МИНИСТЕРСТВО ОБРАЗОВАНИЯ, НАУКИ И МОЛОДЕЖИ РЕСПУБЛИКИ КРЫМ

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ РЕСПУБЛИКИ КРЫМ «ФЕОДОСИЙСКИЙ ПОЛИТЕХНИЧЕСКИЙ ТЕХНИКУМ»

КУРСОВОЙ ПРОЕКТ

по профессиональному модулю ПМ 01 «Разработка программных модулей программного обеспечения для компьютерных систем»

по междисциплинарному курсуМДК 01.01 «Разработка программных модулей»

На тему: Разработка программного модуля «Спорт магазин»

Выполнил:

Студент 3 курса группы ИСПП 22 1/9

Караев Алим Сафтерович

Руководитель проекта:

Ульяницкая Н.Н.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Феодосия, 2024

РЕФЕРАТ

Караев Алим Сафтерович: Разработка программного модуля «Спорт магазин».

Курсовой проект по МДК 01.01 Разработка программных модулей. ГБ ПОУ РК «ФПТ». Группа ИСПП 22 1/9. 2024 г.

Пояснительная записка 37 листов.

Таблицы 2 таблицы.

Иллюстрации 15 иллюстраций.

Ссылки 0 ссылок.

Демонстрационная часть 14 слайдов презентации.

В рамках курсового проекта была рассмотрена концепция электронных хранилищ данных для спортивного магазина, проведен анализ информационно-документационного обеспечения управления спортивным магазином, обоснован выбор оптимальной среды разработки, создана реляционная база данных, представлено подробное описание пользовательского интерфейса и разработан программный модуль, упрощающий управление данными о товарах, поставщиках и запасах спортивного инвентаря.

СОДЕРЖАНИЕ

[**ВВЕДЕНИЕ** 4](#_Toc188846806)

[**ТЕОРЕТИЧЕСКАЯ ЧАСТЬ** 6](#_Toc188846807)

[**1.1** **Анализ предметной области** 6](#_Toc188846808)

[**1.2 Обзор существующих аналогов** 7](#_Toc188846809)

[**1.3 Выбор средств разработки** 8](#_Toc188846810)

[**1.3.1 Обоснование выбора среды разработки** 8](#_Toc188846811)

[**РАЗРАБОТКА ПРИЛОЖЕНИЯ «СПОРТ МАГАЗИН»** 14](#_Toc188846812)

[**2.1 Концептуальное, логическое и физическое моделирование базы данных** 14](#_Toc188846813)

[**2.2 Проектирование интерфейса** 18](#_Toc188846814)

[**ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ «СПОРТ МАГАЗИН»** 22](#_Toc188846815)

[**3.1 Тестирование рабочей программы «Информационная система составления тестирования»** 22](#_Toc188846816)

[**3.2 Руководство по использованию** 24](#_Toc188846817)

[**3.2.1 Системные требования** 24](#_Toc188846818)

[**3.2.2 Руководство пользователя** 24](#_Toc188846819)

[**ЗАКЛЮЧЕНИЕ** 28](#_Toc188846820)

[**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ** 29](#_Toc188846821)

[**ПРИЛОЖЕНИЕ А – Листинг кода** 31](#_Toc188846822)



# **ВВЕДЕНИЕ**

С незапамятных времен люди обменивались информацией. Сначала это происходило устно, через речь и слушание. Затем появилось письмо, позволившее записывать и сохранять информацию. Книгопечатание сделало возможным ее массовое распространение. Настоящая революция произошла с появлением компьютеров. Внедрение электронных вычислительных машин (ЭВМ) позволило не только решать отдельные задачи, но и автоматизировать целые процессы обработки информации, выполняемые ранее вручную. Этот прогресс оказал огромное влияние на то, как мы собираем, обрабатываем и распространяем информацию сегодня. В современном цифровом веке информация стала неотъемлемой частью нашей жизни. От социальных сетей до потоковых сервисов — мы постоянно взаимодействуем с огромными объемами данных. Компьютеры и цифровые технологии стали незаменимыми инструментами для управления, анализа и использования этой информации для улучшения нашей жизни и принятия более обоснованных решений. Эффективность систем обработки информации во многом зависит от их использования автоматизированных информационных баз. Это подразумевает, что в памяти компьютеров хранится необходимая информация, которая помогает решать конкретные задачи, для которых предназначена система. Эта информация формирует основу информационной базы соответствующего программного обеспечения. Современные информационные системы требуют надежного хранилища данных, что позволяет более эффективно управлять и обрабатывать информацию. Каждая система нуждается в доступе к актуальной информации, что способствует целенаправленному анализу и быстрому принятию решений. Надежные автоматизированные базы данных упрощают управление данными и облегчают доступ к ним, что повышает общую продуктивность и точность работы системы. Актуальность данной курсовой работы обусловлена необходимостью автоматизации процессов в управлении спортивным магазином. Объектом курсовой работы является управление данными в сфере спортивного магазина. Предметом работы является разработка программного продукта для управления спортивным магазином. В рамках курсовой работы будет создано программное обеспечение для автоматизации управления, которое будет выполнять следующие функции:

* Предоставление информации о спортивных товарах и поставщиках;
* Формирование отчетов по различным критериям;
* Учет движения спортивных товаров;
* Управление процессами складирования и распределения спортивных товаров.

Для достижения целей поставлены следующие задачи:

* Анализ предметной области;
* Выбор среды разработки;
* Построение модели базы данных;
* Разработка ПО;
* Тестирование программного обеспечения.

В работе будут использованы: MySQL Workbench 8.0, Visual Studio 2022, MS Word 2022. Курсовая работа включает три части: первая посвящена предметной области и существующим решениям, вторая - разработанному ПО, третья - тестированию и руководству пользователя.

## **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

# **Анализ предметной области**

Разработка программного модуля для управления данными в спортивном магазине включает в себя сбор и анализ информации о спортивных товарах, поставщиках, категориях, сотрудниках и процессах складирования и распределения продукции. Эта система позволяет эффективно мониторить и управлять ресурсами, что важно для успешной работы спортивного магазина. Основные сферы применения базы данных:

* Учет движения спортивных товаров и их категорий;
* Управление поставщиками и их контактной информацией;
* Обработка данных о спортивных товарах.

Процесс разработки базы данных обычно проходит через несколько этапов:

* Определение требований — сбор информации о потребностях пользователей и законодательных нормах;
* Проектирование структуры — создание схемы базы данных с сущностями и атрибутами;
* Разработка и тестирование системы — создание базы данных и финальная проверка ее работы.

Эффективная реализация позволит достичь точности и надежности в управлении данными о спортивных товарах. Предметом курсовой работы является разработка программного продукта для управления спортивным магазином. Программный продукт представляет собой систему, которая позволит пользователям просматривать и добавлять данные, связанные со спортивными товарами, включая учет движения товаров и обработку информации о поставщиках. Пользователь системы сможет выбрать категорию спортивного товара и другие параметры. Каждый процесс управления включает в себя определенные действия, которые можно легко просмотреть через интерфейс. Пользователь также будет иметь возможность просматривать данные всех таблиц, анализировать данные за указанный период и формировать отчетность для принятия управленческих решений. Система будет направлена на оптимизацию процессов и повышение эффективности в управлении спортивным магазином.

# **1.2 Обзор существующих аналогов**

В ходе выполнения курсовой работы был проведен анализ аналогичных систем для управления данными в спортивных магазинах. Рассмотрены интернет-магазины Спортмастер и Decathlon.

**Спортмастер** — один из ведущих интернет-магазинов спортивных товаров. Позволяет:

* Вести учет движения товаров с отслеживанием категорий и количества товаров.
* Управлять данными о поставщиках и их контактной информацией.
* Анализировать данные и генерировать отчеты.

**Decathlon** — еще один крупный спортивный магазин, предлагающий широкий ассортимент спортивных товаров и инвентаря. Позволяет:

* Централизованно учитывать спортивные товары;
* Обеспечивать многопользовательский доступ через web-интерфейс;
* Генерировать отчеты и анализировать данные.

Резюмируя, оба решения эффективны, но требуют значительных вложений и времени на настройку:

Недостатки платных систем:

* Высокая цена;
* Сложность настройки;
* Необходимость мощного сервера.

Недостатки бесплатных систем:

* Ограниченность возможностей;
* Необходимость доработки.

Преимущества разрабатываемой системы для спортивного магазина:

* Доступ к базе данных с любой машины;
* Легкость в настройке и использовании;
* Низкие требования к системным ресурсам;
* Возможность расширения собственными силами.

Система будет интуитивно понятной и малозатратной на обучение пользователей.

# **1.3 Выбор средств разработки**

# **1.3.1 Обоснование выбора среды разработки**

Для создания программного обеспечения для создания тестирования использовалась среда разработки Microsoft Visual Studio 2022 и язык программирования С#.

Microsoft Visual Studio — продукт компании [Майкрософт](http://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B9%D0%BA%D1%80%D0%BE%D1%81%D0%BE%D1%84%D1%82), который включает [интегрированную среду разработки](http://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) программного обеспечения и ряд других инструментальных средств.

Преимущества Visual Studio 2022:

* Поддержка современных языков и технологий Поддержка множества языков (C#, [VB.NET](https://en.wikipedia.org/wiki/Comparison_of_C_Sharp_and_Visual_Basic_.NET), C++, Python и др.) и технологий, таких как .NET Core и Azure.
* Инструменты отладки Мощные инструменты для отладки кода с возможностью пошагового выполнения и анализа.
* Поддержка Git и других систем контроля версий Встроенные инструменты для работы с Git, что упрощает управление версиями кода.
* Кроссплатформенная разработка Возможность создания приложений для Windows, Android и iOS с использованием Xamarin.

C# — объектно-ориентированный язык программирования, который относится к семье языков с Cи-подобным синтаксисом. Язык имеет поддерживает перегрузку операторов, полиморфизм, атрибуты, делегаты, обобщённые типы и методы, итераторы, анонимные функции.

Основные достоинства С#:

1. C# создавался параллельно с каркасом Framework .Net и полностью учитывает все его возможности.
2. C# - объектно-ориентированный язык, в котором типы представлены в виде классов.
3. C# является объектно-ориентированным языком, в котором существует возможность универсализации и наследования.
4. Программисты C# получают такие же возможности работы с виртуальной машиной, что и программисты Java, благодаря платформе Framework .Net.
5. Библиотека поддерживает удобное построение разных типов программных продуктов на C#, очень просто сохранять и получать информацию из БД.

Реализация, которая сочетает построение надежного и эффективного кода.

**1.3.2 Обоснование выбора СУБД**

Для выбора СУБД необходимо выбрать критерии для сравнения. Критерии выбора современных СУБД: особенности архитектуры; разграничение прав доступа; механизм ролей; репликация; целостность данных; защита данных; ссылочная целостность. Для сравнения будут рассмотрены СУБД: MS SQL Server и MySQL Workbench.

MS SQL Server – реляционная СУБД, построенная на архитектуре клиент-сервер. MS SQL Server ориентирован на использование в операционных системах Windows NT/2000 и использует системные функции. Интерфейс конструктора таблиц, ввод данных, описание схемы данных в MS SQL Server в значительной мере напоминает MySQL Workbench. MySQL Workbench может использоваться и в качестве клиентской части MS SQL Server. Между СУБД есть огромная разница. MySQL Workbench базируется на архитектуре клиент-сервер и приемлем для различных баз данных и разного числа пользователей. SQL Server базируется на архитектуре клиент-сервер и применяется в базах данных среднего и крупного размера с огромным числом пользователей. В MySQL Workbench и SQL Server реализована система разграничения прав доступа. В SQL Server ограничения доступа можно выставлять на отдельные столбцы таблицы. SQL Server позволяет пользователям, которые правильно указали свой пароль при входе в сеть, при доступе к базе данных повторно не вводить его. Существенное отличие состоит в том, что в MySQL Workbench процедуры и функции реализовываются на SQL и других языках, таких как Python и Lua, а в SQL Server основным языком является Transact SQL. Легко импортируются в SQL Server базы данных MySQL. MS SQL Server поддерживает несколько баз данных. Управление SQL Server 2015 осуществляется через Enterprise Manager, который напоминает проводник.

Достоинства СУБД MS SQL Server 2019:

1. Высокая доступность

2. Масштабируемость

3. Безопасность

4. Управляемость

5. Новый набор средств управления для повышения производительности труда администраторов баз данных

Недостатки СУБД MS SQL Server 2019:

1. Моноплатформенность

2. Нельзя определить для пользователя максимальный объём ресурсов СУБД

3. Не целесообразно использовать в системах с большим количеством пользователей и требовательными SQL-запросами

4. Слабость языка TSQL, отсутствие массивов, структур и ООП начала

MySQL Workbench - это инструмент, предоставляемый компанией Oracle, предназначенный для проектирования, разработки и администрирования баз данных MySQL. Он поддерживает архитектуру клиент-сервер и может работать на различных операционных системах, включая Windows, macOS и Linux.

Достоинства:

1. Бесплатность

2. Кроссплатформенность

3. Интуитивный интерфейс

4. Гибкость

5. Репликация и масштабируемость

Недостатки:

1. Производительность: В некоторых случаях производительность может уступать коммерческим решениям, особенно при работе с большими объемами данных.

2. Ограниченная поддержка: В бесплатной версии отсутствует официальная поддержка, что может быть проблемой для крупных предприятий.

3.Совместимость: Возможны проблемы совместимости с другими системами и приложениями, использующими специфичные для других СУБД функции.

MS SQL Server и MySQL Workbench имеют свои преимущества и недостатки. MS SQL Server обеспечивает высокую производительность и безопасность, но может быть дорогим и сложным в настройке. MySQL Workbench предлагает гибкость и доступность, что делает его отличным выбором для небольших и средних компаний.

В таблице 1.1 приведены сравнительные характеристики систем управления базами данных.

Таблица 1.1 – Сравнение СУБД

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название характеристики | СУБД Access | СУБД MS SQL Server |
| 1 | Архитектура | Клиент-сервер | Клиент-сервер |
| 2 | Разграничение прав доступа | Присутствует | Ограничения доступа можно на отдельные столбцы |
| 3 | Механизм ролей | Присутствует | Присутствует |
| 4 | Язык программирования | SQL, PL/SQL | Transact SQL, XML |
| 5 | Репликация | Присутствует | Присутствует |
| 6 | Целостность данных | Поддерживается с помощью ограничений стандарта ANSI. | Поддерживается с помощью ограничений стандарта ANSI. |
| 7 | Защита данных | Защита данных осуществляется с помощью шифрования. | Защита данных осуществляется с помощью шифрования. |

Из вышесказанного можно сделать вывод, что выбор MySQL Workbench объясняется следующими причинами:

* Интуитивный интерфейс;
* Репликация и масштабируемость;
* Совместимость и интеграция.

## **РАЗРАБОТКА ПРИЛОЖЕНИЯ «СПОРТ МАГАЗИН»**

## **2.1 Концептуальное, логическое и физическое моделирование базы данных**

Для проектирования структуры базы данных используется метод «сущность – связь».

При проектировании базы данных необходимо придерживаться этапов проектирования.

Первый этап проектирования – выделение сущностей и связей между ними.

Выделены следующие сущности:

* ДеталиЗаказов (Ключ – ID\_ДеталиЗаказа)
* Заказы (Ключ – ID\_Заказа)
* Заказчики (Ключ – ID\_Заказчика)
* Категории (Ключ – ID\_Категории)
* Товары (Ключ – ID\_Товара)

Выделены связи между сущностями:

* Товары относятся к Категории.
* ДеталиЗаказа относятся к Заказу.
* Заказ связан с Заказчиком.
* Заказ включает Товары.

Второй этап проектирования – построение диаграммы ER-типа с учетом всех сущностей и связей между ними.

Каждый товар принадлежит определенной категории (ID\_Категории):



Рисунок 2.1 – Диаграмма ER-типа для связи товар относится к категории.

Каждая деталь в заказе относится к заказу (ID\_Заказа):

Рисунок 2.2 – Диаграмма ER-типа для связи детали заказов относятся заказу.

Каждый заказ связан с заказчиком (ID\_Заказчика):

Рисунок 2.3 – Диаграмма ER-типа для связи заказ связан с заказчиком.

Каждый заказ включает товары (ID\_Товара):

Рисунок 2.4 – Диаграмма ER-типа для связи заказ включает товары.

**Третий этап проектирования – формирование набора предварительных отношений:**

Для связи **товар относится к категории**:

* Товары (ID\_Товара)
* Категории (ID\_Категории)

Для связи **детали заказов относятся заказу**:

* Товары (ID\_Товара)
* Заказ (ID\_Заказа)

Для связи **заказ связан с заказчиком**:

* Заказ (ID\_Заказа)
* Заказчик (ID\_Заказчика)

Для связи **заказ включает товар**:

* Заказ (ID\_Заказа)
* Товары (ID\_Товара)

**Четвертый этап проектирования – формирование набора результирующих отношений и добавление в них неключевых атрибутов:**

После добавления неключевых атрибутов схемы отношений примут следующий вид:

* **Заказчики** (ID, Имя, E-mail)
* **Категории** (ID, Название)
* **ДеталиЗаказа** (ID, ID\_Заказа, ID\_Товара, Количество, Цена)
* **Заказы** (ID, ID\_Заказчика, Дата\_заказа, Сумма\_заказа)
* **Товары** (ID, ID\_Категории, Цена, Количество\_на\_складе)

На заключительном, пятом этапе проектирования, необходимо составить логическую модель базы данных с указанием первичных и внешних ключей. Логическая модель базы данных представлена на рисунке 2.5.

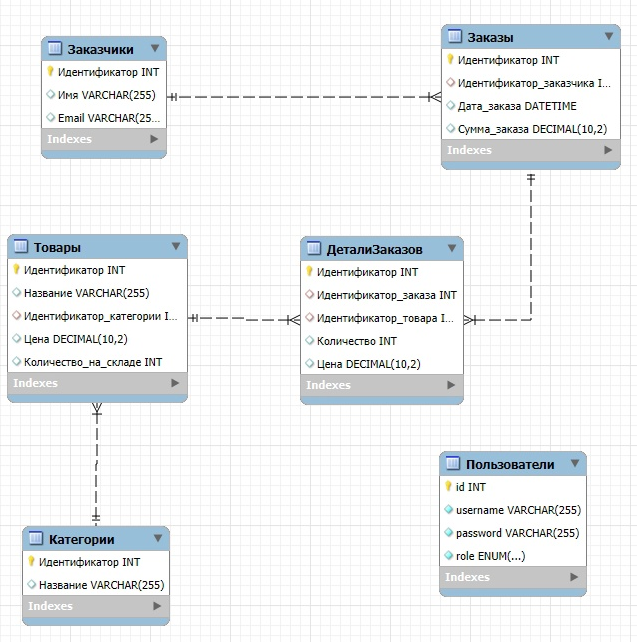


Рисунок 2.5 – Логическая модель базы данных

Под физической организацией БД понимают совокупность методов и средств размещения данных во внешней памяти и созданную на их основе внутреннюю (физическую) модель данных. Внутренняя модель является средством отображения логической модели данных в физическую среду хранения. В отличие от логической модели физическая модель данных связана со способами организации данных на носителях, методами доступа к данным. Эта модель указывает, каким образом записи размещаются в БД, как они упорядочены, как организуются связи и выборка данных.

Конечным итогом разработки физической организации БД являются файлы данных — файл БД и файлы поисковых структур.

На рисунке 2.6 представлена диаграмма разработанной базы данных, которая отображает взаимосвязь таблиц и ключевые поля в таблицах.

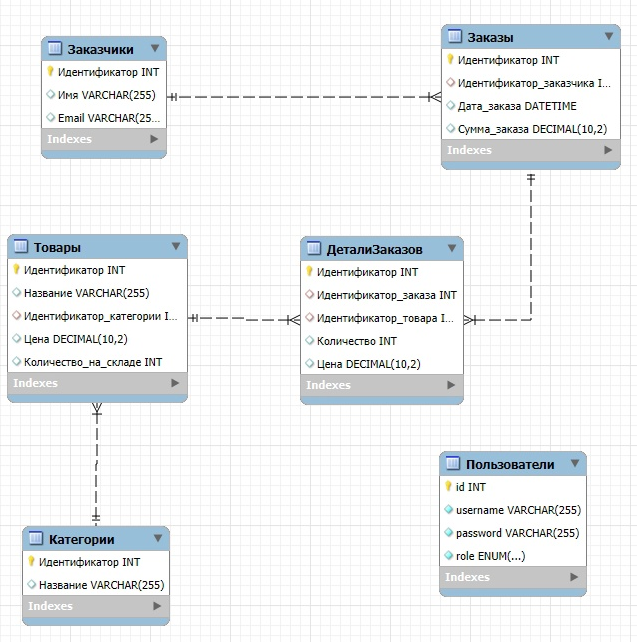


Рисунок 2.6 – Физическая модель базы данных

# **2.2 Проектирование интерфейса**

Разработка программного обеспечения осуществляется посредство системы разработки программного обеспечения – Visual Studio 2022. Вся база данных хранится на отдельном сервере.

При запуске программы пользователь попадает на окно авторизации, где (в случае необходимости) можно зарегистрировать новый аккаунт.

Вид формы авторизации приведен на рисунке 2.7.

Вид формы регистрации приведен на рисунке 2.8.

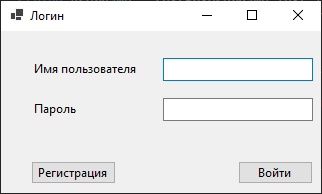


Рисунок 2.6 – Вид формы авторизации

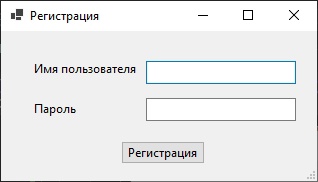


Рисунок 2.7 – Вид формы регистрации

Форма авторизации включает в себя следующие элементы управления:

* Button – представляет элемент управления «кнопка Windows». Используется для подтверждения введенных данных и отправки их на сервер для проверки.
* TextBox – текстовые поля для ввода данных. В данной форме используются два текстовых поля: одно для ввода имени пользователя, другое для ввода пароля.
* Label – метки, которые указывают на поля ввода и помогают пользователю понять, какие данные необходимо вводить. В данной форме используются следующие метки:

Пространство имен MySql.Data.MySqlClient используется для взаимодействия с базой данных MySQL. Класс MySqlConnection устанавливает соединение с базой данных, а класс MySqlCommand выполняет SQL-запросы для проверки введенных пользователем данных.

Форма регистрации включает в себя следующие элементы управления:

* TextBox – текстовые поля для ввода данных. В данной форме используются 2 текстовых поля:
  1. Логин
  2. Пароль
* Label – метки, которые указывают на поля ввода и помогают пользователю понять, какие данные необходимо вводить. В данной форме используются следующие метки:
  1. Имя пользователя – указывает на текстовое поле для ввода логина.
  2. Пароль – указывает на текстовое поле для ввода пароля.
* Button – кнопка, которая используется для подтверждения введенных данных и записи их в таблицу базы данных MySQL.

Пространство имен MySql.Data.MySqlClient используется для взаимодействия с базой данных MySQL. Класс MySqlConnection устанавливает соединение с базой данных, а класс MySqlCommand выполняет SQL-запросы для записи данных пользователей в таблицу.

После авторизации, мы попадаем на окно базы данных. Вид формы базы данных приведен на рисунке 2.9.

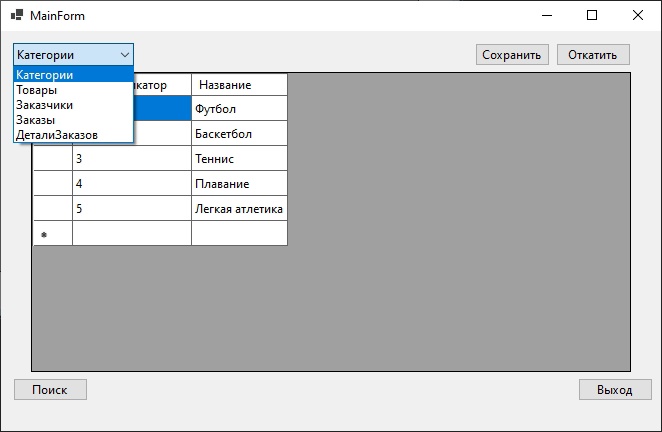


Рис 2.9 – Вид формы базы данных

Форма с базой данных включает в себя следующие элементы управления:

* ComboBox – комбинированное поле для выбора таблицы.
* DataGridView – элемент управления для отображения данных из выбранной таблицы в виде таблицы.
* Button – кнопки для выполнения различных действий:
  1. Сохранить – используется для сохранения изменений в таблице. Данная кнопка активна только для пользователей, вошедших с аккаунта администратора.
  2. Выйти – кнопка для выхода из системы.
  3. Поиск – используется для поиска элементов.
  4. Откатить – используется для отката ошибочных изменений, если не было сохранено.

## **ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ «СПОРТ МАГАЗИН»**

## **3.1 Тестирование рабочей программы «Информационная система составления тестирования»**

Тестовый случай — это алгоритмы проверки функциональности программы. Каждый тестовый случай должен обладать следующими свойствами: четкой целью проверки, известными начальными условиями тестирования, строго определенной средой тестирования, тестовыми данными и ожидаемым результатом тестирования.

Результаты тестирования модулей представлены в таблице 3.1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название  модуля/экрана | Описание тестового случая | Ожидаемые результаты | Тестовый случай пройден? | Комментарии |
| Кнопка «Войти» | Одинарный клик левой кнопкой мыши | Вход в систему | Да | Работает корректно |
| Кнопка «Регистрация» на форме авторизации | Одинарный клик левой кнопкой мыши | Открытие формы регистрации | Да | Работает корректно |
| Кнопка «Регистрация» на форме регистрации | Одинарный клик левой кнопкой мыши | Регистрация в системе, закрытие формы регистрации, открытие формы авторизации. | Да | Работает корректно |
| Кнопка «Поиск» | Одинарный клик левой кнопкой мыши | Открывает отдельную форму, где надо выбрать значения из 3-х ComboBox | Да | Работает корректно |
| Кнопка «Сохранить» | Одинарный клик левой кнопкой мыши | Сохраняет изменения в таблице | Да | Работает корректно |
| Кнопка «Откатить» | Одинарный клик левой кнопкой мыши | Откат ошибочных изменений, если не нажата кнопка сохранить | Да | Работает корректно |
| Кнопка «Выйти» | Одинарный клик левой кнопкой мыши | Выход из системы, закрытие формы базы данных, открытие формы авторизации. | Да | Работает корректно |

Так же в результате реализации тестирования было зафиксировано, что все условные переходы выполняются в каждом направлении, не происходит «зацикливания» в модулях при граничных значениях индексов циклов, а также не обнаружено сбоев в работе при невыполнении тела какого-либо из циклов.

Комплексное тестирование. Комплексное тестирование – процесс поиска несоответствия приложения его исходным целям.

Оно состоит из следующих шагов:

1. Стресс-тестирование. Обычная проблема крупных систем в том, что они могут работать без сбоев при низкой или средней нагрузке, но начинают ломаться под высоким давлением или в стрессовых условиях реального использования. Стресс-тестирование включает в себя проверку системы на устойчивость к экстремальным нагрузкам и условиям, чтобы выявить её слабые места.

Для проведения тестов осуществлялось большое количество запросов к БД (12 запросов). В результате теста не было зафиксировано никаких отклонений в работе программы, но было отмечено определенное замедление работы БД с запросами.

1. Тестирование безопасности. С увеличением внимания к защите конфиденциальности, большинству систем предъявляются строгие требования по предотвращению несанкционированного доступа. Цель тестирования безопасности заключается в выявлении уязвимостей системы, которые могут привести к нарушению конфиденциальности данных.

В результате проведения теста было зафиксировано, в систему может зайти любой пользователь. В саму же базу данных без доступа никто проникнуть не может.

## **3.2 Руководство по использованию**

### **3.2.1 Системные требования**

Данное приложение функционирует на операционной системе Windows 10 и выше.

Минимальные характеристики ПК:

* процессор x64: Тактовая частота 1,4 ГГц; AMD Opteron; AMD Athlon 64; Intel Xeon с поддержкой Intel EM64T; Intel Pentium IV с поддержкой EM64T;
* процессор x86: Тактовая частота 1,0 ГГц; процессор совместимый с Pentium III или выше;
* платформа .NET 4,0;
* жесткий диск 10 ГБ.

### **3.2.2 Руководство пользователя**

При успешном запуске приложения «Спорт магазин» появляется форма авторизации, представленная на рисунке 3.1.

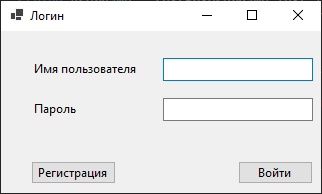


Рисунок 3.1 – Форма авторизации.

Для перехода к базе данных надо или войти, используя данные, или зарегистрироваться (рис 3.2).

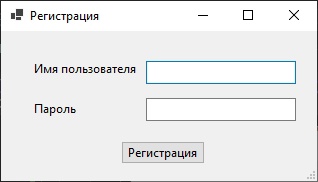


Рисунок 3.2 – Форма регистрации.

После входа в систему мы попадаем на окно с базой данных. В зависимости от роли пользователя, форма будет немного отличаться. Формы за разные роли представлены на рисунках 3.3 и 3.4.

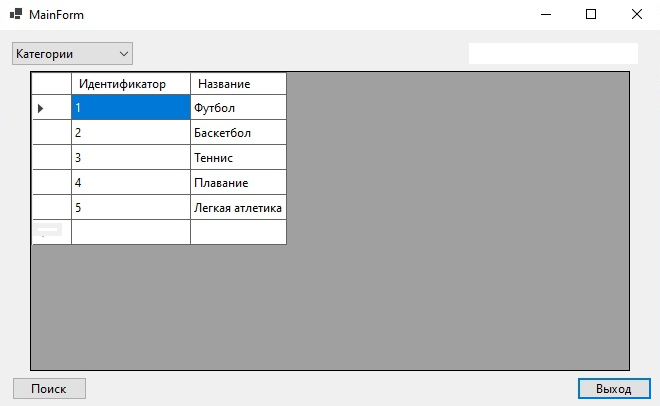


Рисунок 3.3 – Форма с базой данных за обычного пользователя.

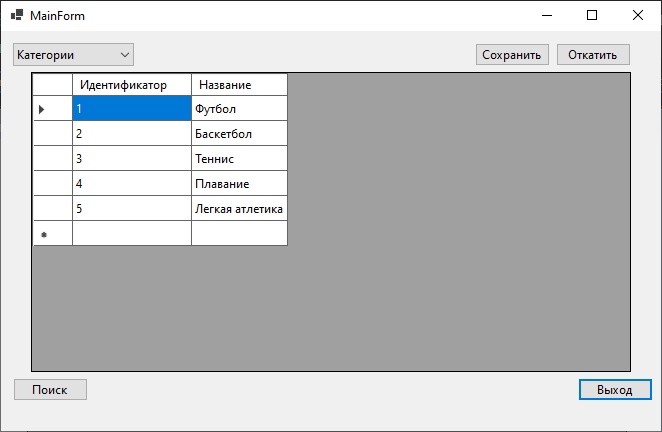


Рисунок 3.4 – Форма с базой данных за администратора.

Следует заметить, что редактирование доступно только для пользователя с правами администратора. Также есть отдельная кнопка, чтобы сохранить изменения. Произвольные значения также невозможно вписать, иначе появится ошибка (рисунок 3.5).

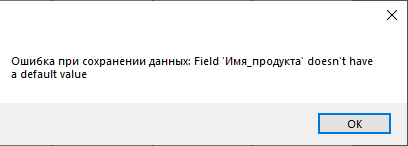


Рисунок 3.5 – Ошибка при сохранении неверных данных.

Если все значения будут введены верно, то таблица успешно сохранится (рисунок 3.6).

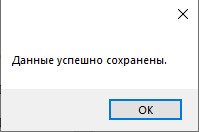


Рисунок 3.6 – Данные успешно сохранены.

# **ЗАКЛЮЧЕНИЕ**

В данной курсовой работе были достигнуты все поставленные задачи и цели.

Разработано программное средство для автоматизации процессов в работе спорт магазина.

Для достижения результата были выполнены следующие задачи:

* проведен сравнительный анализ аналогов системы;
* проанализирована предметная область;
* выделены сущности, их атрибуты и связи между ними;
* разработана логическая и физическая модель данных;
* создан программный интерфейс в среде MS Visual Studio 2022.

Программное обеспечение позволяет вносить новые данные, удалять записи, изменять существующую информацию и искать необходимую информацию. Анализ данных обеспечивает удобство и быстроту работы. Дружественный интерфейс снижает затраты на обучение персонала.

Курсовая работа подчеркивает важность автоматизации при работе в спорт магазине, что улучшает управление данными и обработку данных.

# **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. Крёнке, Д. Теория и практика построения баз данных. 8-е изд./ Д. Крёнке. — СПб.: Питер, 2013. — 800 с.: ил.
2. Шумаков П.В. ADO.NET и создание приложений баз данных в среде Microsoft Visual Studio.NET/ П.В. Шумаков. –: М.: ДИАЛОГ-МИФИ, 2013. – 258 с.
3. Либерти Д. Программирование на C#/ Д. Либерти. – М.: Издательский дом «Символ-Плюс», 2014. – 688 с.
4. Рихтер Д. Программирование на платформе Microsoft .NET Framework/ Д. Рихтер. – СПб.: Русская редакция, 2013. – 512 с.
5. Дейт, К. Дж. SQL и реляционная теория/ К. Дж. Дейт. – М.: Издательский дом «Вильямс», 2015. – 474 с.
6. Visual Studio 2017 [Электронный ресурс]. – 2003-2017. – Электрон дан. – Режим доступа: <https://www.visualstudio.com/ru-ru/visual-studio-homepage-vs.aspx> (дата обращения: 10.04.2018)
7. Вирт, Никлаус. Алгоритмы и структуры данных/ Н. Вирт. – СПб.: БХВ-Петербург, 2015. – 274 с.
8. Кватрани Т. RationalRose 2000 и UML. Визуальное моделирование/ Т. Кватрани. – М.: Издательский дом «Вильямс», 2013. – 583 с.
9. Кумар, В. NET Сетевое программирование/ В. Кумар, Э. Кровчик, Н. Лагари. – М.: Лори, 2014. – 500 с.: ил.
10. Рихтер, Дж. Программирование на платформе Microsoft .NET Framework 4.0 на языке C#/ Дж. Рихтер. – 3-е издание. – СПб.: Питер, 2014. – 400 с.: ил.
11. Микелсен К. Язык программирования C#. Лекции и упражнения/ К. Микелсен. – М.: Издательский дом «ДиаСофт», 2015. – 656 с.
12. Петзольд Ч. Программирование для Microsoft Windows на С#. В 2 томах/ Ч. Петзольд. – СПб.: Русская редакция, 2013. – 536 с.
13. Троелсен Э. C# и платформа .NET. Библиотека программиста/ Э. Троелсен – СПб.: Русская редакция, 2014. – 796 с.
14. Тестирование программного обеспечения [Электронный ресурс] / Информация о тестировании программного обеспечения. – 2003-2017. – Электрон дан. – Режим доступа: <http://www.it-courses.by/sqa/> (дата обращения: 10.04.2018).
15. Тейер, Т. Надежность программного обеспечения / Т. Тейер, М. Липов, Э. Нельсон. – М.: Мир, 2015. – 323 с.
16. Тестирование модуля [Электронный ресурс] / Информация о тестировании модуля. – 2003-2017. – Электрон дан. – Режим доступа: http://mgumoscow.blogspot.ru/2012/10/blog-post\_2400.html. (дата обращения: 10.04.2018)

# **ПРИЛОЖЕНИЕ А – Листинг кода**

DatabaseConnection.cs

using MySql.Data.MySqlClient;

namespace KursAlim

{

internal class DatabaseConnection

{

private static readonly string connectionString = "server=95.183.12.18;port=3306;user=Alim;database=AlimDB;";

public static MySqlConnection GetConnection()

{

return new MySqlConnection(connectionString);

}

}

}

FormLogin.cs

using System;

using System.Security.Cryptography;

using System.Text;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace KursAlim

{

public partial class FormLogin : Form

{

public FormLogin()

{

InitializeComponent();

}

private void buttonLogin\_Click(object sender, EventArgs e)

{

string username = textBoxUsername.Text.Trim();

string password = textBoxPassword.Text.Trim();

// Проверка на пустые поля

if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(password))

{

MessageBox.Show("Пожалуйста, заполните все поля.");

return;

}

password = HashPassword(password); // Используем хеширование пароля

using (MySqlConnection connection = DatabaseConnection.GetConnection())

{

connection.Open();

string query = "SELECT \* FROM Пользователи WHERE username = @username AND password = @password";

MySqlCommand command = new MySqlCommand(query, connection);

command.Parameters.AddWithValue("@username", username);

command.Parameters.AddWithValue("@password", password);

MySqlDataReader reader = command.ExecuteReader();

if (reader.Read())

{

string role = reader["role"].ToString();

MessageBox.Show("Авторизация успешна!");

// Перейти к главной форме приложения

this.Hide();

MainForm mainForm = new MainForm(role);

mainForm.Show();

}

else

{

MessageBox.Show("Неверное имя пользователя или пароль");

}

reader.Close();

}

}

private void buttonRegister\_Click(object sender, EventArgs e)

{

FormRegister registerForm = new FormRegister();

registerForm.ShowDialog();

}

private string HashPassword(string password)

{

using (SHA256 sha256 = SHA256.Create())

{

byte[] bytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(password));

StringBuilder builder = new StringBuilder();

foreach (byte b in bytes)

{

builder.Append(b.ToString("x2"));

}

return builder.ToString();

}

}

}

}

FormRegister.cs

using System;

using System.Security.Cryptography;

using System.Text;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace KursAlim

{

public partial class FormRegister : Form

{

public FormRegister()

{

InitializeComponent();

}

private void buttonRegister\_Click(object sender, EventArgs e)

{

string username = textBoxUsername.Text.Trim();

string password = textBoxPassword.Text.Trim();

// Проверка на пустые поля

if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(password))

{

MessageBox.Show("Пожалуйста, заполните все поля.");

return;

}

password = HashPassword(password);

using (MySqlConnection connection = DatabaseConnection.GetConnection())

{

connection.Open();

string query = "INSERT INTO Пользователи (username, password, role) VALUES (@username, @password, 'user')";

MySqlCommand command = new MySqlCommand(query, connection);

command.Parameters.AddWithValue("@username", username);

command.Parameters.AddWithValue("@password", password);

try

{

command.ExecuteNonQuery();

MessageBox.Show("Регистрация прошла успешно!");

this.Close();

}

catch (MySqlException ex) when (ex.Number == 1062)

{

MessageBox.Show("Имя пользователя уже существует. Пожалуйста, выберите другое имя.");

}

catch (Exception ex)

{

MessageBox.Show($"Произошла ошибка: {ex.Message}");

}

}

}

private string HashPassword(string password)

{

using (SHA256 sha256 = SHA256.Create())

{

byte[] bytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(password));

StringBuilder builder = new StringBuilder();

foreach (byte b in bytes)

{

builder.Append(b.ToString("x2"));

}

return builder.ToString();

}

}

}

}

FormSearch.cs

using System;

using System.Data;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace KursAlim

{

public partial class FormSearch : Form

{

private MySqlConnection connection;

public FormSearch()

{

InitializeComponent();

connection = DatabaseConnection.GetConnection();

LoadTables();

}

private void LoadTables()

{

comboBoxTables.Items.Add("Категории");

comboBoxTables.Items.Add("Товары");

comboBoxTables.Items.Add("Заказчики");

comboBoxTables.Items.Add("Заказы");

comboBoxTables.Items.Add("ДеталиЗаказов");

comboBoxTables.SelectedIndex = 0;

LoadColumns();

}

private void LoadColumns()

{

string selectedTable = comboBoxTables.SelectedItem.ToString();

string query = $"SELECT \* FROM {selectedTable} LIMIT 1";

MySqlDataAdapter adapter = new MySqlDataAdapter(query, connection);

DataTable table = new DataTable();

adapter.Fill(table);

comboBoxColumns.Items.Clear();

foreach (DataColumn column in table.Columns)

{

comboBoxColumns.Items.Add(column.ColumnName);

}

comboBoxColumns.SelectedIndex = 0;

LoadValues();

}

private void LoadValues()

{

string selectedTable = comboBoxTables.SelectedItem.ToString();

string selectedColumn = comboBoxColumns.SelectedItem.ToString();

string query = $"SELECT DISTINCT {selectedColumn} FROM {selectedTable}";

MySqlDataAdapter adapter = new MySqlDataAdapter(query, connection);

DataTable table = new DataTable();

adapter.Fill(table);

comboBoxValues.Items.Clear();

foreach (DataRow row in table.Rows)

{

comboBoxValues.Items.Add(row[selectedColumn].ToString());

}

comboBoxValues.SelectedIndex = 0;

}

private void buttonSearch\_Click(object sender, EventArgs e)

{

string selectedTable = comboBoxTables.SelectedItem.ToString();

string selectedColumn = comboBoxColumns.SelectedItem.ToString();

string selectedValue = comboBoxValues.SelectedItem.ToString();

string query = $"SELECT \* FROM {selectedTable} WHERE {selectedColumn} = '{selectedValue}'";

MySqlDataAdapter adapter = new MySqlDataAdapter(query, connection);

DataTable table = new DataTable();

adapter.Fill(table);

dataGridViewResults.DataSource = table;

}

private void comboBoxTables\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadColumns();

}

private void comboBoxColumns\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadValues();

}

}

}

MainForm.cs

using System;

using System.Data;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace KursAlim

{

public partial class MainForm : Form

{

private MySqlConnection connection;

private string currentUserRole;

private DataTable originalDataTable;

private DataTable currentDataTable;

public MainForm(string currentUserRole)

{

InitializeComponent();

this.currentUserRole = currentUserRole;

connection = DatabaseConnection.GetConnection();

LoadTables();

ConfigureButtons();

}

private void ConfigureButtons()

{

buttonSaveChanges.Visible = currentUserRole == "admin";

buttonUndoChanges.Visible = currentUserRole == "admin";

}

private void LoadTables()

{

comboBoxTables.Items.Add("Категории");

comboBoxTables.Items.Add("Товары");

comboBoxTables.Items.Add("Заказчики");

comboBoxTables.Items.Add("Заказы");

comboBoxTables.Items.Add("ДеталиЗаказов");

comboBoxTables.SelectedIndex = 0;

LoadData();

}

private void LoadData()

{

string selectedTable = comboBoxTables.SelectedItem.ToString();

string query = $"SELECT \* FROM {selectedTable}";

MySqlDataAdapter adapter = new MySqlDataAdapter(query, connection);

originalDataTable = new DataTable();

adapter.Fill(originalDataTable);

currentDataTable = originalDataTable.Copy();

dataGridView.DataSource = currentDataTable;

dataGridView.ReadOnly = currentUserRole != "admin";

buttonSaveChanges.Enabled = currentUserRole == "admin";

buttonUndoChanges.Enabled = currentUserRole == "admin";

dataGridView.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;

this.ClientSize = new System.Drawing.Size(dataGridView.Width + 60, dataGridView.Height + 100);

}

private void buttonLogout\_Click(object sender, EventArgs e)

{

this.Hide();

FormLogin loginForm = new FormLogin();

loginForm.Show();

}

private void buttonSearch\_Click(object sender, EventArgs e)

{

FormSearch searchForm = new FormSearch();

searchForm.ShowDialog();

}

private void comboBoxTables\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadData(); // Обновление данных при изменении таблицы в ComboBox

}

private void buttonSaveChanges\_Click(object sender, EventArgs e)

{

if (currentUserRole == "admin")

{

bool changesMade = false;

string selectedTable = comboBoxTables.SelectedItem.ToString();

foreach (DataRow row in currentDataTable.Rows)

{

if (row.RowState == DataRowState.Modified)

{

changesMade = true;

string updateQuery = $"UPDATE {selectedTable} SET ";

foreach (DataColumn column in currentDataTable.Columns)

{

if (column.ColumnName != "Идентификатор")

{

updateQuery += $"{column.ColumnName} = '{row[column.ColumnName]}', ";

}

}

updateQuery = updateQuery.TrimEnd(',', ' ');

updateQuery += $" WHERE Идентификатор = {row["Идентификатор"]}";

using (MySqlCommand command = new MySqlCommand(updateQuery, connection))

{

connection.Open();

command.ExecuteNonQuery();

connection.Close();

}

}

}

if (changesMade)

{

MessageBox.Show("Изменения успешно сохранены!");

}

else

{

MessageBox.Show("Нет изменений для сохранения.");

}

LoadData();

}

}

private void buttonUndoChanges\_Click(object sender, EventArgs e)

{

if (currentUserRole == "admin")

{

var result = MessageBox.Show("Вы уверены, что хотите откатить все изменения?", "Подтверждение отката изменений", MessageBoxButtons.YesNo);

if (result == DialogResult.Yes)

{

currentDataTable = originalDataTable.Copy();

dataGridView.DataSource = currentDataTable;

MessageBox.Show("Изменения откатились.");

}

}

}

}

}