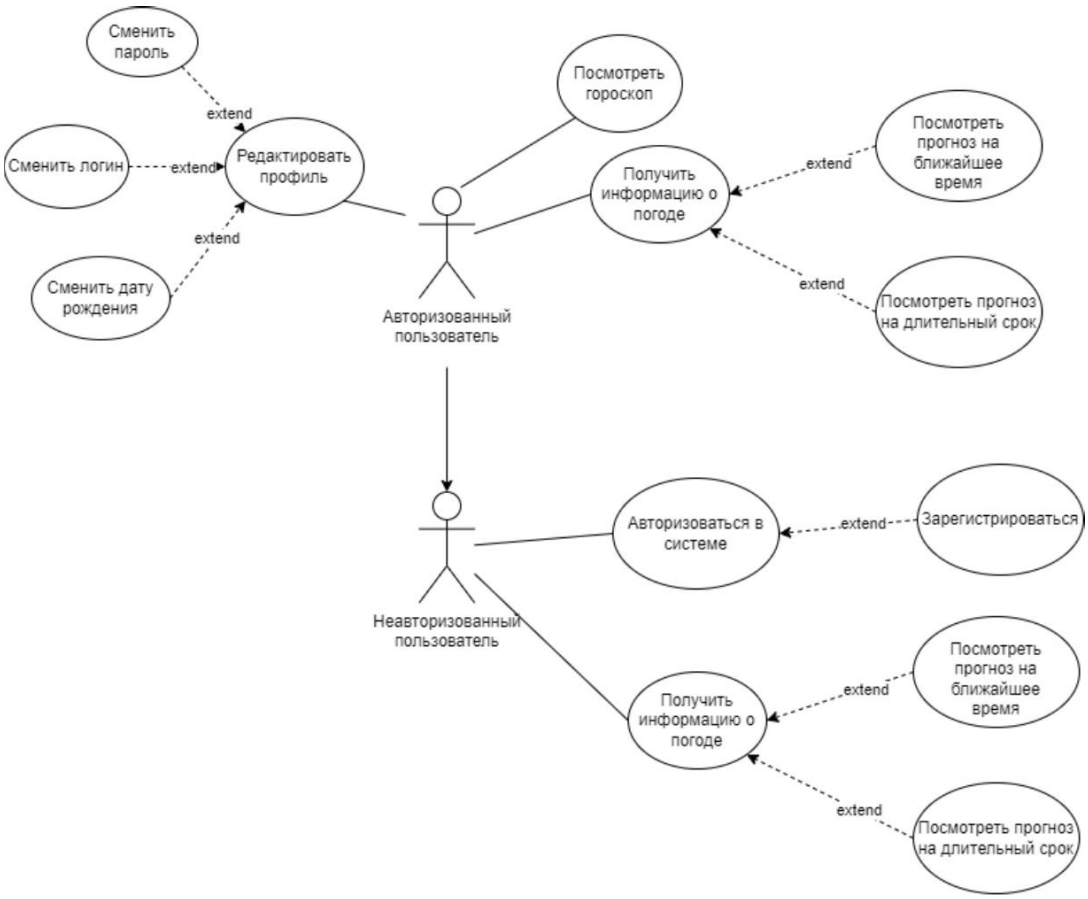

Рисунок 1 - Диаграмма прецедентов

Авторизованный пользователь может:

— Редактировать профиль

— Сменить пароль

— Сменить логин

— Сменить дату рождения

— Смотреть гороскоп

— Получить информацию о погоде

— Посмотреть прогноз на ближайшее время

— Посмотреть прогноз на длительный срок

Неавторизованный пользователь может:

— Зарегистрироваться

— Авторизоваться в системе

— Получить информацию о погоде

— Посмотреть прогноз на ближайшее время

— Посмотреть прогноз на длительный срок

Диаграмма объектов (Рисунок 2) представляет собой статическую структуру системы в конкретный момент времени и показывает экземпляры классов (объекты), их внутреннее состояние и отношения между ними. Диаграмма объектов позволяет наглядно представить реальные экземпляры системы и их взаимосвязи, что может быть полезно при анализе и отладке конкретных сценариев работы системы.

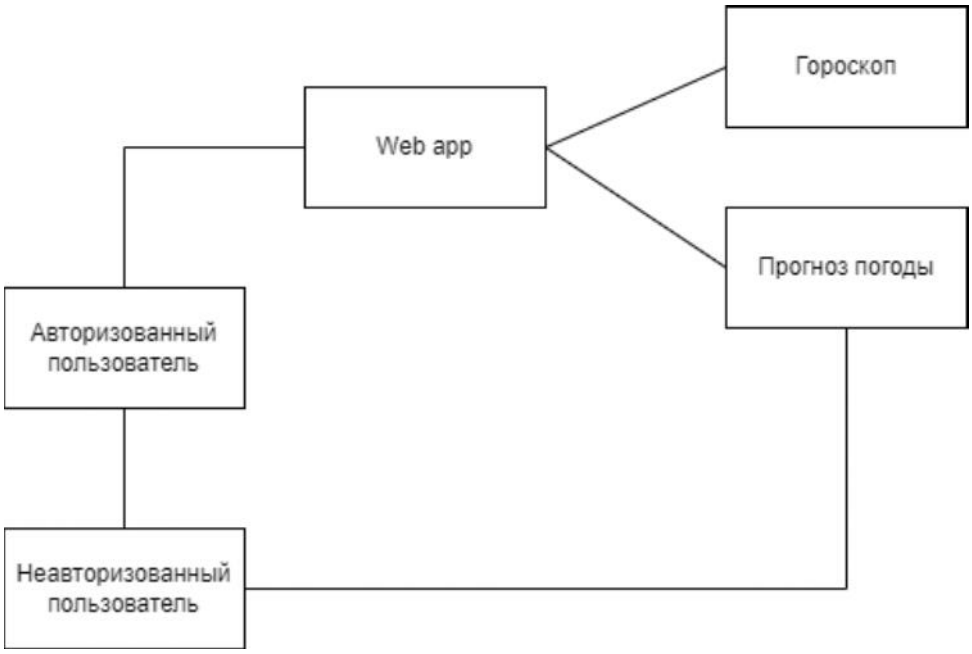

Рисунок 2 - Диаграмма объектов

Неавторизованный пользователь может лишь смотреть прогноз погоды, в то время как авторизованный имеет доступ и к просмотру прогноза, и гороскопа.

Диаграмма активности - это тип диаграммы в методологии UML (Unified Modeling Language), которая предназначена для визуализации последовательности действий в рамках определенного процесса или рабочего потока. Она отображает, как работа переходит от одного действия к другому в системе, и какие условия управляют этими переходами.

Диаграмма активности (Рисунок 3) полезна в случаях, когда важно понимание порядка и условий выполнения действий. Она может быть использована для анализа существующих процессов, для проектирования новых процессов, а также для общения и документирования этих процессов в команде или организации.

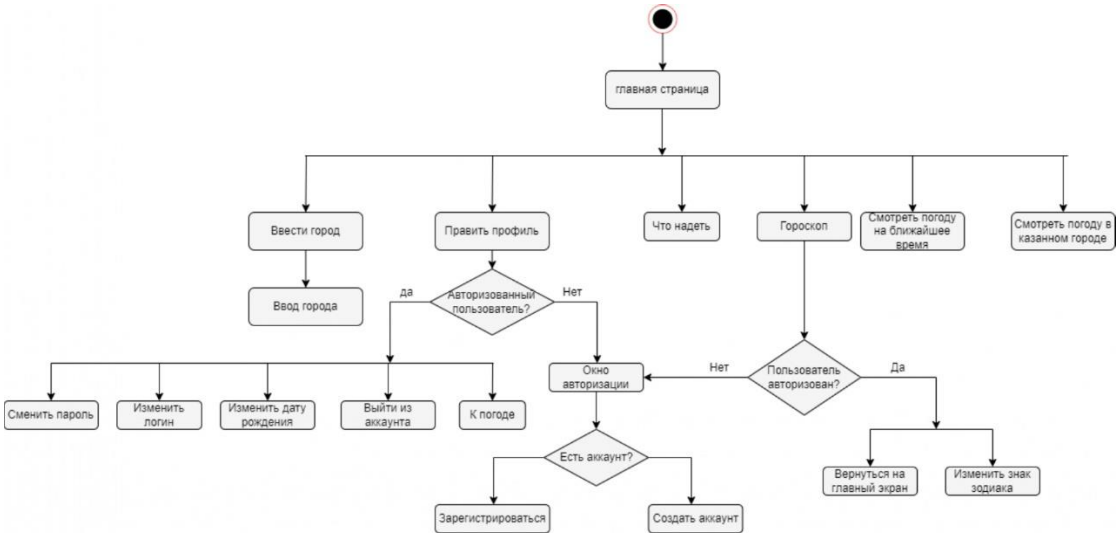

Рисунок 3 - Диаграмма активности

На диаграмме отображено, что при входе пользователь имеет возможность ввести город для просмотра прогноза погоды, править профиль, посмотреть рекомендацию по одежде, посмотреть гороскоп, посмотреть погоду на ближайшее время.

2.3 Концептуальная модель

Концептуальная модель сайта для прогноза погоды, гороскопов и рекомендаций по одежде включает несколько важных аспектов взаимодействия между пользователем и сайтом.

На сайте существуют два типа пользователей - авторизованные и неавторизованные. Авторизованные пользователи проходят процесс регистрации или входа в систему с помощью имеющихся учетных данных, что позволяет им получить доступ к большему количеству функций. Неавторизованные пользователи, или гости, могут просматривать некоторые части сайта без необходимости регистрации или входа в систему.

Самой основной функцией сайта является предоставление прогноза погоды. Это важная информация, доступная для всех посетителей сайта, независимо от того, авторизованы они или нет. Пользователи могут просматривать текущую погоду, погоду на ближайшие дни или даже долгосрочный прогноз.

Кроме того, сайт предлагает рекомендации по одежде, основанные на прогнозе погоды. Эти советы помогают пользователям определить, как лучше одеться в зависимости от текущих или предсказанных погодных условий. Эта информация также доступна всем пользователям сайта.

Для авторизованных пользователей на сайте доступна дополнительная функция - просмотр гороскопа. Пользователи могут перейти на отдельную страницу, чтобы увидеть свой дневной, недельный или месячный гороскоп. Эта функция доступна только после входа в систему, поскольку требуется некоторая персонализация в зависимости от знака зодиака пользователя.

2.4 Выводы

В целом, данная теоретическая часть курсовой работы представляет обзор основных аспектов создания сайта с прогнозом погоды. Были рассмотрены различные аспекты, начиная от выбора источника данных о погоде и методов их получения, до функциональности самого веб-сайта.

В процессе исследования было выяснено, что существует множество источников данных о погоде, таких как государственные метеорологические службы, коммерческие API и общедоступные базы данных. Выбор оптимального источника зависит от конкретных требований и целей проекта.

Функциональность сайта с прогнозом погоды может варьироваться от простого отображения текущей погоды до более сложных функций, таких как прогноз на несколько дней, интерактивные карты и уведомления о погодных условиях. Важно учитывать потребности пользователей и предоставлять им информацию о погоде в удобном и понятном формате.

В заключение, создание сайта с прогнозом погоды требует учета различных факторов, включая выбор источника данных, методы их получения, архитектуру сайта и функциональность. Тщательное планирование и разработка являются ключевыми аспектами успешного проекта. Сайт с прогнозом погоды может быть полезным инструментом для пользователей, предоставляющим им актуальную информацию о погоде и помогая им

3 Реализация

3.1 Используемые инструменты и средства разработки

Для создания сайта с прогнозом погоды были использованы различные средства и технологии, включая HTML, JavaScript (JS), Node.js, axios и express:

— HTML (HyperText Markup Language): HTML является основным языком разметки, используемым для создания структуры и содержимого веб-страницы. С помощью HTML можно создавать элементы, такие как заголовки, параграфы, списки, таблицы и другие компоненты, необходимые для оформления сайта.

— JavaScript (JS): JS является языком программирования, который используется для добавления интерактивности и динамического поведения на веб-страницах. С помощью JS можно обрабатывать события, выполнять асинхронные запросы к серверу, изменять содержимое страницы и многое другое. В контексте сайта с прогнозом погоды, JS может быть использован для отправки запросов к погодным API (Application Programming Interface) и обработки полученных данных.

— Bootstrap - это популярный фреймворк для разработки веб-приложений и создания пользовательского интерфейса (UI). Он предоставляет готовые CSS-стили, JavaScript-компоненты и инструменты для создания стильных и отзывчивых веб-страниц. Bootstrap значительно упрощает и ускоряет процесс разработки веб-страниц, позволяя создавать привлекательные и отзывчивые пользовательские интерфейсы с минимальным количеством кода. Он широко используется в индустрии и имеет большое сообщество разработчиков, что обеспечивает поддержку, обновления и расширение фреймворка.

— Node.js: Node.js является средой выполнения JavaScript, которая позволяет запускать JS-код на стороне сервера. Он предоставляет мощные инструменты для создания веб-приложений, включая

работу с сетью, файловой системой и другими системными ресурсами. В контексте создания сайта с прогнозом погоды, Node.js может использоваться для создания серверной части приложения, обработки запросов и отправки данных на клиентскую сторону.

— Axios: Axios является библиотекой JavaScript, которая предоставляет удобный интерфейс для выполнения HTTP-запросов из браузера или Node.js. Она позволяет отправлять асинхронные запросы к серверу и обрабатывать полученные данные. В контексте сайта с прогнозом погоды, axios может использоваться для отправки запросов к погодным API и получения прогнозов погоды.

— Express: Express.js является минималистичным и гибким фреймворком для создания веб-приложений на Node.js. Он предоставляет простой и удобный способ определения маршрутов, обработки запросов и создания API. В контексте сайта с прогнозом погоды, Express может использоваться для создания сервера, обработки маршрутов и отправки данных клиенту.

— Сочетание этих средств и технологий позволяет создавать сайты с прогнозом погоды, которые могут получать данные о погоде с погодных сервисов через API, обрабатывать эти данные и отображать их пользователю в удобном формате.

— Webstorm: IDE для разработки веб-приложений.

3.2 Реализация backend

В данном проекте backend отвечает за обработку запросов, взаимодействие с погодным API и парсинг XML страницы для получения гороскопов, а также предоставление этих данных фронтенду.

Backend разработка использует Node.js, который является платформой, позволяющей запускать JavaScript на стороне сервера. Node.js обладает

мощными инструментами и библиотеками для создания серверных приложений. В данном случае, Node.js используется для создания сервера на котором будет работать backend.

Для обработки запросов и создания API сервера на Node.js используется фреймворк Express.js. Express.js предоставляет простой и удобный способ определения маршрутов, обработки запросов и создания API. Он позволяет настроить обработку HTTP-запросов, определить маршруты для получения погодных данных и гороскопов, и передать эти данные обратно клиенту.

Для получения погодных данных используется погодное API, такое как Weather API. Этот API предоставляют актуальную информацию о погоде на основе географических координат или названия города. С помощью backend разработки, можно отправлять запросы к погодному API и получать ответы с данными о погоде. Затем эти данные могут быть обработаны и переданы на фронтенд.

Для получения гороскопов, сервер на Node.js может выполнять парсинг XML страницы, содержащей информацию о гороскопах. Для этого используются библиотеки парсинга XML, такая как xml2js. С помощью этой библиотеки можно извлекать необходимую информацию из XML и передавать её обратно на клиентскую часть сайта.

В результате, backend разработка в проекте "Создание сайта с прогнозом погоды" с функцией просмотра гороскопа и рекомендуемой одежды на сервере Node.js позволяет обработать запросы, получить погодные данные с использованием погодного API, парсить XML страницу для получения гороскопов, и предоставить эти данные клиентской части сайта.

Описание основных функций сервера:

— `app.get('/')`: Эта функция обрабатывает GET-запрос на главную страницу. Она получает знак зодиака из запроса и отправляет запрос к внешнему API для получения гороскопа в

формате XML. Затем происходит парсинг полученного XML и отображение гороскопа на странице.

— `app.get('/api/horoscope/:zodiacSign')`: Эта функция обрабатывает GET-запрос для получения гороскопа через API. Она получает знак зодиака из параметров запроса и отправляет запрос к внешнему API для получения гороскопа в формате XML. Затем происходит парсинг полученного XML и возвращение гороскопа в формате JSON.

— `app.get('/save')`: Эта функция обрабатывает GET-запрос для сохранения гороскопа в базе данных. Она получает знак зодиака из запроса и отправляет запрос к внешнему API для получения гороскопа в формате XML. Затем происходит парсинг полученного XML и сохранение гороскопа в базе данных MongoDB с использованием модели `Horoscope`. После успешного сохранения гороскопа, сервер возвращает JSON-ответ с сообщением об успешном сохранении.

— `app.use(express.static(__dirname + '/views'))`: Эта функция указывает серверу использовать статические файлы, расположенные в папке `views`, включая файлы CSS, JavaScript и изображения.

— `app.use('/scripts', express.static(__dirname + '/scripts'))`: Эта функция указывает серверу использовать статические файлы, расположенные в папке `scripts`, которые могут быть связаны с клиентской частью сайта.

— `app.use('/images', express.static(__dirname + '/images'))`: Эта функция указывает серверу использовать статические файлы, расположенные в папке `images`, которые могут быть использованы для отображения изображений на сайте.

— `app.listen(3000)`: Эта функция запускает сервер на порту 3000 и выводит сообщение в консоль о том, что сервер прослушивает соответствующий порт.

Вышеприведенные функции обеспечивают обработку запросов, взаимодействие с внешним API для получения гороскопа в формате XML, парсинг XML, сохранение гороскопа в базе данных и предоставление статических файлов на сервере Node.js.

3.3 Реализация frontend

Frontend разработка включает создание пользовательского интерфейса (UI) для главной страницы, где присутствуют различные окна и функциональность, такие как ввод города, отображение погоды, почасовой прогноз, погода в других городах и рекомендации по одежде. Также реализуется функциональность отображения гороскопа при нажатии на соответствующую кнопку (Рисунок).

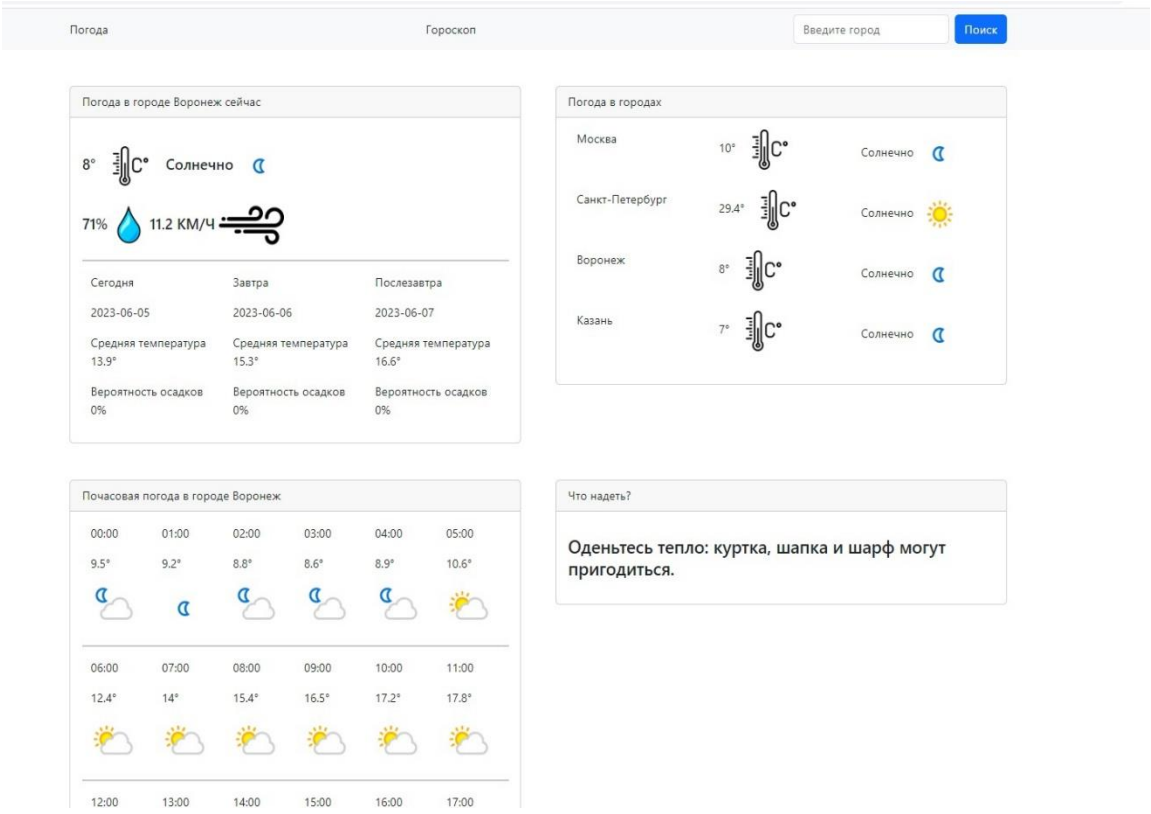

Рисунок 4 - Главная страница

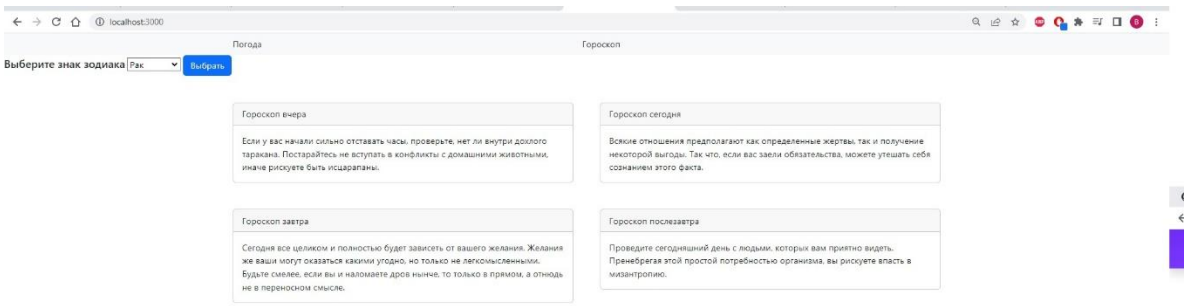

Рисунок 5 - Страница с гороскопом

Окно для ввода города: В этом окне пользователь может ввести название города, для которого хочет получить погоду. Реализуется с помощью текстового поля и кнопки "Поиск". Пользователь может ввести город и нажать на кнопку "Поиск", чтобы получить погоду для выбранного города.

Погода сейчас: В этом окне отображается текущая погода для выбранного города. Информация о погоде включает данные, такие как температура, влажность, скорость ветра, иконка погоды и описание текущих условий. Эти данные получаются с помощью запроса к серверу, который обрабатывает запросы погодного API и возвращает соответствующую информацию.

Почасовая погода: это окно отображает прогноз погоды на ближайшие несколько часов. Он представлен в виде таблицы или графика, где указаны временные интервалы и соответствующие данные о погоде, такие как температура, осадки, ветер и другие параметры.

Погода в других городах: в этом окне отображается погода для других городов. Можно предоставить список популярных городов или дать возможность пользователю выбрать свои города. При выборе города из списка, отображается информация о погоде в выбранном городе.

"Что надеть" с рекомендациями по одежде: В этом окне предоставляются рекомендации по одежде, основанные на текущей погоде. Например, если температура низкая, могут быть рекомендации о ношении теплой одежды, шапки и перчаток. Если погода солнечная и теплая, могут быть рекомендации о ношении легкой одежды и солнцезащитных очков.

Кнопка "Гороскоп": При нажатии на эту кнопку открывается окно с гороскопом. Гороскоп отображается на сегодняшний день, вчера, завтра и послезавтра. Информация о гороскопе может быть получена из внутреннего API сервера, который обрабатывает запросы и возвращает соответствующие данные.

Для реализации frontend разработки используются языки HTML, JavaScript и bootstrap. Bootstrap значительно упрощает и ускоряет процесс разработки веб-страниц, позволяя создавать привлекательные и отзывчивые пользовательские интерфейсы с минимальным количеством кода. HTML используется для создания структуры и разметки страницы. Bootstrap используется для стилизации элементов и создания привлекательного пользовательского интерфейса. JavaScript используется для взаимодействия с пользователем, обработки событий, отправки запросов на сервер и обновления информации на странице в реальном времени.

Все окна и функциональность на главной странице создаются с использованием соответствующих HTML-элементов, Bootstrap и JavaScript-кода. Полученные данные могут быть обработаны и отображены на странице с помощью JavaScript.

Таким образом, frontend разработка в проекте "Создание сайта с прогнозом погоды" включает создание интерактивного и привлекательного пользовательского интерфейса с различными окнами и функциональностью, такими как ввод города, отображение погоды, почасовой прогноз, погода в других городах и рекомендации по одежде. Отображение гороскопа осуществляется при нажатии на кнопку "Гороскоп".

HTML-код является структурой сайта прогноза погоды и обеспечивает интерфейс пользователя. Рассмотрим некоторые ключевые программные решения, реализованные в этом коде:

- Doctype и тег <html>: Задает тип документа и определяет корневой элемент страницы.
- <head>: Секция, содержащая метаинформацию и ссылки на внешние ресурсы, такие как CSS-стили и JavaScript-скрипты.
- <meta charset="utf-8">: Определяет кодировку документа как UTF-8.
- <meta name="viewport" content="width=device-width, initial-scale=1">: Определяет настройки просмотра для мобильных устройств.
- <title>WeatherProject</title>: Задает заголовок страницы, который отображается во вкладке браузера.
- Подключение стилей и скриптов: В данном коде подключены CSS-стили и JavaScript-скрипты Bootstrap и jQuery.
- <body>: Определяет основное содержимое страницы.
- <nav class="navbar sticky-top bg-body-tertiary">: Создает навигационную панель с классом Bootstrap "navbar".
- : Создает контейнер для размещения элементов страницы.
- Ссылки <a>: Создают ссылки для перехода на страницы "Погода" и "Гороскоп".
- <form class="d-flex">: Создает форму для ввода города с классом Bootstrap "d-flex".
- <input>: Создает поле ввода для города.
- <button>: Создает кнопку отправки формы.

— Комментарий `<!-- <a class="nav-link" href="#">Личный Кабинет</a>-->`: Закомментированная ссылка на личный кабинет.

— `<div class="container my-5">`: Создает контейнер для размещения остального содержимого страницы.

— `<div class="row row-cols 1 row-cols-md-1 row-cols-lg-2 g-5">`: Создает строку с гибкой сеткой, содержащей 2 колонки.

— `<div id="firstCol" class="col">...</div>`: Создает колонку с уникальным идентификатором и соответствующим содержимым.

— `<script src="../scripts/conditions.js" type="module"></script>`: Подключает JavaScript-скрипт для работы с условиями погоды.

— `<script src="../scripts/main.js" type="module"></script>`: Подключает основной JavaScript-скрипт для работы с функциональностью сайта.

Эти элементы обеспечивают структуру и функциональность сайта с прогнозом погоды.

Рассмотрим некоторые функции из main.js:

"RemoveCard", удаляет карточки или элементы с определенными идентификаторами из HTML-документа, если они существуют. В частности, она удаляет элементы с идентификаторами "firstCard", "thirdCard" и "fourthCard". Это может быть полезно для обновления или очистки интерфейса перед добавлением новых элементов или информации.

Функция “showError’’ добавляет HTML-код на страницу при возникновении ошибки. Это происходит путем вставки HTML-строки в первый столбец (firstCol), предположительно на веб-странице.

Этот код представляет функцию showCard, которая принимает два аргумента: data и info. Внутри функции создается HTML-разметка, которая будет добавлена в определенный элемент на веб-странице.

HTML-разметка создает блок с классом "card", который содержит заголовок и тело карточки. Заголовок содержит текст "Погода в городе {data.location.name} сейчас", где {data.location.name} будет заменено на фактическое имя города из переданного объекта data.

Тело карточки содержит несколько параграфов с информацией о погоде. Здесь используются данные из объекта data. Например, data.current.temp_c представляет текущую температуру в градусах Цельсия, info.languages[23]['day_text'] представляет текстовое описание погоды, и data.current.condition.icon представляет URL-адрес иконки погодных условий.

Функция “getWeather(city)”. Это асинхронная функция, которая принимает city в качестве параметра. Она создает URL-адрес для запроса погодных данных, используя переданный город и глобальную переменную apiKey. Затем функция выполняет запрос с помощью fetch и ожидает ответа с помощью оператора await. После получения ответа, функция возвращает распарсенный JSON-объект с данными о погоде.

Функция logicCard(city). Это также асинхронная функция, которая принимает city в качестве параметра. Она вызывает getWeather(city) для получения данных о погоде для заданного города. Затем функция проверяет наличие ошибки в полученных данных (data.error). Если ошибка присутствует, функция вызывает removeCard() для удаления существующей карточки (если такая имеется) и showError() для отображения сообщения об ошибке.

Если ошибки нет, функция ищет информацию о погодных условиях для текущего состояния (data.current.condition.code) с помощью conditions.find(). Затем функция вызывает removeCard() для удаления существующей карточки (если такая имеется), showCard(data, info) для отображения карточки с данными о погоде.

Заключение

В ходе разработки сайта с прогнозом погоды были успешно реализованы основные функциональности, такие как получение информации о погоде через API, интеграция гороскопа и предоставление рекомендаций по одежде.

Для получения данных о погоде было использовано API, что позволило получать актуальные данные о погоде в режиме реального времени. С использованием асинхронных запросов, библиотеки axios и языка JavaScript, сайт мог динамически отображать текущую погоду, почасовой прогноз и прогноз погоды в других городах.

Также на сайте была реализована функция просмотра гороскопа. Для получения данных о гороскопе была использована техника парсинга XML. С помощью библиотеки xml2js и языка JavaScript, сайт мог получать и отображать гороскоп на сегодня, вчера, завтра и послезавтра.

Окно с рекомендациями по одежде дополняло функциональности сайта, предоставляя пользователю информацию о том, какой вид одежды рекомендуется носить в соответствии с погодными условиями. Стилизация элементов интерфейса с использованием Bootstrap позволяла создавать привлекательный и отзывчивый внешний вид сайта.

В результате разработки сайта с прогнозом погоды были достигнуты поставленные цели и выполнены основные требования к функциональности и дизайну. Созданный сайт предоставляет пользователю актуальную информацию о погоде, гороскопе и рекомендациях по одежде, делая его полезным и удобным инструментом для планирования активностей на каждый день.

Разработка данного сайта с прогнозом погоды позволила углубить знания в области веб-разработки, освоить работу с внешними API и техниками парсинга данных, а также ознакомиться с применением фреймворка Bootstrap для создания стильного и отзывчивого пользовательского интерфейса.

В целом, создание сайта с прогнозом погоды оказалось интересным и практически значимым проектом, демонстрирующим применение знаний веб-разработки для решения реальных задач и создания полезных сервисов для пользователей.

Список использованных источников

1. Crockford, D. (2008). JavaScript: The Good Parts. O'Reilly Media.
2. Duckett, J. (2014). JavaScript and JQuery: Interactive Front-End Web Development. Wiley.
3. Flanagan, D. (2011). JavaScript: The Definitive Guide: Activate Your Web Pages. O'Reilly Media.
4. Freeman, E., Robson, E. (2014). Head First HTML and CSS: A Learner's Guide to Creating Standards-Based Web Pages. O'Reilly Media.
5. Freeman, E., Robson, E. (2014). Head First JavaScript Programming: A Brain-Friendly Guide. O'Reilly Media.
6. Grannel, C. (2006). The Essential Guide to CSS and HTML Web Design. Apress.
7. Hogan, B. (2014). HTML5 and CSS3: Level Up with Today’s Web Technologies. Pragmatic Bookshelf.
8. Powell, T. A. (2010). HTML & CSS: The Complete Reference. McGraw-Hill.
9. WeatherAPI Documentation. (2023). Retrieved May 25, 2023, from https://www.weatherapi.com/docs/
10. W3Schools Online Web Tutorials. (2023). Retrieved May 25, 2023, from https://www.w3schools.com/
11. Mozilla Developer Network. (2023). Retrieved May 25, 2023, from https://developer.mozilla.org/
12. Eloquent JavaScript: A Modern Introduction to Programming. Haverbeke, M. (2018). No Starch Press.