МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Минский КОЛЛЕДЖ предпринимательства

Отделение «Программное обеспечение информационных технологий»

Пояснительная записка

к курсовому проекту

на тему

**Сайт детской литературы «Comics»**

МКП КП 2-40 01 01 35 004 ПЗ

**Автор проекта**

Учащийся 4 курса группы П-401 /А.В. Казакевич/

**Руководитель**

Преподаватель /И.Ю. Дроздова /

Минск, 2018

**МИНСКИЙ КОЛЛЕДЖ ПРЕДПРИНИМАТЕЛЬСТВА**

Отделение

«Программное обеспечение информационных технологий»

Специальность

«Программное обеспечение информационных технологий» 2-40 01 01

УТВЕРЖДАЮ

Председатель цикловой комиссии

\_\_\_\_\_\_\_\_\_\_/ \_Ю.А. Доманова\_\_\_\_\_\_\_\_\_\_\_\_/

**ЗАДАНИЕ**

**на курсовой проект**

Учащемуся\_\_Казакевичу Андрею Владимировичу\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

курса\_\_\_4\_\_\_\_ группы\_\_\_\_\_П-401\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

по учебной дисциплине \_Базы данных и системы управления базами данных\_\_\_\_\_\_

Тема курсового проекта Сайт детской литературы «Comics»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Исходные данные: \_\_комиксы, пользователи, выпуски \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Состав проекта**

|  |  |
| --- | --- |
| **Пояснительная записка** | |
| Содержание разделов | Срок выполнения |
| Введение | 08.10 – 15.10 |
| Постановка задачи | 16.10 – 22.10 |
| Вычислительная система | 23.10 – 29.10 |
| Проектирование | 30.10 – 12.11 |
| Описание программы | 13.11 – 26.11 |
| Отладка и испытание программы | 27.11 – 03.12 |
| Описание применения | 04.12 – 10.12 |
| Заключение | 11.12 – 17.12 |
| **Графическая часть проекта** | |
| ER-диаграмма - плакат | 18.12 – 20.12 |
| Вариантов использования- плакат | 21.12 – 24.12 |

Дата выдачи « 08 »\_октября\_\_2018 г.

Срок завершения « 24 » декабря 2018 г.

**Преподаватель-руководитель курсового проекта** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись учащегося**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись учащегося**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*y*

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

МКП КП 2-40 01 01 35 013 ПЗ

Разраб.закевич

Казакевич

Пров.

*Дроздова*

Реценз.

Н. контр.

Утв.

*Сайт детской литературы «Comics»*

Лит.

Листов

64

МКП

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ……….……….……….……….……….……….……….……….……….…5](#_Toc533245692)

[1 ПОСТАНОВКА ЗАДАЧИ……….……….……….……….……….……….……….8](#_Toc533245693)

[1.1 Формулировка задания…………………………………………………………….8](#_Toc533245694)

[1.2 Обоснование необходимости автоматизации……………………………………8](#_Toc533245695)

[1.3 Определение данных и их представление………………………………………..9](#_Toc533245696)

[2 ВЫЧИСЛИТЕЛЬНАЯ СИСТЕМА……….……….……….……….……….….….11](#_Toc533245697)

[2.1 Обоснование выбора языка программирования………………………………..11](#_Toc533245698)

[2.2 Обоснование выбора среды разработки…………..……….……….………...…12](#_Toc533245699)

[2.3 Требования к конфигурации программного и аппаратного обеспечения……13](#_Toc533245700)

[3 ПРОЕКТИРОВАНИЕ……….……….……….……….……….……….……….….15](#_Toc533245701)

[3.1 Проектирование интерфейса……….……….……….……….……….…………15](#_Toc533245702)

[3.2 Инфологическая модель предметной области……….……….……….………..15](#_Toc533245703)

[3.3 Физическая модель базы данных……….……….……….……….……….…….16](#_Toc533245704)

[3.4 Физическая модель базы данный……….……….……….……….……….…….22](#_Toc533245705)

[4 Описание программы……….……….……….……….……….……….……….…….26](#_Toc533245706)

[4.1 Логическая структура……….……….……….……….……….……….…………26](#_Toc533245707)

[4.2 Физическая структура……….……….……….……….……….……….………...26](#_Toc533245708)

[5 Отладка и испытание программы……….……….……….……….……….…………28](#_Toc533245709)

[5.1 Тестовые примеры……….……….……….……….……….……….……….……28](#_Toc533245710)

[5.2 Анализ полученных результатов……….……….……….……….……….……..29](#_Toc533245711)

[6 Описание применения программы……….……….……….……….……….………..36](#_Toc533245712)

[6.1 Назначение программы и условия использования……….……….……….……36](#_Toc533245713)

[6.2 Описание работы программы……….……….……….……….……….…………36](#_Toc533245714)

[Заключение……….……….……….……….……….……….……….……….…………38](#_Toc533245715)

[Список использованных источников……….……….……….……….……….……….39](#_Toc533245716)

[Приложение А……….……….……….……….……….……….……….……….……...40](#_Toc533245717)

# ВВЕДЕНИЕ

В наше время разработка сайта является одним из главных атрибутов многих компаний, которые работают не только в Интернете. Сайт – веб-страница или группа веб-страниц, объединённых единой темой. Грамотно разработанный сайт может стать хорошим методом распространения информации.

Первым сайтом в мире стал [info.cern.ch](http://mydiv.net/arts/info.cern.ch), который был создан 6 августа 1991 года. Создателем считается Тим Бернерс-Ли, который разместил на нем описание новой технологии World Wide Web. Данная технология базируется на протоколе передачи данных HTTP, а также на языке гипертекстовой разметки HTML и системе адресации URL. Позднее Тим Бернерс-Ли решил разместить на сайте список ссылок на другие ресурсы, что позволило преобразовать сайт в первый в мире интернет-каталог.

Главной целью сайта является распространение информации. С учётом того, что существует множество способов передачи информации, сайт является лучшим способом, так как в настоящее время Интернет-сайтами пользуются большое количество людей, что позволяет охватить огромную аудиторию.

Сайты имеют множество сильных сторон. Всегда существует возможность модернизации и увеличения функционала. Доверие к компании, имеющей качественно разработанный сайт, как правило выше, чем к компании, что не имеет сайта вообще. Сайт работает 24 часа в сутки, 7 дней в неделю, 365 дней в году. На сайте можно расположить отдельную информацию, которая будет защищена и не будет видна посетителям. Анализ посещаемости позволит получить статистику, благодаря которой можно проанализировать, что интересно пользователю.

Как и у всех проектов, у сайтов также имеются слабые стороны. Серьёзные веб-проекты нуждаются в дорогой поддержке. Требуется хостинг сайта, что влечёт за собой небольшие, но постоянные затраты. На сайт или сервер могут производится взломы.

В сети Интернет существует множество различных сайтов, а именно:

1. Информационный сайт.
2. Игровой портал.
3. Тематический сайт.
4. Интернет-магазин.
5. Блог и так далее.

Данный проект относится к категории интернет-магазин. Интернет-магазин, это сайт, торгующий товарами посредством сети Интернет. Позволяет пользователям онлайн, в своём браузере или через мобильное приложение, сформировать заказ на покупку, выбрать способ оплаты и доставки заказа, оплатить заказ.

Целью курсового проекта является разработка программного средства являющегося аналогом сайта «https://acomics.ru.

Задачами курсового проекта являются:

1. Создание условий для просмотра информации о комиксах.
2. Использование фреймворка Angular и дополнительных библиотек.
3. Овладение современными методами поиска, обработки и использования информации.

Данный проект содержит следующие разделы:

1) Раздел «Постановка задачи» – содержит формулировку задания, обоснование необходимости автоматизации, описание входных и выходных данных.

2) Раздел «Вычислительная система», обоснование выбора языка программирования среды разработки, требования к аппаратным и операционным ресурсам.

3) Раздел «Проектирование», описание пользовательского интерфейса, описание алгоритма, справочная система приложения.

4) Раздел «Описание программы», описание всех модулей и функций, входящих в состав приложения, особенности интерфейса разработанной программы.

5) Раздел «Отладка и испытание программы», проверка работоспособности программы на различных тестах, результаты тестирования, перечень ошибок и пути их исправления.

6) Раздел «Описание применения», назначение программы, область применения, описание применения.

7) Раздел «Заключение», подведение итогов, описание степени соответствие проекта поставленной задаче.

# 1 ПОСТАНОВКА ЗАДАЧИ

* 1. **Формулировка задания**

Необходимо разработать сайт «Comics» для продажи детской литературы. Для хранения информации использовать базу данных.

На сайте реализовать возможность регистрации и авторизации пользователей. Предусмотреть следующие роли: администратор, пользователь.

Для роли «администратор» реализовать следующие функции: добавление информации в базу данных.

Для роли «пользователь» реализовать следующие функции: просмотр комиксов, поиск по заданным критериям, покупка комиксов.

Предусмотреть обработку исключительных ситуаций при добавлении информации.

1. **Обоснование необходимости автоматизации**

Полнометражные фильмы и сериалы по комиксам в наше время являются очень популярными, и поэтому в странах СНГ сами комиксы становятся все более популярными. Наверняка каждому после просмотра фильмов от кинокомпании «MARVEL» или «DC» хотел узнать, что будет дальше, и если ты не хочешь ждать пару лет, то прочитать комикс –лучший выбор. Предметной областью данного сайта является продажа комиксов. Если ты не можешь найти нужный выпуск «на бумаге», то ты всегда сможешь купить его онлайн. Для облегчения этой трудной задачи и существуют сайты, где пользователь может найти нужный выпуск любимого комикса.

Существует множество аналогов сайта «Comics», например, такие как:

1. comicskrama.by– сайт предоставляющий информацию о комиксах, местах их продажи.
2. comix.by – сайт также предоставляет информацию о комиксах и занимается доставкой комиксов по РБ
3. chookandgeek.ru – сайт занимается продажей комиксов, атрибутики.

Преимуществами данного курсового проекта являются:

1. Большой архив комиксов.
2. Постоянное обновление комиксов.
3. Возможность купить электронную версию.

Данный сайт необходимо разработать для предоставления информации о комиксах и их продаже.

## **1.3 Определение данных и их представление**

Данные – поддающееся многократной интерпретации представление информации в формализованном виде, пригодном для передачи, связи или обработки.

Для долговременного хранения данных обычно используются базы данных. Данные в памяти могут быть организованы в различные виды структур данных, таких как массивы, строки, связанные списки или объекты.

Структуры данных могут хранить данные различных типов, включая числа, строки и другие структуры данных. Ввод и вывод данных в компьютеры производится через периферийные устройства.

Входная информация: данные, которые вводятся пользователем в процессе работы с приложением

Входными данными в данном курсовом проекте являются действия пользователя, такие как: ввод логина, пароля, информации добавляемой в базу данных, параметры для поиска информации. Данные заносятся в таблицу и хранятся в базе данных.

Выходная информация: данные, которые вычислительная система посылает пользователю после их обработки.

Выходными данными сайта являются результаты поиска, сообщения о некорректных действиях пользователя.

# 2 ВЫЧИСЛИТЕЛЬНАЯ СИСТЕМА

## **2.1 Обоснование выбора языка программирования**

TypeScript – язык программирования, представленный Microsoft в 2012 году и позиционируемый как средство разработки веб-приложений, расширяющее возможности JavaScript. Разработчиком языка TypeScript является Андерс Хейлсберг, создавший ранее Turbo Pascal, Delphi и C#. Спецификации языка открыты и опубликованы в рамках соглашения Open Web Foundation Specification Agreement. TypeScript является обратно совместимым с JavaScript и компилируется в последний. Фактически, после компиляции программу на TypeScript можно выполнять в любом современном браузере или использовать совместно с серверной платформой Node.js. Код экспериментального компилятора, транслирующего TypeScript в JavaScript, распространяется под лицензией Apache. Его разработка ведётся в публичном репозитории через сервис GitHub. TypeScript отличается от JavaScript возможностью явного статического назначения типов, поддержкой использования полноценных классов (как в традиционных объектно-ориентированных языках), а также поддержкой подключения модулей, что призвано повысить скорость

Преимущества языка TypeScript:

* поддержка многими популярными IDE:
* строгая типизация, которая проще для освоения Java и C# программистами;
* реализует многие концепции ООП, такие как наследование, полиморфизм, инкапсуляция и модификаторы доступа;
* TypeScript – надмножество JavaScript, поэтому любой код на JavaScript будет выполнен и в TypeScript.

Недостатки языка TypeScript:

* слишком много дополнительных файлов, что бывает неудобно, если ваш проект небольшой;
* не все браузеры поддерживают отладку TypeScript в консоли без лишних настроек;
* множество нетривиальных классов.

## **2.2 Обоснование выбора среды разработки**

Visual Studio Code — редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией. Visual Studio Code основан на Electron — фреймворк, позволяющий с использованием Node.js разрабатывать настольные приложения, которые работают на движке Blink. Несмотря на то, что редактор основан на Electron, он не использует редактор Atom. Вместо него реализуется веб-редактор Monaco, разработанный для Visual Studio Online.

Платформа Angular (версия 2 и выше) - это открытая и свободная платформа для разработки веб-приложений, написанная на языке TypeScript, разрабатываемая командой из компании Google, а также сообществом разработчиков из различных компаний. Angular - это полностью переписанный фреймворк от той же команды, которая написала AngularJS.

Преимущества Angular:

* поддержка Google, Microsoft;
* инструменты разработчика (CLI);
* единая структура проекта;
* реактивное программирование с RxJS;
* единственный фреймворк с Dependency Injection из "коробки";
* шаблоны, основанные на расширении HTML.

Недостатки Angular:

* выше порог вхождения из-за Observable (RxJS) и Dependency Injeciton;
* чтобы все работало хорошо и быстро нужно тратить время на дополнительные оптимизации;
* динамическое создание компонентов оказывается нетривиальной задачей.

## **2.3 Требования к конфигурации программного и аппаратного обеспечения**

Программа была разработана на ОС Windows 10. Основанием для разработки приложения в которой реализовался курсовой проект, послужило то что, ОС является наиболее оптимальной для работы как профессионалов, так и обычных пользователей.

Преимущества:

1) Система достаточно проста в использовании.

2) Привлекательный внешний вид с различными темами оформления.

3) Большинство программ как платных, так и бесплатных поддерживают именно эту операционную систему.

4) Легко восстанавливаемая удаленная информация.

Минусы: большая уязвимость от вирусов, атак хакеров и скриптов.

Минимальные требования:

1) Процессор: не менее 1 ГГц с поддержкой PAE, NX и SSE2.

2) ОЗУ: 1 ГБ (для 32-разрядной) или 2 ГБ (для 64-разрядной).

3) Свободное место на жестком диске: 2 ГБ (для 32-разрядной) или 4 ГБ (для 64-разрядной);

4) Широкополосный доступ к интернету.

5) Видеоадаптер: графическое устройство Microsoft DirectX 9 с драйвером WDDM.

# 3 ПРОЕКТИРОВАНИЕ

## **3.1 Проектирование интерфейса**

Интерфейс – общая граница между двумя функциональными объектами, требования к которой определяются стандартом; совокупность средств, методов и правил взаимодействия (управления, контроля и т.д.) между элементами системы.

Главная страница будет содержать: информацию о сайте.

Страница «Каталог» будет содержать: информацию о комиксах.

Страница «Личный кабинет» будет содержать: информацию о пользователе.

Страница «Вход» будет содержать поля для авторизации.

Страница «Регистрация» будет содержать поля для регистрации.

Сайт будет реализован в серо-белом стиле.

## **3.2 Инфологическая модель предметной области**

Предварительное бронирование услуг является очень удобным и востребованным во всем мире. Для регистрации услуги необходимо выбрать услугу и обсудить с мастером в какое время он сможет её выполнить. Сайт помогает клиентам просмотреть предоставляемые услуги и время, которое будет удобно и им и мастеру, также узнать о новинках и акциях. Данную информацию проще всего хранить в базе данных, так как она обеспечивает безопасное хранение.

Необходимо спроектировать базу данных маникюрный-салон «Diamond Nails», информация их которых будет использоваться для предоставления информации о названиях и времени предоставления услуг.

Для контроля услуг в базе данных необходимо хранить список всех имеющихся услуг, а также информацию о их стоимости и расходе материалов.

Для контроля за поступлением и расходом материалов необходимо ввести учет материалов их поставки и траты на определенный вид услуги.

Для контроля записей необходимо хранить информацию о клиенте, мастере к которому он записан, желаемой услуге, а также дату и время записи.

В базе данных необходимо хранить информацию о материалах, о времени оказания услуг и времени, когда возможно провести услугу, пользователях и акциях.

## **3.3 Физическая модель базы данных**

Данный курсовой проект состоит из следующих таблиц. Информация о пользователях хранится в таблице 3.1. Информация о фильмах содержится в таблице 3.2. Информация о сериалах содержится в таблице 3.3. Информация об избранных фильмах содержится в таблице 3.4. Информация об избранных сериалах содержится в таблице 3.5.

Таблица 3.1 – Authors

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| Country | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит страну автора. |
| name | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит имя автора. |
| email | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит email. |

Таблица 3.2 – Available\_comics

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| Number\_available\_comics | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер доступного комикса. |
| releaseId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id выпуска. |
| userId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id пользователя. |

Таблица 3.3 – Comics

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Числовой | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| name | Текстовый | Не является ключевым полем, имеет маски ввода, является обязательным полем, содержит описание название пользователя. |
| rating | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит рейтинг комикса. |
| image | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит путь к картинке. |

Таблица 3.4 – Ganre\_comics

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| ganreId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер жанра. |
| comicsId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер комикса. |

Таблица 3.5 – Ganres

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| nane | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит название жанра. |

Таблица 3.6 – Order\_details

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| number\_order | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит уникальный номер покупки. |
| userId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер пользователя. |
| releaseId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер комикса. |
| orderId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер заказа. |

Таблица 3.7 – Orders

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком.. |
| number\_order | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер заказа. |
| purchase\_date | Дата | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит дату заказа. |

Таблица 3.8 – Releases

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| name | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит название выпуска. |
| price | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит цену выпуска. |
| data\_issue | Дата | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит дату публикации. |
| author\_Id | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id автора. |
| path | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит путь к изображениям. |
| comicsId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id комикса. |
| pages | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем,содержит количество страниц. |

Таблица 3.9 – Reviews

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| Number\_review | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит номер коментария. |
| text | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит текст комментария |
| userId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id пользователя. |
| releaseId | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id выпуска. |

Таблица 3.10 – Role\_user

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| Role\_id | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id роли. |
| User\_id | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит id пользователя |

Таблица 3.11 – Roles

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| Role\_name | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит название роли. |

Таблица 3.12 – User

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| Id | Счетчик | Является ключевым полем, не имеет маски ввода, является обязательным полем, является счетчиком. |
| name | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит логин пользователя. |
| password | Числовой | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит пароль пользователя. |
| email | Текстовый | Не является ключевым полем, не имеет маски ввода, является обязательным полем, содержит email пользователя. |

## **3.4 Физическая модель базы данный**

Физическая модель базы данных разработана на MySQL Workbench.

Структура таблицы Author представлена на рисунке 3.1.

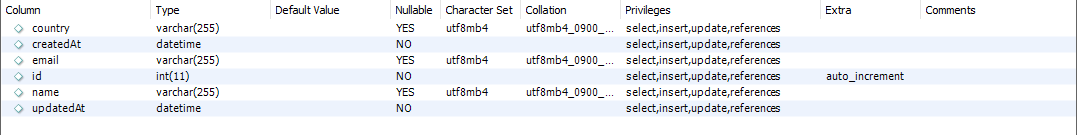


Рисунок 3.1 – Структура таблицы Author

Структура таблицы Available\_comics представлена на рисунке 3.2.

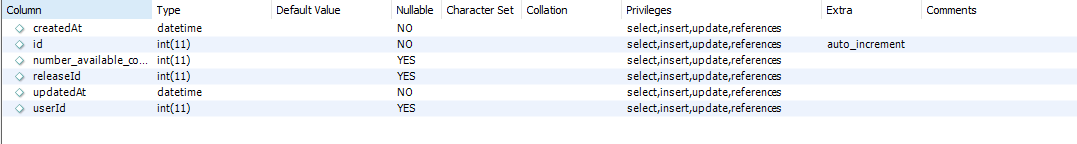


Рисунок 3.2 – Структура таблицы Available\_comics

Структура таблицы Comics представлена на рисунке 3.3.

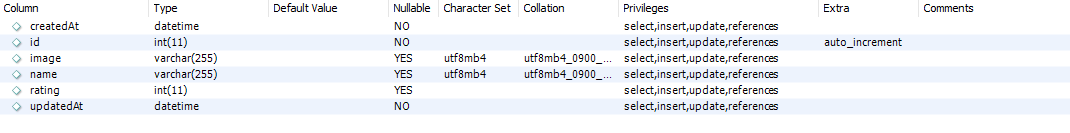


Рисунок 3.3 – Структура таблицы Comics

Структура таблицы Ganre\_comics представлена на рисунке 3.4.

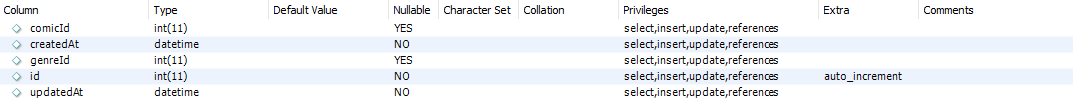


Рисунок 3.4 – Структура таблицы Ganre\_comics

Структура таблицы Ganres представлена на рисунке 3.5.

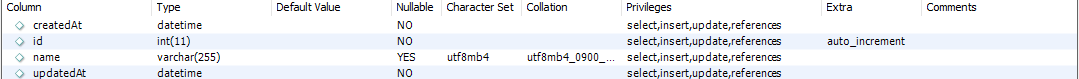


Рисунок 3.5 – Структура таблицы Ganres

Структура таблицы Order\_details представлена на рисунке 3.6.

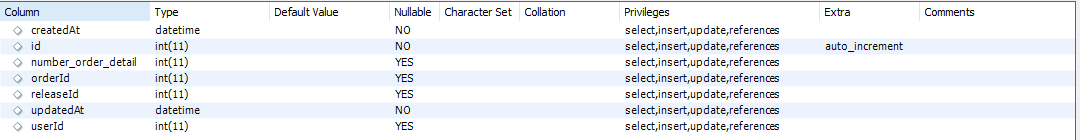


Рисунок 3.6 – Структура таблицы Order\_details

Структура таблицы Orders представлена на рисунке 3.7.

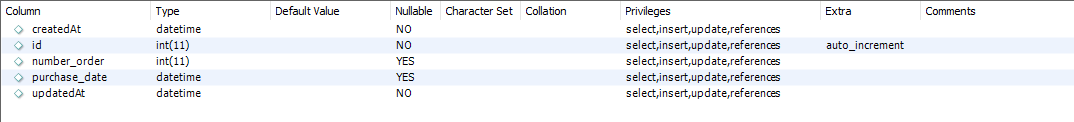


Рисунок 3.7 – Структура таблицы Orders

Структура таблицы Releases представлена на рисунке 3.8.

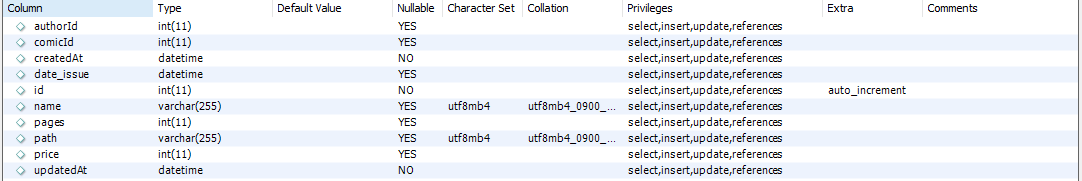


Рисунок 3.8 – Структура таблицы Releases

Структура таблицы Review представлена на рисунке 3.9.

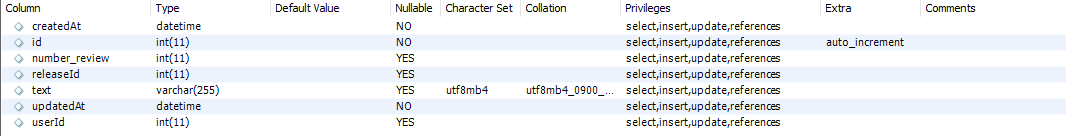


Рисунок 3.9 – Структура таблицы Review

Структура таблицы Role\_users представлена на рисунке 3.10.

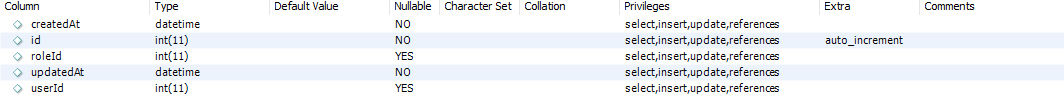


Рисунок 3.10 – Структура таблицы Role\_users

Структура таблицы Roles представлена на рисунке 3.11.

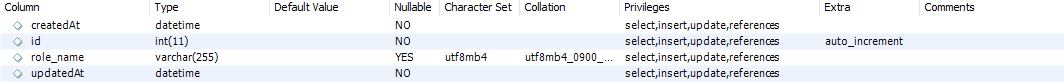


Рисунок 3.11 – Структура таблицы Roles

Структура таблицы Users представлена на рисунке 3.12.

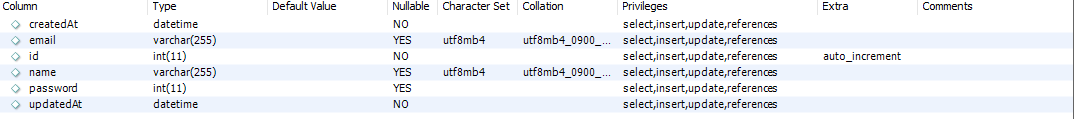


Рисунок 3.12 – Структура таблицы Users

4 Описание программы

4.1 Логическая структура

Процедуры и функции:

1. initForm() – функция, отвечающая за инициацию формы.
2. AddUser() – функция, отвечающая за добавление нового пользователя.
3. IsControlInvalid()– функция, отвечающая за валидацию полей.
4. viewRelease() – функция просмотра выпусков.
5. getNew – функция вывода самых новых комиксов.
6. netTop – функция вывода самых популярных комиксов.
7. getDrama – функция вывода комиксов с жанром «Драма».
8. getFantastic – функция вывода комиксов с жанром «Фантастика».
9. getMarvel– функция вывода комиксов с жанром «Марвел».
10. getDc – функция вывода комиксов с жанром «ДС».
11. getKriminal – функция вывода комиксов с жанром «Криминал».
12. logint – функция авторизация.
13. getGanre – функция вывода жанров комиксов.

4.2 Физическая структура

Страницами данного сайта являются:

* 1. «Главная»:

– общее описание сайта;

– вывод самых популярных комиксов;

– вывод самых новых комиксов.

* 1. «Каталог»:

– поиск комиксов по заданным критериям;

– просмотр выпусков;

* 1. «Личный кабинет»:

– просмотр личного кабинета;

– просмотр доступных комиксов;

* 1. «Вход»:

– ввод данных для авторизации;

* 1. «Регистрация»:

– ввод данных для регистрации;

5 Отладка и испытание программы

5.1 Тестовые примеры

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, а также это оценка разрабатываемого программного обеспечения/продукта, чтобы проверить его возможности, способности и соответствие ожидаемым результатам.

1. Тестирование методом черного ящика – осуществляется без каких-либо знаний внутренней работы системы.
2. Тестирование методом белого ящика – учитывает внутреннее функционирование и логику работы кода.
3. Тестирование методом серого ящика – где тестер обладает лишь общими знаниями данного продукта, необходимыми для выполнения теста. Тестирование проводится конечным пользователем, или пользователям, которые представляются как конечные.

Для того чтобы конечный продукт был готов к выпуску он нуждается в тестировании, чтобы не заставить пользователя столкнуться с трудностями в процесс использования программы. Программа будет тестироваться на основе метода серого ящика, так при разработке качественного программного средства необходимо учитывать мнение конечного пользователя, которому предстоит работа с данной программой.

Методы тестирования:

Для проверки корректной работы перехода по якорный ссылкам, приведены следующие тесты:

1. Попытка перехода на страницу «Главная».
2. Попытка перехода на страницу «Каталог».
3. Попытка перехода на страницу «Регистрация».
4. Попытка перехода на страницу «Вход».

Для проверки корректной работы адаптивной верстки приведен следующий тест появление страницы на малых экранах.

Для проверки корректной работы сайта приведены следующие тесты:

1. Вход с правильными данными.
2. Регистрация с неправильными данными.
3. Поиск по категориям.
4. Просмотр комиксов.
5. Просмотр выпусков.

5.2 Анализ полученных результатов

Для тестирования группы корректной работы перехода по якорным ссылкам, приведены следующие тесты:

1. Попытка перехода на страницу «Главная», результат приведен на рисунке 5.1.

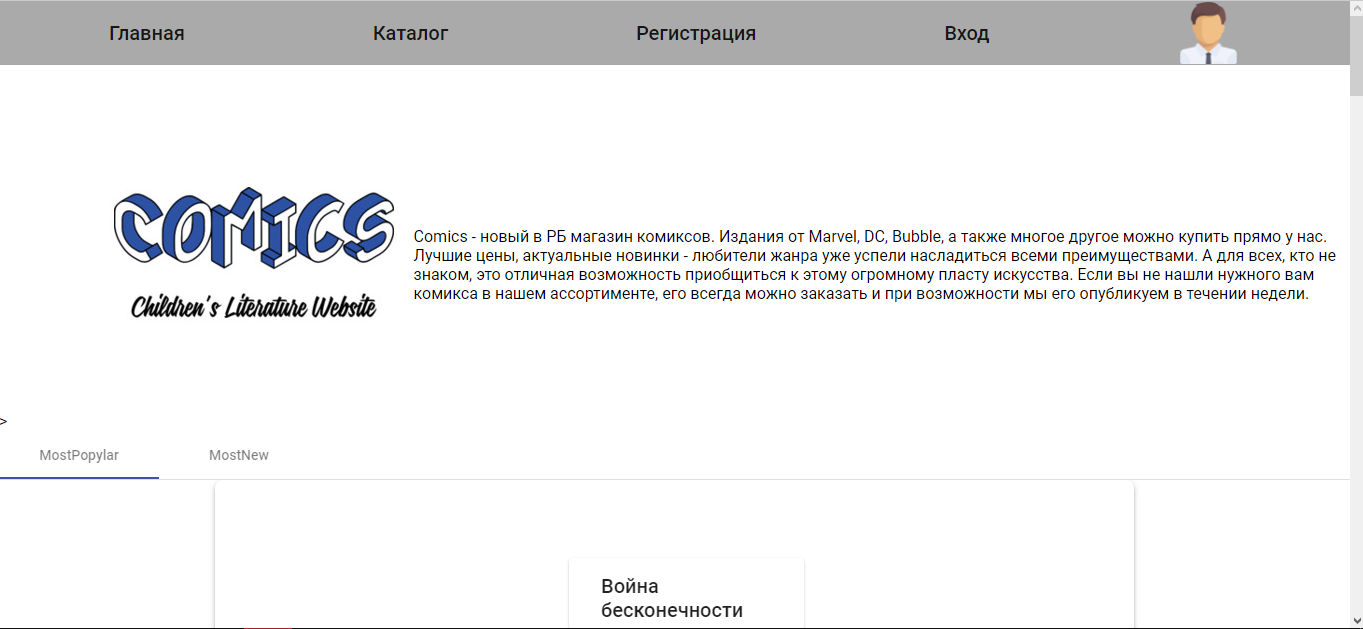


Рисунок 5.1 – Страница «Главная»

Тест показал положительный результат.

1. Попытка перехода на страницу «Каталог», результат приведен на рисунках 5.2.



Рисунок 5.2 – Страница «Каталог»

Тест показал положительный результат.

1. Попытка перехода на страницу «Регистрация», результат приведен на рисунках 5.3.

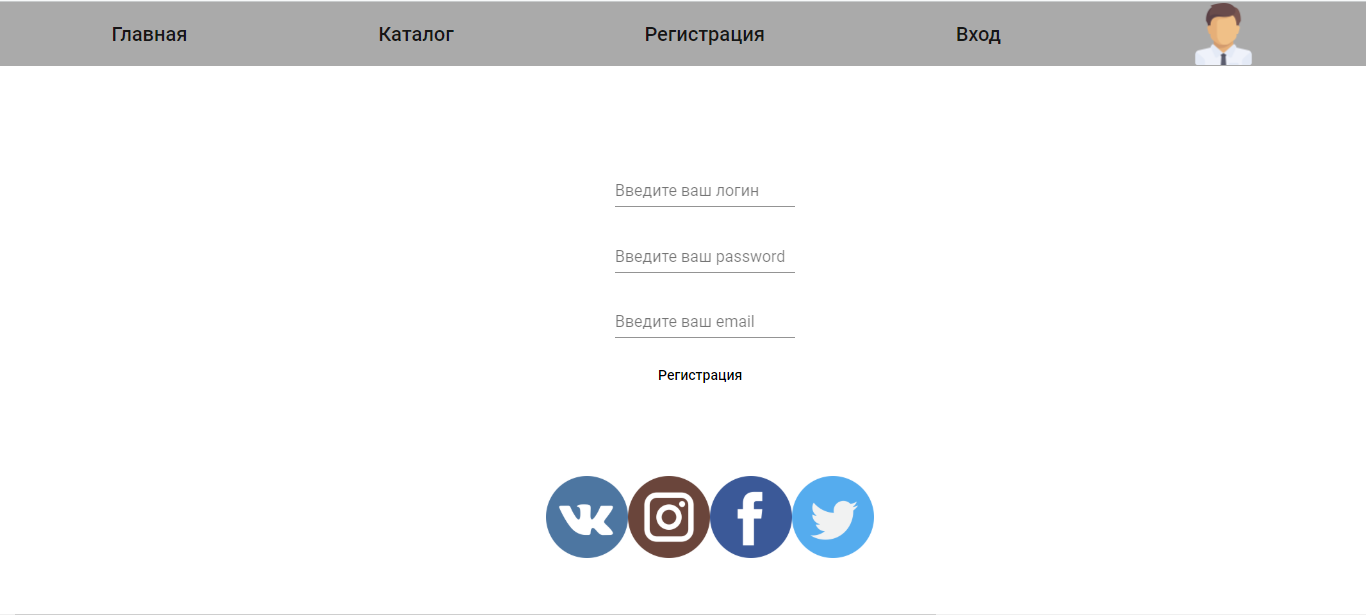


Рисунок 5.3 – Страница «Регистрация»

Тест показал положительный результат.

1. Попытка перехода на страницу «Вход», результат приведен на рисунках 5.4.

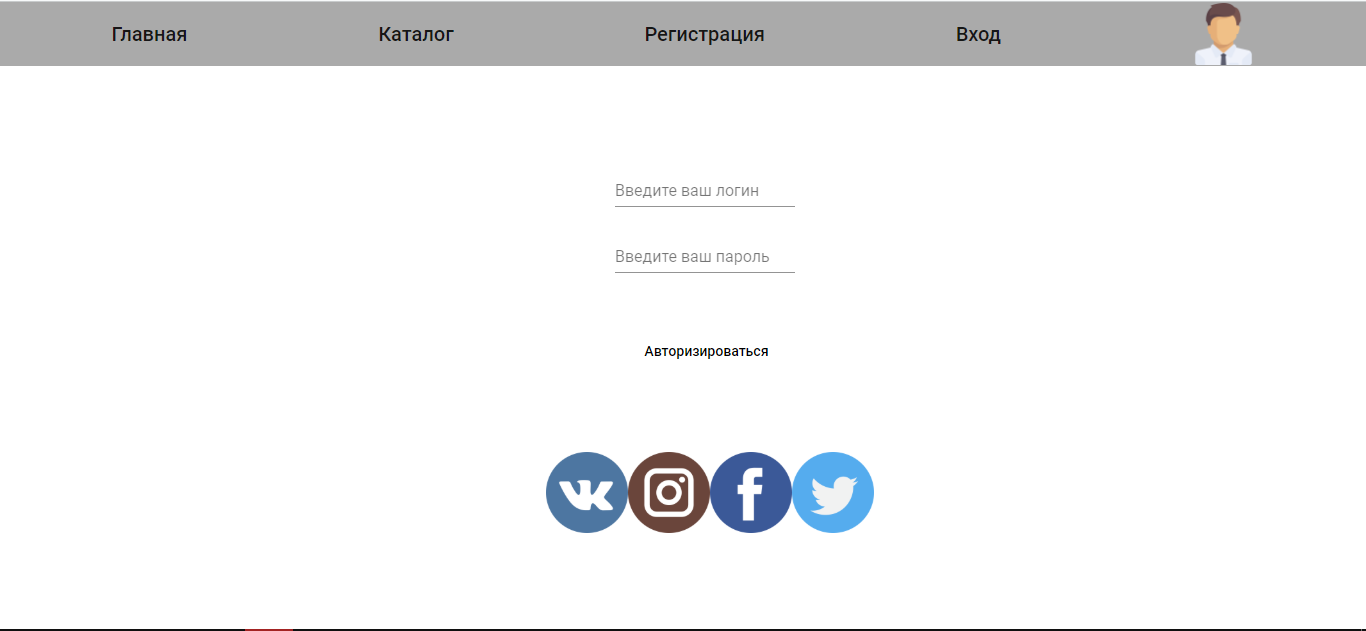


Рисунок 5.4 – Страница «Вход»

Тест показал положительный результат.

По окончании тестирования группы корректной работы перехода по якорным ссылкам, было выявлено что все функции работают стабильно, значит группа корректной работы перехода по якорным ссылкам была выполнена в соответствии с формулировкой задания.

Для тестирования группы корректной работы адаптивной верстки, приведены следующие тесты – появление страницы на больших и на малых экранах, результаты приведены на рисунке 5.5, на рисунке 5.6.

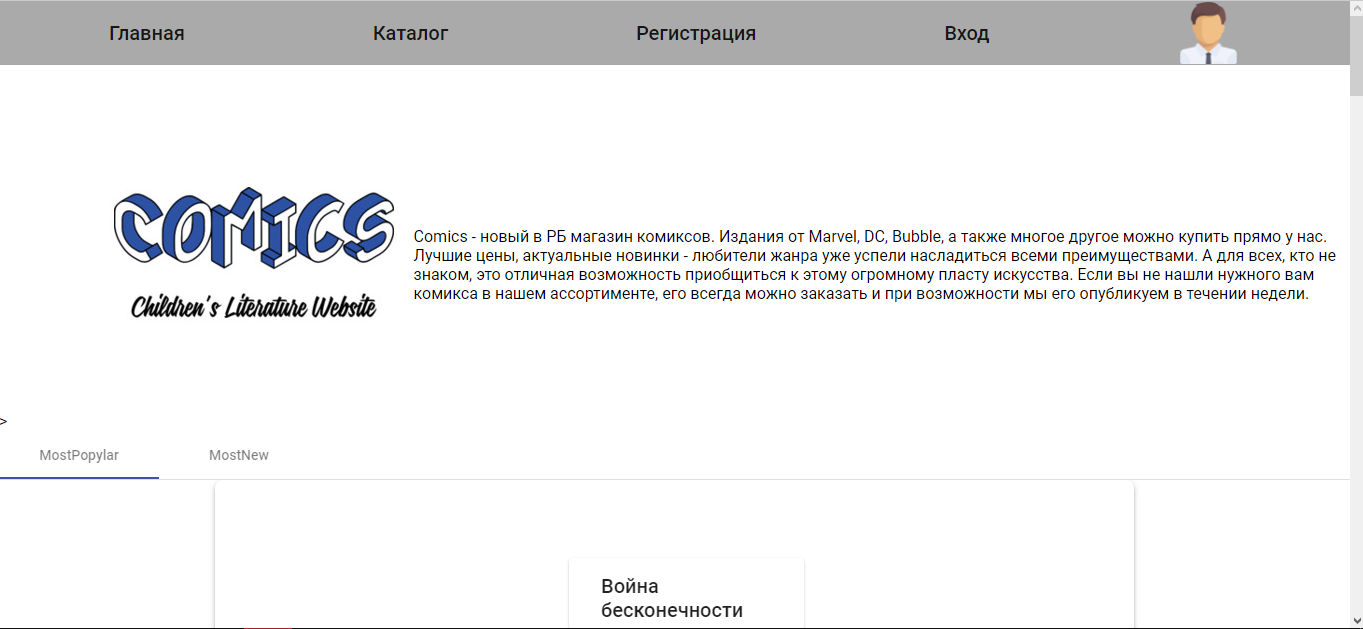


Рисунок 5.5 – Появление страницы на большом экране



Рисунок 5.6 – Появление страницы на малом экране

Тест показал положительный результат.

По окончании тестирования группы корректной работы адаптивной верстки, было выявлено что все функции работают стабильно, значит группа корректной работы адаптивной верстки была выполнена в соответствии с формулировкой задания.

Для проверки корректной работы сайта приведены следующие тесты – вход с правильными данными, регистрация с неправильными данными, поиск по категориям, просмотр комиксов, просмотр выпусков. Результаты приведены на рисунке 5.7, на рисунке 5.8, на рисунке 5.9, на рисунке 5.10, на рисунке 5.11.

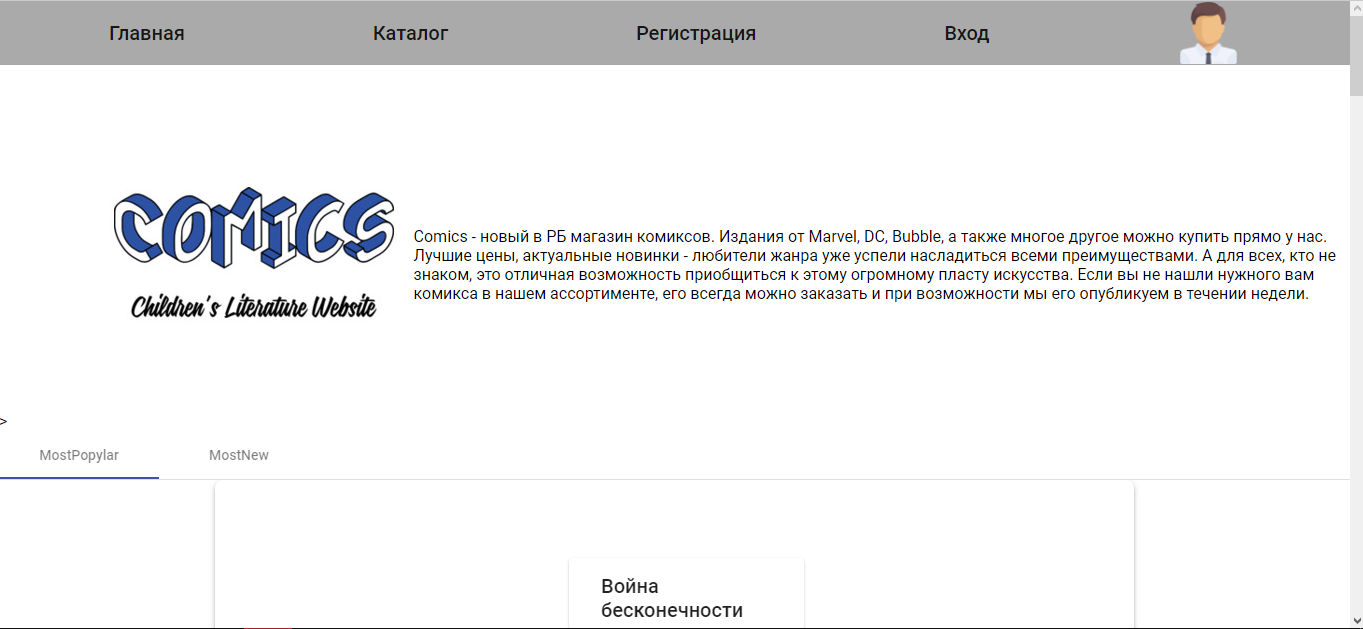


Рисунок 5.7 – Страница, появляющаяся при успешной авторизации.

Тест показал положительный результат.

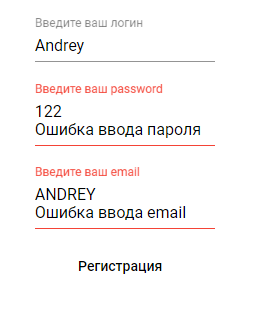


Рисунок 5.8 – Регистрация с неверными данными.

Тест показал положительный результат.

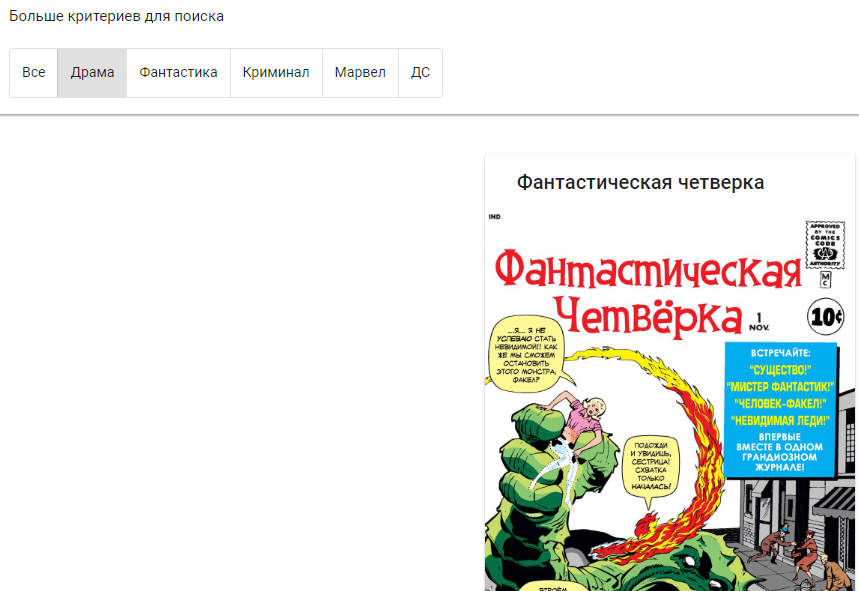


Рисунок 5.9 – Поиск по категориям.

Тест показал положительный результат.



Рисунок 5.10 – Просмотр комикса.

Тест показал положительный результат.



Рисунок 5.11 – Просмотр выпуска.

Тест показал положительный результат.

6 Описание применения программы

6.1 Назначение программы и условия использования

Сайт предназначен для просмотра и записи на консультации.

Требования для корректной работы программы:

1. Процессор: Intel Core 2 Duo E6600, AMD Phenom X3 8750 или выше.
2. ОС «Window 10».
3. Оперативная память: 4 Гб.
4. Видеокарта: Видеокарта должна быть на 256 Мб или выше.
5. Интернет соединение.
6. Веб браузер с поддержкой Html 5.

Для запуска программного средства необходимо ввести в строку поиска в браузере следующую строку http://localhost:4200 После этого пользователя встречает главная страница сайта.

6.2 Описание работы программы

После перехода на сайт появляется главная страница, с которой можно перейти на страницу каталог, вход, регистрация и личный кабинет, если вы авторизированы.

При запуске сайта, пользователь оказывается на главной странице сайта. Пример главной страницы сайта на рисунке 6.1.

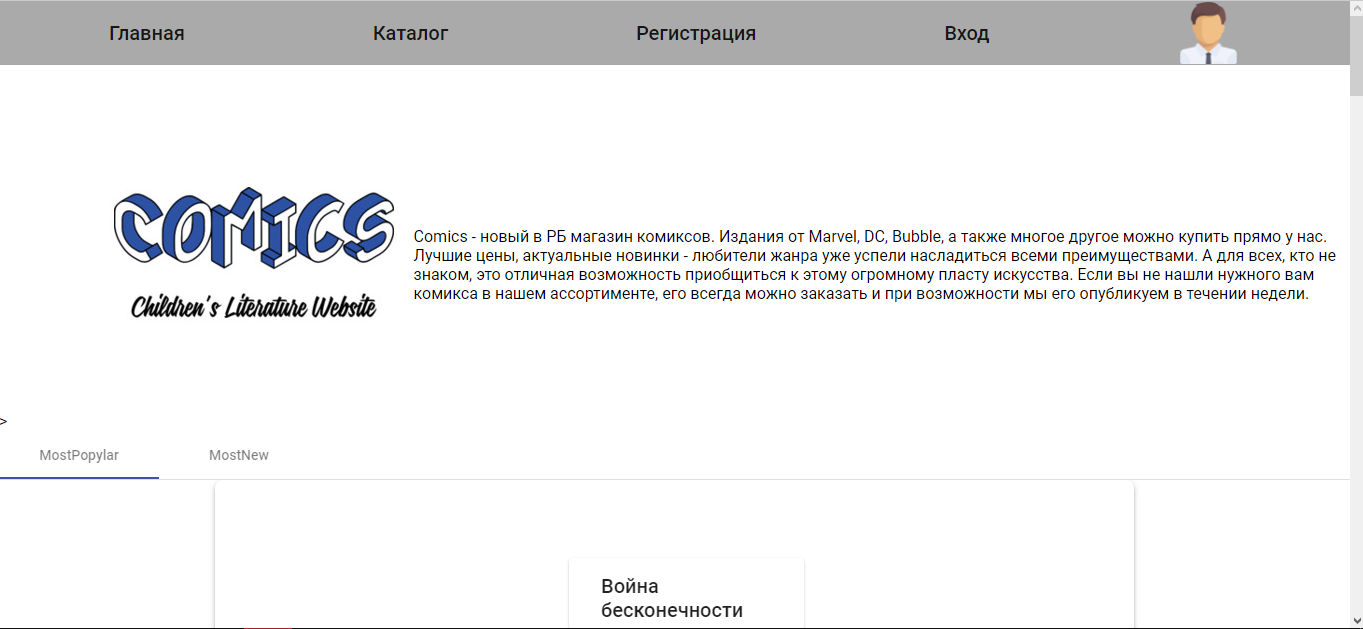


Рисунок 6.1 – Главная страница.

При нажатии на каталог пользователь будет перенаправлен на страницу с контактами сайта. Вверху поле для поиска, а также критерии поиска. Пример страницы на рисунке 6.2.

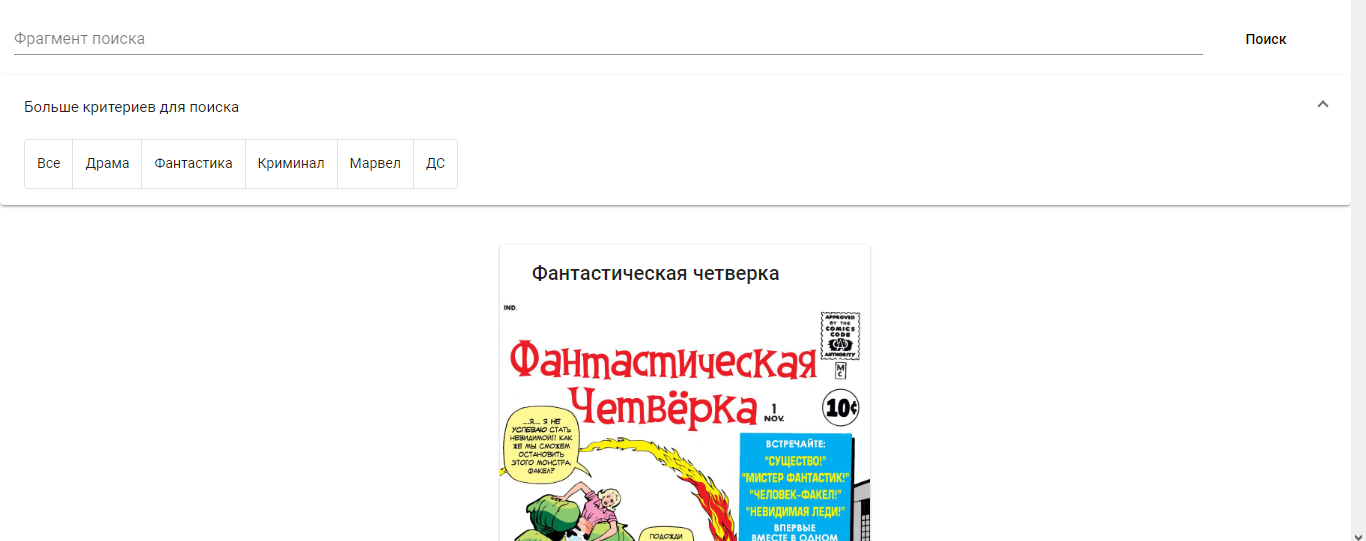


Рисунок 6.2 – Каталог.

При выборе конкретной категории пользователю выдаются комиксы с конкретным жанром. Пример страницы на рисунке 6.3.

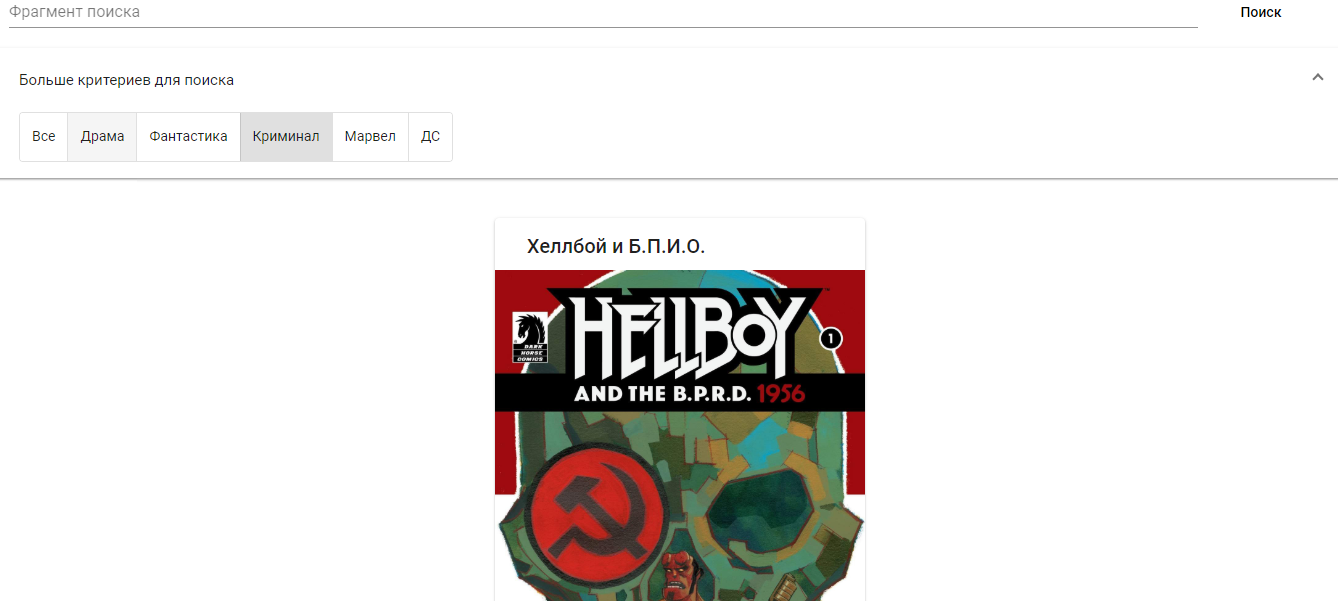


Рисунок 6.3 – Вывод комиксов с жанром «Криминал».

Заключение

В данном курсовом проекте необходимо было разработать сайт детской литературы «Comics».

В процессе разработки были систематизированы и улучшены навыки в области программирования и работы с базами данных, закреплены умения и навыки работы с средой Visual Studio Code, СУБД MySQL, а также с языками JavaScript, TypeScript.

В программе были реализованы следующие функции:

1. Просмотр комиксов.
2. Авторизация.
3. Регистрация
4. Фильтрация данных таблицы.
5. Поиск информации.
6. Просмотр выпусков
7. Покупка выпусков

Дополнить данный проект добавив возможность комментировать выпуски, добавить покупательскую корзину, привязку к реальным деньгам.

Список использованных источников

1. JavaScript [Электронный ресурс]. Режим доступа: http://java-script.com – Дата доступа: 10.11.2018.
2. JavaScript Advanced [Электронный ресурс]. Режим доступа: http://java-script-advanced.com – Дата доступа: 12.11.2018.
3. Linux [Электронный ресурс]. Режим доступа:https://www.linux-linus.com – Дата доступа: 13.11.2018.
4. Ubuntu for desktop [Электронный ресурс] Ubuntu 18.04 – описание. Режим доступа: https://ubuntu-18-04.com. – Дата доступа: 15.11.2018;
5. MySql [Электронный ресурс]. Режим доступа: https://dev.mysql.com/doc/ – Дата доступа: 16.11.2018.

Приложение А

**(обязательное)**

**Листинг программы**

import { Component, OnInit } from "@angular/core";

import { trigger, transition, useAnimation } from "@angular/animations";

import { tada } from "ng-animate";

import { UserRoleService } from "src/app/services/user\_role/user-role.service";

import { LinkService } from "src/app/services/links/link.service";

@Component({

selector: "app-toolbar",

templateUrl: "./toolbar.component.html",

styleUrls: ["./toolbar.component.sass"],

animations: [trigger("tada", [transition("on <=> in", useAnimation(tada))])]

})

export class ToolbarComponent implements OnInit {

tada: any;

links = [];

constructor(

private linkService: LinkService,

private service: UserRoleService

) {}

ngOnInit() {}

getLinks(): any[] {

this.linkService.getLinks().subscribe(data => (this.links = data));

return this.links;

}

ChackAut(): boolean {

return this.service.loggedIn;

}

}

<mat-toolbar>

<span

routerLink="{{ link.path }}"

\*ngFor="let link of getLinks()"

(click)="link.isClicked = !link.isClicked"

[@tada]="link.isClicked ? 'in' : 'on'"

>{{ link.title }}

</span>

<div \*ngIf="ChackAut()">

<img src="../../../../assets/image/man.png" /> <br />

</div>

<div \*ngIf="!ChackAut()">

<img src="../../../../assets/image/un\_man.png" /> <br />

</div>

</mat-toolbar>

<router-outlet></router-outlet>

<app-footer></app-footer>

import { Component, OnInit } from "@angular/core";

import { MainService } from "src/app/services/page/main/main.service";

import { Router } from "@angular/router";

@Component({

selector: "app-main",

templateUrl: "./main.component.html",

styleUrls: ["./main.component.sass"]

})

export class MainComponent implements OnInit {

bd: any[];

bd2: any[];

constructor(private service: MainService, private route: Router) {}

ngOnInit() {

this.getNew();

this.getTop();

}

getTop() {

this.service.getTop().subscribe(data => (this.bd = data));

}

getNew() {

this.service.getNew().subscribe(data => (this.bd2 = data));

}

view\_release(id\_comics: number) {

this.route.navigate(["release", id\_comics]);

}

}

<div id="page">

<div class="text">

<img class="logo" src="../../../../assets/image/new-logo.jpg" />

<p id="p">

Comics - новый в РБ магазин комиксов. Издания от Marvel, DC, Bubble, а

также многое другое можно купить прямо у нас. Лучшие цены, актуальные

новинки - любители жанра уже успели насладиться всеми преимуществами. А

для всех, кто не знаком, это отличная возможность приобщиться к этому

огромному пласту искусства. Если вы не нашли нужного вам комикса в нашем

ассортименте, его всегда можно заказать и при возможности мы его

опубликуем в течении недели.

</p>

</div>

>

<mat-tab-group dynamicHeight>

<mat-tab label="MostPopylar">

<div class="MostPopylarBox mat-elevation-z4">

<mat-card \*ngFor="let comics of bd">

<mat-card-header>

<mat-card-title>{{ comics.name }}</mat-card-title>

</mat-card-header>

<img mat-card-image [src]="comics.image" />

<mat-card-content>

<p>Рейтинг комикса: {{ comics.rating }}</p>

</mat-card-content>

<mat-card-actions>

<button mat-button (click)="view\_release(comics.id)">

Просмотреть

</button>

</mat-card-actions>

</mat-card>

</div>

</mat-tab>

<mat-tab label="MostNew">

<div class="MostNewBox mat-elevation-z4">

<mat-card \*ngFor="let comics of bd2">

<mat-card-header>

<mat-card-title>{{ comics.name }}</mat-card-title>

</mat-card-header>

<img mat-card-image [src]="comics.image" />

<mat-card-content>

<p>Рейтинг комикса: {{ comics.rating }}</p>

</mat-card-content>

<mat-card-actions>

<button mat-button (click)="view\_release()">Просмотреть</button>

</mat-card-actions>

</mat-card>

</div>

</mat-tab>

</mat-tab-group>

</div>

import { Component, OnInit } from "@angular/core";

import { CatalogService } from "src/app/services/page/catalog/catalog.service";

import { trigger, transition, useAnimation } from "@angular/animations";

import { fadeIn } from "ng-animate";

import { Router } from "@angular/router";

@Component({

selector: "app-catalog",

templateUrl: "./catalog.component.html",

styleUrls: ["./catalog.component.sass"],

animations: [trigger("fadeIn", [transition("\* => \*", useAnimation(fadeIn))])]

})

export class CatalogComponent implements OnInit {

req: any[];

fadeIn: any;

constructor(private service: CatalogService, private route: Router) {}

ngOnInit() {

this.getAll();

}

getAll() {

this.service.getAll().subscribe(data => (this.req = data));

}

getFantastic() {

this.service.getFantastic().subscribe(data => (this.req = data));

}

getDrama() {

this.service.getDrama().subscribe(data => (this.req = data));

}

getMarvel() {

this.service.getMarvel().subscribe(data => (this.req = data));

}

getDc() {

this.service.getDc().subscribe(data => (this.req = data));

}

getKriminal() {

this.service.getKriminal().subscribe(data => (this.req = data));

}

view\_release(id\_comics: number) {

this.route.navigate(["release", id\_comics]);

}

getGanre(ganre): any {

if (!ganre) return "load";

if (!ganre.name) {

let ganres = [];

for (let i = 0; i < ganre.length; i++) ganres.push(ganre[i].name);

return ganres;

}

return ganre.name;

}

} <div id="searsh">

<mat-form-field class="searshPanel">

<input id="firstInput" matInput placeholder="Фрагмент поиска" />

</mat-form-field>

<button id="firstButton" mat-button>Поиск</button>

</div>

<mat-accordion>

<mat-expansion-panel>

<mat-expansion-panel-header>

<mat-panel-title> Больше критериев для поиска </mat-panel-title>

</mat-expansion-panel-header>

<mat-button-toggle-group name="fontStyle" aria-label="Font Style">

<mat-button-toggle (click)="getAll()">Все</mat-button-toggle>

<mat-button-toggle (click)="getDrama()">Драма</mat-button-toggle>

<mat-button-toggle (click)="getFantastic()">Фантастика</mat-button-toggle>

<mat-button-toggle (click)="getKriminal()">Криминал</mat-button-toggle>

<mat-button-toggle (click)="getMarvel()">Марвел</mat-button-toggle>

<mat-button-toggle (click)="getDc()">ДС</mat-button-toggle>

</mat-button-toggle-group>

</mat-expansion-panel>

</mat-accordion>

<div [@fadeIn]="fadeIn" id="card">

<mat-card \*ngFor="let comics of req">

<mat-card-header>

<div id="image"></div>

<mat-card-title>

<span>{{ comics.comic.name }}</span>

</mat-card-title>

</mat-card-header>

<img mat-card-image [src]="comics.comic.image" />

<mat-card-content>

<p>Рейтинг комикса: {{ comics.comic.rating }}</p>

<p>Жанр: {{ getGanre(comics.genre) }}</p>

</mat-card-content>

<mat-card-actions>

<button mat-button (click)="view\_release(comics.comic.id)">

Просмотреть

</button>

</mat-card-actions>

</mat-card>

<div></div>

</div>

import { Injectable } from "@angular/core";

import { HttpClient } from "@angular/common/http";

import { Observable } from "rxjs";

import { map } from "rxjs/operators";

import { JwtHelperService } from "@auth0/angular-jwt";

@Injectable({

providedIn: "root"

})

export class UserRoleService {

commonUrl: string = "http://localhost:8080/role\_users";

currentUser: any;

constructor(private http: HttpClient, private jwt: JwtHelperService) {}

getAll(): Observable<any[]> {

return this.http.get<any[]>(this.commonUrl + "/all");

}

getUser(userId: number): Observable<any> {

return this.http.get<any[]>(this.commonUrl + "/user", {

params: {

id: userId.toString()

}

});

}

login(user: any): Observable<boolean> {

return this.http

.post<{ token: string }>(this.commonUrl + "/logIn", user)

.pipe(

map(result => {

localStorage.setItem("token", result.token);

return true;

})

);

}

logout() {

localStorage.removeItem("token");

}

get loggedIn(): boolean {

return localStorage.getItem("token") !== null;

}

getCurrentUser() {

return this.jwt.decodeToken(localStorage.getItem("token"));

}

add(role\_users: any) {

this.http.post<any[]>(this.commonUrl + "/add", role\_users).subscribe();

}

}

var express = require("express");

var app = express();

var bodyParser = require("body-parser");

var cors = require("cors");

var connection = require("./connect");

var bd = require("./bd");

const jwt = require("jsonwebtoken");

const expressJwt = require("express-jwt");

app.use(bodyParser.json());

app.use(

cors({

origin: "http://localhost:4200",

optionsSuccessStatus: 200

})

);

app.use(

expressJwt({ secret: "secret" }).unless({

path: [

"/comics\_strip/all",

"/comics\_strip/top",

"/comics\_strip/new",

"/role\_users/logIn",

"/role\_users/add",

"/ganre\_comics/all/filter",

"/ganre\_comics/drama",

"/ganre\_comics/fantastic",

"/ganre\_comics/dc",

"/ganre\_comics/marvel",

"/ganre\_comics/kriminal",

"/available\_comics/all",

"/available\_comics/avail",

"/users/addUser",

/\/releases\*/,

/\/favicon\*/

]

})

);

connection.sequelize

.authenticate()

.then(() => console.log("YSE NORMAS,MOGNO PIT"))

.catch(err => console.log(err));

//bd.sequelize.sync({ alter: true });

app.listen(8080, "localhost", () => {

console.log("Server startanyl))");

});

require("./routing/controllers")(app);