Министерство науки и высшего образования Российской Федерации

Муромский институт (филиал)

Федерального государственного бюджетного образовательного учреждения высшего образования

«Владимирский государственный университет   
имени Александра Григорьевича и Николая Григорьевича Столетовых»

Факультет СПО

Кафедра ПИн

Производственная практика

Руководитель

Мортин К.В.

(фамилия, инициалы)

(подпись) (дата)

Студент ПКС - 120

(группа)

Точилин И.А.

(фамилия, инициалы)

(подпись) (дата)

Муром 2023

**Front-end**

Разработка интерфейса – это разработка визуальных и интерактивных элементов веб сайта. Это комбинация HTML, CSS и JavaScript, где HTML обеспечивает структуру, CSS – стиль и верстку, а JavaScript – динамическое поведение и интерактивность.

Front-end разработчики отвечают за создание пользовательского интерфейса веб-сайта, за то, чтобы он хорошо выглядел и был прост в использовании, уделяя особое внимание принципам дизайна и пользовательскому опыту. Они тесно сотрудничают с дизайнерами, серверными разработчикам и менеджерами проектов, чтобы убедится, что конечный продукт соответствует потребностям клиента и обеспечивает наилучший возможный опыт для конечных пользователей.

**Интернет**

Интернет – это глобальная сеть компьютеров, соединенных друг с другом, которые обмениваются данными с помощью стандартизированного набора протоколов.

**Что такое HTTP?**

HTTP – это TCP/IP основанный на протоколе связи прикладного уровня, который стандартизирует способ взаимодействия клиента и сервера друг с другом. HTTP следует классической “клиент-северной модели”, когда клиент открывает запрос на подключение, а затем ожидает получения ответа. HTTP – это протокол без состояния, это означает, что сервер не хранит никаких данных между двумя запросами.

**Браузеры**

Веб-браузеры – это программное приложение, которое позволяет пользователю получать доступ к веб-страницам или другому онлайновому контенту и отображать их через свой графический пользовательский интерфейс.

**DNS**

Система доменных имен (DNS) – это телефонная книга Интернета. Люди получают доступ к информации онлайн через доменные имена, такие как nytimes.com или espn.com. Веб-браузеры взаимодействуют через адреса интернет-протокола(IP). DNS преобразует доменные имена в IP-адреса, чтобы браузеры могли загружать интернет-ресурсы.

**Доменное имя**

Доменное имя – это уникальный, легко запоминающийся адрес, используемый для доступа к веб-сайтам, таким как ‘google.com’ и ‘facebook.com’. Пользователи могут подключится к веб-сайту, используя доменные имена, благодаря системе доменных именных имен (DNS).

**Хостинг**

Веб-хостинг – это онлайн-сервис, который позволяет вам публиковать файлы вашего веб-сайта в Интернете. Таким образом, любой, у кого есть доступ к Интернету, имеет доступ к вашему веб-сайту.

**HTML**

HTML расшифровывается как HyperText Markup Language. Он используется во внешнем интерфейсе и придает структуру веб-странице, которую вы можете стилизовать с помощью CSS и сделать интерактивной с помощью JavaScript

**Семантический HTML**

Семантический элемент четко описывает его значение как для браузера, так и для разработчика. В HTML семантический элемент – это тип элементов, которые могут использовать для определения различных частей вкб-страницы, таких как <from>, <table>, <article>, <header> <footer> … и т.д.

**Формы и проверки**

Перед отправкой данных на сервер важно убедиться, что все необходимые элементы управления формой заполнены в правильной формате. Это называется проверкой формы на стороне клиента и помогает убедится, что преодоление данные соответствуют требованиям, изложенным в различных элементах управления формой.

**Лучшие практики**

Научится следовать лучшим практикам написания поддерживаемых и масштабируемых HTML-документов.

**Специальные возможности**

Доступность веб-сайтов означает, что веб-сайты, инструменты и технологии спроектированы таким образом, чтобы люди с ограниченными возможностями могли легко ими пользоваться.

**Основы SEO**

SEO или поисковая оптимизация – это метод, используемый для оптимизации вашего сайта для повышения рейтинга в поисковых системах, таких как Google, Bing и т.д.

**CSS**

CSS или каскадные таблицы стилей это язык, используемый для оформления итерфейса любого веб-сайта. CSS является краеугольной технологией всемирной паутины, наряду с HTML и JavaScript.

**Создание макетов**

Float, grid, flexbox, позиционирование, отображение и коробчатая модель – вот некоторые из ключевых тем, которые используются для создания макетов.

**Адаптивный веб-дизайн**

Адаптивный веб-дизайн – это метод, позволяющий сделать ваши веб-страницы хорошо выглядящими на экранах любого размера. Для достижения этого используются определенные методы, например, медиазапросы CSS, процентная ширина, минимальная или максимальная ширина и т.д.

**JfvaScript**

JfvaScript позволяет вам добавлять интерактивность на ваши страницы. Распространенными примерами, которые вы возможно, видели на веб-сайтах, являются ползунки, взаимодействие по нажатию, всплывающие окна и так далее.

**Манипуляции с DOM**

Объектная модель документа (DOM) – Это программный интерфейс, созданный для документов HTML и XML. Он представляет страницу, которая позволяет программам и скриптам динамически обновлять структуру, содержимое и стиль документа. С помощью DOM мы можем легко получать доступ к тегам, идентификаторам, классам, атрибутам и т.д. И манипулировать ими.

**Fetch API**

Ajax – это технология, которая позволяет нам асинхронно отправлять и получать данные с серверов, например, обновлять профиль пользователя или асинхронно извлекать список искомых продуктов без перезагрузки страницы.

**Современный Java Script**

ECMAScript 2015 или ES2015 – это значительно обновление языка программирования JavaScript. Это первое крупное обновление языка после ES5, который был стандартезирован в 2009 году. Вам следует взглянуть на функции, представленные в ES6 и последующих версиях.

**Концепции JavaScript**

Изучить и понять такие концепции, как подъем, создание пузырьков событий, области видимости, прототип, Shadow DOM и strict.

**Системы контроля версий**

Системы контроля версий позволяют вам отслеживать изменения в вашей кодовой базе / файлах с течением времени. Они позволяют вам вернуться к некоторой предыдущей версии кодовой базы без каких-либо проблем. Кроме того, они помогают в сотрудничестве с людьми, работающими над одним и тем же кодом – если вы когда-либо сотрудничали с другими людьми в проекте, вы, возможно, уже знакомы с разочарованием от копирования и объединения изменений от кого-то другого в вашу кодовую базу; системы контроля версий позволяют вам избавиться от этой проблемы.

**Услуги репо-хостинга**

Существуют различные сервисы размещения репозиториев, наиболее известными из которых являются GitHub, GitLab и BitBucket. Я бы рекомендовал создать учетную запись на GitHub, потому что именно там выполняется большая часть работы с открытым исходным кодом и работают большинство разработчиков.

**GitHub**

[GitHub](https://github.com/) - поставщик интернет-хостинга для разработки программного обеспечения и контроля версий с использованием Git. Он предлагает функции распределенного контроля версий и управления исходным кодом Git, а также свои собственные функции.

**GitLab**

[GitLab](https://gitlab.com/) - поставщик интернет-хостинга для разработки программного обеспечения и контроля версий с использованием Git. Он предлагает функции распределенного контроля версий и управления исходным кодом Git, а также свои собственные функции.

**BitBucket**

[BitBucket](https://bitbucket.com/) - поставщик интернет-хостинга для разработки программного обеспечения и контроля версий с использованием Git. Он предлагает функции распределенного контроля версий и управления исходным кодом Git, а также свои собственные функции.

**Git**

[Git](https://git-scm.com/) - это бесплатная распределенная система контроля версий с открытым исходным кодом, предназначенная для быстрой и эффективной работы с любыми проектами, от небольших до очень крупных.

**HTTPS**

**HTTPS** - это безопасный способ передачи данных между веб-сервером и браузером.

Защищенный протокол передачи гипертекста (HTTPS) - это защищенная версия HTTP, который является основным протоколом, используемым для передачи данных между веб-браузером и веб-сайтом. HTTPS зашифрован для повышения безопасности передачи данных. Это особенно важно, когда пользователи передают конфиденциальные данные, например, при входе в банковский счет, службу электронной почты или поставщика медицинского страхования

**CORS**

Совместное использование ресурсов из разных источников (CORS) - это механизм на основе HTTP-заголовка, который позволяет серверу указывать любые источники (домен, схему или порт), отличные от его собственных, с которых браузер должен разрешать загрузку ресурсов.

**Политика безопасности контента**

Политика безопасности контента - это стандарт компьютерной безопасности, введенный для предотвращения межсайтового скриптинга, взлома кликов и других атак с внедрением кода, возникающих в результате выполнения вредоносного контента в контексте доверенной веб-страницы.

**Риски безопасности OWASP**

OWASP или Open Web Application Security Project - это онлайн-сообщество, которое выпускает свободно доступные статьи, методологии, документацию, инструменты и технологии в области безопасности веб-приложений.

**Менеджеры пакетов**

Менеджеры пакетов позволяют вам управлять зависимостями (внешним кодом, написанным вами или кем-то еще), которые необходимы вашему проекту для правильной работы.

**npm**

npm - это менеджер пакетов для языка программирования JavaScript, поддерживаемый npm, Inc. npm - это менеджер пакетов по умолчанию для среды выполнения JavaScript Node.js.

**Пряжа**

Yarn - это система упаковки программного обеспечения, разработанная Facebook в 2016 году для Node.js Среды выполнения JavaScript, которая обеспечивает скорость, согласованность, стабильность и безопасность в качестве альтернативы npm (package manager).

**pnpm**

PNPM - это альтернативный менеджер пакетов для Node. js, что расшифровывается как “Performant NPM”. Основная цель PNPM - хранить все пакеты в глобальном (централизованном) хранилище и использовать их при необходимости и в других проектах, создавая на них жесткие ссылки.

**Архитектура CSS**

Известно, что CSS сложно управлять в больших, сложных системах с быстрыми итерациями. Существуют различные способы написания CSS, которые позволяют писать более удобный CSS.

**BEM**

Методология блоков, элементов, модификаторов (обычно называемая BEM) - популярное соглашение об именовании классов в HTML и CSS. Разработана командой Яндекса, ее цель - помочь разработчикам лучше понять взаимосвязь между HTML и CSS в данном проекте.

**Препроцессоры CSS**

Препроцессоры CSS - это языки сценариев, которые расширяют возможности CSS по умолчанию. Они позволяют нам использовать логику в нашем CSS-коде, такую как переменные, вложенность, наследование, микшины, функции и математические операции.

**Дерзость**

Sass - это язык сценариев с препроцессором, который интерпретируется или компилируется в каскадные таблицы стилей. Она позволяет вам писать поддерживаемый CSS и предоставляет такие функции, как переменные, вложенность, микшины, расширение, функции, циклы, условные обозначения и так далее.

**PostCSS**

PostCSS - это инструмент для преобразования стилей с помощью плагинов JS. Эти плагины могут изменять ваш CSS, поддерживать переменные и микшины, транспилировать будущий синтаксис CSS, встроенные изображения и многое другое.

**Инструменты сборки**

Исполнители задач автоматически выполняют команды и выполняют процессы за кулисами. Это помогает автоматизировать ваш рабочий процесс за счет выполнения рутинных повторяющихся задач, на повторение которых вы в противном случае потратили бы огромное количество времени.

Обычные способы выполнения задач включают в себя многочисленные задачи разработки, такие как: развертывание серверов разработки, компиляция кода (например, SCSS в CSS), запуск линтеров, отправка файлов с локального порта на вашем компьютере и многое другое!

**Линтеры форматеры**

Линтер - это инструмент, используемый для анализа кода и обнаружения ошибок, синтаксических погрешностей, стилистических несоответствий и подозрительных конструкций. Популярные линтеры для JavaScript включают ESLint, JSLint и JSHint.

**Красивее**

Prettier - это самоуверенный программист для форматирования кода с поддержкой JavaScript, HTML, CSS, YAML, Markdown, схем GraphQL. Безусловно, главная причина для принятия Prettier - прекратить все продолжающиеся дебаты о стилях.

**ESLint**

С ESLint вы можете навязать стандарт кодирования, используя определенный набор автономных правил.

**Исполнители задач**

Task Runner - это инструменты для упрощения некоторых утомительных задач разработки, таких как автоматизация компиляции sass / scss, объединение ресурсов, компоновка исходного кода и горячая перезагрузка локального сервера.

**Сценарии npm**

сценарии npm - это записи в поле scripts файла package.json. Поле scripts содержит объект, в котором вы можете указать различные команды и сценарии, которые вы хотите предоставить.

**Пакеты модулей**

Пакет модулей - это инструмент, который берет фрагменты JavaScript и их зависимости и объединяет их в один файл, обычно для использования в браузере. Возможно, вы использовали такие инструменты, как Browserify, Webpack, Rollup или один из многих других.

Обычно она начинается с файла ввода, и оттуда она объединяет весь код, необходимый для этого файла ввода.

**Vite**

Vite - это инструмент для сборки, целью которого является обеспечение более быстрой и компактной разработки для современных веб-проектов.

**esbuild**

Наши текущие инструменты сборки для Интернета работают в 10-100 раз медленнее, чем могли бы быть. Основная цель проекта esbuild bundler - открыть новую эру производительности инструментов сборки и попутно создать простой в использовании современный пакет.

**Webpack**

Webpack - это модульный пакет. Его основная цель - связывать файлы JavaScript для использования в браузере, но он также способен преобразовывать, объединять или упаковывать практически любой ресурс или актив.

**Свертка**

Rollup - это модульный пакет для JavaScript, который компилирует небольшие фрагменты кода во что-то большее и более сложное, такое как библиотека или приложение.

**Посылка**

Parcel - это пакет веб-приложений, отличающийся опытом разработки. Он обеспечивает невероятно высокую производительность с использованием многоядерной обработки и не требует никакой настройки.

**Выберите фреймворк**

Веб-фреймворки предназначены для написания веб-приложений. Фреймворки представляют собой наборы библиотек, которые помогают в разработке программного продукта или веб-сайта. Фреймворки для разработки веб-приложений представляют собой наборы различных инструментов. Фреймворки различаются по своим возможностям и функциям в зависимости от поставленных задач. Они определяют структуру, устанавливают правила и предоставляют необходимые инструменты разработки.

**Реагировать**

React - самая популярная интерфейсная библиотека JavaScript для создания пользовательских интерфейсов. React также может выполнять рендеринг на сервере с помощью Node и запускать мобильные приложения с помощью React Native.

**Angular**

Angular - это основанная на компонентах платформа интерфейсной разработки, построенная на TypeScript, которая включает в себя набор хорошо интегрированных библиотек, включающих такие функции, как маршрутизация, управление формами, взаимодействие клиент-сервер и многое другое.

**Vue.js**

Vue.js это фреймворк JavaScript с открытым исходным кодом для создания пользовательских интерфейсов и одностраничных приложений. В основном он ориентирован на разработку внешнего интерфейса.

**Стройный**

Svelte - это фреймворк javascript, который, в отличие от Vue и React, не использует вертикальное разделение DOM, а вместо этого точно знает, что и где обновлять при изменении состояния. В основном он ориентирован на интерфейс и построение пользовательских интерфейсов.

**SolidJS**

Solid - это реактивный JavaScript-инструментарий для создания пользовательских интерфейсов без виртуального DOM. Чтобы гарантировать, что при обновлении состояния выполняется только соответствующий код, он компилирует шаблоны до реальных узлов DOM один раз и преобразует изменения в детализированные реакции.

**Современный CSS**

Способ, которым мы пишем CSS в наших современных интерфейсных приложениях, полностью отличается от того, как мы писали CSS раньше. Существуют такие методы, как стилизованные компоненты, модули CSS, стилизованный JSX, Emotion и т. Д

**Стилизованные компоненты**

Styled-components - это библиотека CSS в JS, которая позволяет вам писать обычный CSS и присоединять его к компонентам JavaScript. С помощью styled-components вы можете использовать CSS, с которым вы уже знакомы, вместо того, чтобы изучать новую структуру стиля.

**CSS-модули**

CSS-файлы, в которых все имена классов и анимации по умолчанию имеют локальную область видимости.

**Стиль JSX**

Styled JSX - это библиотека CSS в JS, которая позволяет вам писать инкапсулированный CSS с ограниченной областью действия для стилизации ваших компонентов. Стили, которые вы вводите для одного компонента, не влияют на другие компоненты, позволяя вам добавлять, изменять и удалять стили, не беспокоясь о непреднамеренных побочных эффектах.

**Эмоция**

Emotion - это библиотека, предназначенная для написания стилей css с помощью JavaScript. Она обеспечивает мощную и предсказуемую стилевую композицию в дополнение к большому опыту разработчика с такими функциями, как карты исходных текстов, метки и утилиты тестирования. Поддерживаются как строковые, так и объектные стили.

**CSS фреймворки**

CSS framework предоставляет пользователю полностью функциональную таблицу стилей CSS, позволяя им создавать веб-страницы, просто кодируя HTML с соответствующими классами, структурой и идентификаторами. Классы для популярных функций веб-сайта, таких как нижний колонтитул, слайдер, панель навигации, меню гамбургеров, макеты на основе столбцов и так Далее, Уже включены в фреймворк.

**Мантин**

Mantine - это библиотека компонентов React, содержащая более 100 настраиваемых компонентов и 40 хуков, которые пригодятся вам в любой ситуации.

**Tailwind CSS**

CSS Framework, предоставляющий атомарные классы CSS, помогающие вам стилизовать компоненты, например, **flex**, **pt-4** **text-center** и **rotate-90**, которые могут быть скомпонованы для создания любого дизайна непосредственно в вашей разметке.

**Пользовательский интерфейс Chakra**

Chakra UI - это простая, модульная и доступная библиотека компонентов, которая предоставляет вам строительные блоки, необходимые для создания ваших приложений React.

**Материальный пользовательский интерфейс**

Material-UI - это платформа с открытым исходным кодом, которая включает компоненты React, реализующие Material Design от Google.

**Пользовательский интерфейс Daisy**

Библиотека компонентов на основе Tailwind CSS, которая поставляется с несколькими встроенными компонентами.

**Базовый пользовательский интерфейс**

Библиотека компонентов пользовательского интерфейса с открытым исходным кодом для создания высококачественных, доступных систем проектирования и веб-приложений.

**Bootstrap**

Быстро создавайте и настраивайте адаптивные сайты для мобильных устройств с помощью Bootstrap, самого популярного в мире интерфейсного инструментария с открытым исходным кодом, включающего переменные Sass и mixins, адаптивную сеточную систему, обширные готовые компоненты и мощные плагины JavaScript.

**Bulma**

Bulma - это бесплатный фреймворк с открытым исходным кодом, предоставляющий готовые к использованию компоненты интерфейса, которые вы можете легко комбинировать для создания адаптивных веб-интерфейсов.

**Тестирование ваших приложений**

Прежде чем предоставлять свое приложение пользователям, вы должны быть уверены, что ваше приложение соответствует требованиям, для которых оно было разработано, и что оно не совершает никаких странных, непреднамеренных действий (называемых ‘ошибками’). Для достижения этой цели мы ‘тестируем’ наши приложения различными способами.

**Jest**

Jest - это восхитительный фреймворк для тестирования JavaScript с акцентом на простоту. Он работает с проектами, использующими: Babel, TypeScript, Node, React, Angular, Vue и другими!

**Библиотека тестирования React**

Библиотека тестирования React - это очень легкое решение для тестирования компонентов React. Она предоставляет легкие служебные функции поверх react-dom и react-dom / test-utils таким образом, чтобы стимулировать совершенствование методов тестирования. Ее основной руководящий принцип таков: чем больше ваши тесты напоминают способ использования вашего программного обеспечения, тем больше уверенности они могут вам придать.

**Cypress**

Cypress framework - это платформа комплексного тестирования на основе JavaScript, построенная поверх Mocha – многофункциональной платформы тестирования JavaScript, работающей в браузере и в нем, что делает асинхронное тестирование простым и удобным. Он также использует библиотеку утверждений BDD / TDD и браузер для сопряжения с любой платформой тестирования JavaScript.

**Драматург**

Playwright - это библиотека автоматизации тестирования с открытым исходным кодом, первоначально разработанная авторами Microsoft. Она поддерживает такие языки программирования, как Java, Python, C # и NodeJS. Playwright поставляется с лицензией Apache 2.0 и наиболее популярен среди NodeJS с Javascript / Typescript.

**Стратегии аутентификации**

Стратегии аутентификации - это методы, используемые для проверки личности пользователя или системы с целью предоставления доступа к защищенному ресурсу. Существует несколько различных стратегий аутентификации, которые могут быть использованы, в том числе:

Базовая аутентификация

Аутентификация на основе сеанса

Аутентификация на основе токенов

Аутентификация JWT

OAuth

Единый вход

Вам не обязательно изучать все это, как внедрять и все тонкости с самого начала. Но важно знать, что это такое и как они работают. Это поможет вам принимать более обоснованные решения при выборе стратегии аутентификации для вашего приложения.

**Веб-компоненты**

Веб-компоненты - это набор различных технологий, позволяющих вам создавать пользовательские элементы многократного использования — с их функциональностью, инкапсулированной вдали от остального вашего кода, — и использовать их в ваших веб-приложениях.

**HTML-шаблоны**

**<template>** Элемент HTML - это механизм для хранения HTML, который не должен отображаться сразу при загрузке страницы, но может быть создан впоследствии во время выполнения с использованием JavaScript. Представьте шаблон как фрагмент контента, который сохраняется для последующего использования в документе.

**Пользовательские элементы**

Одной из ключевых особенностей стандарта веб-компонентов является возможность создавать пользовательские элементы, которые инкапсулируют вашу функциональность на HTML-странице, вместо того, чтобы довольствоваться длинным вложенным набором элементов, которые вместе обеспечивают функцию пользовательской страницы.

**Shadow DOM**

Важным аспектом веб-компонентов является инкапсуляция — способность скрывать структуру разметки, стиль и поведение и отделять их от остального кода на странице, чтобы разные части не конфликтовали, а код можно было поддерживать красивым и чистым. API Shadow DOM является ключевой частью этого, предоставляя способ прикрепить скрытый разделенный DOM к элементу.

**Введите средства проверки**

Средство проверки типов помогает разработчикам писать код с меньшим количеством ошибок, добавляя типы в свой код, пытаясь отловить ошибки типов в вашем коде, а затем удаляя их во время компиляции. Flow и TypeScript - два популярных средства проверки статических типов для JavaScript.

**Машинописный текст**

TypeScript - это строго типизированный язык программирования, основанный на JavaScript, предоставляющий вам лучшие инструменты в любом масштабе.

**Прогрессивные веб-приложения**

Прогрессивные веб-приложения (PWAS) - это веб-сайты, которые постепенно совершенствуются, чтобы функционировать как установленные собственные приложения на поддерживающих платформах, функционируя при этом как обычные веб-сайты в других браузерах.

**API веб-хранилища**

Web Storage API предоставляет механизмы для хранения пар ключ-значение в веб-браузере. Она включает в себя два объекта хранения: localStorage и sessionStorage, которые позволяют сохранять данные на стороне клиента и сохранять их в нескольких сеансах браузера соответственно.

API веб-хранилища разработан таким образом, чтобы быть простым и непринужденным в использовании, и он широко поддерживается в современных веб-браузерах. Он часто используется в качестве альтернативы файлам cookie, поскольку позволяет хранить большие объемы данных и является более эффективным с точки зрения производительности.

**Веб-сокеты**

Web Sockets - это технология, которая обеспечивает полнодуплексную связь по одному TCP-соединению. Она обеспечивает двунаправленную связь в режиме реального времени между клиентом и сервером и обычно используется в приложениях, требующих высокоскоростной связи с низкой задержкой, таких как онлайн-игры и потоковая передача данных в режиме реального времени.

Web Sockets использует постоянное соединение между клиентом и сервером, обеспечивающее непрерывный обмен данными без необходимости для клиента отправлять дополнительные запросы на сервер. Это делает его более эффективным и быстрым по сравнению с другими технологиями, такими как HTTP, которые требуют отправки нового запроса для каждой части данных.

Web Sockets поддерживается большинством современных веб-браузеров и может использоваться с различными языками программирования и фреймворками.

**События, отправленные сервером**

События, отправляемые сервером (SSE), - это технология, которая позволяет веб-серверу передавать данные клиенту в режиме реального времени. Он использует HTTP-соединение для отправки потока данных с сервера клиенту, и клиент может прослушивать эти события и предпринимать действия при их получении.

SSE полезен для приложений, которым требуются обновления в режиме реального времени, таких как системы чатов, биржевые тикеры и ленты социальных сетей. Это простой и эффективный способ установить долговременное соединение между клиентом и сервером, и он поддерживается большинством современных веб-браузеров.

Чтобы использовать SSE, клиент должен создать объект EventSource и указать URL серверного скрипта, который будет отправлять события. Затем сервер может отправлять события, записывая их в поток ответов с надлежащим форматированием.

**Работники сферы обслуживания**

Сервисные работники - это тип веб-работников, которые действуют как посредник между веб-страницей и сетью, позволяя веб-разработчикам создавать автономные и надежные приложения. Сервисные работники могут перехватывать сетевые запросы, получать доступ к кэшу и принимать решения о том, как реагировать на запрос, на основе доступных ресурсов.

Сервисные работники написаны на JavaScript и регистрируются на веб-странице. После регистрации они могут управлять страницей и всеми ее запросами, даже если страница не открыта в браузере. Это позволяет сервисным работникам включать такие функции, как push-уведомления, фоновая синхронизация и автономная поддержка.

Сервисные работники поддерживаются большинством современных веб-браузеров, и они являются важным компонентом прогрессивных веб-приложений (PWA).

**Location API**

API геолокации - это веб-API, который предоставляет доступ к данным о местоположении устройства, таким как широта и долгота. Это позволяет веб-разработчикам создавать приложения на основе определения местоположения, такие как картографирование и совместное использование местоположения, используя GPS устройства, Wi-Fi и другие датчики для определения местоположения пользователя.

Чтобы использовать API геолокации, веб-страница должна сначала запросить у пользователя разрешение на доступ к своему местоположению. Если разрешение предоставлено, страница может затем использовать **navigator.geolocation** объект для доступа к данным о местоположении устройства. API предоставляет несколько методов для получения текущего местоположения пользователя, отслеживания изменений местоположения и вычисления расстояний между двумя местоположениями.

**API уведомлений**

API уведомлений - это веб-API, который позволяет веб-страницам отображать уведомления системного уровня для пользователя. Эти уведомления можно использовать для оповещения пользователя о важных событиях, таких как новые сообщения или обновления, даже если веб-страница не открыта в браузере.

Чтобы использовать API уведомлений, веб-страница должна сначала запросить у пользователя разрешение на отображение уведомлений. Если разрешение предоставлено, страница может затем использовать **Notification** конструктор для создания нового уведомления и отображения его пользователю. Уведомление может включать заголовок, основной текст и значок, и его можно настроить с помощью таких параметров, как время ожидания и действие щелчка.

**API ориентации устройства**

API ориентации устройства - это веб-API, который предоставляет доступ к данным об ориентации устройства и его движении, таким как его тангаж, крен и рыскание. Это позволяет веб-разработчикам создавать приложения, которые могут реагировать на ориентацию и движение устройства, такие как дополненная реальность и игры с управлением движением.

Чтобы использовать API ориентации устройства, веб-страница должна сначала запросить у пользователя разрешение на доступ к данным ориентации устройства. Если разрешение предоставлено, страница может затем использовать объект DeviceOrientationEvent для доступа к данным ориентации устройства и реагирования на изменения в ориентации. API предоставляет несколько свойств для доступа к данным об ориентации устройства и движении, а также события для обнаружения изменений в ориентации.

API ориентации устройства поддерживается большинством современных веб-браузеров и часто используется в сочетании с другими API, такими как API геолокации, для создания приложений на основе определения местоположения.

**Платежи**

API платежных запросов - это веб-API, который позволяет веб-разработчикам создавать потоки оформления заказа в своих веб-приложениях. Он предоставляет стандартизированный интерфейс на основе браузера для сбора информации об оплате и доставке от пользователя и поддерживает широкий спектр способов оплаты, включая кредитные карты, дебетовые карты и цифровые кошельки.

Чтобы использовать API запроса платежа, веб-страница должна сначала создать **PaymentRequest** объект и указать доступные пользователю параметры оплаты и доставки. Затем страница может вызвать пользовательский интерфейс запроса платежа, вызвав **show()** метод для **PaymentRequest** объекта. Затем пользователь может выбрать предпочитаемые им варианты оплаты и доставки и подтвердить платеж, после чего API-интерфейс запроса платежа вернет объект ответа на платеж, который можно использовать для завершения транзакции.

API платежных запросов поддерживается большинством современных веб-браузеров и разработан таким образом, чтобы быть простым и эффективным как для разработчика, так и для пользователя.

**API учетных данных**

API управления учетными данными - это веб-API, который позволяет веб-разработчикам интегрировать потоки входа на основе паролей и федеративного входа в свои приложения. Она предоставляет стандартизированный интерфейс на основе браузера для хранения и извлечения учетных данных пользователя, таких как комбинации имени пользователя и пароля и токены OAuth.

Чтобы использовать API управления учетными данными, веб-страница должна сначала создать объект учетных данных и указать учетные данные, которые пользователь желает сохранить. Затем страница может использовать объект navigator.credentials для хранения и извлечения учетных данных пользователя. API предоставляет несколько методов для хранения и извлечения учетных данных, а также для запроса пользователя на вход или регистрацию.

API управления учетными данными поддерживается большинством современных веб-браузеров и предназначен для повышения безопасности и удобства входа в систему, позволяя пользователю хранить и повторно использовать свои учетные данные на нескольких сайтах и устройствах.

**Шаблон PRPL**

Шаблон PRPL - это шаблон оптимизации производительности для веб-приложений, который фокусируется на минимизации времени, необходимого для первоначального рендеринга веб-страницы. Она расшифровывается как Push, Render, Pre-cache и отложенная загрузка и включает в себя следующие шаги:

Продвижение: Расставьте приоритеты в доставке критически важных ресурсов, таких как HTML, CSS и JavaScript, клиенту как можно раньше.

Визуализация: начните визуализацию веб-страницы, как только будут получены критически важные ресурсы, даже если некоторые некритичные ресурсы все еще загружаются.

Предварительное кэширование: предварительное кэширование некритичных ресурсов в фоновом режиме, чтобы они были доступны при необходимости.

Отложенная загрузка: отложите загрузку некритичных ресурсов до тех пор, пока они не понадобятся, например, когда пользователь переходит к ним или взаимодействует с ними.

Шаблон PRPL разработан для повышения воспринимаемой производительности веб-страницы за счет сокращения времени, необходимого для того, чтобы страница стала интерактивной. Это особенно полезно для приложений, которые обслуживаются по медленным или ненадежным сетям, поскольку позволяет странице отображаться как можно быстрее, а затем постепенно загружать оставшиеся ресурсы.

**Модель рельса**

Модель RAIL - это модель оптимизации производительности для веб-приложений, которая фокусируется на повышении воспринимаемой производительности веб-страницы. Она обозначает отклик, анимацию, бездействие и загрузку и включает в себя следующие четыре цели производительности:

Ответ: Время, необходимое веб-странице для ответа на вводимые пользователем данные, должно составлять менее 100 миллисекунд.

Анимация: время, необходимое для запуска анимации, должно составлять менее 10 миллисекунд.

Простой: веб-страница должна использовать время простоя для выполнения некритичных задач.

Загрузка: время, необходимое для полной загрузки веб-страницы, должно составлять менее 1 секунды.

Модель RAIL предназначена для обеспечения основы для оптимизации производительности веб-страницы путем сосредоточения внимания на четырех ключевых областях, которые влияют на восприятие производительности пользователем. Это особенно полезно для приложений, требующих высокого уровня интерактивности, таких как игры и приложения для социальных сетей.

**Показатели производительности**

Показатели веб-производительности - это количественные показатели производительности веб-страницы или приложения. Они используются для оценки скорости и эффективности веб-страницы и могут помочь определить области для улучшения. Некоторые распространенные показатели веб-производительности включают:

Время загрузки: время, необходимое веб-странице для полной загрузки и перехода в интерактивное состояние.

Первое наполнение контентом (FCP): время, необходимое для появления первого контента на странице.

Время перехода к интерактивности (TTI): время, необходимое для того, чтобы страница стала полностью интерактивной.

Задержка первого ввода (FID): время, необходимое странице для ответа на первый пользовательский ввод.

Общее время блокировки (TBT): время, необходимое для того, чтобы страница стала полностью интерактивной, с учетом времени, затраченного на блокировку основного потока.

Для измерения показателей веб-производительности доступно множество инструментов и методик, включая инструменты разработки браузеров, инструменты мониторинга производительности и API веб-производительности. Отслеживая эти показатели и анализируя результаты, веб-разработчики могут определить области для улучшения и оптимизировать производительность своих веб-страниц.

**Маяк**

Lighthouse - это инструмент с открытым исходным кодом, разработанный Google, который используется для аудита производительности, доступности и SEO веб-страниц. Она доступна как расширение для браузера и как инструмент командной строки, и ее можно запустить на любой веб-странице для создания отчета с рекомендациями по улучшению.

Lighthouse работает путем моделирования загрузки и взаимодействия веб-страницы и измерения различных показателей производительности, таких как время загрузки, время первого рисования и время интерактивного взаимодействия. Она также проверяет наличие распространенных проблем, таких как неправильные размеры изображений, отсутствие текста alt и неработающие ссылки.

Lighthouse предоставляет всеобъемлющий и простой в использовании инструмент для выявления и устранения проблем с производительностью и доступностью веб-страниц. Он широко используется веб-разработчиками и интегрирован во многие популярные инструменты разработки.

**Инструменты разработки браузера**

Большинство браузеров имеют встроенные инструменты разработчика, которые помогают вам выполнять конкретные задачи веб-разработки на веб-странице. Моя личная рекомендация заключалась бы в использовании Chrome DevTools, поскольку он самый продвинутый и обладает наибольшим количеством функций. Однако, если вы используете другой браузер, вы все равно можете использовать DevTools для отладки вашего веб-приложения.

Chrome DevTools - это набор инструментов веб-разработки, встроенных в браузер Google Chrome. Это позволяет веб-разработчикам отлаживать и оптимизировать веб-страницы, предоставляя ряд функций для проверки HTML, CSS и JavaScript страниц и манипулирования ими.

Некоторые функции, предоставляемые Chrome DevTools, включают:

**Панель элементов**: проверьте и измените HTML и CSS страницы.

**Панель консоли**: просмотр и отладка ошибок JavaScript и сообщений журнала.

**Сетевая панель**: отслеживайте сетевые запросы и ответы и анализируйте проблемы с производительностью.

**Панель производительности**: проанализируйте производительность страницы и определите узкие места.

**Панель приложения**: Проверьте ресурсы страницы, такие как файлы cookie и локальное хранилище.

Chrome DevTools - мощный и необходимый инструмент для веб-разработчиков, широко используемый для отладки и оптимизации веб-страниц. Он постоянно обновляется новыми функциями и улучшениями и доступен во всех современных веб-браузерах.

**Рендеринг на стороне сервера**

Рендеринг на стороне сервера относится к процессу, в ходе которого сервисная сторона завершает объединение HTML-структуры страницы, отправляет ее в браузер, а затем привязывает статус и события, чтобы она стала полностью интерактивной страницей.

**Реагировать**

React - самая популярная интерфейсная библиотека JavaScript для создания пользовательских интерфейсов. React также может выполнять рендеринг на сервере с помощью Node и запускать мобильные приложения с помощью React Native.

**Angular**

Angular - это основанная на компонентах платформа интерфейсной разработки, построенная на TypeScript, которая включает в себя набор хорошо интегрированных библиотек, включающих такие функции, как маршрутизация, управление формами, взаимодействие клиент-сервер и многое другое.

**Vue.js**

Vue.js это фреймворк JavaScript с открытым исходным кодом для создания пользовательских интерфейсов и одностраничных приложений. В основном он ориентирован на разработку интерфейса.

**Стройный**

Svelte - это фреймворк javascript, который, в отличие от Vue и React, не использует вертикальное разделение DOM, а вместо этого точно знает, что и где обновлять при изменении состояния. В основном он ориентирован на интерфейс и построение пользовательских интерфейсов.

**Ремикс**

Remix - это веб-платформа с полным стеком, которая позволяет вам сосредоточиться на пользовательском интерфейсе и работать с веб-стандартами, чтобы обеспечить быстрый, гладкий и устойчивый пользовательский интерфейс. Людям понравится использовать ваши материалы.

**Next.js**

Next.js это платформа разработки с открытым исходным кодом, построенная поверх Node.js предоставляющая веб-приложениям на основе React функциональные возможности, такие как рендеринг на стороне сервера и создание статических веб-сайтов.

**Angular Universal**

Проект Angular Universal - это проект сообщества, направленный на расширение основных API-интерфейсов Angular (платформа-сервер), чтобы разработчики могли выполнять рендеринг приложений Angular на стороне сервера. Он в основном использует экспресс для отображения страниц на страницы node.js сервер.

**Nuxt.js**

Nuxt.js это бесплатная библиотека JavaScript с открытым исходным кодом, основанная на Vue.js, Node.js, Webpack и Babel.js. Nuxt вдохновлен Next.js , который является фреймворком аналогичного назначения, основанным на React.js.

**Изящный комплект**

Svelte Kit - это платформа для создания веб-приложений с использованием Svelte JavaScript framework. Он разработан как комплексное решение для создания веб-приложений с поддержкой рендеринга на стороне сервера, автоматического разделения кода и оптимизированной производительности.

**Генераторы статических сайтов**

Генератор статических сайтов - это инструмент, который генерирует полноценный статический HTML-сайт на основе необработанных данных и набора шаблонов. По сути, генератор статических сайтов автоматизирует задачу кодирования отдельных HTML-страниц и подготавливает эти страницы к предоставлению пользователям заранее. Поскольку эти HTML-страницы предварительно созданы, они могут очень быстро загружаться в браузеры пользователей.

**Одиннадцать**

Eleventy (11ty) - это простой в использовании, легко настраиваемый, высокопроизводительный и мощный генератор статических сайтов с полезным набором плагинов (например, навигация, преобразования изображений во время сборки, ресурсы кэша). Страницы могут быть созданы и написаны с использованием различных языков шаблонов (HTML, Markdown, JavaScript, Liquid, Nunjucks, Handlebars, Mustache, литералы шаблонов EJS, Haml, Pug или JS). Но она также предлагает возможность динамического создания страниц из локальных данных или внешних источников, которые компилируются во время сборки. Она не имеет зависимостей JavaScript на стороне клиента.

**Astro**

Astro - это универсальный веб-фреймворк для создания быстрых веб-сайтов, ориентированных на контент. Astro сочетает в себе мощь современного компонентного фреймворка с производительностью и гибкостью генератора статических сайтов.

Острова компонентов: новая веб-архитектура для создания более быстрых веб-сайтов.

Разработка API для первого сервера: уберите дорогостоящую гидратацию с устройств ваших пользователей.

По умолчанию нулевой JS: нет накладных расходов во время выполнения JavaScript, замедляющих работу.

Готовность к работе на Edge: развертывание в любом месте, даже в глобальной среде выполнения Edge, такой как Deno или Cloudflare.

Настраиваемый: Tailwind, MDX и более 100 других интеграций на выбор.

Не зависит от пользовательского интерфейса: поддерживает React, Preact, Svelte, Vue, Solid, Lit и другие.

**Next.js**

Next.js это платформа разработки с открытым исходным кодом, построенная поверх Node.js предоставляющая веб-приложениям на основе React функциональные возможности, такие как рендеринг на стороне сервера и создание статических веб-сайтов.

**Gatsby**

Gatsby - это платформа с открытым исходным кодом на основе React со встроенными функциями производительности, масштабируемости и безопасности.

**Vuepress**

VuePress состоит из двух частей: минималистичного генератора статических сайтов (открывается в новом окне) с системой тематизации на базе Vue и API плагинов, а также темы по умолчанию, оптимизированной для написания технической документации. Она была создана для поддержки потребностей в документации собственных подпроектов Vue.

**Джекилл**

Jekyll - это генератор статических сайтов. Он берет текст, написанный на вашем любимом языке разметки, и использует макеты для создания статического веб-сайта. Вы можете настроить внешний вид сайта, URL-адреса, данные, отображаемые на странице, и многое другое.

**Хьюго**

Hugo - самый быстрый в мире движок для статических веб-сайтов. Он написан на Go (он же Golang) и разработан bep, spf13 и друзьями.

**Nuxt js**

Nuxt.js это платформа разработки с открытым исходным кодом, построенная поверх Node.js предоставляющая веб-приложениям на основе Vue такие функциональные возможности, как рендеринг на стороне сервера и создание статических веб-сайтов.

**Ремикс**

Remix - это веб-платформа с полным стеком, которая позволяет вам сосредоточиться на пользовательском интерфейсе и работать с веб-стандартами, чтобы обеспечить быстрый, гладкий и устойчивый пользовательский интерфейс. Людям понравится использовать ваши материалы.

**Graphql**

GraphQL - это язык запросов для API и среда выполнения для выполнения этих запросов с вашими существующими данными. GraphQL предоставляет полное и понятное описание данных в вашем API, дает клиентам возможность запрашивать именно то, что им нужно, и ничего больше, упрощает разработку API с течением времени и предоставляет мощные инструменты разработчика.

**Apollo**

Apollo - это платформа для построения унифицированного графика, коммуникационного уровня, который помогает вам управлять потоком данных между вашими клиентскими приложениями (такими как веб и собственные приложения) и вашими внутренними сервисами.

**Реле современное**

Relay - это JavaScript-клиент, используемый в браузере для извлечения данных GraphQL. Это JavaScript-фреймворк, разработанный Facebook для управления и извлечения данных в приложениях React. Она построена с учетом масштабируемости для обеспечения работы сложных приложений, таких как Facebook. Конечная цель GraphQL и Relay - обеспечить мгновенное взаимодействие пользовательского интерфейса с ответом.

**Мобильные приложения**

Некоторое время назад разработка мобильного приложения с использованием JavaScript была невозможна. Но теперь разработчики JavaScript могут создавать мобильные приложения, используя свои знания в области веб-разработки. Вот список опций для создания мобильных приложений на JavaScript.

**React Native**

React Native - это популярный фреймворк для мобильных приложений на основе JavaScript, который позволяет создавать мобильные приложения с встроенной визуализацией для iOS и Android. Фреймворк позволяет создавать приложения для различных платформ, используя одну и ту же кодовую базу.

**Flutter**

Flutter - это бесплатный фреймворк мобильного пользовательского интерфейса с открытым исходным кодом, созданный Google и выпущенный в мае 2017 года. В нескольких словах, он позволяет создавать собственное мобильное приложение только с одной кодовой базой. Это означает, что вы можете использовать один язык программирования и одну кодовую базу для создания двух разных приложений (для iOS и Android).

Flutter состоит из двух важных частей:

SDK (Software Development Kit): набор инструментов, которые помогут вам разрабатывать ваши приложения. Сюда входят инструменты для компиляции вашего кода в машинный код (код для iOS и Android).

Фреймворк (библиотека пользовательского интерфейса на основе виджетов): набор повторно используемых элементов пользовательского интерфейса (кнопок, текстовых вводов, ползунков и так далее), которые вы можете персонализировать для своих собственных нужд. Для разработки с помощью Flutter вы будете использовать язык программирования под названием Dart. Язык был создан Google в октябре 2011 года, но за последние годы он значительно улучшился.

Dart фокусируется на интерфейсной разработке, и вы можете использовать его для создания мобильных и веб-приложений.

Если вы немного разбираетесь в программировании, Dart - это типизированный объектный язык программирования. Вы можете сравнить синтаксис Dart с JavaScript.

**Ionic**

Ionic Framework - это инструментарий пользовательского интерфейса с открытым исходным кодом для создания высокопроизводительных высококачественных мобильных приложений, настольных приложений и прогрессивных веб-приложений с использованием таких веб-технологий, как HTML, CSS и JavaScript.

**NativeScript**

NativeScript - это платформа с открытым исходным кодом для создания собственных приложений для iOS и Android на Angular, TypeScript или JavaScript.

**Настольные приложения на JavaScript**

Некоторое время назад разработка настольного приложения с использованием JavaScript была невозможна. Но теперь разработчики JavaScript могут создавать настольные приложения, используя свои знания в области веб-разработки. Вот список опций для создания настольных приложений на JavaScript.

**Electron**

Electron позволяет создавать кроссплатформенные настольные приложения с использованием HTML, CSS и Javascript / Typescript. Он использует Chromium и Node.js , так что по сути это “браузерное” приложение, совместимое с Mac, Windows и Linux.

**Tauri**

Tauri - это инструментарий, который помогает разработчикам создавать приложения для основных настольных платформ, используя практически любой существующий фреймворк интерфейса. Ядро построено на Rust, а интерфейс командной строки использует преимущества Node.js что делает Tauri по-настоящему многоязычным подходом к созданию и поддержке отличных приложений.

**Flutter**

Flutter - это бесплатный мультиплатформенный фреймворк пользовательского интерфейса с открытым исходным кодом, созданный Google и выпущенный в мае 2017 года. В нескольких словах, он позволяет создавать собственное мобильное приложение только с одной кодовой базой. Это означает, что вы можете использовать один язык программирования и одну кодовую базу для создания трех разных приложений (для iOS, Android и настольных компьютеров).

**Бонусный контент**

Дополнять бонусный контент не требуется, он просто расширяет темы, затронутые в дорожной карте, и содержит элементы, которые были пропущены в дорожной карте. Просмотрите их, если хотите.

**Знания пользовательского интерфейса / UX** - Получение знаний пользовательского интерфейса / UX не является обязательным для разработчика интерфейса, но разборчивость в дизайне и хорошее понимание пользовательского интерфейса / UX помогут вам в вашей карьере.

[Системы проектирования](https://roadmap.sh/design-system) - Узнайте, как создавать системы проектирования и работать с ними.

**Визуальное программирование** — вполне возможно быть успешным, компетентным разработчиком интерфейса и без этого, но для многих разработчиков интерфейса этот путь представляет собой серьезную часть их рабочей нагрузки и представляет собой очень востребованный вариант разработки интерфейса – особенно для разработчиков, обладающих визуальными способностями.

**Для анимации**: [GSAP](https://greensock.com/gsap/), [Lottie](https://airbnb.design/lottie/), [Framer Motion](https://github.com/framer/motion) или [MoJs](https://mojs.github.io/" \t "_blank)

**Библиотека веб-графики (WebGL)**: [WebGL MDN](https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API/Tutorial), [основы WebGL](https://webglfundamentals.org/webgl/lessons/webgl-fundamentals.html)

**2D графика**: [Canvas](https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API), [PixiJS](https://pixijs.com/" \t "_blank), [PhaserJS](https://phaser.io/" \t "_blank)

**3D графика**: [ThreeJS](https://threejs.org/" \t "_blank), [BabylonJS](https://www.babylonjs.com/" \t "_blank)

Взгляните на [Qwik](https://qwik.builder.io/" \t "_blank) и [Partytown](https://partytown.builder.io/" \t "_blank).

[Houdini](https://developer.mozilla.org/en-US/docs/Web/Guide/Houdini) — Вот хорошая [вводная статья к houdini](https://www.smashingmagazine.com/2020/03/practical-overview-css-houdini/).

[WebRTC](https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API) - WebRTC - это бесплатный открытый проект, который предоставляет браузерам и мобильным приложениям возможности обмена данными в реальном времени (RTC) через простые API. Компоненты WebRTC были оптимизированы, чтобы наилучшим образом служить этой цели.

[IDL](https://developer.mozilla.org/en-US/docs/Glossary/IDL) - Не каждому разработчику нужно разбираться в [IDL](https://developer.mozilla.org/en-US/docs/Glossary/IDL), но есть веский аргумент в пользу того, чтобы больше разработчиков JavaScript знали об их существовании и, надеюсь, могли их использовать.

[WebAssembly](https://developer.mozilla.org/en-US/docs/WebAssembly) - WebAssembly - это сравнительно новая технология, которая позволяет запускать код, написанный на языках, отличных от JavaScript, в Интернете. Это низкоуровневый язык, подобный ассемблеру, с компактным двоичным форматом, который работает с почти родной производительностью и предоставляет таким языкам, как C / C ++ и Rust, цель компиляции, чтобы они могли работать в Интернете.