СОДЕРЖАНИЕ

|  |  |  |
| --- | --- | --- |
| 1. | ВВЕДЕНИЕ…………………………………………………………………………………………… | 0 |
| 1. | Описание задачи на проект…………………………………….…………..………….. | 1 |
|  | 1.1. План разработки дипломного проекта……………………….………………………….… | 4 |
|  | 1.2. Выбор программного обеспечения……………………….………………………….… | 5 |
| 2. | Этапы разработки проекта……………………….………………………….………… | 6 |
|  | 2.1. Проектирование базы данных……………………….………………………….… | 7 |
|  | 2.2. Создание веб страницы проекта. Его структура…………….………………………….… | 8 |
|  | 2.3. Реализация интерактивных элементов на сайте……………………………… | 6 |
|  | 2.4. Реестр новых поступлений……………………………………………………………… | 7 |
|  | 2.5. Адаптивная часть веб страницы……………….…………………….…………………….…… | 9 |
|  | 2.6. Создание мобильного приложения……………….…………………….…….. | 11 |
| 3. | Экономическая часть……………….…………………….…………………….………………… | 12 |
| 4. | Охрана труда……………….…………………….…………………….…………………….…… | 13 |
| 5. | ЗАКЛЮЧЕНИЕ……………….…………………….…………………….…………………….…… | 14 |
| 6. | СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ…………………….……………………. | 15 |
| 7. | ПРИЛОЖЕНИЯ……………….…………………….…………………….…………………….…… | 16 |

ВВЕДЕНИЕ

В современном мире информационные технологии занимают важное место во многих сферах деятельности. В частности, создание и развитие интернет-порталов и мобильных приложений является одной из наиболее динамично развивающихся отраслей в IT-секторе. В этой связи, создание мультиязычного интернет-портала и мобильного приложения для библиотеки Костанайского Инженерно-Экономического Университета является актуальной и перспективной темой для дипломной работы.

Данная работа направлена на разработку и реализацию мультиязычного интернет-портала для библиотеки КИнЭУ, с целью улучшения доступности и качества обслуживания пользователей из разных стран и культур. В рамках проекта будет решена задача создания интуитивно понятного и удобного интерфейса, позволяющего быстро и легко находить необходимые книги и другие ресурсы, а также получать информацию о работе библиотеки и ее услугах.

В работе будут использованы современные технологии и инструменты для создания веб-приложений и мобильных приложений, в том числе HTML, CSS, JavaScript, и другие. Также будет произведен анализ требований пользователей и разработана оптимальная архитектура приложения, учитывающая функциональность, безопасность и производительность системы.

Полученные результаты могут быть использованы как в рамках университетской библиотеки, так и в других организациях и учреждениях, заинтересованных в развитии мультиязычных интернет-порталов и мобильных приложений.

Таким образом, данная дипломная работа имеет высокую практическую значимость и может привести к созданию новых возможностей для пользователей и развитию современных технологий в области информационных технологий

Кроме того, данная дипломная работа имеет научную и теоретическую значимость, так как в процессе ее выполнения будут изучены и анализированы существующие методы и подходы к созданию мультиязычных интернет-порталов и мобильных приложений. Также будут рассмотрены особенности работы с многоязычными данными, в том числе проблемы локализации и адаптации контента к различным культурам и языкам.

В ходе выполнения дипломной работы будут решены следующие задачи:

* Анализ требований пользователей и определение функциональных и нефункциональных требований для интернет-портала и мобильного приложения.
* Проектирование архитектуры интернет-портала, учитывающей требования пользователя, безопасность и производительность.
* Разработка веб-интерфейса и мобильного приложения с использованием современных технологий и инструментов.
* Тестирование и оптимизация созданного сайта и его мобильного приложения.
* Оценка эффективности созданной системы и ее возможной дальнейшей разработки и улучшения.

В целом, данная дипломная работа является актуальной и перспективной задачей в области разработки сайтов библиотеки и мобильных приложений, и может привести к созданию новых возможностей для пользователей и развитию современных технологий в этой области.

1. Описание задачи на проект

1.1. План разработки дипломного проекта

Целью моей дипломной работы стало создание мульти язычного интернет - портала и его мобильного приложения. Объектом разработки стал сайт библиотеки Костанайского Инженерно-Экономического Университета.

План разработки:

1. Разработка веб-страницы библиотеки

* Определение дизайна и интерфейса веб-страницы
* Создание шаблонов и макетов веб-страницы
* Разработка HTML и CSS кода для веб-страницы
* Создание графических элементов (например, логотипа и изображений) для веб-страницы
* Разработка навигации и функциональных элементов на веб-странице

1. Создание базы данных. Заполнение контента в БД

* Выбор базы данных и определение структуры БД
* Создание схемы базы данных
* Разработка SQL-запросов для создания и заполнения БД
* Определение категорий книг и описаний книг для БД
* Внесение информации в БД

1. Реализация мультиязычности сайта

* Определение языков, которые должен поддерживать сайт
* Создание и перевод текстового контента для каждого языка
* Разработка механизма переключения языков на веб-странице
* Разработка механизма локализации контента в мобильном приложении

1. Реализация адаптивности сайта под различные экраны

* Определение устройств и экранов, которые должны поддерживаться
* Разработка респонсивного дизайна веб-страницы
* Использование медиазапросов и технологий CSS для адаптивности
* Тестирование сайта на различных устройствах и экранах

1. Разработка мобильного приложения

* Определение платформ, на которых должно работать приложение
* Разработка макетов и интерфейса мобильного приложения
* Создание механизма локализации контента в приложении
* Использование технологий и инструментов для разработки мобильного приложения

1. Загрузка сайта на сервер

* Определение хостинга для сайта
* Выбор доменного имени для сайта
* Размещение сайта на сервере
* Тестирование сайта после загрузки на сервер

1. Тестирование проекта

* Определение методов и средств тестирования
* Тестирование функциональности и надежности веб-страницы, базы данных и мобильного приложения
* Проведение тестов на различных устройствах и операционных системах
* Оценка производительности веб-страницы и мобильного приложения
* Определение уязвимостей и устранение обнаруженных ошибок
* Тестирование механизма локализации контента веб-страницы и мобильного приложения
* Проведение финального тестирования перед запуском проекта.

1.2. Выбор программного обеспечения

Выбор программного обеспечения является важным этапом в разработке проекта, поскольку от него зависит работоспособность и функциональность сайта. Для создания мультиязычного сайта для КИнЭУ были выбраны следующие языки программирования и технологии:

* HTML (HyperText Markup Language) - это язык разметки, который используется для создания содержимого веб-страниц. Он является стандартом для создания веб-страниц и позволяет создавать структуру документа, определять заголовки, текст, таблицы, изображения и другие элементы страницы.
* CSS (Cascading Style Sheets) - это язык описания стилей, который позволяет определять внешний вид веб-страниц. С помощью CSS можно определять цвета, шрифты, размеры элементов, расположение блоков и другие аспекты внешнего вида веб-страницы.
* JavaScript - это язык программирования, который используется для создания интерактивных элементов на веб-страницах. Он позволяет создавать динамические эффекты, обрабатывать события, валидировать формы, и многое другое.
* PHP (Hypertext Preprocessor) - это скриптовый язык программирования, который используется для создания динамических веб-страниц. Он позволяет создавать интерактивные элементы, обрабатывать формы, работать с базами данных, и многое другое.

Выбор этих языков программирования был основан на ряде факторов. HTML и CSS широко используются для создания веб-страниц и имеют простой синтаксис, который легко понять и использовать. JavaScript позволяет создавать интерактивность на странице и работать с элементами страницы, что полезно для создания многих функций сайта.

PHP используется для создания динамических элементов и работает хорошо с базами данных, что является важным фактором для создания мультиязычного портала.

Таким образом, выбор HTML, CSS, JavaScript и PHP для создания моего проекта был основан на технических и функциональных потребностях проекта.

А также, необходимо создание мобильного приложения для этого были использованы данная технология:

Android Studio: это интегрированная среда разработки (IDE) для создания приложений для устройств на операционной системе Android.

Android Studio использует языки программирования Kotlin и Java, которые предоставляют разработчикам мощные инструменты для создания функциональных и эффективных приложений. Она также имеет множество интегрированных инструментов, таких как Android Emulator для тестирования приложений на виртуальных устройствах, Android Debug Bridge (ADB) для отладки приложений и множество других инструментов для создания высококачественных приложений

Одним из преимуществ Android Studio является то, что она постоянно обновляется и улучшается, предоставляя разработчикам новые инструменты и функции. Также она позволяет разрабатывать приложения для широкого спектра устройств, начиная от небольших смартфонов до больших планшетов и устройств с большими экранами.

Android Studio также обеспечивает интеграцию с другими инструментами и технологиями, такими как Firebase, которая предоставляет разработчикам инструменты для создания облачных сервисов и аналитики приложений. Также в Android Studio есть возможность подключения к системам контроля версий, таким как Git, что упрощает совместную работу разработчиков над проектом.

В целом, Android Studio является мощной и эффективной средой разработки для создания приложений для Android, которая обеспечивает разработчикам все необходимые инструменты для создания высококачественных и функциональных приложений. Она использует языки программирования Kotlin и Java.

2. Этапы разработки проекта

2.1. Проектирование базы данных

Проектирование баз данных является одним из важных этапов разработки информационной системы. База данных представляет собой организованную коллекцию данных, которые хранятся и обрабатываются в целях получения необходимой информации. Для проектирования базы данных необходимо определить ее структуру и связи между таблицами.

В вашей базе данных присутствуют таблицы books (Книги, таблица в которой хранятся книги), department, editions, faculty, specialities и rubrics, которые представляют собой основные объекты хранения данных. Таблица books содержит информацию о книгах, а таблицы department, editions, faculty, specialities и rubrics содержат справочную информацию, которая связана с книгами.

Для обеспечения связей между таблицами, в таблице books используются внешние ключи, которые указывают на соответствующие записи в других таблицах. Например, поле faculty в таблице books связано с таблицей faculty посредством внешнего ключа. Это позволяет связывать данные, упрощает поиск и обработку информации.

Важно учитывать, что при проектировании базы данных необходимо определить типы данных, ограничения и связи между таблицами. Также необходимо учитывать принципы нормализации данных, которые обеспечивают минимальную избыточность данных и улучшают производительность системы.

Таким образом, проектирование базы данных играет важную роль в разработке информационной системы и позволяет эффективно организовать хранение и обработку данных. В вашей базе данных используются основные принципы проектирования баз данных, что обеспечивает ее эффективную работу и удобство использования.

База данных создавалась в сервисе PhpMyAdmin в нём удобно создавать и заполнять таблицы, а также тестировать SQL запросы в пару кликов. Ещё одним немаловажным плюсом PhpMyAdmin является простота подключения базы данных к нашему проекту, ну об этом позже.

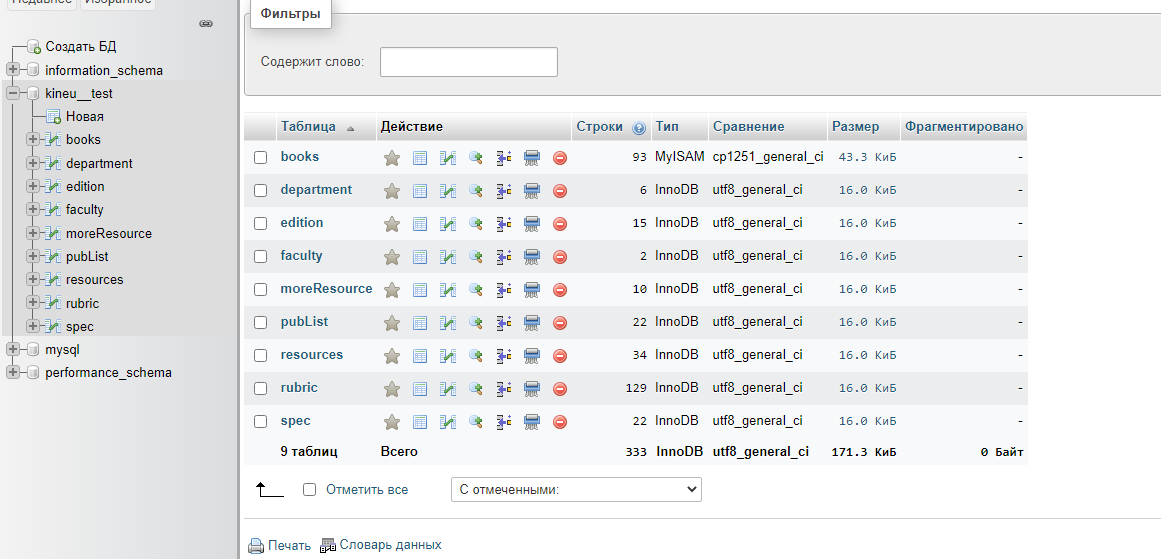


Рисунок 1. Внешний вид Базы данных PhpMyAdmin

Как видно на рис.1 была создана база данных kineu\_\_test и в ней находится 9 таблиц, подробнее о каждой:

1. «books» - Основная таблица, в которой хранятся данные обо всех книгах в библиотеке. Внешний вид таблицы:

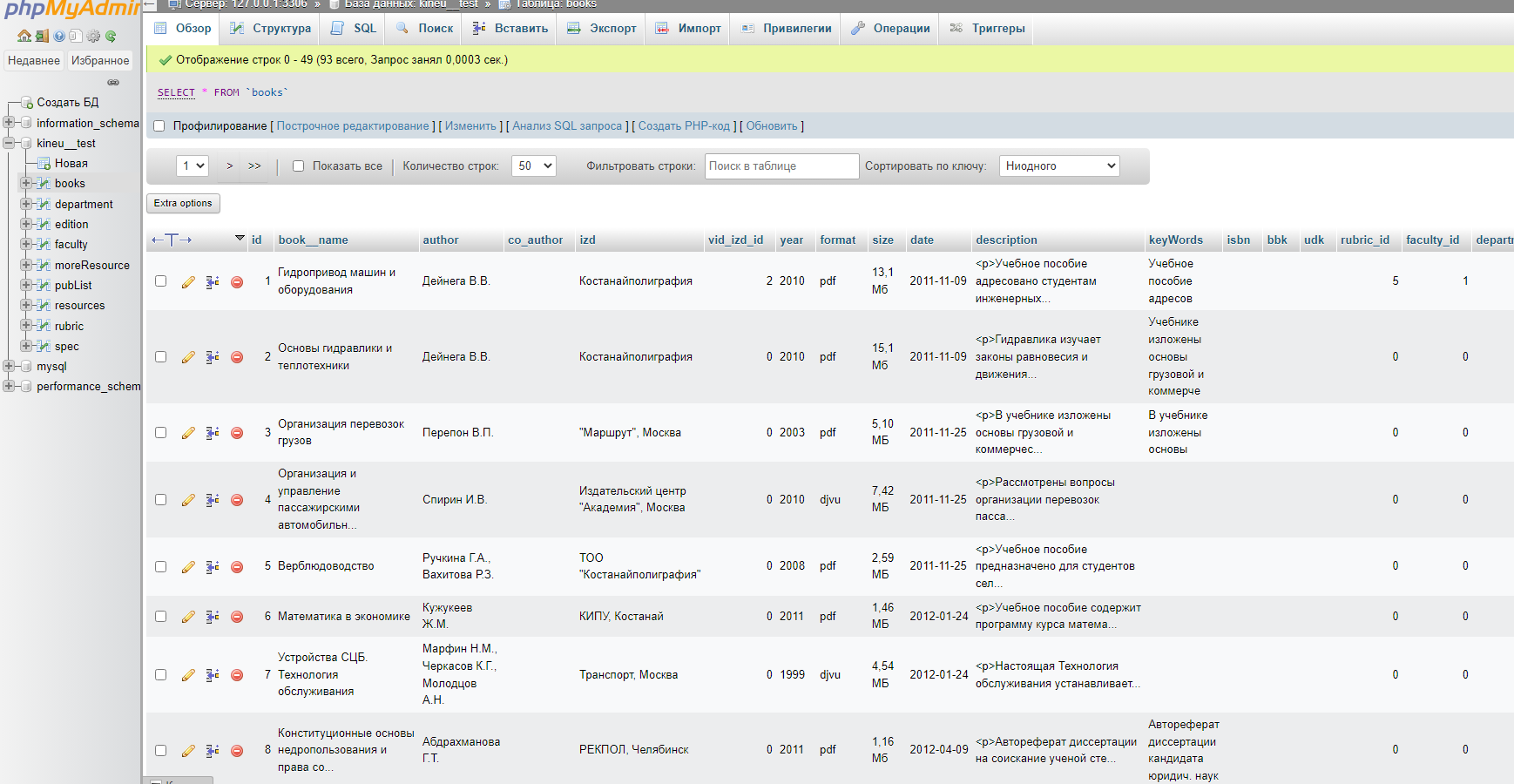


Рисунок 2. Таблица «books»

1. «department» - (от англ. «Кафедра») Таблица, что хранит в себе данные обо всех кафедрах в университете. К этой таблице обращается таблица books по идентификатору, чтобы получить данные о кафедре для конкретной книги. Внешний вид таблицы:

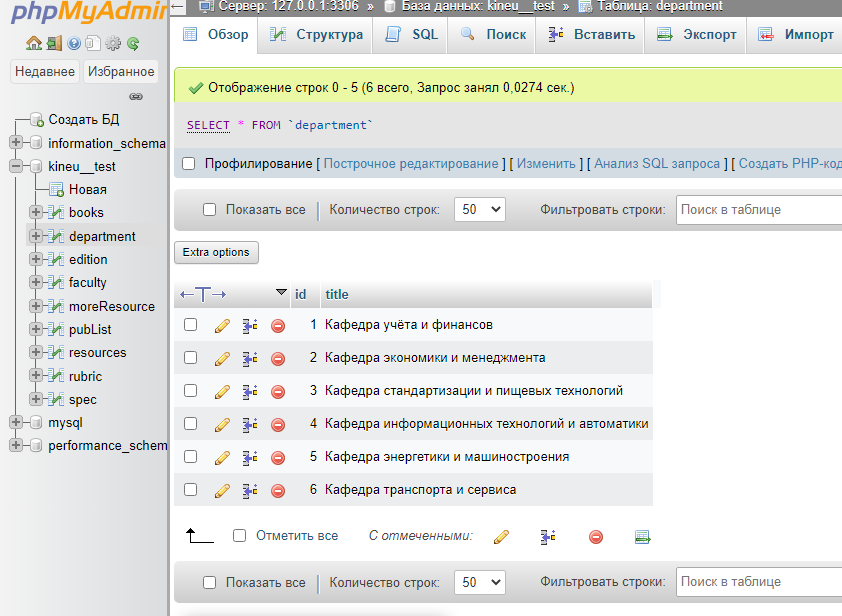


Рисунок 3. Таблица «department»

1. «edition» - (от англ. «Издание») Таблица, в которой содержится информация о видах изданий, к которым тоже идёт обращение с таблицы books. Внешний вид таблицы:

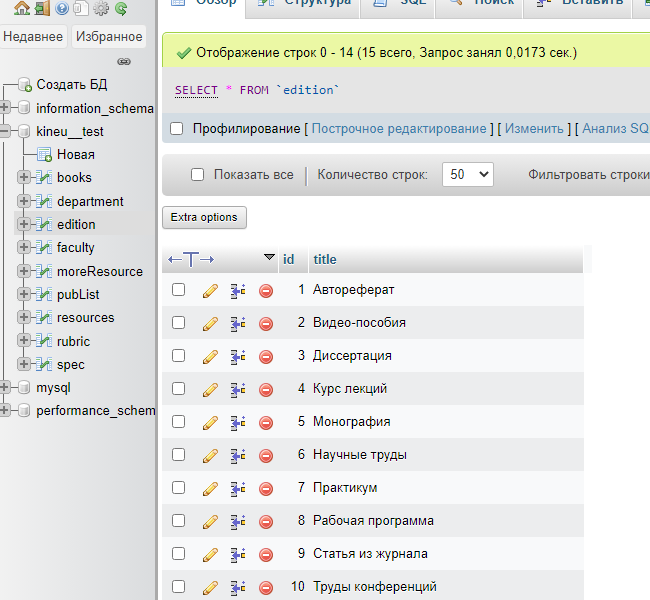


Рисунок 4. Таблица «Edition»

1. «faculty» - (от англ. «Факультет») Таблица, что содержит информацию о факультетах и точно также обрабатывает информацию по id и используется в таблице books. Внешний вид таблицы:

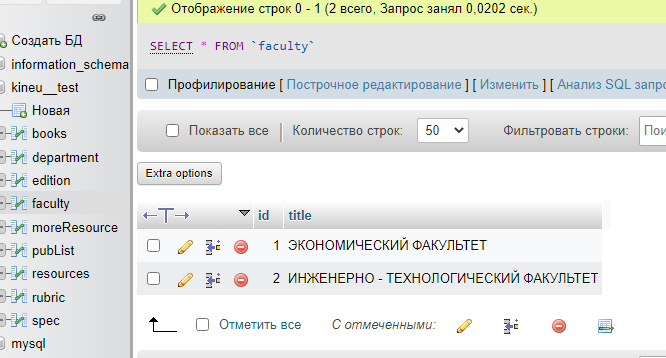


Рисунок 5. Таблица «Faculty»

1. «moreResources» - Это таблица что содержит дополнительную информацию для страницы на сайте. Весь ознакомительный текст на сайте является статичным т.к. он не меняется, а есть информация, что может меняться и для динамической замены, эта информация была записана в базу данных. Внешний вид таблицы:

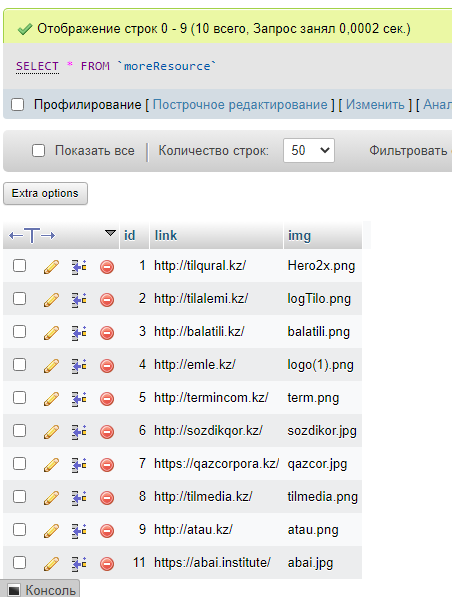


Рисунок 6. Таблица «More Resources»

1. «pubList» - Таблица, что также содержит вспомогательную информацию для сайта. Чтобы динамически можно было её поменять и не изменять всё внутри кода, т.к. это требует больших усилий и обновлений проекта на сервере, в отличие от серверного изменения. Внешний вид таблицы:

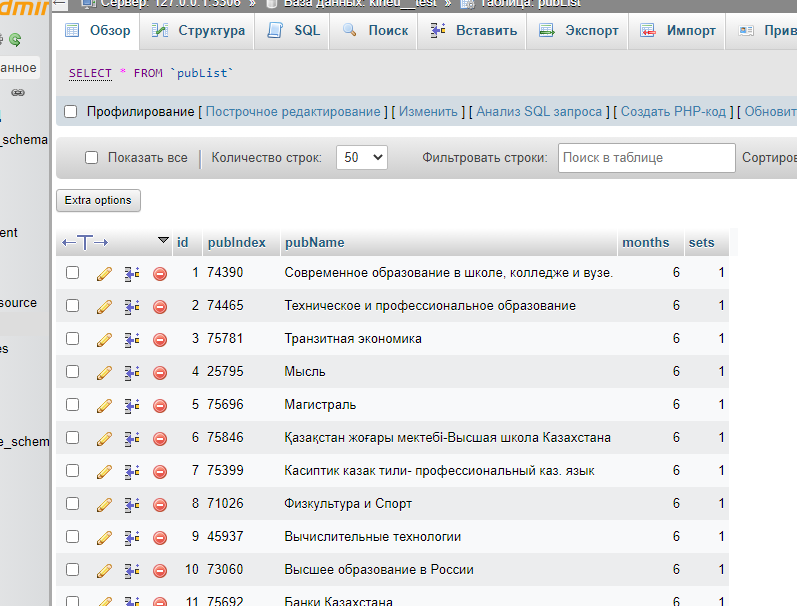


Рисунок 7. Таблица «pubList»

1. «resources» - Это таблица работает вместе с «moreResources» для отображения списка внешних электронных ресурсов библиотеки. Данные таблицы содержат в себе адрес сайта каждого из ресурсов и название его изображения. Внешний вид таблицы:

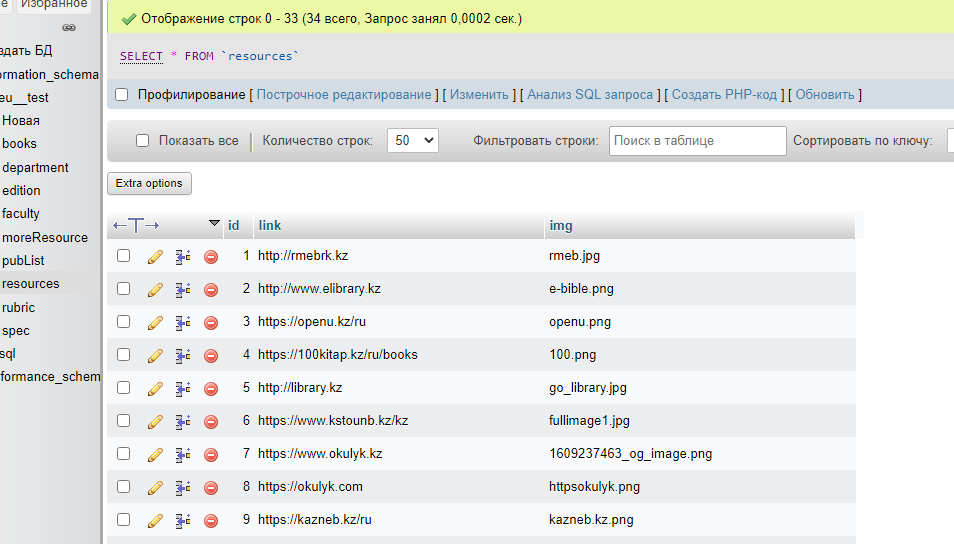


Рисунок 8. Таблица «Resources»

1. «rubric» - (от англ. «Рубрика») Это таблица такая же, как faculty или editions. Она хранит в себе данные о рубриках наших книг. Также она “отправляет” свои данные в таблицу books по id. Внешний вид таблицы:

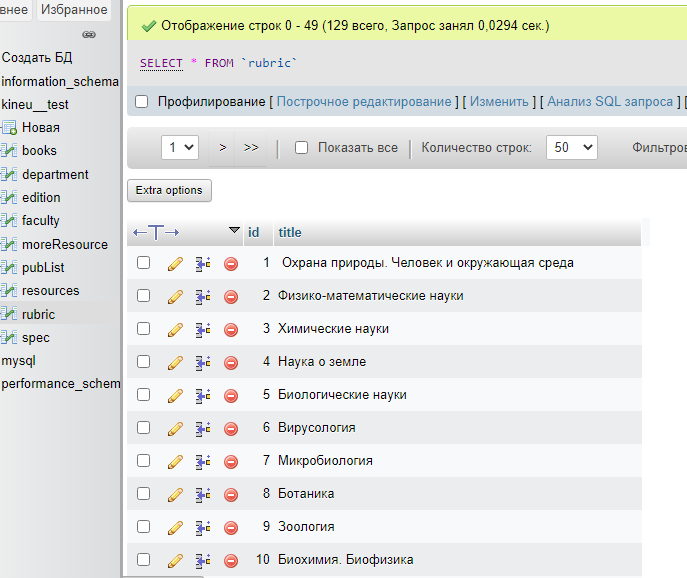


Рисунок 9. Таблица «Rubric»

1. «spec» - (сокр. «specialties» - «специальности») Таблица, что точно также хранит в себе информацию о специальностях и делится информацией с таблицей «books». Внешний вид таблицы:

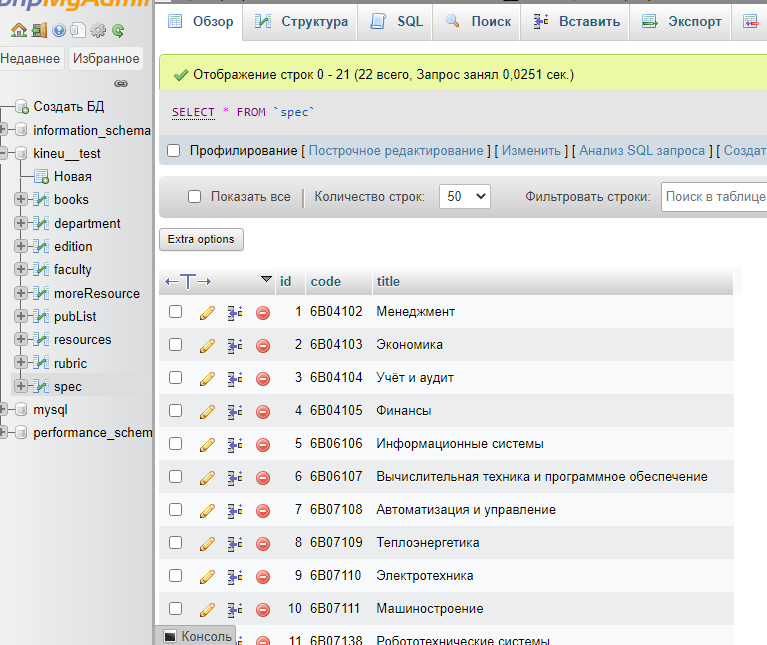


Рисунок 10. Таблица «spec»

Когда все таблицы и структура была создана, необходимо только заполнить их данными. И если с рубриками и факультетами с этим проблем нет, то с таблицей «books» всё будет сложней, ибо материала действительно много. В базу данных входят книги, как с внешних источников, так и преподавателей университета начиная с 2011 года по сегодняшний день.

2.2. Создание веб страницы проекта. Его структура

Создание веб-страницы - это процесс разработки и создания интернет - страницы, которая может содержать различные элементы, такие как текст, изображения, видео, формы и другие элементы, необходимые для обеспечения функциональности сайта. Этот процесс включает в себя несколько этапов, начиная с планирования и проектирования до фактической реализации и публикации страницы в Интернете.

При проектировании веб-страницы важно учитывать ее цели и задачи, а также потребности пользователей. Ключевые аспекты проектирования веб-страницы включают в себя выбор цветовой гаммы, типографию, расположение элементов на странице, используемые шрифты и графические элементы.

Создание веб-страницы обычно включает в себя использование языков программирования, таких как HTML (Hypertext Markup Language) для разметки содержимого, CSS (Cascading Style Sheets) для оформления страницы, а также JavaScript для создания интерактивных элементов и динамического контента.

Важно также учесть адаптивность веб-страницы для отображения на разных устройствах, таких как мобильные телефоны и планшеты. В этом случае используется технология Responsive Web Design, которая позволяет создавать сайты, которые адаптируются к различным устройствам, сохраняя свой вид и функциональность.

Наконец, после создания веб-страницы необходимо проверить ее на наличие ошибок и недочетов, а также оптимизировать ее для быстрой загрузки и эффективной работы.

Но, в данном проекте, так как мы используем базу данных MySQL, то для этого нам необходимо будет воспользоваться версткой на языке PHP.

Создание веб-страницы на языке PHP с подключением базы данных можно разделить на следующие этапы:

1. Планирование структуры базы данных: определение таблиц и их связей, выбор типов данных и ограничений.
2. Создание базы данных: создание базы данных и необходимых таблиц в СУБД.
3. Наполнение базы данных: добавление начальных данных в таблицы, если они уже известны.
4. Написание скриптов на языке PHP: создание файлов скриптов для работы с базой данных. Эти скрипты будут отвечать за добавление, удаление, редактирование и получение данных из базы данных.
5. Подключение скриптов к веб-странице: подключение созданных скриптов к веб-странице, чтобы они могли взаимодействовать с базой данных.
6. Тестирование и отладка: проверка работоспособности веб-страницы и скриптов на языке PHP, исправление возможных ошибок.
7. Развертывание: загрузка веб-страницы и скриптов на сервер для общего доступа пользователей.

Пройдёмся по каждому пункту по порядку, если с базами данных уже всё понятно и уже было всё сказано. То, на четвертом и пятом пункте хотелось бы остановиться поподробнее.

Написание скриптов на PHP является более усложнённой задачей чем просто верстка на HTML, нам необходимо установить специальное программное обеспечение для обработки PHP скриптов.

«OpenServer Panel» - Это приложение для создания локального сервера на компьютере, с помощью него мы можем компилировать PHP код и конвертировать его в HTML понятный браузеру код. Скачиваем приложение, устанавливаем. В настройках обязательно необходимо указать версию MySQL (нам это понадобиться при работе с БД).

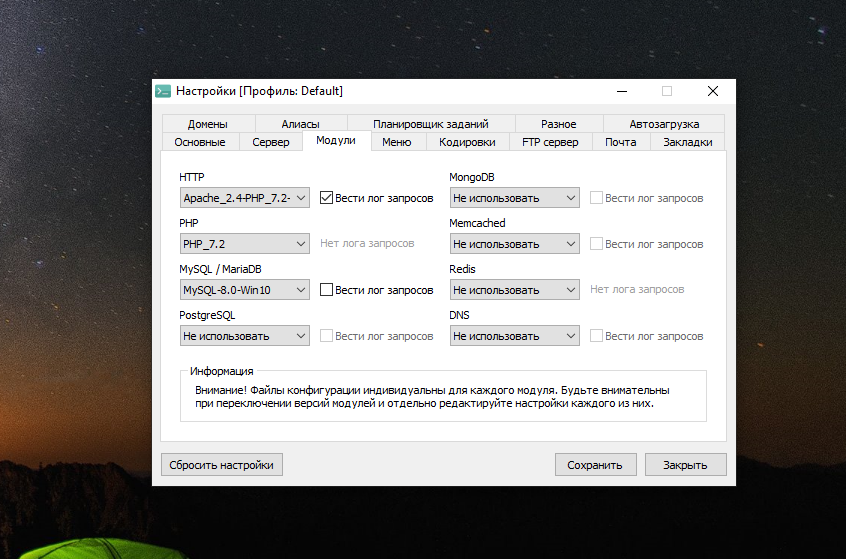


Рисунок 11. Панель настроек MySQL в «OpenServer»

Уже сейчас мы можем просмотреть нашу среду разработки баз данных которая называется «PhpMyAdmin». Для этого нажмём на иконку «OpenServer» (Зеленый, Красный либо Желтый флажок) в правом нижнем углу на панели задач. В открывшемся меню выбираем вкладку «Дополнительно» и выбираем «PhpMyAdmin».

Нам открывается окно как на (рис. 12)

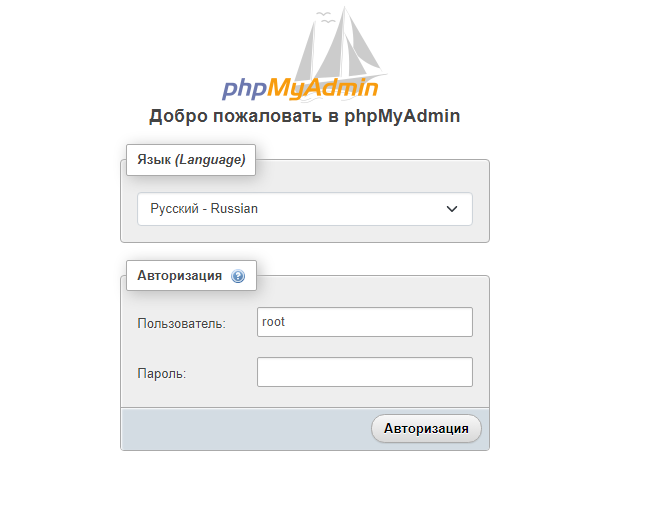


Рисунок 12.Меню входа в PhpMyAdmin

По умолчанию у вас будет стоять логин «root» и пароль пустой. Так и должно быть, просто нажимаем «Авторизация» и открывается окно PhpMyAdmin.

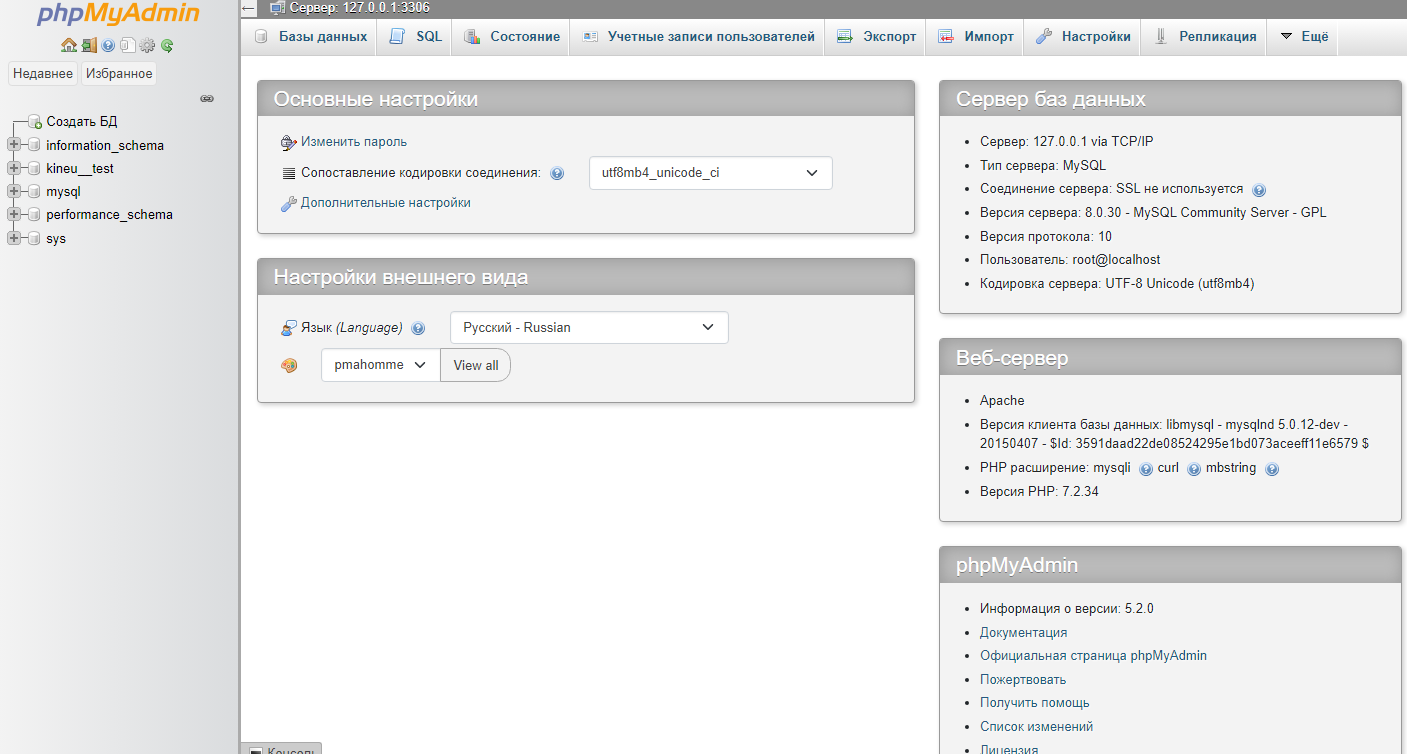


Рисунок 13. Окно приложения PhpMyAdmin

Слева, на панели, мы можем видеть список наших баз данных. В данных БД хранятся наши таблицы, которые мы можем создавать сколько угодно. PhpMyAdmin очень удобен в своём функционале, в нём все автоматизировано и упрощено до понятного вида, поэтому я и выбрал его для разработки своего проекта.

OpenServer может компилировать код PHP, полностью выполнять всё что там происходит, затем результат выводить в основной файл с HTML версткой и уже потом отдавать его браузеру на обработку.

Структура проекта. Сайт представляет собой 1 главный файл «index.php»

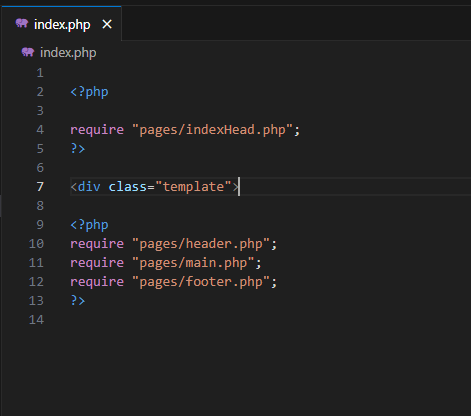


Рисунок 14. Листинг кода файла «index.php»

Во время разработки я придерживался компонентного подхода создания. Чтобы не писать весь код в одном файле я писал все по блокам и в разных файлах. Здесь мы можем видеть открытие PHP тега «<?php?>».

А внутри прописываем команду «require» что означает подключение какого-то файла в данном случае это файл «indexHead.php» и так далее. Мы просто подключаем файлы, таким образом, загружается главная страница проекта.

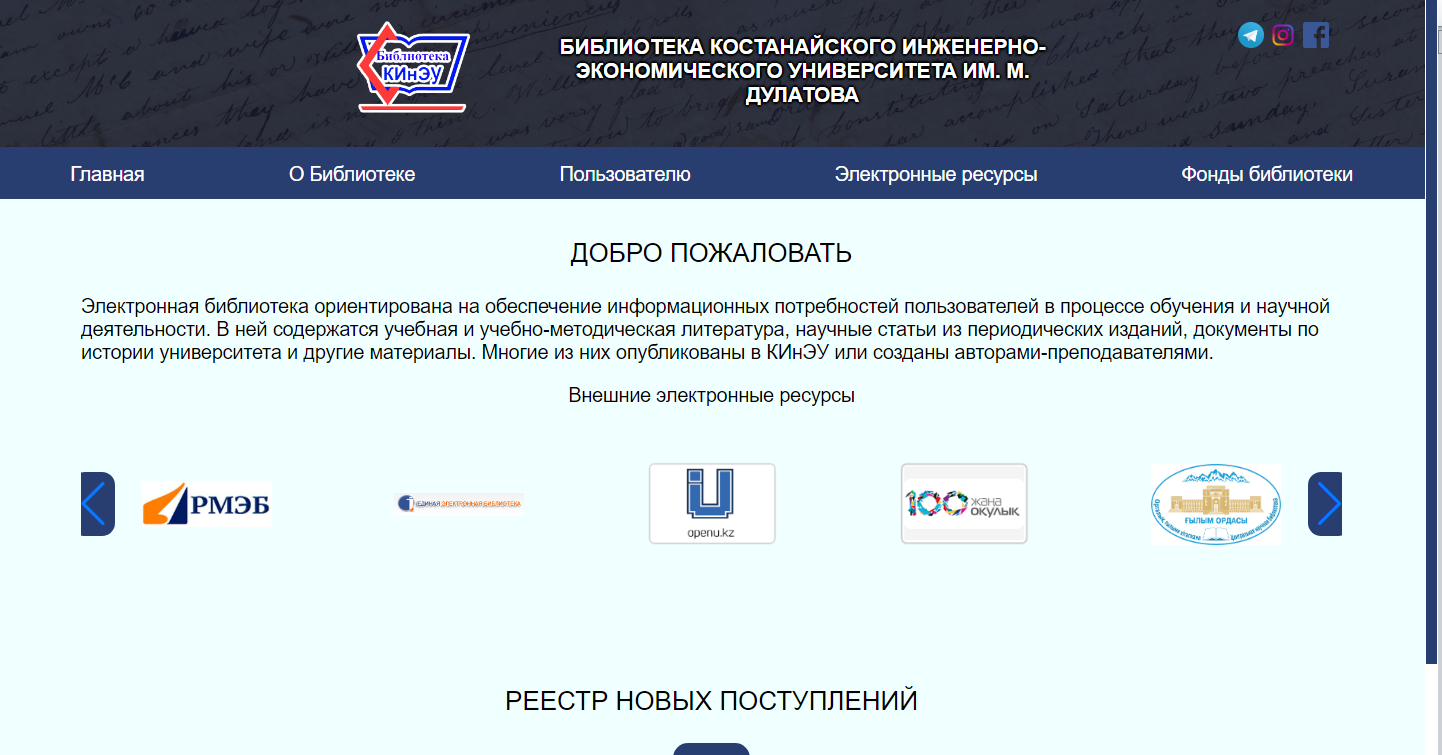


Рисунок 15. Главная страница сайта

И так наш сайт приобретает вид красивой веб страницы, мы можем видеть, что шапка сайта и навигационное меню, будто одна часть с остальным контентом, но это не так. Все что идёт от приветствия «Добро пожаловать» эта часть контента уже является файлом «main.php».

В этом и есть одна из прелестей языка PHP то, что в нём можно так легко и просто создавать страницы по частям.

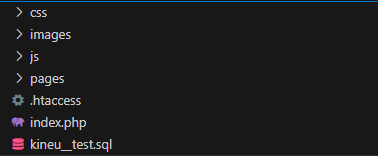


Рисунок 16. Структура сайта

В файловой системе проекта, у нас есть папка «css» которая хранит в себе 1 файл «style.css»

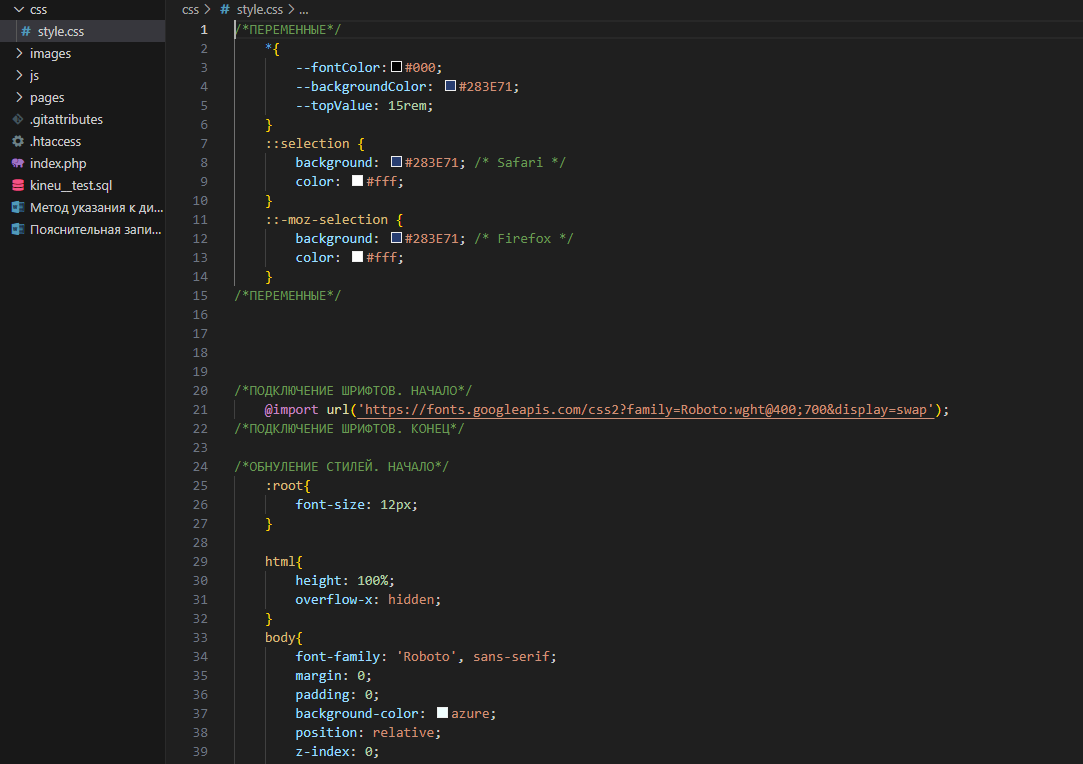


Рисунок 17. Пример листинга кода в файле «style.css»

Далее, у нас есть папка «images» что хранит в себе все изображения, что есть на сайте. Папка «js» хранит в себе скрипты на языке JavaScript для реализации анимации и интерактивных элементов на сайте.

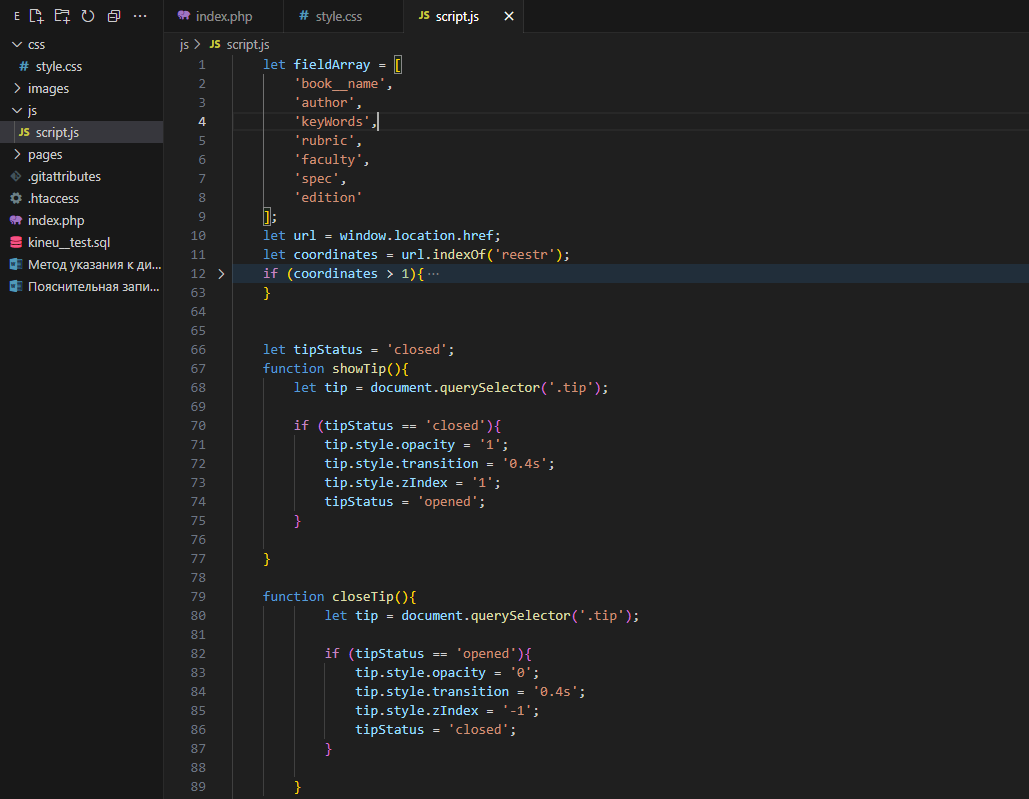


Рисунок 18. Пример листинга кода в файле «script.js»

И одна из важнейших папок в проекте это «pages». Папка что хранит файлы всех страниц на сайте. Которые подключаются посредством той же самой команды require. Ниже приведена иллюстрация всех файлов в папке «pages».

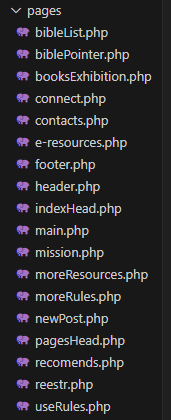


Рисунок 19. Папка «pages»

На этом о структуре сайта мы закончили.

2.3. Реализация интерактивных элементов на сайте

Во время разработки, была написана шапка сайта и его навигационное меню. И после его создания началась разработка механика работы нажатия и появления меню каждого пункта этого меню. По нажатию на кнопку должна срабатывать функция «showDropDown» (Показать выпадающее меню) на языке JavaScript. Функция работала на основе манипуляции JS на DOM дерево HTML и меняла стили конкретного блока меню. А именно, функция меняла его CSS свойство «top» что позволяло опускать элемент до уровня, где его видно.

Но, впоследствии, я отказался от данного кода. Так как это можно сделать более удобнее, и менее затратным способом для браузера с точки зрения затраты ресурсов на подгрузку дополнительных файлов и обработку дополнительных функций.

Альтернативой стал способ обработки открытия меню не по клику мыши, а по наведению курсора. Для этого был использован псевдокласс hover в файле стилей CSS. Псевдокласс hover позволяет обрабатывать стили на событие наведения курсора мыши на элемент браузера. Поэтому было принято такое простое решение, возвести окончательную позицию top для выпадающего меню именно на той позиции, где его видно, в переменную. А в дальнейшем мы можем менять значение этой переменной, когда будем писать адаптивность сайта.

Пример кода ниже:

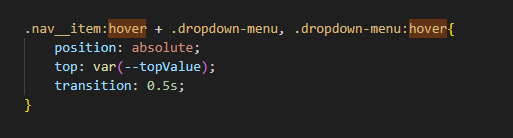


Рисунок 20. Фрагмент кода с событием hover.

Здесь мы видим довольно длинный селектор. Описание элементов селектора:

* «nav\_\_item» - это класс заголовка меню, что мы видим постоянно. Например: «О библиотеке» или «Электронные ресурсы».
* «dropdown-menu» - это класс выпадающего меню, что по умолчанию скрыто. Но по событию hover, что мы записали чуть позже, мы задаём этому скрытому меню абсолютное позиционирование и в свойство top что отвечает за отступ сверху, мы задаём значение переменной «topValue». Что по умолчанию является 15rem.

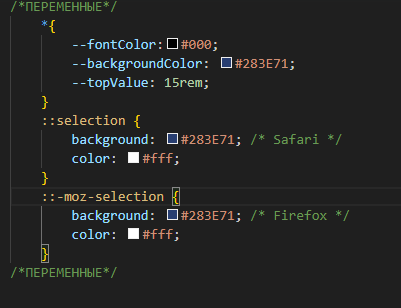


Рисунок 21. Код объявления переменных в CSS.

Здесь мы можем видеть, что мы объявляем 3 переменные в самом начале CSS файла. Первые две это наши цвета, что везде используются на сайте, а третье это наша переменная «topValue», в которой хранится по умолчанию значение 15rem.

Что за «15rem». Rem - это одна из множества единиц измерений в CSS которой пользуются для создания отзывчиво-адаптивной страницы. Она ссылается на размер шрифта на странице, таким образом, 1rem равен размеру шрифта на странице, для стандартного размера экрана монитора размер шрифта указан 16 пикселей.

Значит, в таком случае 15 rem будет равняться, 15 умножить на 16, то есть 240 пикселей. А в дальнейшем, при создании адаптивности сайта это значение меняется в зависимости от размеров экрана.

Таким образом, у нас работают выпадающие списки.

Следующим интерактивным элементом на сайте является слайдеры. Слайдеры или же карусели данных это блок на странице, в котором представлены несколько объектов, в нашем случае это логотипы компаний которые являются партнерами библиотеки КИнЭУ. Которые, можно переключать по нажатию кнопки. Внешний вид слайдера на главной странице сайта:



Рисунок 22. Слайдер на главном экране

Данные иконки можно листать как слайдом по экрану, так и нажатием на управляющие кнопки.

Работа этот слайдер благодаря библиотеке для JavaScript под названием «Swiper». Суть заключается в том, что мы создаём простую HTML верстку и к ней подключаем наш Swiper. Код представлен ниже:

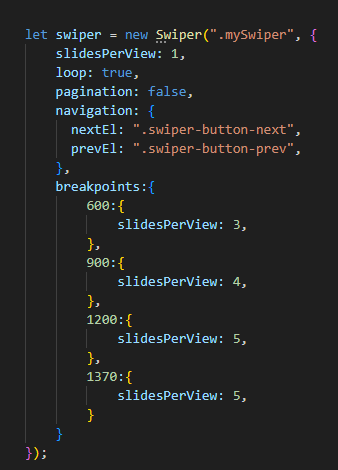


Рисунок 23. Фрагмент кода библиотеки «Swiper»

Здесь, мы задаём стандартные параметры для слайдера, такие как, сколько будет показываться слайдеров на экране, будут ли они зацикливаться по кругу, видимы ли кнопки управления, адаптивность слайдера и так далее.

В данном случае дополнительная библиотека сделала всё за нас, и сэкономила много кода.

Далее, на сайте находится много статичного текста и интерактива в нём мало. Но тут мы переходим к главной части проекта это «Реестр новых поступлений»

2.4. Реестр новых поступлений

Реестр новых поступлений это та часть проекта, что заняла большую часть времени на разработку. Реестр новых поступлений подразумевает страницу, в которой пользователь может видеть последние добавленные книги в базу данных, может пролистнуть сайт вниз и использовать навигационное меню для выбора страницы и просмотреть каждую книгу.

Также, сбоку, у нас располагается блок с сортировкой книг, для выборки необходимых среди большого количества материала.

План разработки был таков:

1. Реализовать вывод всех статей на страницу, путём подключения её к базе данных. А также сделать вывод именно тех характеристик книги, которые указаны в базе.
2. Создать пагинацию страниц.
3. Создать сортировку страниц.

Пойдём по порядку, попутно описывая каждый этап разработки и проблемы, с которыми я столкнулся.

1. Вывод статей на страницу. Необходимо для начала подключить страницу к базе данных, для этого было создан файл «connect.php» он хранится в папке pages с остальными страницами.



Рисунок 24. Подключение к базе данных.

И за себя этот файл представляет лишь PHP код, который вставляется в HTML страницу перед использованием базы данных. После подключения к БД мы можем создать ассоциативный массив из столбцов таблицы что нам нужна. Выбираем таблицу мы с помощью SQL запроса.

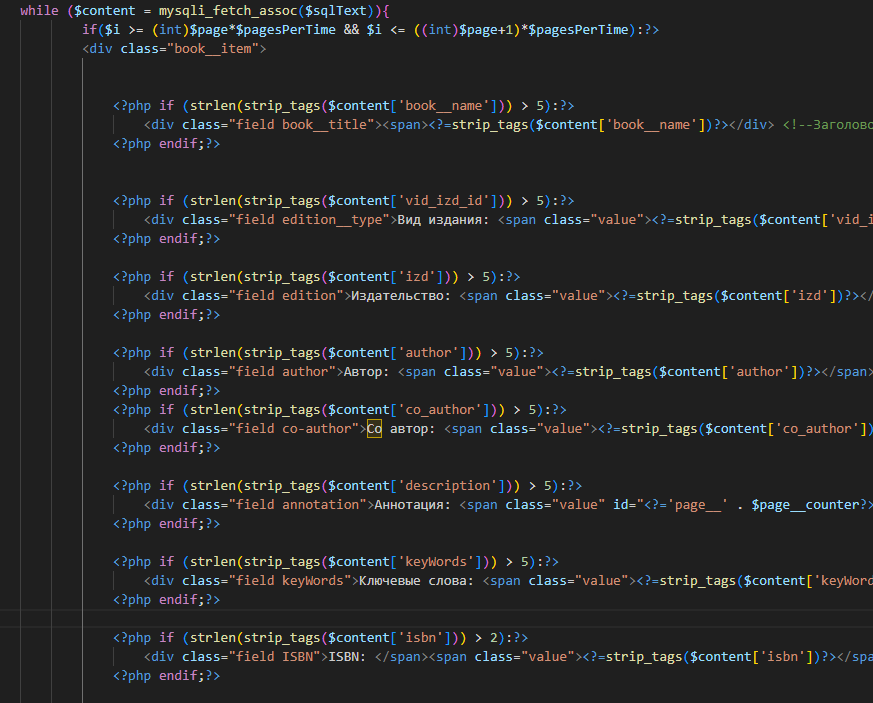


Рисунок 25. Фрагмент кода функции, что вызывает каждую книгу из базы данных.

С помощью цикла while мы можем вывести данные из таблицы, к таблице мы подключаемся с помощью SQL запроса, этот запрос хранится в переменной «$sqlText». Он хранится в переменной, так как запрос может меняться в зависимости от параметров сортировки.

Итак, с помощью функции «mysqli\_fetch\_assoc» мы можем создать ассоциативный массив, в котором по названию полей у нас хранятся значения из таблицы.

Цикл нам нужен для того чтобы вывести каждое значение из таблицы по порядку. При этом цикл сам создаёт все элементы HTML верстки.

Следующим этапом, у нас идёт проверка условий через оператор «if», проверка заключается в том, чтобы вывести книги из правильного диапазона страниц, ну это уже второй вопрос о пагинации.

Ну и заключением нашей функции по выводу данных из БД является проверка каждой строки наших характеристик, на наличие каких либо данных. Т.е. если, у нас ничего не записано в строку «Соавтор» в таком случае нам не нужно даже выводить ее, ибо она будет пустая, а если же там что то есть, то выводим всё. Собственно, внешний вид книг и всей странице вы можете увидеть ниже.

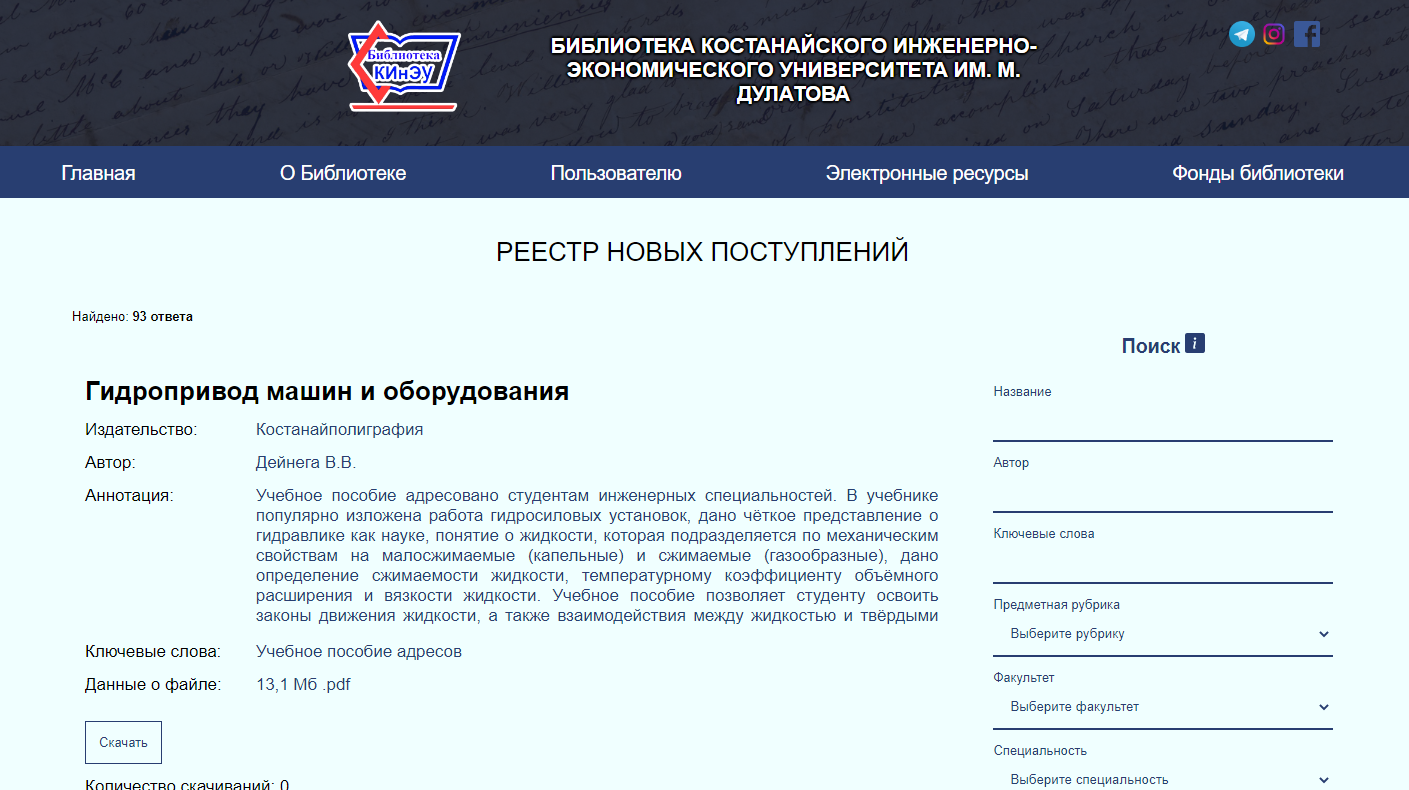


Рисунок 26. Внешний вид страницы «Реестр новых поступлений»

Таким образом, все страницы выводятся в полной мере и соблюдая все правила. Но сейчас встала такая проблема, на момент создания, в базе находится 95 книг, и все они выходят подряд друг за другом, это сильно увеличивает длину страницы и время её загрузки.

Поэтому нам приходит на помощь такой способ навигации по данным как «пагинация».

1. Создать пагинацию страниц.

Пагинация страниц - это метод организации контента на сайте, при котором большое количество данных разбивается на отдельные страницы, которые доступны для просмотра пользователю по одной или нескольким страницам одновременно.

Пагинация используется в тех случаях, когда необходимо предоставить большой объем информации, но отображение всех данных на одной странице слишком загружает ее и затрудняет навигацию для пользователя. Разбивая информацию на страницы, пользователь может более удобно просматривать содержимое и быстро находить необходимые данные.

Пагинация также позволяет сократить время загрузки страницы, так как она загружается только та часть информации, которая отображается на текущей странице. Это особенно актуально для пользователей с медленным интернет - соединением или для мобильных устройств с ограниченной скоростью передачи данных.

Веб-разработчики часто используют пагинацию в сочетании с запросами к базам данных, чтобы выводить большие объемы данных на веб-странице, таких как списки новостей, блогов, товаров в интернет - магазинах и т.д.

Как я создавал пагинацию страниц. Для начала была проведена работа в коде без отображения на странице, создана переменная, в которой хранится информация о том, сколько должны быть страниц на сайте, т.е. общее число статей поделено на количество статей, что отображается на экране.



Рисунок 27. Внешний вид Пагинации на сайте.

Потом, цикл создаёт 4 кнопки, предыдущая страница, текущая страница, следующая страница и последняя страница. Также 2 кнопки для навигации по стрелочкам, которые перебрасывают тебя на 1 страницу вперёд либо назад.

Вся проблема заключалась в том, что я не знал, как вывести данные конкретной страницы, но я понял, что можно настроить диапазон значений вывода данных. А найти этот диапазон можно простым способом, где начальная точка является текущей страницей умноженной на количество статей на странице. А конечная точка это текущая страница плюс 1 и также умноженная на количество статей.

Но тогда встал вопрос, как получить текущую страницу. Вопрос решился GET запросом на PHP, нажатием на кнопку в пагинации, отправляется GET запрос на страницу и передаётся 1 параметр, например page-1, что указывает на страницу 2, так как отчёт идёт с нуля.

Далее, строка конкатенируется и обрезается до числа, полученное число уже играет роль начального индекса для цикла и это число плюс 1 как конечный индекс.

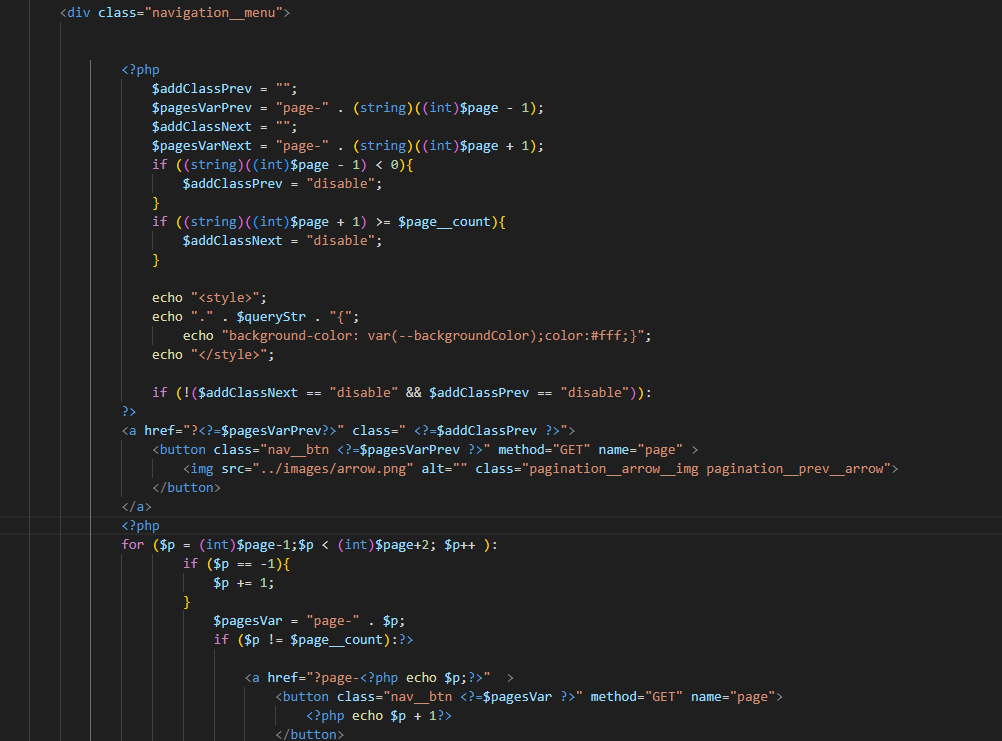


Рисунок 28. Фрагмент кода меню пагинации.

Таким образом, у нас в зависимости от выбранной страницы выводится вся информация.

1. Создать сортировку страниц. Большая часть создания реестра уже выполнена, все книги выводятся и верно отображаются. Теперь необходимо реализовать сортировку страниц в зависимости от параметров полей сортировки.

Вот мы создали все поля сортировки

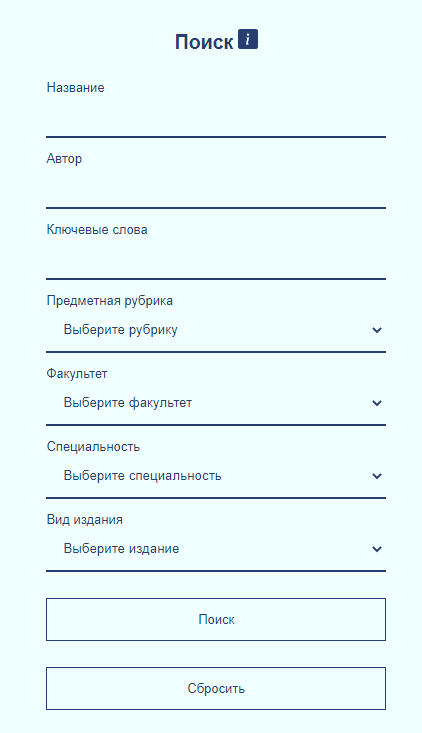


Рисунок 29. Внешний вид полей сортировки.

В выпадающие списки мы добавили данные из базы данных. Так, например поле «Предметная рубрика» хранит в себе множество значений

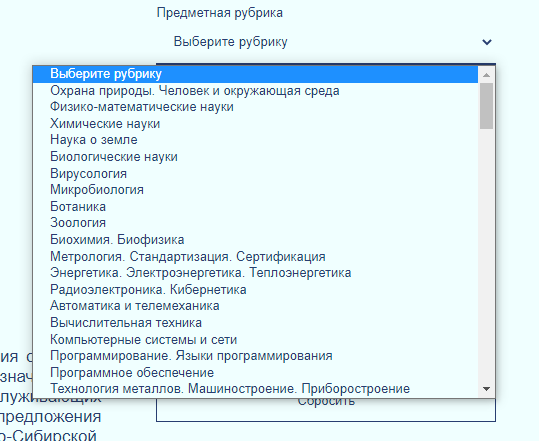


Рисунок 30. Выпадающий список поля «Предметная рубрика»

И записывать их в HTML верстке не вариант, так как это неудобно и в случае если какая, то рубрика удалится или возникнет надобность добавить новую, это будет неудобно редактировать внутри кода в файле, поэтому всё делается в базе данных. Мы подключаемся к таблице «rubric» (Рис. 9)

И из этой таблицы также, циклом выписываем все данные и вставляем в выпадающий список «Предметная рубрика». Повторяем эти манипуляции со всеми остальными полями кроме тех, в которых пользователь сам вводит значения как, например поле «Автор».

Перед нами теперь стоит задача, необходимо реализовать механизм работы так чтобы пользователь мог выбрать либо одно, либо два поля для сортировки максимум.

Для этого, мы используем событие ввода информации на странице - «input». И обрабатываем его, если пользователь ввёл данные в два поля, то мы блокируем все остальные поля, а если одно поле он стер либо отменил, то возвращаем всё, как было, за это отвечает функция «addEventListener»

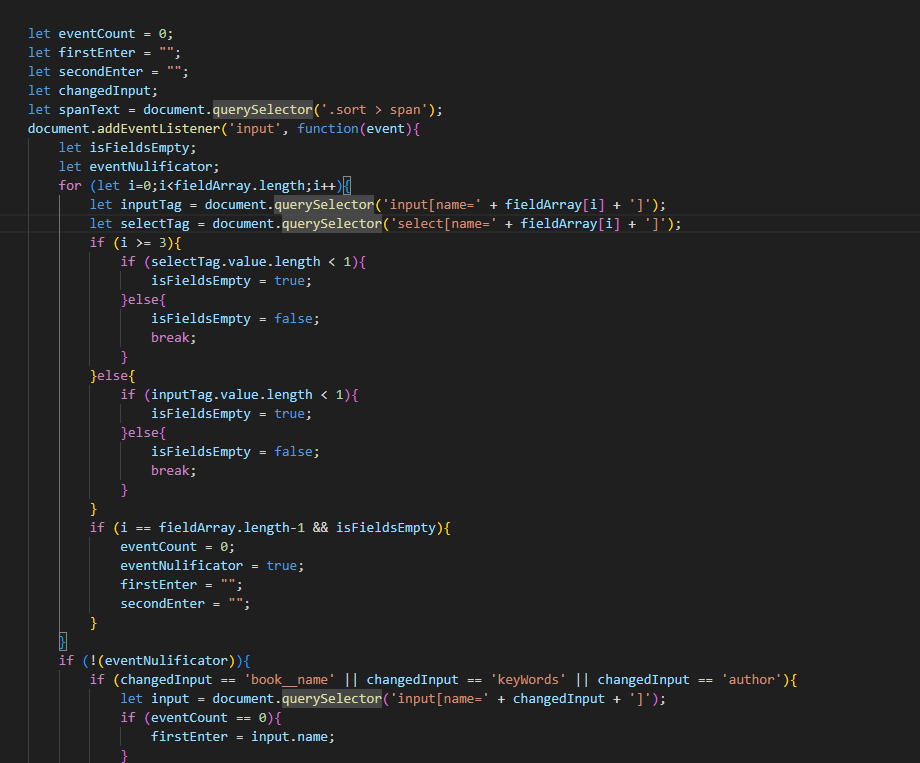


Рисунок 31. Фрагмент кода для обработчика события ввода текста

А также функция «disableFields», т.е. во время исполнения кода выше, мы в конце получаем две либо одну переменную с названием поля, в которое пользователь ввёл текст. А затем, вызываем функцию для отключения всех полей кроме тех, что мы получили.

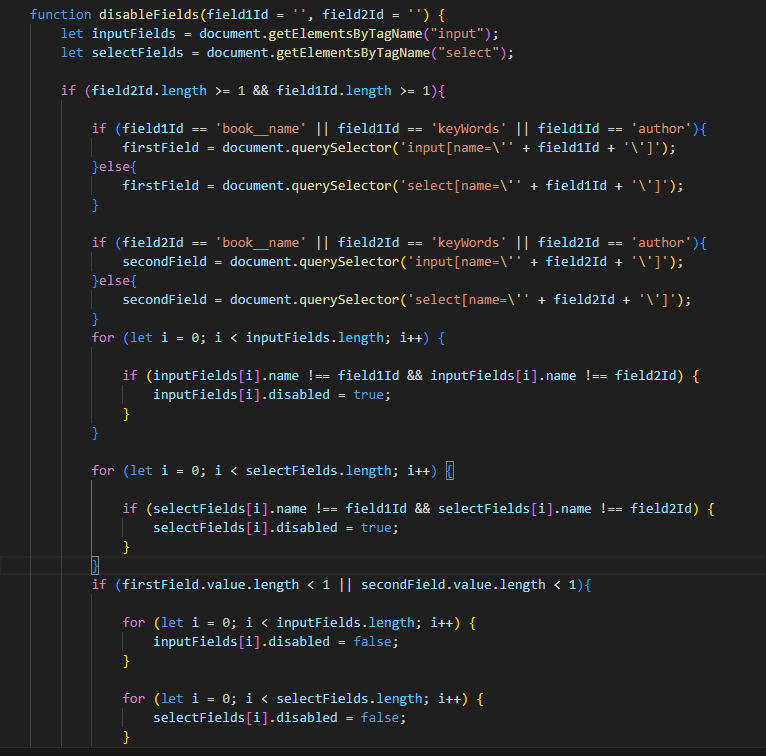


Рисунок 32. Фрагмент кода функции «disableFields»

И эта функция получает два параметра, это те самые две переменные, что были переданы после выполнения события «input». И после выполнения функции наши поля были отключены. Как это выглядит на сайте:

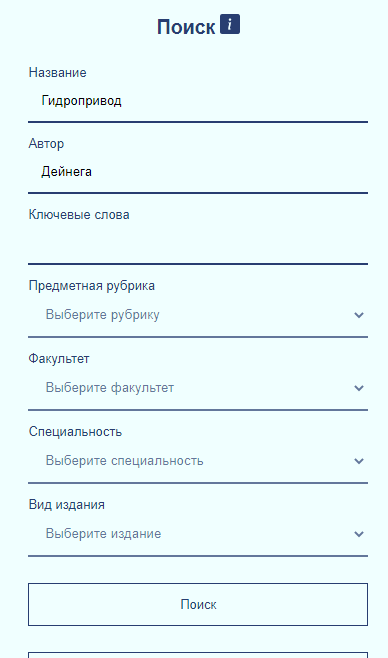


Рисунок 33. Демонстрация отключения полей сортировки.

Здесь мы можем увидеть то, как поля, что не были выбраны стали окрашены немного серым цветом, что подчеркивает их нынешнюю позицию, а именно, они отключены.

После того, как мы выбрали поля и знаем какие заполнены, а какие отключены, необходимо переходить к следующему этапу, к сортировке.