**Раздел 2. Обзор библиотек, онлайн-сервисов и инструментов необходимых для создания сайта предприятия**

**О Figma.**

Figma — это онлайн-редактор, который позволяет удобно создавать интерфейсы, макеты сайтов и мобильных приложений, презентации, иллюстрации, логотипы и анимацию.

Этим инструментом часто пользуются дизайнеры, но также он может быть полезен для продакт-менеджеров и разработчиков. Когда все участники проекта хорошо разбираются в программе, им легче понимать друг друга и эффективно работать над продуктом.

Особенности программы:

* Макет доступен для просмотра и редактирования в режиме онлайн. Все участники проекта сразу видят изменения, вносимые другими участниками. Менеджеру или разработчику больше не нужно беспокоить дизайнера и узнавать, как продвигается работа над проектом. Они могут зайти в программу и увидеть последние изменения самостоятельно.
* Сервис открывается в браузере, поэтому не нужно устанавливать дополнительные приложения. Это удобно, если вам нужно показать макет клиенту или коллегам. Вы можете просто отправить им ссылку и настроить права доступа.
* В платном тарифе есть полезные инструменты для командной работы. Например, голосовой чат. Участники проекта могут созвониться в редакторе, обсудить текущий макет и внести правки. Это удобно, так как не нужно включать демонстрацию экрана и переключаться между разными программами.

Конечно, менеджеру по продукту не нужно владеть программой на уровне дизайнера. Однако знание основных инструментов графического редактора может значительно ускорить работу над созданием сайта или приложения.

[Прототипы](https://education.yandex.ru/journal/chto-takoe-figma#prototipy).

Чтобы не тратить время на объяснение, как должно работать приложение или сайт, можно создать его прототип — модель, которая будет отражать будущий продукт. Это поможет разработчику и клиенту лучше понять, как будет выглядеть макет на реальном устройстве.

В прототип можно добавить переходы между экранами, анимацию и выпадающее меню. Такой подход позволит заранее обнаружить ошибки в сценарии и исправить их ещё до начала разработки проекта.

Интерфейс.

Интерфейс — это внешний вид продукта, который включает в себя кнопки, формы и иконки. Их можно создавать непосредственно в Figma и добавлять различные эффекты, такие как анимация для баннеров или возможность сделать кнопки кликабельными.

**О среде разработки Visual Studio Code.**

Visual Studio Code — это продукт компании Microsoft, который предназначен для разработчиков и верстальщиков. Он является одним из самых популярных инструментов в своей нише.

Это редактор кода от Microsoft, который представляет собой «облегчённую» версию Visual Studio. С его помощью можно не только создавать приложения, но и использовать множество плагинов, которые позволят превратить редактор в полноценную среду программирования.

Подходит для работы на операционных системах:

* Windows;
* MacOS;
* Linux.

Этот редактор доступен бесплатно, поэтому он приобрёл огромную популярность. Он одинаково хорошо работает как на старых устройствах, так и на современных компьютерах.

Visual Studio Code — это редактор, который особенно удобен для новичков благодаря своей простоте. Однако он также может быть полезен и опытным разработчикам при создании сложных кодов. Поддерживающий множество языков программирования. Он подходит для кроссплатформенной разработки и широко используется веб-программистами.

Поддерживает такие языки как:

* JavaScript;
* HTML;
* CSS;
* PHP;
* Go;
* Ruby;
* Python;
* C#;
* TypeScript.

Этот инструмент подходит для работы почти со всеми современными языками программирования. Он поддерживает популярные фреймворки и библиотеки, такие как React JS, Vue.js, LESS и SCSS.

Теоретически Visual Studio Code можно использовать для любых целей при разработке программного обеспечения. Однако на практике он чаще всего применяется, когда для создания программы не требуется полный функционал интегрированной среды разработки (IDE).

Особенность Visual Studio Code в том, что его легко расширить. Для этого достаточно скачать и установить плагин или дополнение из официального каталога. Все инструменты и сам редактор распространяются бесплатно.

Visual Studio Code поддерживает:

1. Создание и редактирование кода приложения – вот основное предназначение этого бесплатного продукта от Microsoft. С его помощью пользователи смогут не только написать код, но и отредактировать его, а также сохранить. Редактор предлагает функции корректировки, которые автоматически расставляют отступы в коде для повышения его читаемости. Кроме того, он обеспечивает автодополнение, что позволяет ускорить разработку программы, особенно на начальных этапах. Visual Studio Code – это инструмент, в котором есть целая система помощи при редактировании исходного кода, называемая IntelliSense.

2. Функция автоматического редактирования — полезный инструмент, который позволяет автоматически исправлять и корректировать исходный код приложения. Это осуществляется с помощью специальных плагинов и расширений.В Visual Studio Code с их помощью можно исправить незначительные ошибки, такие как неправильные отступы, ошибки в именовании, несоответствие стиля и другие. Однако для решения более сложных проблем, которые требуют глубокого понимания кода, эти расширения не подходят, поскольку система не сможет самостоятельно определить, как их исправить.Тем не менее, для мелких недочётов в VS Code есть специальные возможности, которые работают очень хорошо.

3. В Visual Studio Code есть удобная строка поиска с поддержкой регулярных выражений, что позволяет быстро находить нужную информацию. Также в редакторе доступно автодополнение кода, что значительно ускоряет процесс разработки. Кроме того, можно выделить код и закомментировать его с помощью простого сочетания клавиш.

4. Горячие клавиши. Используются для быстрого применения функций без предварительного входа в меню редактора. Можно воспользоваться встроенными сочетаниями Visual Studio Code или настроить их самостоятельно.

5. Контроль версий. Визуал Студия позволяет сразу из редактора провести откат, добавить комментарий, отправить коммит в удаленный репозиторий. В консоль для этого заходить не придется, как и набирать те или иные команды – все осуществляется через встроенный визуальный интерфейс.

6. Установку дополнений. Это – ключевая особенность Visual Studio Code. Из редактора можно перейти в каталог дополнение и расширений, подобрать необходимый плагин и установить его в несколько кликов. Пример – поддержка различных языков, автоматическое исправление, конфигураторы. Данная опция дает возможность настройки редактора под нужды каждого конкретного разработчика.

7. Отладчик. Он используется для языка JavaScript и базирующихся на нем технологий. Пример – TypeScript. Для остальных языков разработки встроенных отладчиков нет – они устанавливаются в виде плагинов. Интерактивная [отладка](https://otus.ru/journal/otladka-prilozheniya-chto-dolzhen-pomnit-razrabotchik/) пошагово помогает выполнять код, а также на каждом этапе отслеживать изменение данных.

К преимуществам Visual Studio Code относят следующие моменты:

1. Простое освоение. Редактор оснащен интуитивно понятным интерфейсом, а также возможностью настройки. Единственная проблема для новичков – это горячие клавиши, которые организованы необычным образом.
2. Небольшой вес.
3. Кроссплатформенность.
4. Наличие online версии. В случае ее использования для написания программного кода не придется вообще ничего устанавливать. Visual Studio Online работает через браузер. Расположен редактор [здесь](https://vscode.dev/).
5. Поддержка разных языков. Опция активируется за счет установки плагинов.
6. Гибкость.
7. Данная технология позволяет работать со строками (lines), а также приводить исходный код к единому стилю в несколько кликов. Ей легко пользоваться, но недостатки у VS Code тоже есть.

К минусам Visual Studio Code относят:

1. Ограниченную функциональность, особенно в сравнении с полноценной IDE от Майкрософт.
2. Медленную работу, особенно на старых устройствах. Связано это с тем, что редактор написан на Election.

**О HTML и CSS.**

HTML (HyperText Markup Language) — это язык разметки гипертекста, который используется для создания документов в интернете. Он был создан в начале 90-х годов как простой инструмент для разработки веб-страниц, но сегодня уже сложно представить себе интернет без него. Большинство сайтов используют HTML в той или иной форме.

В 2014 году был официально завершён процесс разработки нового стандарта — HTML5. Этот стандарт привнёс множество инноваций в HTML и фактически произвёл революцию в этой области.

Что именно привнес HTML5?

* HTML5 определяет новый алгоритм парсинга для создания структуры DOM
* добавление новых элементов и тегов, как например, элементы video, audio и ряд других
* переопределение правил и семантики уже существовавших элементов HTML

Фактически, с добавлением новых функций HTML5 превратился не просто в новую версию языка разметки для создания веб-страниц, а стал полноценной платформой для разработки приложений. Область его применения вышла далеко за пределы интернета: HTML5 используется для создания мобильных приложений под Android, iOS, Windows Mobile, а также для разработки десктопных приложений для обычных компьютеров с операционными системами Windows 8/8.1/10.

В итоге, как правило, HTML 5 применяется преимущественно в двух значениях:

* HTML 5 как обновленный язык разметки гипертекста, некоторое развитие предыдущей версии HTML 4
* HTML 5 как мощная платформа для создания веб-приложений, которая включает не только непосредственно язык разметки гипертекста, обновленный HTML, но и язык программирования JavaScript и каскадные таблицы стилей CSS 3.

За развитие HTML5 отвечает World Wide Web Consortium, или сокращённо W3C — это независимая международная организация, устанавливающая стандарты для Всемирной паутины.

Актуальную полную спецификацию на английском языке вы можете найти по адресу https://www.w3.org/TR/html5/. Важно отметить, что организация продолжает работу над HTML5 и регулярно выпускает обновления к спецификации.

Поддержка браузерами.

Следует отметить, что между спецификацией HTML5 и её использованием в веб-браузерах всегда был некоторый разрыв. Большинство браузеров начали внедрять стандарты HTML5 ещё до их официальной публикации. На данный момент большинство последних версий браузеров поддерживают большую часть функциональности HTML5, включая Google Chrome, Firefox, Opera, Internet Explorer 11 и Microsoft Edge.

Однако многие старые браузеры, такие как Internet Explorer 8 и более ранние версии, не поддерживают эти стандарты, а IE 9 и 10 поддерживают их только частично. Даже те браузеры, которые в целом поддерживают стандарт, могут не поддерживать некоторые его функции. Это тоже нужно учитывать при работе.

Тем не менее, в целом ситуация с поддержкой этой технологии довольно хорошая. Чтобы проверить, поддерживает ли конкретный браузер HTML5, можно воспользоваться специальным сервисом — html5test.com.

**О JavaScript.**

JavaScript был создан в 1995 году для программирования веб-страниц в браузере Netscape Navigator. Позднее этот язык был реализован и в других популярных графических браузерах. Благодаря этому стали возможны современные веб-приложения, с которыми можно взаимодействовать непосредственно, без необходимости перезагрузки страницы при каждом действии.

JavaScript — это язык программирования, который используется для создания интерактивных элементов на веб-страницах. Он позволяет добавлять различные эффекты и функции, которые делают сайты более удобными и привлекательными для пользователей.

Важно отметить, что JavaScript не имеет ничего общего с языком программирования Java. Похожее название языка было выбрано не из-за его сходства с Java, а из-за маркетинговых соображений. Когда появился JavaScript, язык Java активно продавался и приобретал популярность. Кто-то решил, что будет хорошей идеей воспользоваться успехом другого языка. Однако это решение привело к путанице и проблемам в будущем.

После того как JavaScript стал популярным, был создан стандартный документ, который описывал, как должен работать этот язык. Это так называемый стандарт ECMAScript, который был разработан организацией Ecma International. На практике термины ECMAScript и JavaScript взаимозаменяемы — они представляют собой два названия одного и того же языка.

Идея не типизированности JavaScript заключается в том, чтобы облегчить программирование на этом языке для новичков. Однако на практике это может затруднять поиск проблем в программах, так как система не указывает на них.

JavaScript претерпел несколько версий. Версия ECMAScript 3 была широко поддерживаема во времена, когда JavaScript только начинал свой путь к господству, примерно между 2000 и 2010 годами. В это время разрабатывалась амбициозная версия 4, которая планировалась как значительное улучшение и расширение языка. Однако такое радикальное изменение живого языка оказалось политически сложным, и в 2008 году работы над версией 4 были прекращены.

В результате появилась менее амбициозная пятая версия, которая включала лишь некоторые улучшения. В 2015 году вышла шестая версия, в которой были реализованы некоторые идеи, запланированные для четвёртой версии. С тех пор каждый год выпускаются небольшие обновления.

Язык развивается, и браузеры должны постоянно следовать за этими изменениями, чтобы поддерживать все функции. Разработчики языка стараются не вносить изменения, которые могут нарушить работу существующих программ. Поэтому старые программы по-прежнему будут работать в новых браузерах.

JavaScript используется не только в браузерах. Некоторые базы данных, такие как MongoDB и CouchDB, используют JavaScript в качестве языка сценариев и запросов. Ряд платформ для программирования настольных ПК и серверов, включая проект Node.js, предоставляют среду для программирования на JavaScript вне браузера. [5]

**О TypeScript.**

TypeScript — это язык, который используется для создания современных веб- и мобильных приложений, проектов на NodeJS и IoT (систем управления устройствами через интернет). Он позволяет разрабатывать более безопасные программы с подробной документацией, которая будет полезна не только вам, но и будущим разработчикам. TypeScript также упрощает процесс рефакторинга кода и избавляет от необходимости писать половину модульных тестов.

Программы представляют собой файлы, содержащие написанный вами текст. Компилятор — специальная программа — считывает этот текст и преобразует его в абстрактное синтаксическое дерево (АСД). АСД — это структура данных, которая игнорирует пустые области, комментарии и ваше мнение о пробелах и табуляции. Затем компилятор преобразует АСД в байт-код низкого уровня, который можно запустить в среде выполнения и получить результат.

Таким образом, когда вы запускаете программу, вы фактически просите среду выполнения прочитать байт-код, созданный компилятором на основе АСД, полученного из исходного кода. Детали этого процесса могут отличаться, но для большинства языков он выглядит так:

1. Программа преобразуется в АСД.

2. АСД компилируется в байт-код.

3. Байт-код считывается средой выполнения.

Особенность TypeScript заключается в том, что он компилирует код не в байт-код, а в код JavaScript. После этого вы можете запустить его в браузере с помощью NodeJS.

Проверка типов — это одна из главных особенностей TypeScript. Благодаря ей компилятор (TSC) может убедиться, что ваша программа работает так, как вы ожидаете.

Итак, если мы добавим проверку типов и преобразование в JavaScript, то процесс компиляции TypeScript будет выглядеть примерно так (рис. 1):

|  |  |
| --- | --- |
| TypeScript | 1. Код на TS -> TypeScript АСД  2. АСД проверяется проверкой типов  3. TypeScript АСД -> Код на JS |
| JavaScript | 4. Код на JS -> JavaScript АСД  5. АСД -> Байт-код  6. Байт-код оценивается по времени выполнения |

**Рис. 1** Компиляция и запуск TypeScript.

Шаги 1-3 производятся компилятором, а шаги 4-6 — средой выполнения JavaScript, находящейся в вашем браузере, или NodeJS, или любым другим JavaScript-движком.

В течение всего процесса шаги 1-2 используют типы программы, а шаг 3 уже этого не делает. Стоит еще раз повториться: когда TSC компилирует код в JavaScript, он не будет смотреть на типы. Это означает, что типы никогда не смогут повлиять на сгенерированный вывод и будут использованы только для проверки типов. Эта особенность позволяет безопасно с ними экспериментировать — обновлять и улучшать их без риска сломать приложение.

JavaScript-компиляторы и среды выполнения, как правило, представляют собой единую программу, называемую движком. Будучи программистом, с ним вы и будете взаимодействовать. Так работают V8 (движок, лежащий в основе NodeJS, Chrome и Opera), SpiderMonkey (Firefox), JSCore (Safari) и Chakra (Edge). Именно поэтому JavaScript и называют интерпретируемым языком.[6]

**О React.**

React — это популярная библиотека, которая используется для создания пользовательских интерфейсов. Она была разработана в Facebook с целью решения проблем, связанных с разработкой крупномасштабных сайтов, управляемых данными.

Когда библиотека была выпущена в 2013 году, она вызвала некоторое недоверие, так как её принципы были довольно необычными. React — это небольшая библиотека, которая поставляется без набора инструментов, необходимых для создания приложений. Однако благодаря React можно создавать код, похожий на HTML, с помощью JavaScript.

Эти теги требуют предварительной обработки перед тем, как они будут запущены в браузере. Для этого, скорее всего, потребуется встроенный инструмент, такой как Webpack.

Если вы, как и мы, прочитали эту статью, то могли быть впечатлены возможностями новой библиотеки JavaScript, которая может решить все проблемы с DOM (объектной моделью документа). Вы могли подумать, что она всегда будет проста в использовании и никогда не вызовет неудобств.

Однако затем у вас могли возникнуть вопросы: как можно будет преобразовать этот JSX? Как загружать данные? Куда девать CSS? И что такое декларативное программирование?

С появлением этой библиотеки возникало всё больше вопросов о том, как её внедрить в повседневную работу. Каждый разговор порождал новую терминологию, технические приёмы и вопросы.

Стоит отметить, что библиотека небольшая по размеру и используется только для одной части работы. Она не предоставляет весь инструментарий, который можно ожидать от традиционного JavaScript-фреймворка. Разработчики сами принимают решения о том, какие средства экосистемы использовать. Кроме того, постоянно появляются новые наборы инструментов, а старые уходят на второй план. Среди такого многообразия библиотек неудивительно, что за ними сложно угнаться.

React всё ещё сохраняет свою новизну. Он достиг стадии стабильности основных функциональных свойств, но они могут измениться и в будущих версиях библиотеки. В следующих версиях будет включено средство Fiber — новая реализация основного алгоритма React, цель которой — увеличить скорость отображения изображений на экране. Пока рано делать предположения о том, как это повлияет на React-разработчиков, но это, несомненно, отразится на скорости, с которой приложения выводят изображение на экран и обновляют его.

Многие из этих изменений связаны с целевой аудиторией. В этой книге рассматриваются способы разработки одностраничных веб-приложений с помощью React, но не стоит думать, что браузеры — это единственное место, где могут работать такие приложения. React Native, выпущенный в 2015 году, позволяет использовать преимущества React-приложений в приложениях для iOS и Android. Хотя об этом ещё рано говорить, но уже появился фреймворк React VR для создания интерактивных приложений с поддержкой виртуальной реальности, что открывает возможности для разработки 360-градусных интерфейсов с использованием React и JavaScript. [7]

**О Next.js.**

За последние несколько лет веб-разработка претерпела значительные изменения. До появления современных фреймворков JavaScript создание динамических веб-приложений было сложным процессом, требующим использования множества различных библиотек и конфигураций.

Однако с появлением таких фреймворков, как Angular, React и Vue, разработка веб-интерфейсов стала более быстрой и эффективной. Эти фреймворки привнесли в сеть множество инновационных идей.

React, созданный Джорданом Уолком в Facebook, стал одним из самых популярных фреймворков. Он был создан под влиянием библиотеки XHP Hack, которая позволяла разработчикам Facebook на PHP и Hack создавать повторно используемые компоненты для интерфейса приложений.

Библиотека JavaScript React стала доступна с открытым исходным кодом в 2013 году и навсегда изменила способы создания веб-сайтов, веб-приложений, нативных приложений (позже с помощью React Native) и даже виртуальной реальности (с помощью React VR). React быстро стала одной из самых любимых и популярных библиотек JavaScript, и миллионы веб-сайтов используют её в рабочей среде для достижения различных целей.

Однако у React была одна проблема: по умолчанию он запускается на стороне клиента, что может негативно сказаться на поисковой оптимизации (SEO) и начальной производительности загрузки. Для корректного отображения веб-приложения, написанного с использованием этой библиотеки, браузеру требуется некоторое время. Фактически, для отображения полного веб-приложения браузер должен загрузить весь пакет приложений, проанализировать его содержимое, затем запустить и отобразить результат, что может занять до нескольких секунд для очень больших приложений.

Многие компании и разработчики начали изучать способы предварительной отрисовки приложения на сервере, позволяя браузеру отображать отрисованное приложение React в виде обычного HTML. Это делает приложение интерактивным, как только бандл JavaScript передаётся клиенту.

Затем Vercel представил Next.js, который изменил правила игры. С момента своего первого выпуска фреймворк предоставил множество инновационных функций, таких как автоматическое разделение кода, рендеринг на стороне сервера, системы маршрутизации на основе файлов, предварительная выборка маршрутов и так далее.

Next.js показал, насколько простым может быть написание универсальных веб-приложений. Фреймворк позволяет разработчикам писать повторно используемый код как для клиентской, так и для серверной части, упрощая реализацию сложных задач, таких как разделение кода и рендеринг на стороне сервера.

Сегодня Next.js предоставляет множество новых функций, таких как:

1. Статическая генерация сайтов

2. Инкрементная статическая генерация

3. Поддержка встроенного TypeScript

4. Автоматическое заполнение полей

5. Оптимизация изображений

6. Поддержка интернационализации

7. Анализ производительности

Сегодня Next.js используется в производстве такими компаниями высшего уровня, как Netflix, Twitch, Tik Tok, Hulu, Nike, Rubber, Elastic и многими другими. Если вам интересно, вы можете ознакомиться с полным списком по ссылке [https://nextjs.org/showcase.[8](https://nextjs.org/showcase.%5b8)]

**О Node.js.**

Node — это веб-сервер, который имеет много общего с другими популярными решениями, такими как IIS от Microsoft или Apache. Однако, в отличие от них, Node отличается своей минималистичностью и простотой в установке и настройке.

Хотя настройка серверов Node для достижения максимальной производительности в условиях промышленной эксплуатации может показаться сложной задачей, конфигурационные опции в Node гораздо более просты и понятны, чем в более традиционных веб-серверах.

Ещё одно важное отличие Node — это его однопоточность. На первый взгляд это может показаться шагом назад, но на самом деле это гениальная идея. Однопоточность значительно упрощает задачу написания веб-приложений. Если вам требуется производительность многопоточного приложения, вы можете просто запустить больше экземпляров Node и получить преимущества многопоточности.

Приложения Node больше похожи на приложения PHP или Ruby, чем на приложения.NET или Java. Движок JavaScript, используемый в Node (V8, разработанный компанией Google), компилирует JavaScript во внутренний машинный код, подобно C или C++, но делает это прозрачным образом. Это уменьшает сложность обслуживания и развёртывания приложений.

Ещё одно преимущество приложений Node — невероятная независимость от платформы. Вы можете запустить приложение Node на любой операционной системе (Windows, OS X и Linux), что облегчает совместную работу команд веб-разработчиков.

В сердцевине экосистемы Node находится программное обеспечение, которое позволяет выполнять JavaScript-код на сервере без участия браузера. Это позволяет использовать фреймворки, написанные на JavaScript, такие как Express. Все веб-приложения, кроме самых простых, потребуют базы данных. Существуют базы данных, которые лучше других подходят для экосистемы Node, такие как MySQL, MariaDB, PostgreSQL, Oracle, SQL Server и MongoDB.

Использование Node открывает новые возможности для разработки под веб-серверы. Его минималистичность и простота в установке и настройке делают его привлекательным выбором для разработчиков. [9]

**О Express.js.**

Ресурс expressjs.com характеризует Express как «минималистичный и гибкий фреймворк для веб-приложений Node.js, обеспечивающий набор возможностей для построения веб- и мобильных приложений».

Достоинства Express:

1. Минималистичный. Один из наиболее привлекательных аспектов Express. Много раз разработчики фреймворков забывали, что лучше меньше, да лучше. Философия Express заключается в обеспечении минимальной прослойки между вами и сервером. Это не говорит о слабой надежности фреймворка или недостаточном количестве его полезных возможностей. Просто он в меньшей степени становится у вас на пути, позволяя более полно выражать свои идеи. Express предоставляет вам минималистичный фреймворк, а вы можете добавлять необходимый функционал в разных его частях, заменяя то, что вам никогда не понадобится. Это глоток свежего воздуха, ведь многие фреймворки создают раздутые, непонятные и сложные проекты еще до того, как вы написали хотя бы одну строчку кода. Зачастую первой задачей становится отсечение ненужного функционала или замена того, который не соответствует требованиям. Express практикует другой подход, позволяя вам добавлять то, что нужно, там, где нужно.

2. Гибкий. Механизм действия Express очень прост: он принимает HTTP-запрос от клиента (которым может быть браузер, мобильное устройство, другой сервер, десктопное приложение — одним словом, все, что использует HTTP) и возвращает HTTP-ответ. Этот базовый шаблон описывает практически все, связанное с Интернетом, что делает приложения Express чрезвычайно гибкими.

3. Фреймворк для веб-приложений. Наверное, более точным описанием будет «серверная часть фреймворка для веб-приложений». На сегодня под «фреймворками для веб-приложений» обычно имеются в виду фреймворки одностраничных приложений, такие как React, Angular или Vue. Тем не менее, за исключением небольшой группы отдельных приложений, большинство веб-приложений должно обмениваться данными и взаимодействовать с другими сервисами. Обычно это происходит через веб-API, который можно рассматривать как серверный компонент фреймворка веб-приложений. Обратите внимание, что построение всего приложения с рендерингом только на стороне сервера по-прежнему возможно (а иногда и желательно). В этом случае Express вполне может быть фреймворком всего веб-приложения!

4. Быстрый. Express, став одним из лучших фреймворков для платформы Node.js, привлек много внимания со стороны больших компаний с высокопроизводительными и сильно нагруженными веб-сайтами. Это вынудило команду по разработке Express сконцентрироваться на производительности, и теперь Express — лидер в этом сегменте.

5. Некатегоричный. Одними из отличительных особенностей экосистемы JavaScript являются ее размер и разнообразие. Ввиду того что Express зачастую занимает центральную позицию в разработке веб-приложений в Node.js, существуют сотни (если не тысячи) страниц сообщества, где подробно рассматриваются приложения Express. Разработчики Express, в свою очередь, предоставили чрезвычайно гибкую систему разработки промежуточного ПО, которая упрощает использование выбранных вами компонентов при создании приложения. В процессе разработки можно заметить, что Express отключает встроенные компоненты в пользу настраиваемого промежуточного ПО. [9]

**О Nest.js.**

Существует огромное количество доступных веб-фреймворков, а с появлением Node.js их стало еще больше. JavaScript-фреймворки очень быстро входят в моду и выходят из нее по мере развития веб-технологий. Nest.js это хорошая отправная точка для многих разработчиков, которые хотят использовать современный веб-фреймворк, потому что он использует язык, который очень похож на язык JavaScript, наиболее часто используемый в Интернете на сегодняшний день. Многих разработчиков обучали программированию с использованием таких языков, как Java или C/C++, которые являются строгими языками, поэтому использование JavaScript может быть немного неудобным, и в нем легко допускать ошибки из-за отсутствия безопасности типов. Nest.js используется TypeScript, который является золотой серединой. Это язык, который обеспечивает простоту и мощь JavaScript при обеспечении безопасности типов других языков, к которым вы, возможно, привыкли. Безопасность типа в Nest.js доступен только во время компиляции, поскольку Nest.js сервер скомпилирован в Node.js Экспресс-сервер, работающий на JavaScript. Однако это по-прежнему является важным преимуществом, поскольку позволяет вам лучше разрабатывать программы без ошибок перед выполнением.

Node.js имеет богатую экосистему пакетов в NPM (Node Package Manager). С более чем 350 000 пакетов, это крупнейший в мире реестр пакетов. Фреймворк Nest.js создан поверх фреймворка Express, что позволяет получить доступ ко всем его возможностям при разработке приложений. Одно из самых больших преимуществ Node.js — это огромный репозиторий модулей, которые можно установить и сразу использовать, вместо того чтобы писать свои собственные. Некоторые из этих модулей уже включены в платформу Nest, например, @nestjs/mongoose, которая использует библиотеку NPM mongoose. До 2009 года JavaScript был в основном интерфейсным языком, но после выпуска Node.js в 2009 году он стимулировал разработку многих проектов на JavaScript и TypeScript, таких как Angular, React и Vue. Angular послужил источником вдохновения для разработки Nest.js, потому что они оба используют «Систему модулей и компонентов», позволяющую использовать их повторно. Они также очень хорошо сочетаются друг с другом, поскольку Nest предоставляет возможность запускать универсальный сервер для обслуживания предварительно отрендеренных веб-страниц Angular, чтобы ускорить доставку веб-сайта, используя упомянутый выше серверный рендеринг (SSR). [10]

**О PostgreSQL.**

На сегодняшний день существует большое количество различных систем управления базами данных - СУБД, от коммерческих до открытых, от реляционных до новомодных NoSQL и аналогичных.

Одним из лидеров направления СУБД является PostgreSQL и ее различные ответвления, о некоторых из которых мы рассмотрим подробнее.

Редакции PostgreSQL:

1. Классический PostgreSQL - является «ванильная» сборка от PGDG, PostgreSQL Global Development Group. PostgreSQL создана на основе некоммерческой СУБД Postgres, разработанной как open-source проект в Калифорнийском университете в Беркли. Название расшифровывалось как «Post Ingres», и при создании Postgres были применены многие ранние наработки БД Ingres;

2. Российский Postgres Professional - сборка Postgres Pro это российская коммерческая СУБД, разработанная компанией Postgres Professional с использованием свободно-распространяемой СУБД PostgreSQL. Но при этом классическая СУБД значительно переработана для соответствия требованиям корпоративных заказчиков и российских регуляторов. Также Postgres Pro входит в реестр российского ПО и имеет действующий сертификат ФСТЭК.;

3. 2nd Quadrant Postgres-XL - для тяжелых и высоконагруженных систем в семействе PostgreSQL есть версия Postgres-XL (2nd Quadrant Postgres-XL), которая отличается от PostgreSQL иной философией и целями развития, в рамках которых стабильность, корректность и производительность ставятся выше функциональности. Данная редакция направлена на поддержку функций PostgreSQL при распределении рабочей нагрузки по кластеру и нескольких серверов БД. Само название "Postgres-XL" означает "Расширяемая решетка". Для выполнения этих задач Postgres-XL включает в себя дополнительные средства для повышения производительности и безопасности, такие как MPP-параллелизм и расширенная модель разграничения доступа, и охватывает область применения обработки больших объёмов данных, в то время как PostgreSQL в основном нацелен на OLTP (Online Transaction Processing).

В основе работы СУБД PostgreSQL лежит серверный процесс базы данных, выполняемый на одном сервере. Доступ из приложений к данным базы PostgreSQL производится с помощью специального процесса базы данных. То есть клиентские программы не могут получать самостоятельный доступ к данным даже в том случае, если они функционируют на том же ПК, на котором осуществляется серверный процесс.

У каждого серверного процесса есть своя локальная память. В ней находится кэш каталога (часто используемая информация о базе данных), планы запросов, рабочее пространство для выполнения запросов и другое.

Экземпляр СУБД работает с несколькими базами данных. Эти базы данных называются кластером. Хранение данных на диске организовано с помощью табличных пространств. Табличное пространство указывает расположение данных (каталог на файловой системе). [11]