МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Алтайский государственный университет»

Колледж АлтГУ

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**(дипломная работа)**

Тема: Разработка веб-сайта блога   
с использованием множества технологий

Выпускную квалификационную работу выполнил(а) студентка 3 курса,

группы к002в   
Третьяков Михаил Владимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
  
 (подпись)  
  
Научный руководитель:  
Жданов Денис Николаевич \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
  
 (подпись)   
  
Выпускная квалификационная работа защищена:   
«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г.   
  
Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
  
Председатель ГЭК:  
Печатнов Владимир Владимирович \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись)

Барнаул 2023 г.

Оглавление

[Введение 5](#_Toc161239009)

[1 ОСНОВЫ ПОСТРОЕНИЯ ВЕБ-САЙТОВ 6](#_Toc161239010)

[1.1 История появления интернета. 6](#_Toc161239011)

[1.2 Технологии, которые используется для построения веб-сайтов 7](#_Toc161239012)

[1.2.1 HTML 7](#_Toc161239013)

[1.2.2 CSS 8](#_Toc161239014)

[1.2.3 JavaScript 8](#_Toc161239015)

[1.2.4 PHP 9](#_Toc161239016)

[1.3 Среды выполнения кода 11](#_Toc161239017)

[1.3.1 UNIX 11](#_Toc161239018)

[1.3.2 Linux 12](#_Toc161239019)

[1.3.3 Docker 12](#_Toc161239020)

[1.3.4 NodeJS 13](#_Toc161239021)

[1.3.5 Bun 13](#_Toc161239022)

[1.4 Базы данных 14](#_Toc161239023)

[1.4.1 MySQL 14](#_Toc161239024)

[1.4.2 Mongo 15](#_Toc161239025)

[1.4.3 PostgreSQL 16](#_Toc161239026)

[1.4.4 Redis 16](#_Toc161239027)

[1.5 Дополнительные инструменты 17](#_Toc161239028)

[1.5.1 Astro 17](#_Toc161239029)

[1.5.2 TailwindCSS 18](#_Toc161239030)

[1.5.3 React 18](#_Toc161239031)

[1.5.4 Qwik 18](#_Toc161239032)

[1.5.5 HTMX 19](#_Toc161239033)

[1.5.6 Svelte 19](#_Toc161239034)

[2. Разработка сайта с использованием разных технологий 20](#_Toc161239035)

[2.1 Верстка сайта 20](#_Toc161239036)

[2.2 Создание страницы React и настройка Nodejs, MongoDB 23](#_Toc161239037)

[2.2.1 Настройка папок и баз данных с помощью Docker для страницы React 24](#_Toc161239038)

[2.2.2 Завершение страницы React 26](#_Toc161239039)

# Введение

# 1 ОСНОВЫ ПОСТРОЕНИЯ ВЕБ-САЙТОВ

# 1.1 История появления интернета.

Интернет — всемирная система объединённых компьютерных сетей, построенная на использовании протокола IP и маршрутизации пакетов данных. Интернет образует глобальное информационное пространство, служит физической основой для Всемирной паутины и множества других систем (протоколов) передачи данных. Часто упоминается как «Всемирная сеть» и «Глобальная сеть». В настоящее время, когда слово «Интернет» употребляется в обиходе, чаще всего имеется в виду Всемирная паутина и доступная в ней информация, а не сама физическая сеть. К середине 2008 года число пользователей, регулярно использующих Интернет, составило около 1,5 млрд человек (около четверти населения Земли). Всемирная паутина (англ. World Wide Web) — распределенная система, предоставляющая доступ к связанным между собой документам, расположенным на различных компьютерах, подключенных к Интернету. Всемирную паутину образуют миллионы web-серверов. Большинство ресурсов всемирной паутины представляет собой гипертекст. Гипертекстовые документы, размещаемые во всемирной паутине, называются web-страницами. Несколько web-страниц, объединенных общей темой, дизайном, а также связанных между собой ссылками и обычно находящихся на одном и том же web-сервере, называются web-сайтом. Всемирная паутина вызвала настоящую революцию в информационных технологиях и бум в развитии Интернета. В 1988 году был разработан протокол Internet Relay Chat (IRC), благодаря чему в Интернете стало возможно общение в реальном времени (чат). В 1989 году в Европе, в стенах Европейского совета по ядерным исследованиям родилась концепция Всемирной паутины. Её предложил знаменитый британский учёный Тим Бернерс-Ли, он же в течение двух лет разработал протокол HTTP, язык HTML и идентификаторы URI. В 1990 году сеть ARPANET прекратила своё существование, полностью проиграв конкуренцию NSFNet. В том же году было зафиксировано первое подключение к Интернету по телефонной линии. В 1991 году Всемирная паутина стала общедоступна в Интернете, а в 1993 году появился знаменитый веб-браузер NCSA Mosaic. Всемирная паутина набирала популярность. Именно сочетание веб-протокола от Тима Бернерс-Ли, который обеспечивал коммуникацию, и браузера (Mosaic) от Марка Андриссена, который предоставил функционально совершенный пользовательский интерфейс, создало условия для наблюдаемого взрыва (интереса к Веб). В 1995 году NSFNet вернулась к роли исследовательской сети, маршрутизацией всего трафика Интернета теперь занимались сетевые провайдеры, а не суперкомпьютеры Национального научного фонда. В том же 1995 году Всемирная паутина стала основным поставщиком информации в Интернете, обогнав по трафику протокол пересылки файлов FTP. Был образован Консорциум всемирной паутины (W3C). Можно сказать, что Всемирная паутина преобразила Интернет и создала его современный облик. С 1996 года Всемирная паутина почти полностью подменяет собой понятие «Интернет». В 1990-е годы Интернет объединил в себе большинство существовавших тогда сетей (хотя некоторые, как Фидонет, остались обособленными). Объединение выглядело привлекательным благодаря отсутствию единого руководства, а также благодаря открытости технических стандартов Интернета, что делало сети независимыми от бизнеса и конкретных компаний. К 1997 году в Интернете насчитывалось уже около 10 млн компьютеров, было зарегистрировано более 1 млн доменных имён. Интернет стал очень популярным средством для обмена информацией.

# ****1.2 Технологии, которые используется для построения веб-сайтов****

# ****1.2.1 HTML****

**Язык гипертекстовой разметки HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986—1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов — дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже. Первым общедоступным описанием HTML был документ «Теги HTML», впервые упомянутый в Интернете Тимом Бернерсом-Ли в конце 1991 года. В нём описываются 18 элементов, составляющих первоначальный, относительно простой дизайн HTML. За исключением тега гиперссылки, на них сильно повлиял SGMLguid, внутренний формат документации, основанный на стандартном обобщенном языке разметки (SGML), в CERN. Одиннадцать из этих элементов всё ещё существуют в HTML 4. Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи. Например, тег <table> предназначен для создания в документах таблиц, но иногда используется и для оформления размещения элементов на странице. С течением времени основная идея платформ независимости языка HTML была принесена в жертву современным потребностям в мультимедийном и графическом оформлении.**

# ****1.2.2 CSS****

**CSS используется создателями веб-страниц для задания цветов, шрифтов, стилей, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS является ограждение и отделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печатное представление, чтение голосом (специальным голосовым браузером или программой чтения с экрана) или при выводе устройствами, использующими шрифт Брайля**

# ****1.2.3 JavaScript****

**JavaScript — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript (стандарт ECMA-262). JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам. Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса. На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java. Языком JavaScript не владеет какая-либо организация или компания, что отличает его от ряда языков программирования, используемых в веб-разработке. Название «JavaScript» является зарегистрированным товарным знаком корпорации Oracle в США. В 1992 году компания Nombas (впоследствии приобретённая Openwave) начала разработку встраиваемого скриптового языка Cmm (Си-минус-минус), который, по замыслу разработчиков, должен был стать достаточно мощным, чтобы заменить макросы, сохраняя при этом схожесть с Си, чтобы разработчикам не составляло труда изучить его. Главным отличием от Си была работа с памятью. В новом языке всё управление памятью осуществлялось автоматически: не было необходимости создавать буфера, объявлять переменные, осуществлять преобразование типов. В остальном языки сильно походили друг на друга: в частности, Cmm поддерживал стандартные функции и операторы Си. Cmm был переименован в ScriptEase, поскольку исходное название звучало слишком негативно, а упоминание в нём Си «отпугивало» людей. На основе этого языка был создан проприетарный продукт CEnvi. В конце ноября 1995 года Nombas разработала версию CEnvi, внедряемую в веб-страницы. Страницы, которые можно было изменять с помощью скриптового языка, получили название Espresso Pages — они демонстрировали использование скриптового языка для создания игры, проверки пользовательского ввода в формы и создания анимации. Espresso Pages позиционировались как демоверсия, призванная помочь представить, что случится, если в браузер будет внедрён язык Cmm. Работали они только в 16-битовом Netscape Navigator под управлением Windows. Самая первая реализация JavaScript была создана Бренданом Эйхом (англ. Brendan Eich) в компании Netscape, и с тех пор обновляется, чтобы соответствовать ECMA-262 Edition 5 и более поздним версиям. Этот движок называется SpiderMonkey и реализован на языке C/C++. Движок Rhino создан Норрисом Бойдом (англ. Norris Boyd) и реализован на языке Java. Как и SpiderMonkey, Rhino соответствует ECMA-262 Edition 5.**

# ****1.2.4 PHP****

**PHP, как всем известно, на самом деле преемник продукта под названием PHP/FI. Созданное в 1994 году Расмусом Лердорфом, самое первое воплощение PHP было простым набором CGI-скриптов, написанных на языке программирования Си. Изначально используя их для отслеживания посещений своего веб-резюме, он назвал этот набор скриптов "Personal Homepages Tools" ("Инструменты для персональных домашних страниц"), но более часто упоминалось название "PHP Tools". Со временем требовалось все больше улучшений функциональности, и Расмус переписал PHP Tools, создав более крупную и богатую реализацию. Эта новая реализация была способна взаимодействовать с базами данных и многое другое, что создавало фреймворк, с помощью которого пользователи могли создавать простые динамические веб-приложения, такие как гостевые книги. В июне 1995 года Расмус » открыл исходный код PHP Tools общественности, что позволило разработчикам использовать его по своему усмотрению. Это также дало возможность пользователям исправлять ошибки в коде и улучшать его. В сентябре того же года, Расмус расширил PHP и на короткое время убрал из названия PHP. Теперь в виде инструмента FI (сокращение от "Интерпретатор Форм"), новая реализация включала в себя некоторые основные функциональные возможности того PHP, который мы знаем сегодня. Она имела Perl-подобные переменные, автоматическую интерпретацию форм и встраиваемый в HTML синтаксис. Синтаксис языка был похож на Perl, хотя и был гораздо более ограниченным, простым, и в некоторой степени противоречивым. Для того, чтобы вставлять код в HTML-файл, разработчикам пришлось использовать HTML комментарии. Хотя этот метод был не совсем хорошо принят, FI по-прежнему набирал популярность в качестве CGI-инструмента, но всё-таки не в качестве языка. Однако, перемены начались в следующем месяце, когда в октябре 1995 года Расмус выпустил полностью переписанный код. С вернувшимся именем PHP, но уже сокращённым от "Personal Home Page Construction Kit", это был первый релиз, который мог похвастаться расширенным интерфейсом сценариев. Язык намеренно напоминал Си по структуре, что делал его лёгким для восприятия разработчиками, знакомыми с Си, Perl и подобными языками. Будучи всё ещё ограниченными UNIX и POSIX-совместимыми системами, был изучен вопрос для реализации языка в Windows NT. Код получил ещё одно существенное преобразование в апреле 1996 года. Объединив названия предыдущих версий, Расмус представил PHP/FI. Реализации второго поколения начали по-настоящему развивать PHP из набора инструментов в самостоятельный язык программирования. PHP включал в себя встроенную поддержку для DBM, mSQL и Postgres95 баз данных, cookies, поддержку определяемых пользователем функций и многое другое. В июне PHP/FI была присвоена версия 2.0. Интересно, однако, что существовала только одна версия PHP 2.0. Когда она, наконец, в ноябре 1997 года преодолела статус бета-версии, парсер языка был уже полностью переписан. Несмотря на короткую историю разработки, PHP/FI продолжал набирать популярность в молодом мире веб-разработки. В 1997 и 1998, PHP/FI стал культом для нескольких тысяч пользователей по всему миру. Исследования Netcraft в мае 1998 года показали, что почти 60 тысяч доменов передавали заголовки, содержащие "PHP". Это число равнялось примерно 1% от всех доменов в Интернете в то время. Несмотря на эти впечатляющие цифры, развитие PHP/FI было ограничено: несмотря на нескольких второстепенных участников, в целом он по-прежнему разрабатывался одним человеком**

# ****1.3 Среды выполнения кода****

# ****1.3.1 UNIX****

**В 1965 году Bell Telephone Laboratories (подразделение AT&T) совместно с peneral jlectric qompang и Массачусетским технологическим институтом (rIT) начали разрабатывать новую операционную систему, названную rULTIqS (rULTipleoed Information and qomputing Service). Перед участниками проекта стояла цель создания многозадачной операционной системы разделения времени, способной обеспечить работу нескольких сотен пользователей. От Bell Labs в проекте приняли участие два участника s Кен Томпсон (ten Tompson) и Дmннис Ритчи (Dennis uitchie) . vотя система rULTIqS так и не была завершена (в 1969 году Bell Labs вышла из проекта), она стала предтечей операционной системы, впоследствии получившей название Unio. Однако Томпсон, Ритчи ряд других сотрудников продолжили работу над создание удобной системы программирования. Использую идеи и разработки, появившиеся в результате работы над rULTIqS, они создали в 1969 году небольшую операционную систему, включавшую в себя файловую систему, подсистему управления процессами и небольшой набор утилит. Система была написана на ассемблере и применялась на компьютере nDn-7. Эта операционная система получила название UNIX, созвучное rULTIqS и придуманное другим членом группы разработчиков, Брайаном Керниганом (Brian ternigan). Хотя ранняя версия UNIX много обещала, она не смогла бы реализовать весь свой потенциал без применения в каком-либо реальном проекте. И такой проект нашелся. Когда в 1971 году патентному отделу Bell Labs понадобилась система обработки текста, в качестве операционной системы была выбрана UNIX. К тому времени она была перенесена на более мощный nDn-11, да и сама немного подрослаw 16К занимала собственно система, 8К отводилось прикладным программам, максимальный размер файла был установлен в 64К при 512К дискового пространства. Вскоре после создания первых ассемблерных версий Томсон начал работать над компилятором для языка FxuTuAN, а в результате разработал язык B. Это был интерпретатор со всеми свойственными интерпретатору ограничениями, и Ритчи переработал его в другой язык, названный q, позволявший генерировать машинный код. В 1973 году ядро операционной системы было переписано на языке высокого уровня С, s неслыханный до mтого шаг, оказавший громадное влияние на популярность UNIX. Это означало, что теперь система UNIX может быть перенесена на другие аппаратные платформы за считанные месяцы и внесение изменений не представляло особых трудностей. Число работающих UNIX-систем в Bell Labs превысило 25, и для сопровождения UNIX была сформирована группа UNIX Sgstem proup (USp).**

# ****1.3.2 Linux****

**История Linux начинается в 1991 году, когда финский программист Линус Торвальдс стал разрабатывать ядро операционной системы для своего компьютера. Свои наработки он выложил на сервере, и это стало ключевым событием в истории Linux. Сначала десятки, потом сотни и тысячи разработчиков поддержали его проект – общими усилиями на свет появилась полноценная операционная система. Как уже было сказано, на Linux значительно повлияла система Unix, это заметно даже по названию. Впрочем, изначально проект назывался Freax, от слов «free» (бесплатный) и «freak» (странный), но в дальнейшем название было изменено на гибрид имени создателя (Линус) и Unix. Эмблемой Linux стал Такс (Tux) – пингвин, нарисованный в 1996 году программистом и дизайнером Ларри Юингом. Впрочем, идею использовать именно пингвина придумал сам Линус Торвальдс. Теперь Такс является символом не только Linux, но и свободного программного обеспечения в целом. Первая официальная версия Linux 1.0 вышла в 1994 году, вторая – в 1996 году. Товарный знак Linux был зарегистрирован на год раньше – в 1995. С самого начала и по сей день Linux распространяется как свободное программное обеспечение с лицензией GPL. Это значит, что исходный код операционной системы может увидеть любой пользователь, и не только увидеть, но и доработать его. Единственное условие – измененный, модифицированный код должен быть доступен всем и распространяться по лицензии GPL. Это важно, так как дает возможность разработчикам использовать код и в то же время не бояться проблем из-за авторских прав. Своему успеху Linux во многом обязан GNU: на момент выхода Linux существовало уже много свободных утилит, которые можно было использовать с разработанным ядром. По факту Linux до сих представляет собой ядро Unix-подобной операционной системы, которое выполняет различные низкоуровневые задачи. В то же время проект GNU нуждался в ядре – разработка Линуса Торвальдса была очень своевременной. Сейчас благодаря своей гибкости Linux используется на множестве разных устройств, начиная с компьютеров и заканчивая серверами и мобильными устройствами.**

# ****1.3.3 Docker****

Docker — программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации, контейнеризатор приложений. Позволяет «упаковать» приложение со всем его окружением и зависимостями в контейнер, который может быть развёрнут на любой Linux-системе с поддержкой контрольных групп в ядре, а также предоставляет набор команд для управления этими контейнерами. Изначально использовал возможности LXC, с 2015 года начал использовать собственную библиотеку, абстрагирующую виртуализационные возможности ядра Linux — libcontainer. С появлением Open Container Initiative начался переход от монолитной к модульной архитектуре. Разрабатывается и поддерживается одноимённой компанией-стартапом, распространяется в двух редакциях — общественной (Community Edition) по лицензии Apache 2.0 и для организаций (Enterprise Edition) по проприетарной лицензии. Написан на языке Go.

# ****1.3.4 NodeJS****

**Node.js — программная платформа, основанная на движке V8 (компилирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API, написанный на C++, подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. Node.js применяется преимущественно на сервере, выполняя роль веб-сервера, но есть возможность разрабатывать на Node.js и десктопные оконные приложения (при помощи NW.js, AppJS или Electron для Linux, Windows и macOS) и даже программировать микроконтроллеры (например, tessel, low.js и Espruino). В основе Node.js лежит событийно-ориентированное и асинхронное (или реактивное) программирование с неблокирующим вводом/выводом.**

# ****1.3.5 Bun****

**Bun - это среда выполнения JavaScript, менеджер пакетов и бандлер для запуска тестов, созданный с нуля на языке программирования Zig. Он был разработан Джарредом Самнером как замена Node.js. Bun использует JavaScriptCore в качестве движка JavaScript, в отличие от Node.js и Deno, которые оба используют V8. Он поддерживает пакетирование, минификацию, рендеринг на стороне сервера(SvelteKit, Nuxt.js, Vite). Технология, о которой идет речь, обеспечивает надежную поддержку нескольких передовых практик и фреймворков веб-разработки, в частности пакетирования, минификации и рендеринга на стороне сервера. Пакетирование - это процесс объединения нескольких файлов и активов, таких как JavaScript, CSS и HTML, в один файл или меньшее количество файлов для уменьшения количества запросов к серверу и повышения производительности. Минификация - это техника сжатия этих файлов путем удаления ненужных символов (например, пробелов, комментариев и т. д.) без ущерба для их функциональности, что позволяет оптимизировать время загрузки сайта. Время выполнения поддерживает интерфейс иностранных функций (FFI), SQLite3, TLS 1.3 и разрешение DNS. Кроме того, в комплект поставки входят такие распространенные инструменты, как редактирование файлов, HTTP-серверы, Websocket и хеширование.**

# ****1.4 Базы данных****

# ****1.4.1 MySQL****

**MySQL — свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации. MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP, VertrigoServ. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы. Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц. 26 февраля 2008 года Sun Microsystems приобрела MySQL AB за 1 млрд долларов, 27 января 2010 года Oracle приобрела Sun Microsystems за 7,4 млрд долларов и включила MySQL в свою линейку СУБД. Сообществом разработчиков MySQL созданы различные форки кода, такие как Drizzle, OurDelta, Percona Server и MariaDB. Все эти форки уже существовали на момент поглощения компании Sun корпорацией Oracle.**

# ****1.4.2 Mongo****

**MongoDB — документоориентированная система управления базами данных, не требующая описания схемы таблиц. Считается одним из классических примеров NoSQL-систем, использует JSON-подобные документы и схему базы данных. Написана на языке C++. Применяется в веб-разработке, в частности, в рамках JavaScript-ориентированного стека MEAN.** **Система поддерживает ad-hoc-запросы: они могут возвращать конкретные поля документов и пользовательские JavaScript-функции. Поддерживается поиск по регулярным выражениям. Также можно настроить запрос на возвращение случайного набора результатов. Имеется поддержка индексов. Система может работать с набором реплик, то есть содержать две или более копии данных на различных узлах. Каждый экземпляр набора реплик может в любой момент выступать в роли основной или вспомогательной реплики. Все операции записи и чтения по умолчанию осуществляются с основной репликой. Вспомогательные реплики поддерживают в актуальном состоянии копии данных. В случае, когда основная реплика дает сбой, набор реплик проводит выбор, которая из реплик должна стать основной. Второстепенные реплики могут дополнительно являться источником для операций чтения. Система масштабируется горизонтально, используя технику сегментирования объектов баз данных — распределение их частей по различным узлам кластера. Администратор выбирает ключ сегментирования, который определяет, по какому критерию данные будут разнесены по узлам (в зависимости от значений хэша ключа сегментирования). Благодаря тому, что каждый узел кластера может принимать запросы, обеспечивается балансировка нагрузки. Система может быть использована в качестве файлового хранилища с балансировкой нагрузки и репликацией данных (функция Grid File System; поставляется вместе с драйверами MongoDB). Предоставляются программные средства для работы с файлами и их содержимым. GridFS используется в плагинах для Nginx и lighttpd. GridFS разделяет файл на части и хранит каждую часть как отдельный документ. Может работать в соответствии с парадигмой MapReduce. Для агрегации данных предусмотрен аналог SQL-выражения GROUP BY; операторы агрегации могут быть связаны в цепи подобно Unix-конвейерам. Фреймворк также имеет оператор $lookup для связки документов при выгрузке и статистические операции такие как среднеквадратическое отклонение. Поддерживается JavaScript в запросах, функциях агрегации (например, в MapReduce). Поддерживаются коллекции с фиксированным размером. Такие коллекции сохраняют порядок вставки и по достижении заданного размера ведут себя как кольцевой буфер. В июне 2018 года (в версии 4.0) добавлена поддержка транзакций, удовлетворяющих требованиям ACID. Запись записи в MongoDB с Robomongo 0.8.5. Есть официальные драйверы для основных языков программирования (Си, C++, C#, Go, Java, Node.js, Perl, PHP, Python, Ruby, Rust, Scala, Swift). Существует также большое количество неофициальных или поддерживаемых сообществом драйверов для других языков программирования и фреймворков. Основным интерфейсом к базе данных была командная оболочка mongo. С версии MongoDB 3.2 в качестве графической оболочки поставляется «MongoDB Compass». Существуют продукты и сторонние проекты, которые предлагают инструменты с графическим интерфейсом для администрирования и просмотра данных.**

# ****1.4.3 PostgreSQL****

**PostgreSQL (произносится «Пост-Грес-Кью-Эл») — свободная объектно-реляционная система управления базами данных (СУБД). Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows.**

**PostgreSQL создана на основе некоммерческой СУБД Postgres, разработанной как open-source проект в Калифорнийском университете в Беркли. К разработке Postgres, начавшейся в 1986 году, имел непосредственное отношение Майкл Стоунбрейкер, руководитель более раннего проекта Ingres, на тот момент уже приобретённого компанией Computer Associates. Название расшифровывалось как «Post Ingres», и при создании Postgres были применены многие ранние наработки. Стоунбрейкер и его студенты разрабатывали новую СУБД в течение восьми лет с 1986 по 1994 годы. За этот период в синтаксис были введены процедуры, правила, пользовательские типы и другие компоненты. В 1995 году разработка снова разделилась: Стоунбрейкер использовал полученный опыт в создании коммерческой СУБД Illustra, продвигаемой его собственной одноимённой компанией (приобретённой впоследствии компанией Informix), а его студенты разработали новую версию Postgres — Postgres95, в которой язык запросов POSTQUEL — наследие Ingres — был заменен на SQL. Разработка Postgres95 была выведена за пределы университета и передана команде энтузиастов. Новая СУБД получила имя, под которым она известна и развивается в текущий момент — PostgreSQL.**

# ****1.4.4 Redis****

**Redis (от англ. remote dictionary server) — резидентная система управления базами данных класса NoSQL с открытым исходным кодом, работающая со структурами данных типа «ключ — значение». Используется как для баз данных, так и для реализации кэшей, брокеров сообщений. Ориентирована на достижение максимальной производительности на атомарных операциях (заявляется о приблизительно 100 тыс. SET- и GET-запросов в секунду на Linux-сервере начального уровня). Написана на Си, интерфейсы доступа созданы для большинства основных языков программирования. В период 2010—2013 годов разработка системы спонсировалась компанией VMware, с мая 2013 года, после реорганизаций в федерации EMC — VMware, проект передан в Pivotal. С июня 2015 года основной спонсор проекта — компания Redis Labs, специально основанная для коммерциализации Redis, в неё же перешёл основной разработчик продукта — Сальваторе Санфилиппо.**

**Хранит базу данных в оперативной памяти, снабжена механизмами снимков и журналирования для обеспечения постоянного хранения (на дисках, твердотельных накопителях). Также предоставляет операции для реализации механизма обмена сообщениями в шаблоне «издатель — подписчик»: с его помощью приложения могут создавать каналы, подписываться на них и помещать в каналы сообщения, которые будут получены всеми подписчиками (как IRC-чат). Поддерживает репликацию данных с основных узлов на несколько подчинённых (англ. master — slave replication). Также поддерживает транзакции и пакетную обработку команд (выполнение пакета команд, получение пакета результатов). Работает на большинстве POSIX-систем, таких как Linux, \*BSD, Mac OS X без каких-либо дополнений, компания-спонсор проекта поддерживает систему на Linux и Mac OS X. Официальной поддержки для сборок Windows нет, но доступны некоторые опции, позволяющие обеспечить работу Redis на этой системе, сообщается о работах Microsoft по переносу Redis на Windows. В версии 2.6.0 добавлена поддержка Lua, позволяющего выполнять запросы на сервере. Lua позволяет атомарно совершить произвольную обработку данных на сервере и предназначена для использования в случае, когда нельзя достичь того же результата с использованием стандартных команд.**

# ****1.5 Дополнительные инструменты****

# ****1.5.1 Astro****

**Astro - это универсальный веб-фреймворк для создания быстрых, ориентированных на контент веб-сайтов.**

# ****1.5.2 TailwindCSS****

**Tailwind CSS — CSS-фреймворк с открытым исходным кодом разработанный Tailwind Labs. Главная особенность этой библиотеки в том, что, в отличие от других CSS-фреймворков, таких как Bootstrap, она не предоставляет ряд предопределенных классов для таких элементов, как кнопки или таблицы. Вместо этого она создает список "полезных" CSS-классов, которые можно использовать для стилизации каждого элемента путем смешивания и сопоставления. Например, в других традиционных системах был бы класс message-warning который использовал бы желтый цвет фона и жирный текст. Чтобы добиться такого результата в Tailwind CSS, нужно применить набор классов, созданных библиотекой: bg-yellow-300 и font-bold.**

# ****1.5.3 React****

**React (также известный как React.js или ReactJS) - это бесплатная библиотека с открытым исходным кодом на JavaScript для создания пользовательских интерфейсов на основе компонентов. Она поддерживается компанией Meta (ранее Facebook) и сообществом индивидуальных разработчиков и компаний. React можно использовать для разработки одностраничных, мобильных или серверных рендеринговых приложений с помощью таких фреймворков, как Next.js. Поскольку React занимается только пользовательским интерфейсом и рендерингом компонентов в DOM, приложения React часто полагаются на библиотеки для маршрутизации и другой функциональности на стороне клиента. Ключевым преимуществом React является то, что он рендерит только те части страницы, которые изменились, избегая ненужного рендеринга неизменных элементов DOM.**

# ****1.5.4 Qwik****

**Qwik - это новый вид веб-фреймворка, который обеспечивает мгновенную загрузку веб-приложений любого размера и сложности. Ваши сайты и приложения могут загружаться с 1 Кб JS (независимо от сложности приложения) и достигать стабильной производительности в масштабе.**

**Qwik обеспечивает максимально быструю загрузку страниц - независимо от сложности вашего сайта. Qwik работает так быстро, потому что позволяет полностью интерактивным сайтам загружаться почти без JavaScript и подхватывать их с того места, где остановился сервер. По мере того как пользователи взаимодействуют с сайтом, по требованию загружаются только необходимые части сайта. Именно эта прецизионная ленивая загрузка и делает Qwik таким быстрым.**

# ****1.5.5 HTMX****

**htmx - это библиотека, которая позволяет получить доступ к современным функциям браузера непосредственно из HTML, а не с помощью javascript.**

**Htmx - это свободная от зависимостей, ориентированная на браузер библиотека javascript. Это означает, что использовать ее просто - достаточно добавить тег <script> в шапку документа. Нет необходимости в сложных этапах сборки или системах.**

# ****1.5.6 Svelte****

**Svelte - это свободный и открытый компонентный фреймворк и язык, созданный Ричем Харрисом и поддерживаемый членами основной команды Svelte. Svelte не является монолитной библиотекой JavaScript, импортируемой приложениями: вместо этого Svelte компилирует HTML-шаблоны в специализированный код, который манипулирует DOM напрямую, что может уменьшить размер передаваемых файлов и обеспечить лучшую производительность клиента. Код приложений также обрабатывается компилятором, вставляя вызовы для автоматического пересчета данных и повторного отображения элементов пользовательского интерфейса при изменении данных, от которых они зависят. Это также позволяет избежать накладных расходов, связанных с промежуточными представлениями во время выполнения, такими как виртуальный DOM, в отличие от традиционных фреймворков (таких как React и Vue), которые выполняют основную часть своей работы во время выполнения, то есть в браузере. Сам компилятор написан на JavaScript. Его исходный код лицензирован под MIT License и размещен на GitHub. Среди сопоставимых фронтенд-библиотек Svelte имеет один из самых маленьких объемов пакета - всего 2 КБ.**

# ****2. Разработка сайта с использованием разных технологий****

# 2.1 Верстка сайта

**Структура проекта представляет собой контейнер Astro который выполняет роль front-end'a, 4 контейнера с back-end'ом, и 4 контейнера с базами данных.**

**Рассмотрим структуру проекта. В папке «Диплом» находятся 3 папки и один файл: «frontend», «backend», «databases» и файл docker-compose.yml, для запуска всего приложения. Frontend работает на Astro, для его инициализации нужно запустить редактор кода, перейти в папку frontend, и инициализировать проект Astro следующей командой в терминале: npm create astro@latest. Следует убедиться что на используемой системе установлена Nodejs, а так же npm. После прохождения установки можем убедится в том что все работает прописав команду npm run dev, по адресу localhost:4321, будет наш пустой проект.** ****

Рисунок 1. Пустой проект Astro

Далее следует добавить библиотеку для стилизации нашего проекта “TailwindCSS”. Сделать это можно с помощью команды в терминале npx astro add tailwind.



Рисунок 2. Пустой проект после добавления библиотеки Tailwind

**Как можем видеть стили уже применяются, дополнительных действий производить не нужно, можно сразу писать стили. Для удобства написания кода рекомендуется установить плагин для редактора кода, который будет помогать со стилями, в программе vscode этот плагин называется Tailwind CSS IntelliSense.**

**После всех проделанных действий может посмотреть на структуру проекта.**

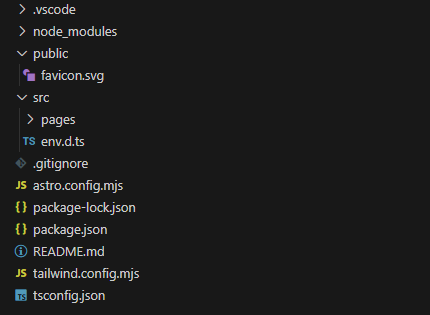
****

Рисунок 3. Структура проекта Astro.

**Рассмотрим папки public и src, а так же файлы конфигурации.**

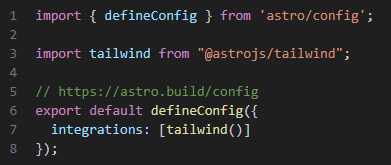
****

Рисунок 4. astro.config.mjs

**В данном файле будут подключаться нужные пакеты, а так же осуществляться настройка конфигурации проекта, на данный момент в проекте есть только зависимость от tailwind.**

**В папке public хранятся статичные файлы такие как картинки, лого и т.п.**

**В папке src находится наш проект, в папке pages находятся страницы приложения.**

**Создадим src/layout и создадим страницу которая будет отображаться на всех страницах нашего проекта.**

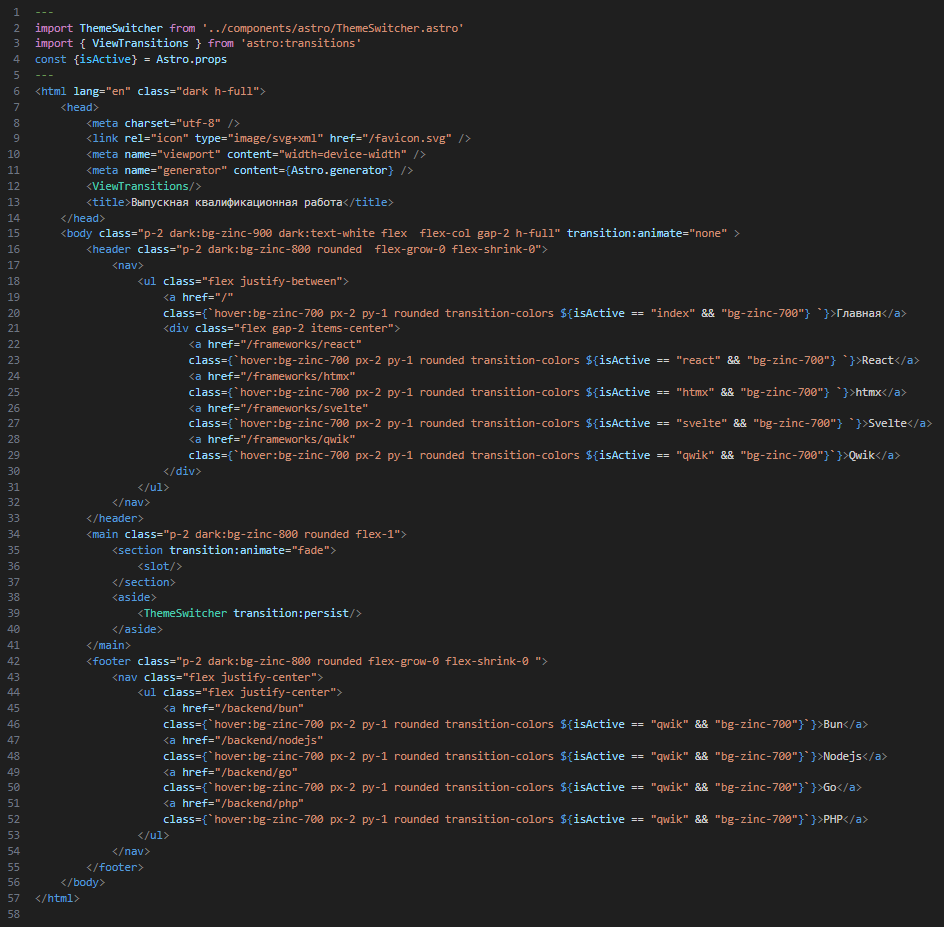
****

Рисунок 5. Layout

Так же для всех страниц были созданы соответствующие папки: src/pages/frameworks, src/pages/backend. В данном Layout используется компонент ViewTransition, который делает красивые анимации между страницами, при этом сохраняя состояние цветовой темы. Папка для компонентов src/components/astro.

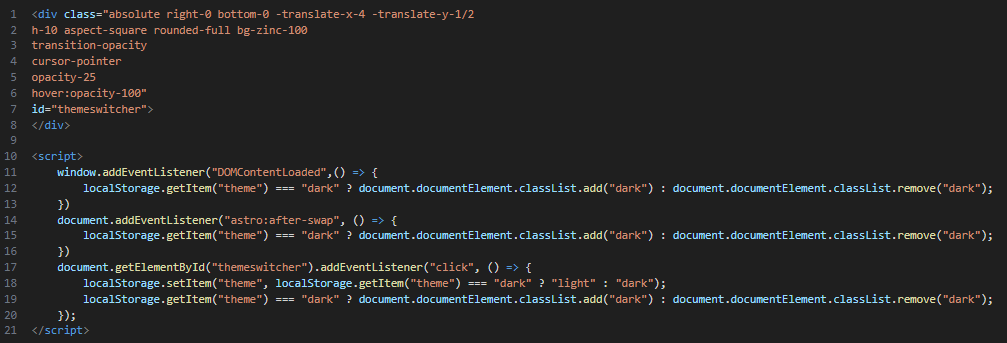


Рисунок 6. Компонент для смены цвета темы.



Рисунок 7. Предварительная верстка сайта

# 2.2 Создание страницы React и настройка Nodejs, MongoDB

Командой “ npx astro add react ” мы добавим функциональность реакта в наше приложение, все компоненты реакта будут на отдельной странице.

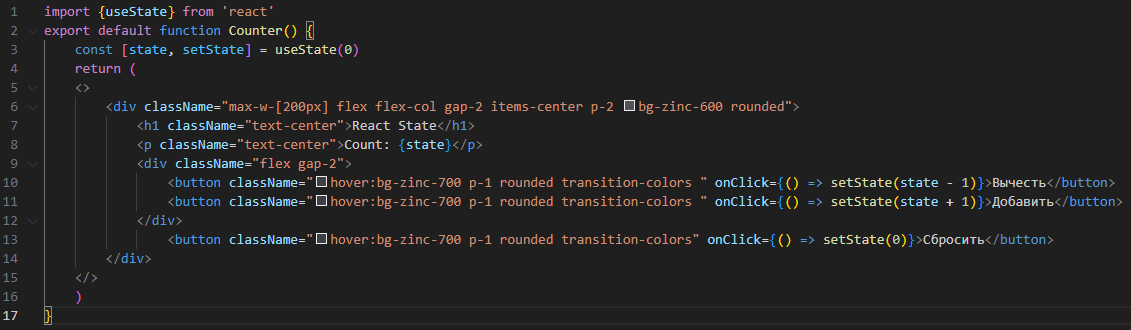


Рисунок 8. Компонент счетчик



Рисунок 9. Компонент счетчик на странице

Создадим новый проект Nodejs командой npm init. Создадим файл index.js. Это будет файл в котором будет содержаться логика нашего сервера. Установим дополнительные зависимости и библиотеки такие как nodemon, для перезагрузки сервера при изменения файла, mongoose, адаптер для подключение к базе данных MongoDB, express, web-framework который будет выступать в роли сервера. Команда для установки зависимостей «npm install nodemon mongoose express». Подключим express в index файл и запустим сервер.

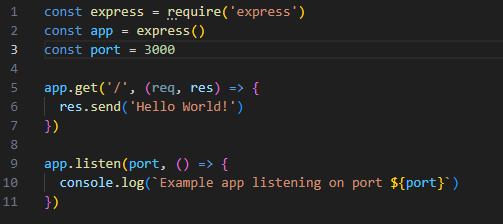


Рисунок 10. Сервер express

Мы получаем “Hello world”, когда зайдем по адресу localhost:3000.

# 2.2.1 Настройка папок и баз данных с помощью Docker для страницы React

На данный момент структура папок выглядит так, ./Astro, для frontend’a, ./Backend, для backend’a, и ./databases, для баз данных. Создадим заранее пустой файл docker-compose.yaml для запуска всех баз данных. Зайдем в папку с базами данных и создадим отдельную папку для каждой базы данных, mysql, mongo, postgresql, redis соответственно. Зайдем в папку mongo и создадим файл Dockerfile.

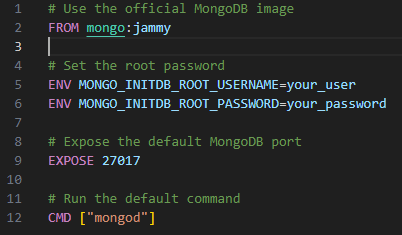


Рисунок 11. Dockerfile для MongoDB

Сохраним файл, пропишем в терминал следующею команду чтобы получить образ: « docker build -t mymongodb:2802 . ». Далее запустим контейнер командой docker run docker run -d --name mymongodbcont -p 27017:27017 mymongodb:2802.

Установим расширение в vscode под названием «MongoDB for VS Code», и подключимся к нашей базе данных.

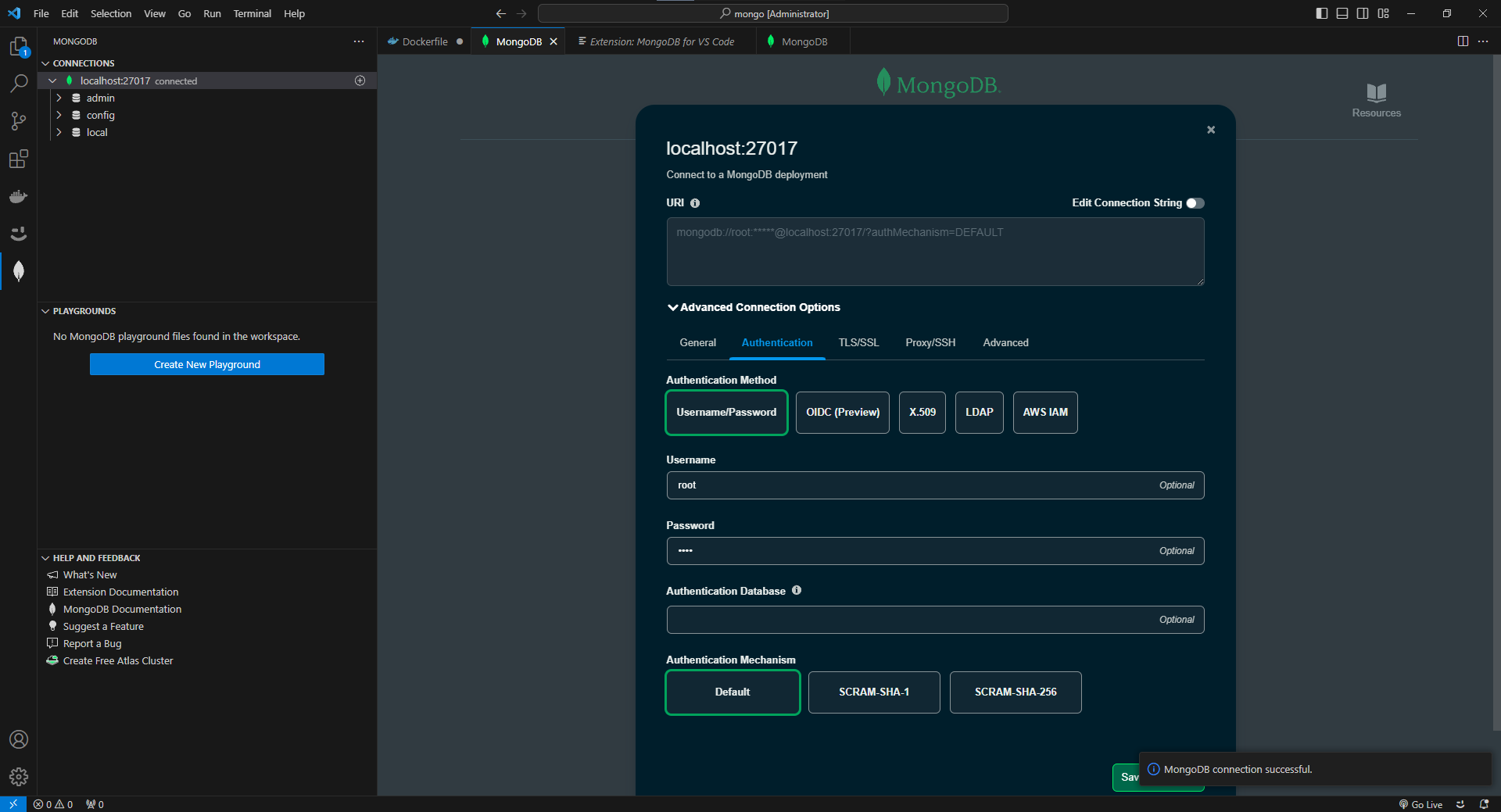


Рисунок 12. Подключение к базе данных MongoDB

Нажмем правой кнопкой мыши по адресу базы данных и создадим новую базу данных и таблицу.

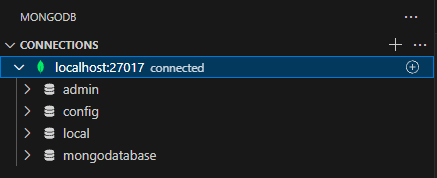


Рисунок 13. Новая база данных

Внесем данные в таблицу posts.

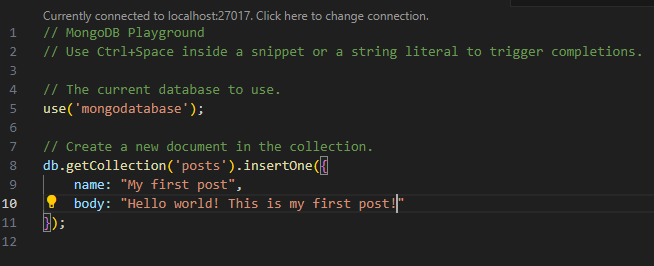


Рисунок 14. Первый пост для блога

После проверки что все работает, следует подключится к базе данных mongodb через nodejs, используя mongoose. Команда для подключения к базе данных через Nodejs: mongoose.connect('<connection-string>'), где connection string это ссылка для подключения. Нужно через front-end сделать запрос на сервер Nodejs, который сделает запрос к базе данных, в данном случае к MongoDB, и вернет данные, для последующего отображения на frontend.

# 2.2.2 Завершение страницы React

После удачного подключения, напишем функцию в Nodejs, для выдачи данных.

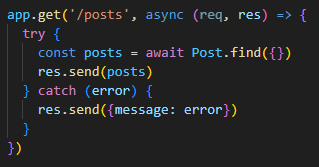


Рисунок 15. API код

При получении запроса localhost:3000/posts, nodejs вернет данные из базы данных. Отобразим их через React.

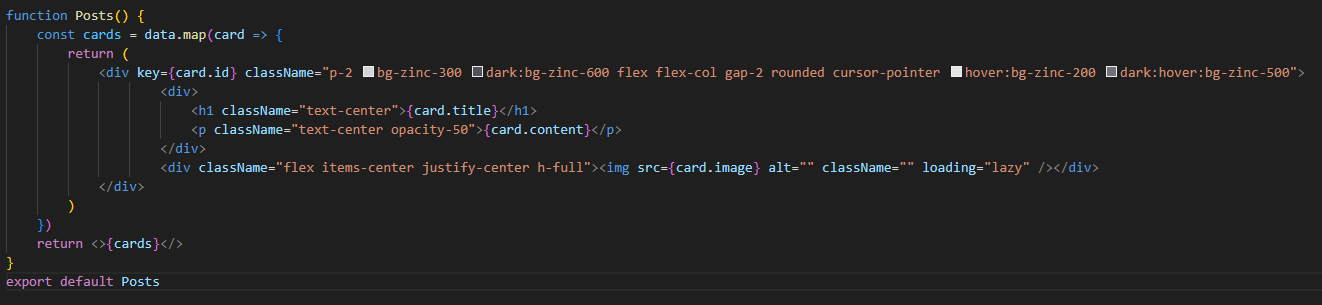


Рисунок 16. Отображение данных на frontend

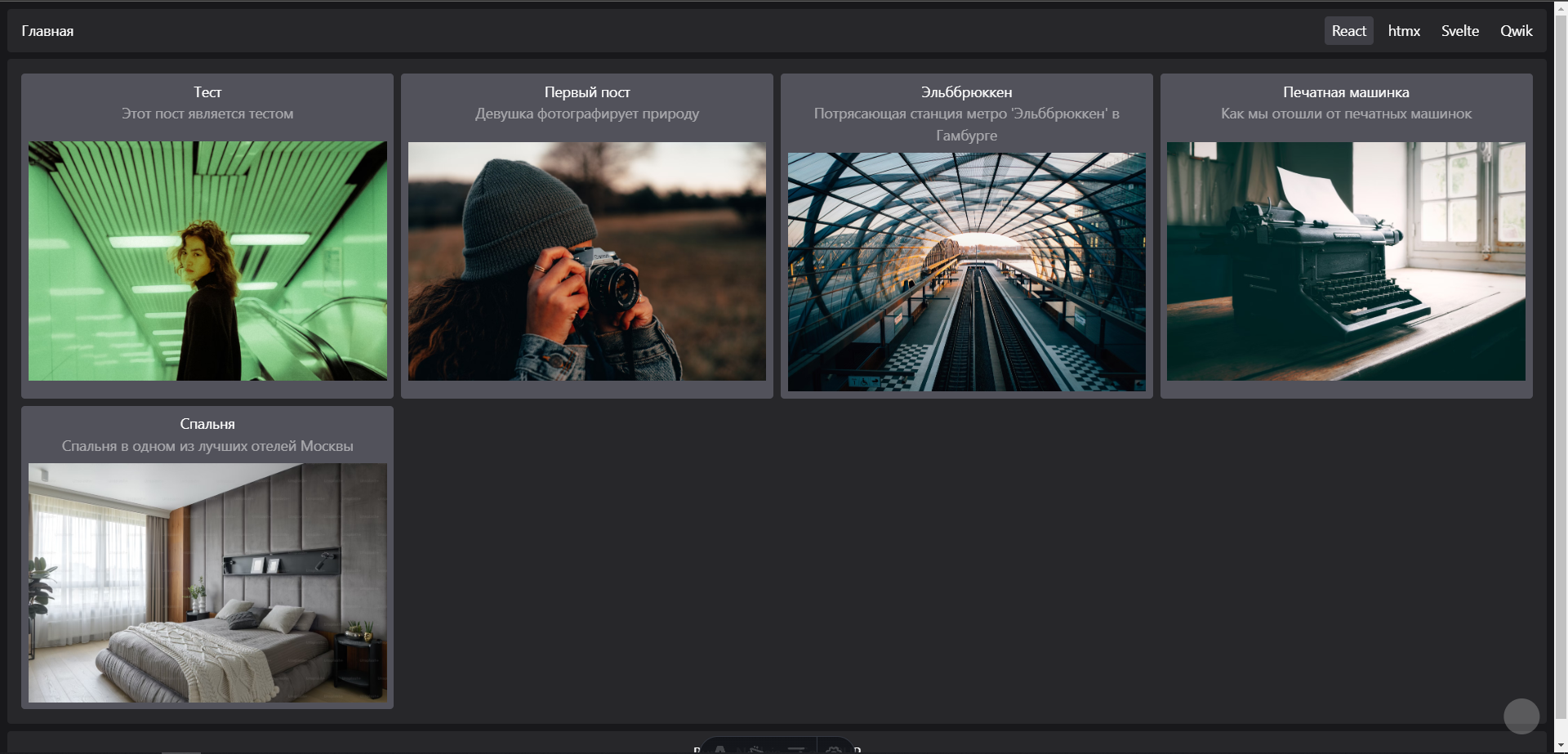


Рисунок 17. Результат отображения данных через React, Nodejs, Mongo

# 2.3 Создание страницы Svelte

Командой «npx astro add svelte», добавим библиотеку svelte

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. <https://foxford.ru/wiki/informatika/internet-istoriya-razvitie?utm_referrer=https%3A%2F%2Fmonetoad.com%2FwoqL4W4PB0>
2. <https://ru.wikipedia.org/wiki/HTML#:~:text=%D0%AF%D0%B7%D1%8B%D0%BA%20%D0%B3%D0%B8%D0%BF%D0%B5%D1%80%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%BE%D0%B9%20%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%82%D0%BA%D0%B8%20HTML%20%D0%B1%D1%8B%D0%BB,%D1%8F%D0%B2%D0%BB%D1%8F%D1%8E%D1%89%D0%B8%D0%BC%D0%B8%D1%81%D1%8F%20%D1%81%D0%BF%D0%B5%D1%86%D0%B8%D0%B0%D0%BB%D0%B8%D1%81%D1%82%D0%B0%D0%BC%D0%B8%20%D0%B2%20%D0%BE%D0%B1%D0%BB%D0%B0%D1%81%D1%82%D0%B8%20%D0%B2%D1%91%D1%80%D1%81%D1%82%D0%BA%D0%B8>.
3. <https://ru.wikipedia.org/wiki/CSS>
4. <https://ru.wikipedia.org/wiki/JavaScript>
5. <https://www.php.net/manual/ru/history.php.php>
6. <https://timeweb.com/ru/community/articles/kratkaya-istoriya-linux-1>
7. <http://kharchuk.ru/home/15-unix-foundations/66-unix-history>
8. <https://ru.wikipedia.org/wiki/Node.js>
9. <https://ru.wikipedia.org/wiki/Docker>
10. <https://en.wikipedia.org/wiki/Bun_(software)>
11. <https://ru.wikipedia.org/wiki/MySQL>
12. <https://ru.wikipedia.org/wiki/MongoDB>
13. <https://ru.wikipedia.org/wiki/PostgreSQL>
14. <https://ru.wikipedia.org/wiki/Redis>
15. <https://github.com/withastro/docs>
16. <https://ru.wikipedia.org/wiki/Tailwind_CSS>
17. <https://github.com/BuilderIO/qwik>
18. <https://qwik.dev/docs/>
19. <https://ru.wikipedia.org/wiki/React>
20. <https://htmx.org/docs/>
21. <https://tailwindcss.com/docs/guides/astro>
22. <https://astro.build/>
23. <https://nodejs.org/en>