**Государственное бюджетное профессиональное образовательное учреждение**

**«Нижегородский Губернский колледж»**

Методическая комиссия «Информатика и вычислительная техника»

Допущен к защите:

преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_ Л.В. Мухина,

\_\_\_\_\_\_\_\_\_\_\_\_ Е.П. Голубева,

«21»\_\_марта\_\_2025 г.

**ОТЧЕТ** **ПО**

**УЧЕБНОЙ ПРАКТИКЕ**

ПМ.01 РАЗРАБОТКА МОДУЛЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОМПЬЮТЕРНЫХ СИСТЕМ

Руководитель: \_\_\_\_\_\_\_\_\_\_ Мухина Л.В., Голубева Е.П. 21.03.2025 г.

Студент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Батракова В.В. 21.03.2025 г.

Специальность, группа: 09.02.07, 43П

Нижний Новгород

2025 г.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc193470298)

[РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 4](#_Toc193470299)

[1.1. Описание предметной области 4](#_Toc193470300)

[1.2. Создание даталогической модели БД 5](#_Toc193470301)

[1.3. Выбор СУБД 6](#_Toc193470302)

[1.4. Создание и заполнение таблиц 9](#_Toc193470303)

[1.5. Паттерн проектирования 11](#_Toc193470304)

[1.6. Функциональные требования информационной системы 12](#_Toc193470305)

[1.7. Нефункциональные требования информационной системы 12](#_Toc193470306)

[1.8. Выбор программного обеспечения для реализации ИС 13](#_Toc193470307)

[1.9. Реализация информационной системы 14](#_Toc193470308)

[1.10. Проектирование 17](#_Toc193470309)

[АНАЛИЗ И РАЗРАБОТКА ПО 20](#_Toc193470310)

[2.1. Описание предметной области 20](#_Toc193470311)

[2.2. Создание даталогической модели БД 20](#_Toc193470312)

[2.3. Создание и заполнение таблиц 21](#_Toc193470313)

[2.4. Реализация информационной системы 25](#_Toc193470314)

[2.5. Тестирование и внедрение информационной системы 27](#_Toc193470315)

[РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ 29](#_Toc193470316)

[3.2. Создание и заполнение базы данных 30](#_Toc193470317)

[3.3. Реализация информационной системы 31](#_Toc193470318)

[ЗАКЛЮЧЕНИЕ 34](#_Toc193470319)

[ПРИЛОЖЕНИЕ А 35](#_Toc193470320)

[ПРИЛОЖЕНИЕ Б 39](#_Toc193470321)

[ПРИЛОЖЕНИЕ В 45](#_Toc193470322)

[ПРИЛОЖЕНИЕ Г 49](#_Toc193470323)

[ПРИЛОЖЕНИЕ Д 55](#_Toc193470324)

[ПРИЛОЖЕНИЕ Е 64](#_Toc193470325)

# ВВЕДЕНИЕ

Учебная практика по ПМ.01 Разработка модулей программного обеспечения для компьютерных систем проходила в ГБПОУ «Нижегородский Губернский колледж» в период с 01.03.2025г. по 21.03.2025г.

Целью учебной практики является овладение общими и профессиональными компетенциями:

ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.

ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием.

ПК 1.3. Выполнять отладку программных модулей с использованием специализированных программных средств.

ПК 1.4. Выполнять тестирование программных модулей.

ПК 1.5. Осуществлять рефакторинг и оптимизацию программного кода.

ПК 1.6. Разрабатывать модули программного обеспечения для мобильных платформ.

Для достижения цели были поставлены следующие задачи:

1. изучить техническое задания и сформировать алгоритма разработки;
2. разработать базу данных в соответствии с техническим заданием;
3. разработать пользовательский интерфейс;
4. реализовать функционал приложения;
5. провести отладку и тестирование готового приложения;
6. провести рефакторинг и оптимизировать код.

Практическим результатом прохождения практики является отчёт по выполнению заданий с представленными в нём рисунками, таблицами и диаграммами.

# РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

## Описание предметной области

Производственная компания «Мастер пол» выпускает напольные покрытия. Компания занимается производством и реализует свою продукцию через партнеров, которые доставляют продукцию компании до конечных потребителей.

В данной области для работы информационной системы «Мастер пол» определены следующие основные составляющие:

Партнеры: тип, наименование компании, юридический адрес, ИНН, ФИО директора, контактные данные (телефон, email), логотип, рейтинг, места продаж, история реализации продукции для последующего расчета скидок. Партнеры покупают продукцию со скидками, которые зависят от общего объема их продаж. Скидки мотивируют партнеров к большему объему продаж. Партнеры реализуют товары в розничных и оптовых магазинах, интернет-магазинах, другим компаниям по продаже товаров и оказанию услуг. Взаиморасчеты производственной компании с партнерами осуществляются в рублях.

Продукция: артикул, тип, наименование, описание, изображение, минимальная стоимость для партнера, размер упаковки (длина, ширина, высота), вес без упаковки, вес с упаковкой, сертификат качества (скан документа), номер стандарта, история изменений минимальной стоимости для партнера, время изготовления, себестоимость, номер цеха, количество человек на производстве, необходимые материалы для производства.

История реализации продукции: компания партнер, продукция, количество поставленной продукции и дата реализации.

Тип материала: наименование и процент брака.

Основные таблицы базы данных перечислены в таблице 1.

Таблица 1 – Список сущностей

|  |  |  |
| --- | --- | --- |
| № | Название | Назначение |
| 1 | Типы материалов | Перечень возможных типов материала |
| 2 | Типы продукции | Перечень возможных типов продукции |
| 3 | Типы партнеров | Перечень возможных типов партнеров |
| 4 | Партнеры | Описание партнеров |
| 5 | Продукция | Описание продукции |
| 6 | Партнеры и продукция | Отражает связь многие-ко-многим между таблицами «Партнеры» и «Продукция» |

## Создание даталогической модели БД

Даталогическая модель отражается графически в виде схемы базы данных, где указываются имена сущностей, их атрибуты и связи между сущностями.

Даталогическая модель для базы данных «Мастер пол» представлена на рисунке 1.

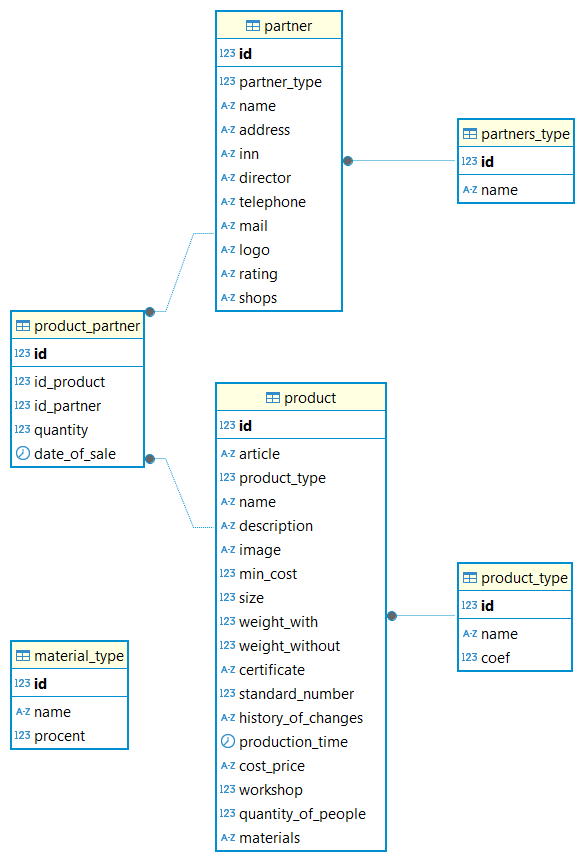


Рисунок 1 – Даталогическая модель

## Выбор СУБД

Система управления базами данных (СУБД) — это совокупность программных и языковых средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

У многих СУБД возможности описания, манипулирования и чтения данных объединены в единых синтаксических рамках - рамках языка SQL, широко используемого как языка баз данных.

Некоторые СУБД располагают такими языками, которые реализуют не только указанные функции, но и обладают структурами и средствами, свойственными традиционным языкам программирования. Благодаря этому они могут использоваться и для обращения к данным базы, и для создания прикладных программ.

Часто пользователю требуется более сложная аналитическая обработка данных базы, не достигаемая через систему меню СУБД. В этом случае приходится разрабатывать прикладные программы. Для их создания СУБД имеют встроенный язык программирования.

Благодаря языковым средствам, обеспечивается доступ пользователей к БД в абстрактных терминах, не связанных со способами хранения данных в компьютере.

Программные средства СУБД обеспечивают работу с физической БД и выполнение всех ее функций.

Функциональные возможности СУБД обширны. СУБД имеют эффективные средства для:

* создания БД, в которой интегрированы данные многих пользователей с целью удовлетворения их информационных потребностей;
* обновления хранящихся в ней данных;
* быстрого извлечения из БД необходимых данных по запросам пользователей;
* выполнения вычислений над данными;
* создания экранных шаблонов - форм, обеспечивающих удобство работы с данными;
* вывода данных из базы в отчетах в виде, удобном для восприятия пользователями;
* разработки приложений;
* экспорта данных в другие БД и импорта данных их других БД;
* публикации данных в сети Интернет.

Эти средства ориентированы на непрофессиональных пользователей. СУБД обеспечивают также управление БД, а именно:

* поддержку целостности БД с помощью механизма транзакций. Транзакция – это совокупность операций с БД, которые должны быть выполнены обязательно до конца, чтобы БД оказалась в непротиворечивом состоянии;
* защиту данных от несанкционированного доступа, от сбоев в работе компьютерной системы;
* восстановление БД в случае ее повреждения.

Благодаря своим развитым функциональным возможностям СУБД используются в качестве мощного инструментального средства для создания и ведения информационной БД автоматизированных информационных систем, позволяющего сокращать сроки их разработки, экономить трудовые, материальные и финансовые ресурсы.

Был проведен анализ следующих СУБД: Microsoft SQL Server и DBeaver для информационной системы «Мастер пол». Согласно критериям выбора, были выделены следующие плюсы и минусы каждой из них.

Microsoft SQL Server – это система управления реляционными базами данных, разработанная компанией Microsoft. Ее активно используют в корпоративном секторе, особенно в крупных компаниях. И это не просто СУБД – это целый комплекс приложений, позволяющий не только хранить и модифицировать данные, но еще и анализировать их, осуществлять безопасность этих данных и многое другое.

Основные преимущества системы управления базами данных Microsoft SQL Server:

* высокая производительность и масштабируемость;
* широкий набор функциональных возможностей для хранения и обработки данных;
* хорошая поддержка инструментов разработки и администрирования;
* надежность и стабильность;
* удобное средство для интеграции с другими продуктами Microsoft и сторонними приложениями.

В свою очередь к недостаткам данной СУБД относят:

* высокая стоимость лицензии;
* ограничения на количество подключений и объем данных в некоторых версиях.
* запросы к базе данных могут быть сложными для оптимизации.

DBeaver — это бесплатная система управления базами данных, позволяющая создавать новые базы, модифицировать данные в существующих, выполнять SQL-запросы.

Преимущества данной СУБД:

* универсальность: в бесплатной версии включена поддержка 80 типов БД;
* доступность: бесплатно можно пользоваться всеми необходимыми для работы с БД функциями — создавать SQL-запросы, визуализировать данные и переносить их, управлять пользовательскими учетными записями. Для установки программы и неограниченного использования достаточно скачать ее с официального сайта разработчиков;
* гибкость: DBeaver поддерживает подключение расширений, позволяя гибко настраивать программу для конкретных задач;
* легкий вес и низкие требования к ресурсам;
* кросс-платформенность;
* поддержка сообщества.

Также можно выделить следующие недостатки:

* ограниченность документации;
* проблемы с производительностью больших баз данных.

Таким образом, DBeaver оптимально подходит для реализации информационной системы «Мастер пол», обеспечивая баланс между стоимостью, удобством эксплуатации и гибкостью конфигурации.

## Создание и заполнение таблиц

Исходя из проведенного анализа в пункте 1.1 необходимо создать 8 таблиц с соответствующими атрибутами и ограничениями. С реализацией запросов на создание и заполнение таблиц в виде программного кода можно ознакомиться в приложении А, а результат их выполнения представлен на рисунках 2 – 8.

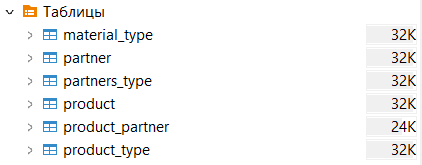


Рисунок 2 – Таблицы БД «Мастер пол»

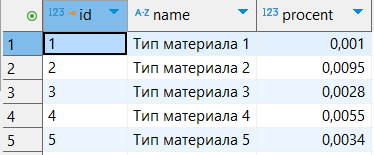


Рисунок 3 – Результат заполнения таблицы «Типы материала»

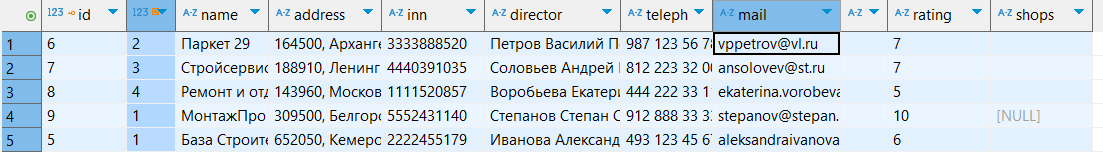


Рисунок 4 – Результат заполнения таблицы «Партнеры»

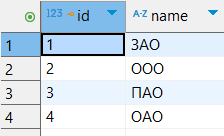


Рисунок 5 – Результат заполнения таблицы «Типы партнеров»

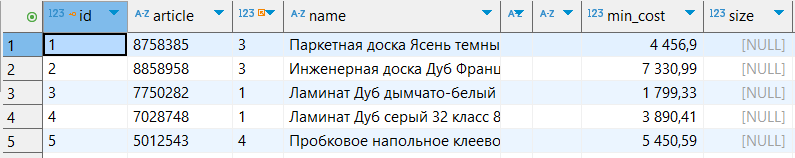


Рисунок 6 – Результат заполнения таблицы «Продукция»

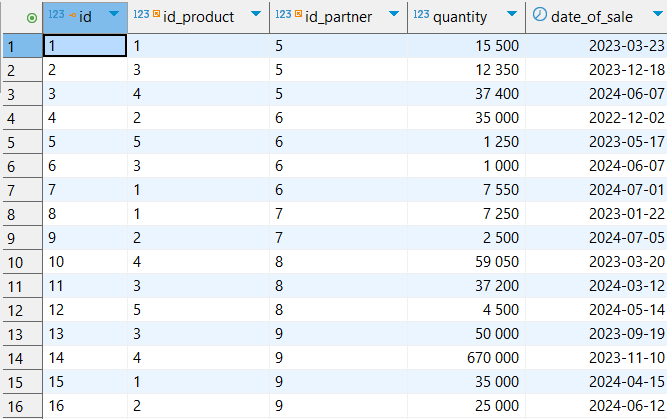


Рисунок 7 – Результат заполнения таблицы «Продукция партнеров»

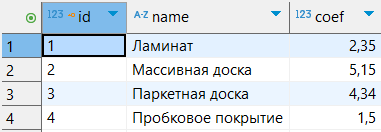


Рисунок 8 – Результат заполнения таблицы «Типы продукции»

## Паттерн проектирования

Для разработки приложения «Мастер пол» был выбран паттерн проектирования Model-View-ViewModel (MVVM). Этот паттерн представляет собой архитектурный шаблон, который используется для отделения логики представления от бизнес-логики и данных, что способствует улучшению тестируемости и упрощению процесса разработки. В рамках прохождения учебной практики использование MVVM обосновано следующими преимуществами:

* Разделение кода на три компонента — модель (Model), представление (View) и модель представления (ViewModel) — позволяет лучше организовать кодовую базу и повысить её поддерживаемость. Каждый компонент отвечает за свою область: View занимается отображением данных, Model хранит данные, а ViewModel связывает эти два слоя вместе и управляет бизнес-логикой.
* MVVM позволяет вносить изменения в один слой без влияния на остальные части приложения. Например, обновление интерфейса (View) не затрагивает бизнес-логику (ViewModel), что делает модификацию проще и быстрее.
* Логика, заложенная в ViewModel, может использоваться повторно в нескольких представлениях, что сокращает дублирование кода и повышает его надежность.
* Модель представления отделяет логику пользовательского интерфейса от бизнес-логики, что делает код более абстрактным и независимым от конкретной платформы или технологии.

Применение паттерна MVVM позволит структурировать кодовую базу приложения «Мастер пол» таким образом, чтобы обеспечить его устойчивость к изменениям, улучшить тестируемость и упростить процесс дальнейшей разработки и поддержки.

## Функциональные требования информационной системы

Согласно полученному техническому заданию, информационная система «Мастер пол» должна содержать следующие программные модули:

* просмотр списка партнеров;
* добавление/редактирование данных о партнере;
* просмотр истории реализации продукции партнером;
* расчет количества материала, требуемого для производства продукции

## Нефункциональные требования информационной системы

Все экранные формы пользовательского интерфейса должны иметь заголовок. Кроме того, на главной форме должен быть установлен логотип (представлен в ресурсах). Логотип не искажать: не менять изображение, пропорции изображения, цвет.

Для приложения должна быть установлена иконка (дана в ресурсах), если это реализуемо в рамках платформы.

Использовать шрифт Segoe UI.

В качестве основного фона используется белый цвет, дополнительного фона используется цвет #F4E8D3. Для акцентирования внимания пользователя на целевом действии интерфейса используется цвет #67BA80.

Необходимо реализовать вывод списка партнеров, информация о которых хранятся в базе данных, согласно макету, представленному на рисунке 9.

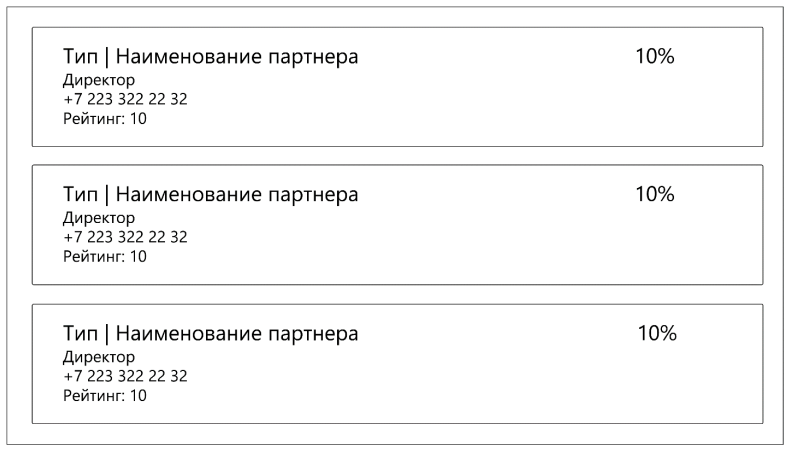


Рисунок 9 – Макет списка

## Выбор программного обеспечения для реализации ИС

Для реализации информационной системы "Мастер пол" были выбраны Visual Studio 2022 в качестве среды разработки и Avalonia в качестве фреймворка для построения графического интерфейса. Эти инструменты обеспечивают высокую степень гибкости, производительности и совместимости, что критически важно для успешного завершения проекта.

Visual Studio предлагает широкий спектр инструментов для разработки, начиная от редактора кода и заканчивая системой контроля версий и средствами профилирования. Что позволяет эффективно управлять всеми аспектами проекта, включая отладку, тестирование и развёртывание.

Visual Studio поддерживает множество языков программирования, включая C#, VB.NET, F# и другие. Это даёт возможность выбирать оптимальный язык для конкретной задачи, что существенно ускоряет процесс разработки.

Avalonia была выбрана в качестве фреймворка для построения графического интерфейса, так как имеет следующие преимущества:

* позволяет создавать приложения, работающие на различных операционных системах, включая Windows, macOS и Linux. Это существенно расширяет потенциальную аудиторию информационной системы "Мастер пол" и делает её доступной для большего числа пользователей.
* предлагает богатый набор элементов интерфейса и инструментов для стилизации, что позволяет создать современный и привлекательный внешний вид приложения. Это важно для привлечения внимания пользователей и повышения их удовлетворённости.
* построена на основе XAML и имеет минималистичный API, что делает её лёгкой в освоении и эффективной в использовании. Это снижает нагрузку на ресурсы компьютера и ускоряет процесс разработки.

Выбор Visual Studio 2022 и Avalonia для реализации информационной системы "Мастер пол" обусловлен необходимостью создания современного, производительного и кросс-платформенного приложения. Эти инструменты предоставляют всё необходимое для успешной разработки, тестирования и развёртывания системы, обеспечивая высокое качество конечного продукта и удовлетворение потребностей пользователей.

## Реализация информационной системы

Согласно проведенному анализу технического задания и возможностей программного обеспечения, было разработано десктопное приложение «Мастер пол», а именно разработан вывод списка партнеров, хранящихся в БД, добавление и редактирование партнера в БД и история реализации продукции партнером, представленные на рисунках   
10-13 соответственно.

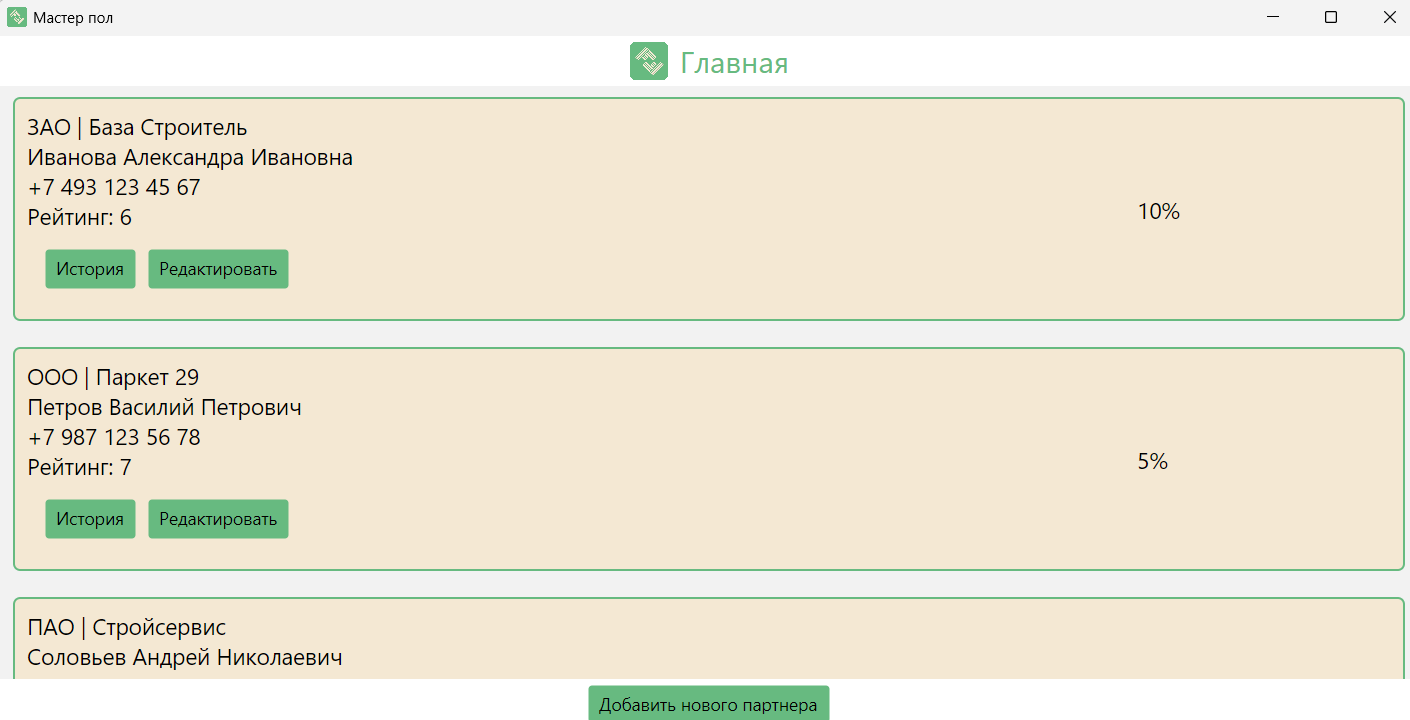


Рисунок 10 – Список партнеров

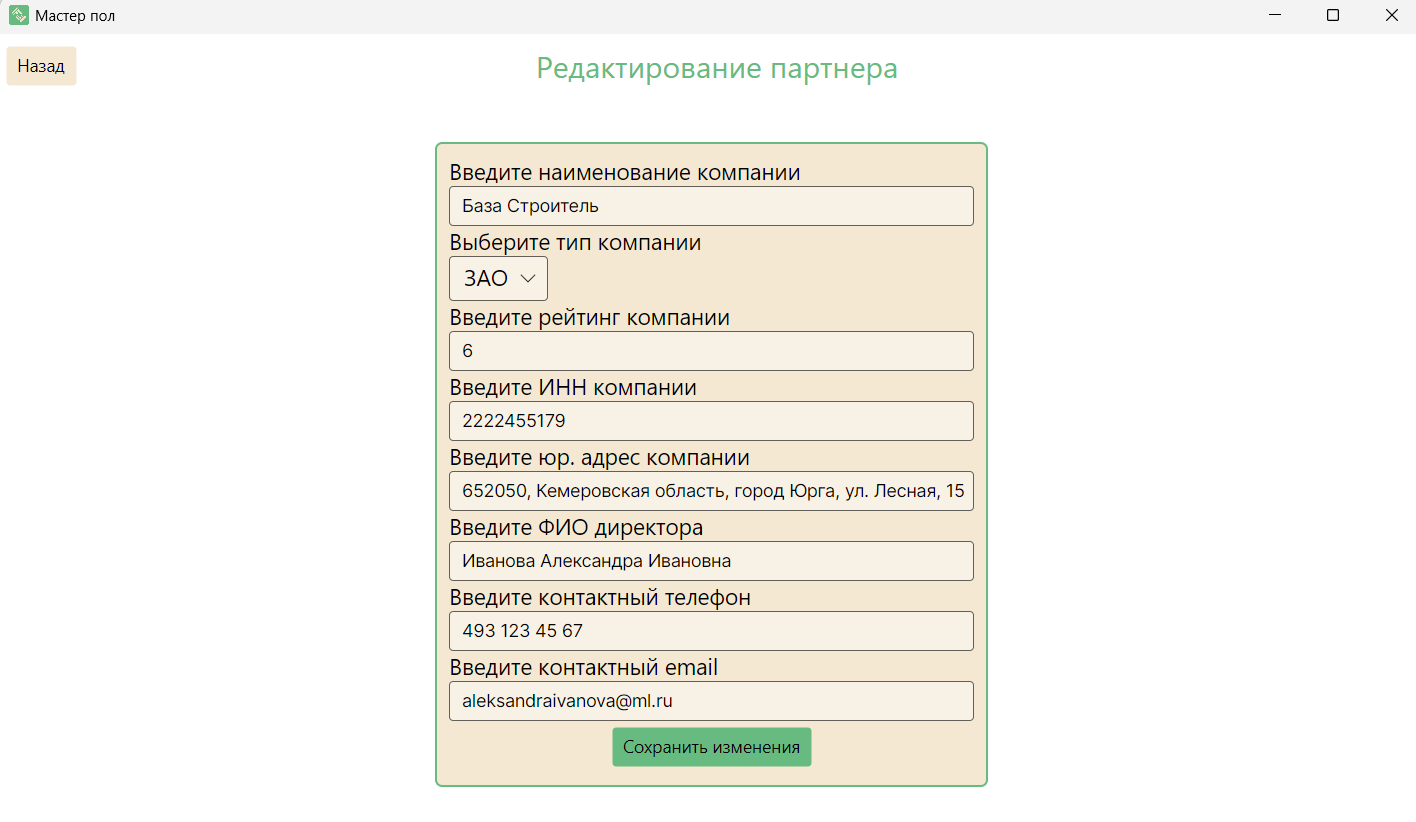


Рисунок 11 – Редактирование партнера

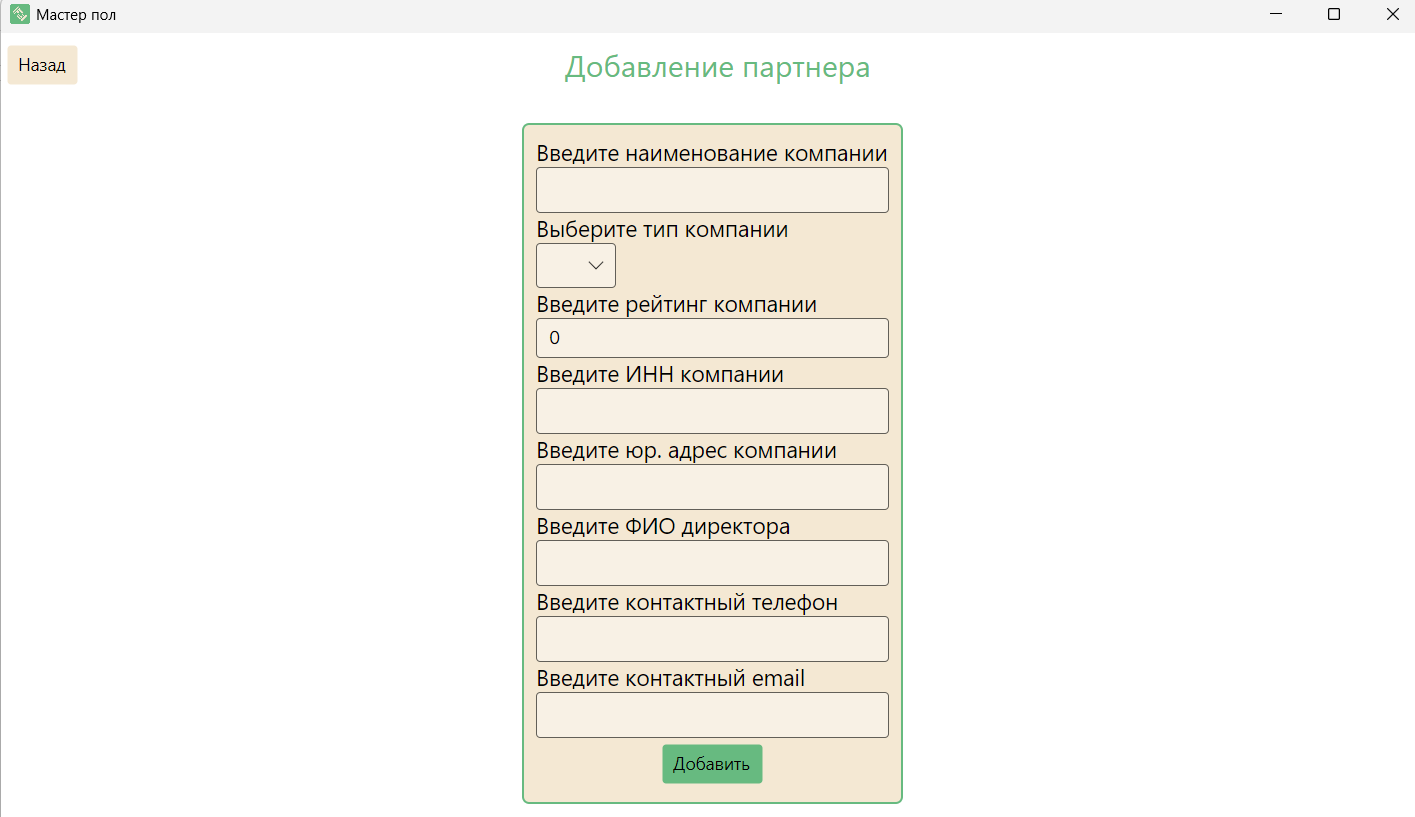


Рисунок 12 – Добавление партнера

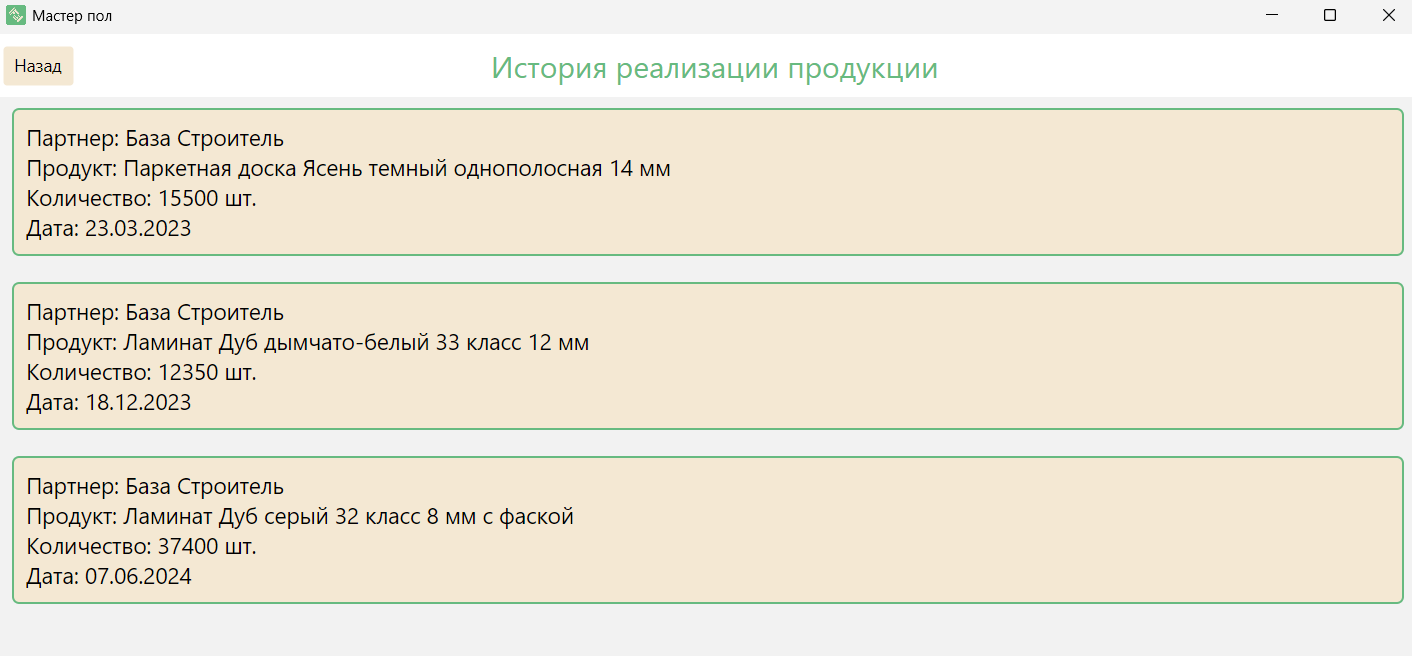


Рисунок 13 – История реализации продукции

Также целью обеспечить одинаковый расчет количества материала, требуемого для производства продукции, был разработан метод в динамической библиотеке, программный код метода представлен в приложении Б.

Метод должен принимать идентификатор типа продукции, идентификатор типа материала, количество получаемой продукции – целые числа, параметры продукции (два параметра) – вещественные, положительные числа, а возвращать целое число – количество необходимого материала с учетом возможного брака материала. Метод должен рассчитывать целое количество материала, необходимого для производства указанного количества продукции, учитывая возможный брак материала. Количество необходимого материала на одну единицу продукции рассчитывается как произведение параметров продукции, умноженное на коэффициент типа продукции. Кроме того, нужно учитывать процент брака материала в зависимости от его типа: с учетом возможного брака материала необходимое количество материала должно быть увеличено. Коэффициент типа продукции и процент брака – вещественные числа. Если в качестве параметров метода будут указаны несуществующие типы продукции, материалов или другие неподходящие данные, то метод должен вернуть -1.

Для обеспечения соответствия заявленным требованиям были разработаны 5 тест-кейсов для модуля редактирования партнера и 10 юнит-тестов низкого уровня сложности и 5 юнит-тестов высокого уровня сложности для метода реализованного в динамической библиотеке, представленные в приложении В и приложении Б программный код 2 соответственно.

## Проектирование

Программа для фитнес-центра по распределению фитнес – расписания и контроля его соблюдения

Предполагается, что в системе фитнес центра будет 3 роли пользователей: клиенты, тренеры, администраторы. Авторизация в системе производится по телефону и паролю.

Клиенты могут зарегистрироваться в системе, указав ФИО, телефон, пароль, дату рождения, фото профиля, пол.

Администраторы – пользователи с уже заполненным профилем. Они могут добавлять новых тренеров и записывать их на различные курсы обучения с целью поддержки и улучшения их профессиональной квалификации. Постоянным клиентам администраторы могут предоставлять скидки на тренировки.

Любой клиент после авторизации может выбрать себе тренера (если у него нет такового). В этом случае клиент видит список тренеров с именем, фото, полом, стажем работы и списком достижений. Клиент может отправить заявку любому из тренеров, написав при этом цель, которую он хочет достигнуть при тренировках.

Тренер после авторизации видит новые заявки от клиентов и их количество (если таковые имеются). Тренер может принять заявку или отклонить. В случае отказа, тренер должен указать причину. В случае подтверждения заявки тренер должен выставить план индивидуальных занятий для клиента. Выбрав из списка клиентов без плана тренировок, тренер видит цель клиента, его возраст и планирует даты тренировочного цикла. Для индивидуальных занятий тренер может выбрать упражнения, указывая при этом его вид (приседания, отжимания и т.д.), частоту выполнения (сколько раз в неделю), число подходов и число повторений в каждом подходе.

Клиент, отправивший заявку, но не получивший ответа, видит список своих заявок с результатами (в том числе с указанием причины при отказе) и количеством дней ожидания ответа. Получив план тренировок, клиент видит экран с 2 вкладками: план тренировок (дата-список упражнений через запятую) и сегодняшний перечень индивидуальных занятий. Для последней выводится список: вид упражнения, количество повторов и Checkbox, позволяющий отметить выполнения, упражнения. Несмотря на это, упражнение не будет засчитано системой до тех пор, пока клиент не укажет показатель своего пульса во время выполнения упражнения. Сверху выводится сегодняшний прогресс (по количеству выполненных упражнений) в процентах с графическим отображением.

Тренер также может посмотреть список своих текущих клиентов с указанием у каждого: проценты выполнения всего цикла тренировок (зависит от длительности цикла) и процента выполненных упражнений (т.к. некоторые упражнения могут быть пропущены). По каждому клиенту выводится средний показатель пульса во время выполнения упражнений.

В соответствии с данной предметной областью был проведен анализ и были реализованы диаграммы вариантов использования и деятельности, представленные на рисунках 14 и 15 соответственно.

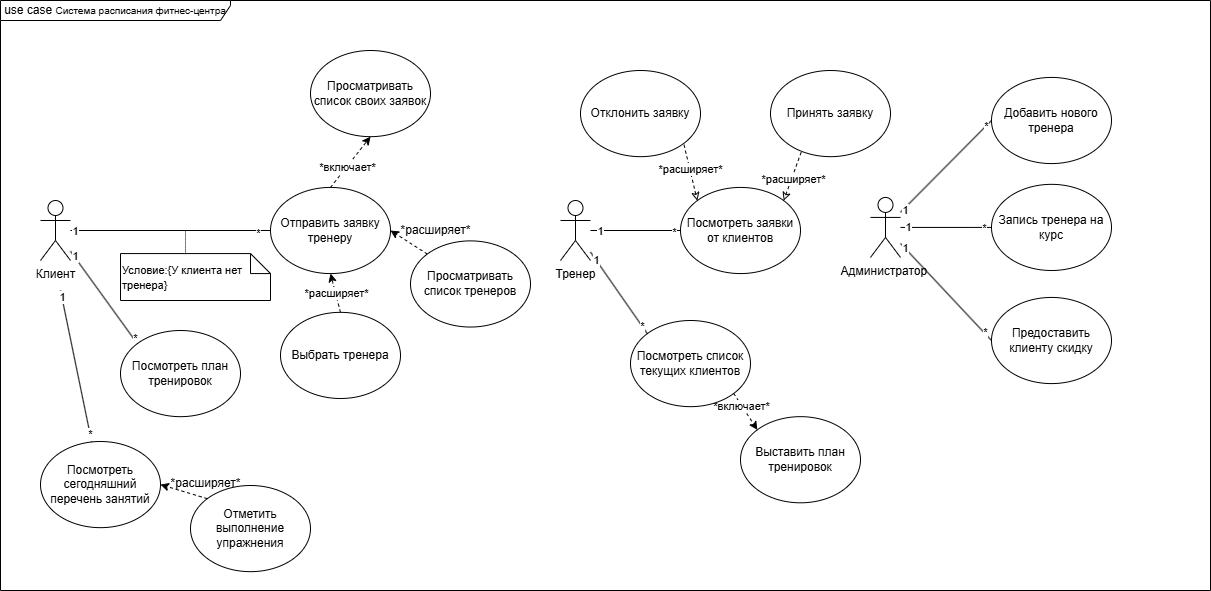


Рисунок 14 – Диаграмма вариантов использования

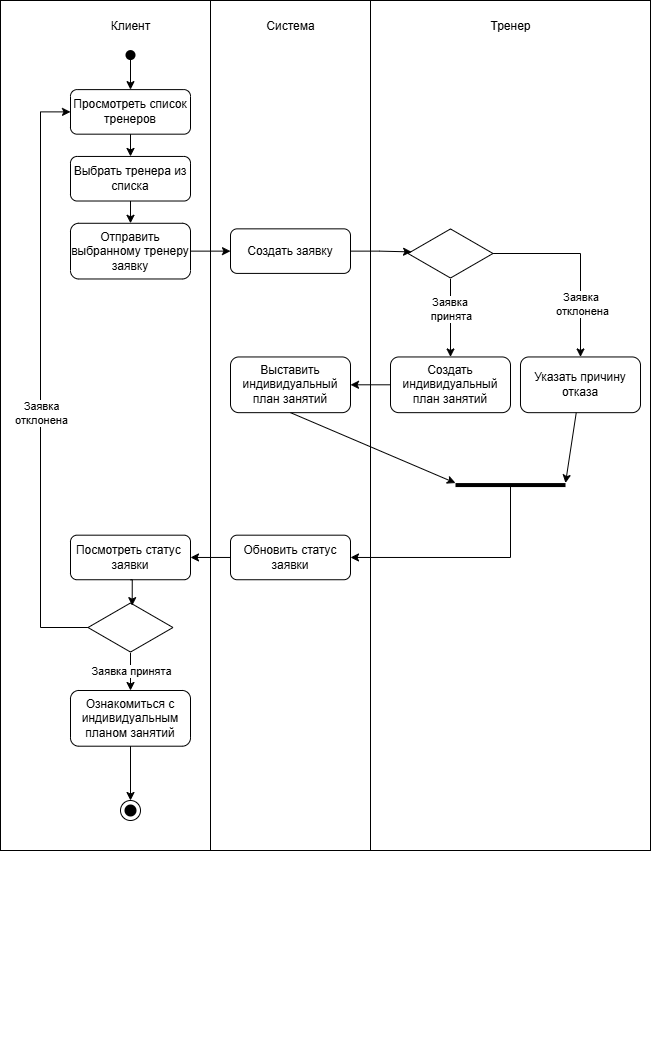


Рисунок 15 – Диаграмма деятельности

# АНАЛИЗ И РАЗРАБОТКА ПО

## Описание предметной области

ООО «Пиши-стирай» - магазин по продаже канцелярских товаров.

В соответствии с предложенным заданием определены следующие функциональные модули ИС «Пиши-стирай»:

* неавторизованный клиент и авторизованный клиент может просматривать товары и формировать заказ;
* менеджер может просматривать товары, формировать и редактировать заказы;
* администратор может добавлять/редактировать/удалять товары, просматривать и редактировать заказы.

Для того, чтобы реализовать данный функционал была спроектирована БД со следующими сущностями: Пользователи, Роли пользователей, Продукты в заказе, Заказы, Продукты, Статус заказа, Вид продукта, Пункты выдачи, Производитель продукции, Поставщик продукции и Категория продукции

## Создание даталогической модели БД

Даталогическая модель отражается графически в виде схемы базы данных, где указываются имена сущностей, их атрибуты и связи между сущностями.

Даталогическая модель для базы данных «Пиши-стирай» представлена на рисунке 16.

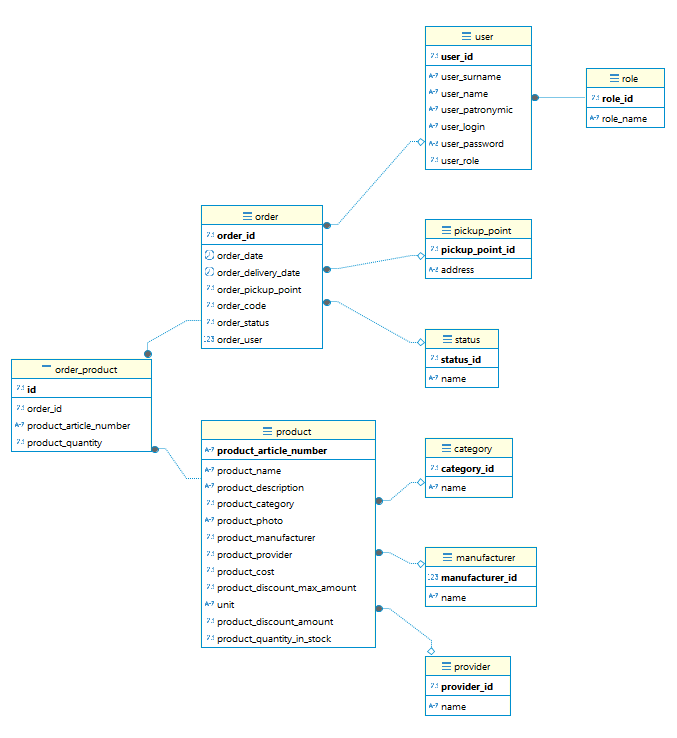


Рисунок 16 – Даталогическая модель

## Создание и заполнение таблиц

Исходя из проведенного анализа в пункте 2.1 необходимо создать 10 таблиц с соответствующими атрибутами и ограничениями. С реализацией запросов на создание и заполнение таблиц в виде программного кода можно ознакомиться в приложении А, а результат их выполнения представлен на рисунках 17 – 26.

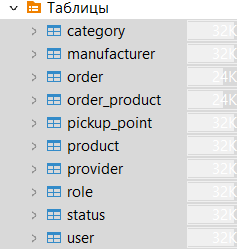


Рисунок 17 – Таблицы БД «Пиши-стирай»

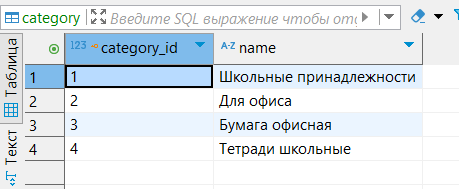


Рисунок 18 – Результат заполнения таблицы Категории

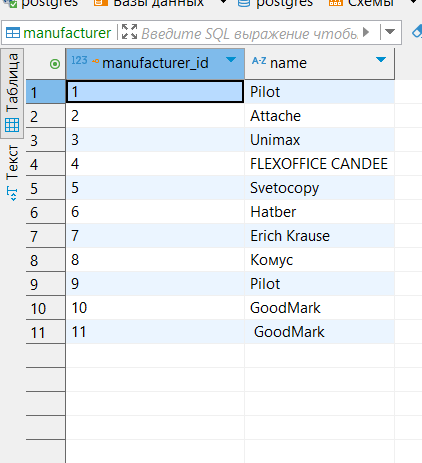


Рисунок 19 – Результат заполнения таблицы Производители

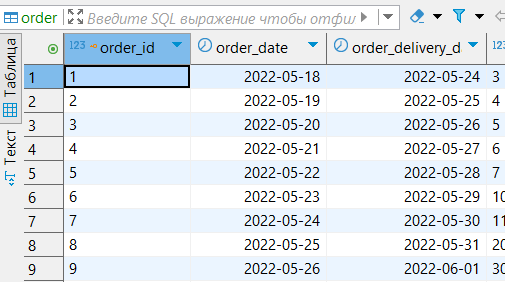


Рисунок 20 – Результат заполнения таблицы Заказы

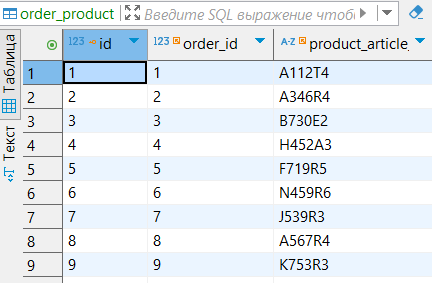


Рисунок 21 – Результат заполнения таблицы Состав заказа

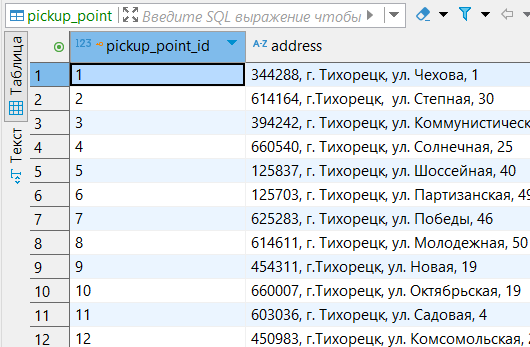


Рисунок 22 – Результат заполнения таблицы Пункты выдачи

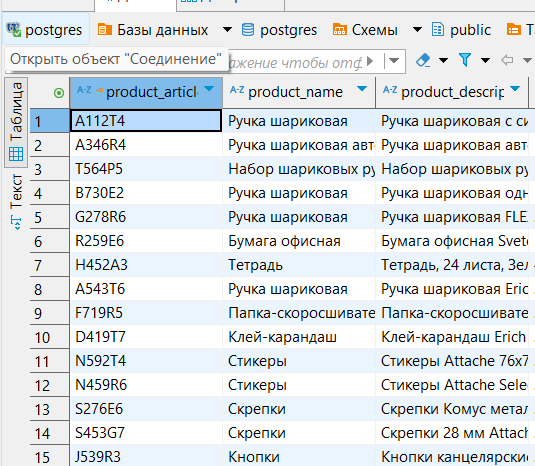


Рисунок 23 – Результат заполнения таблицы Продукция

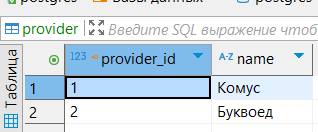


Рисунок 24 - Результат заполнения таблицы Поставщики

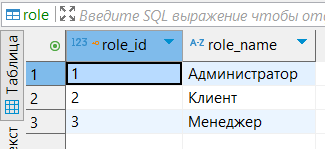
****

Рисунок 25 - Результат заполнения таблицы Роли

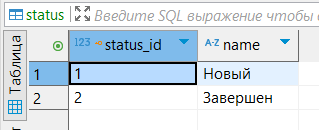


Рисунок 25 - Результат заполнения таблицы Статус заказа

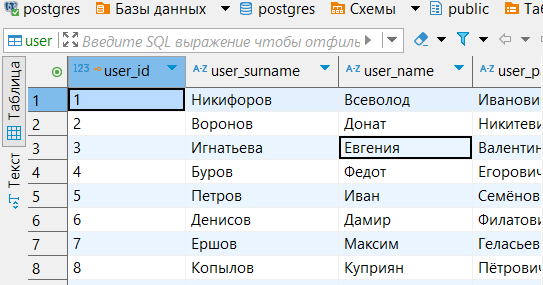


Рисунок 26 - Результат заполнения таблицы Пользователи

## Реализация информационной системы

Согласно проведенному анализу технического задания и возможностей программного обеспечения, было разработано десктопное приложение «Пиши стирай», а именно разработан вывод списка продукции, хранящихся в БД, просмотр и редактирование заказа в БД и авторизация пользователя, представленные на рисунках   
27-29 соответственно.

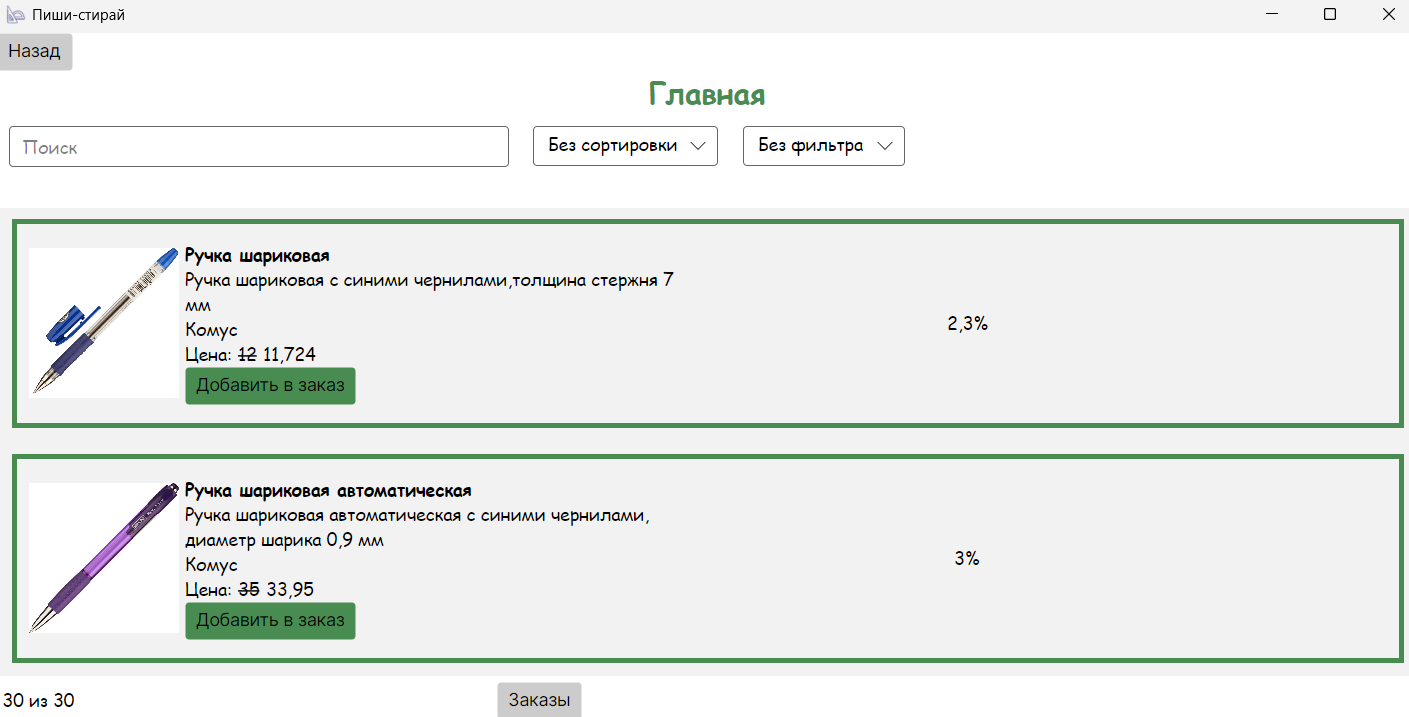


Рисунок 27 – Список продукции

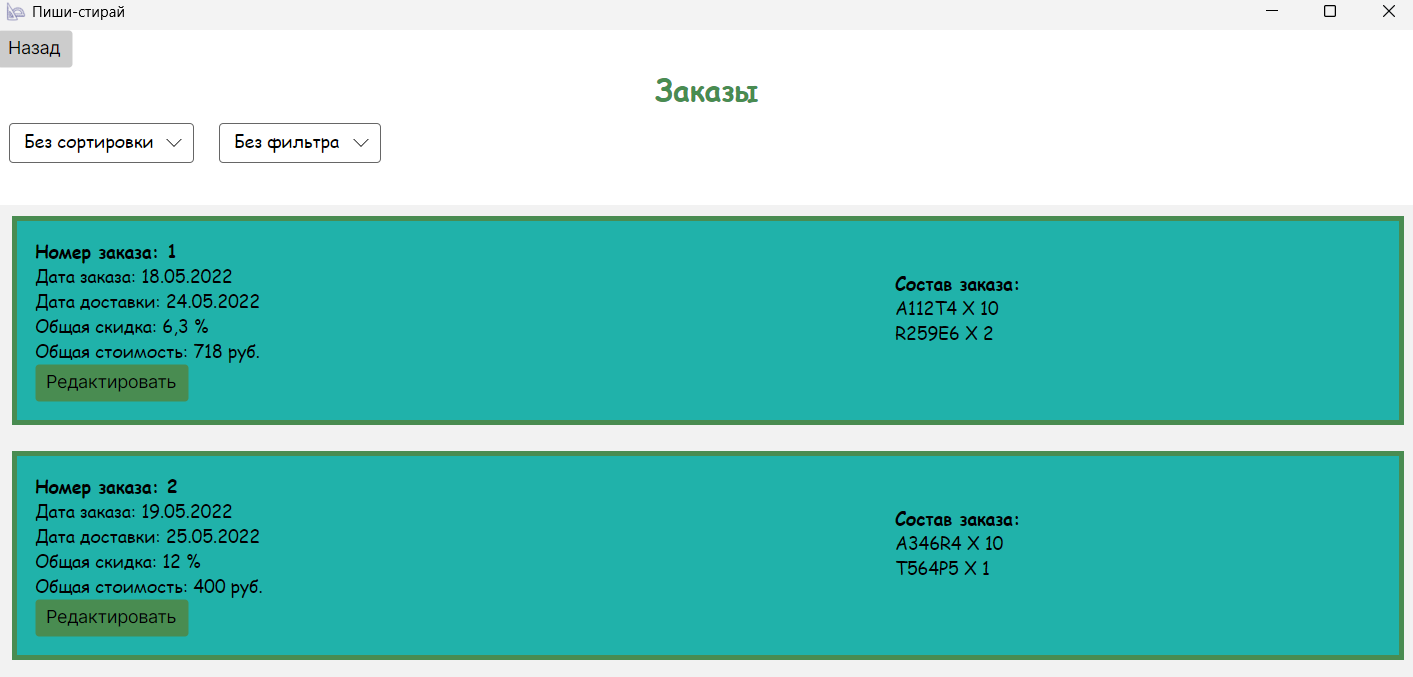


Рисунок 28 – Список заказов

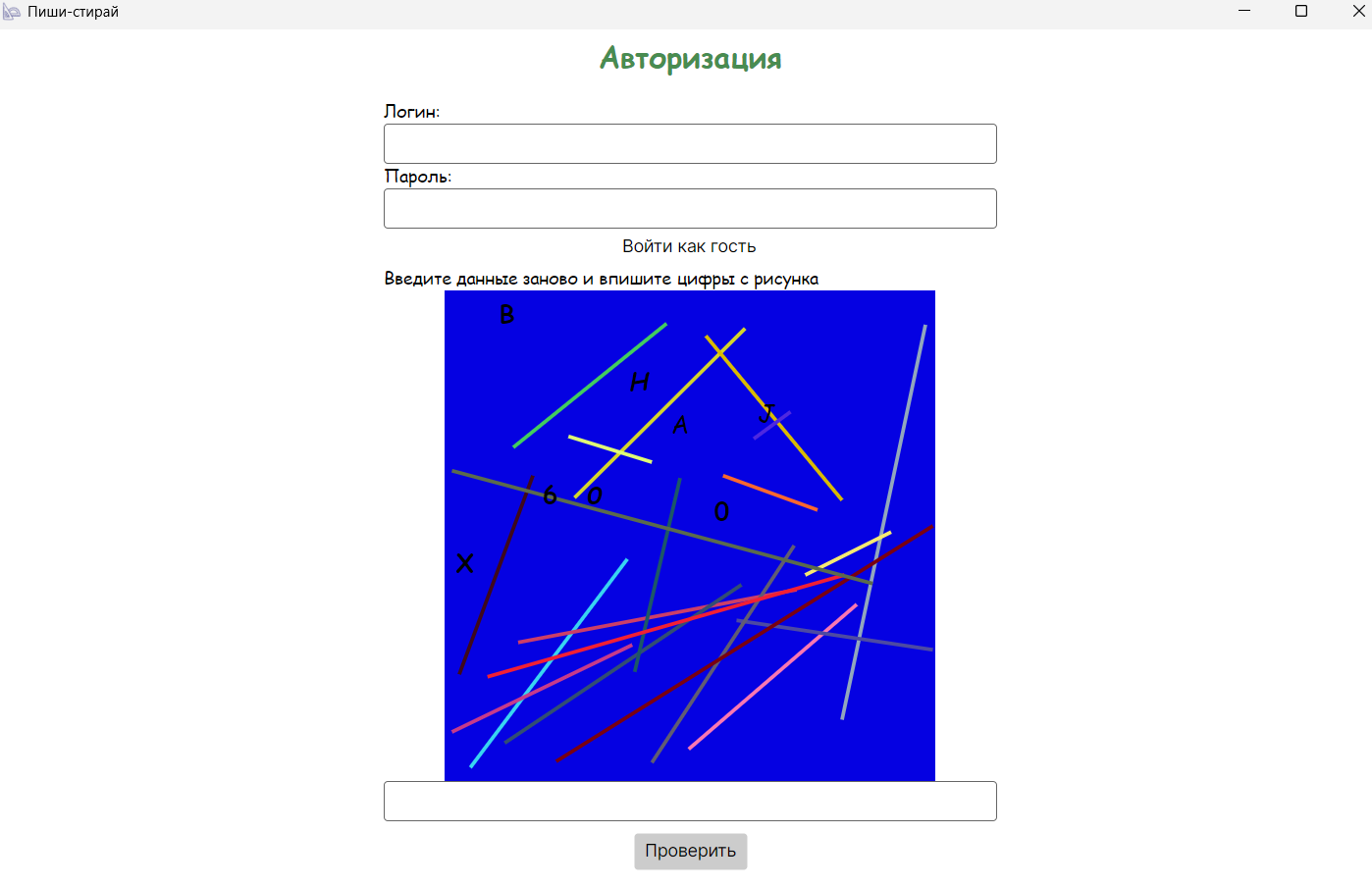


Рисунок 29 – Авторизация пользователя

Для обеспечения соответствия заявленным требованиям были разработаны 5 тест-кейсов для модуля редактирования партнера и 10 юнит-тестов низкого уровня сложности и 5 юнит-тестов высокого уровня сложности для метода реализованного в динамической библиотеке, представленные в приложении В и приложении Б программный код 2 соответственно.

## Тестирование и внедрение информационной системы

Необходимо разработайте библиотек, которая позволит вернуть список свободных временных интервалов в графике сотрудника для формирования оптимального графика работы сотрудников.

Вход: список занятых промежутков времени (в двух массивах: startTimes - начало, durations длительность), минимальное необходимое время для работы менеджера (consultationTime), рабочий день сотрудника (начало - beginWorkingTime и завершение endWorkingTime)

Выход: список подходящих свободных временных промежутков (в массив строк формата HH:mm-HH:mm)

Программный код реализованного метода представлен в приложении Д Программный код Д.1.

Для обеспечения соответствия заявленным требованиям были разработаны 10 юнит-тестов для метода реализованного в динамической библиотеке и 5 тест-кейсов для модуля добавления продукции, представленные в приложении Д программный код Д.2 и Таблицы Д.1 – Д.6 соответственно.

# РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ

* 1. **Выбор программного обеспечения для реализации продукта**

Для разработки мобильного приложения было выбрано Android Studio — это официальная IDE для разработки приложений на платформе Android, созданная компанией Google. Она обладает мощными инструментами для разработки, отладки и тестирования, а также гарантирует высокую производительность и удобство использования. Android Studio поддерживает различные языки программирования, включая Java и Kotlin, и имеет глубокую интеграцию с SDK и другими инструментами. Это делает её идеальным выбором для разработки мобильных приложений на Android.

Выбор сервиса Supabase для создания и хранения базы данных мобильного приложения обоснован следующими преимуществами этого сервиса:

* Простота использования: Supabase предлагает простой и интуитивно понятный интерфейс, что облегчает работу с базой данных.
* Интеграция с другими сервисами: Supabase легко интегрируется с популярными инструментами и языками программирования, такими как JavaScript, Python и PHP.
* Встроенная панель управления: Supabase поставляется с панелью управления, которая позволяет управлять приложениями и базами данных.
* Поддержка популярных баз данных: Supabase использует базу данных PostgreSQL, что обеспечивает надёжность и масштабируемость.

Для создания макета мобильного приложения был выбран сервис Figma. Figma — онлайн-редактор, в котором удобно проектировать интерфейсы, создавать макеты сайтов, мобильных приложений, презентации, иллюстрации, логотипы и анимацию. Figma предлагает ряд преимуществ, которые делают его отличным выбором для дизайнеров мобильных приложений. Вот несколько основных причин, почему многие профессионалы выбирают Figma:

* Интуитивный интерфейс и простота использования. Figma обладает интуитивно понятным интерфейсом, что делает его легким в освоении даже для новичков. В нем можно легко создавать и редактировать элементы дизайна, использовать шаблоны и библиотеки, добавлять взаимодействия и анимации, а также экспортировать проекты в различные форматы. Все это делает процесс разработки мобильного приложения более эффективным и продуктивным.
* Плагины — дополнительные приложения внутри редактора, которые расширяют его возможности. Например, плагин Unsplash позволяет добавлять изображения на макет.
* Режим демонстрации работы дизайна приложения для лучшего понимания поведения элементов на странице.
* Высокая производительность и оптимизация работы даже при разработке крупных проектов.

Для интеграции нейросети в мобильное приложение «Pensilmuse» была выбрана нейросеть Gigachad. Нейросеть Gigachad — это уникальная разработка Сбера, которая позволяет решать множество задач в разных сферах жизни и деятельности. Вот некоторые преимущества использования этой нейросети:

* Мультимодальность: нейросеть может работать с разными типами данных: текстом, звуком, изображением, видео и т. д.
* Генерация данных: нейросеть способна генерировать новые данные по запросу пользователя.
* Работа с разными языками: нейросеть может общаться на русском языке.
* Создание текстов и картинок: нейросеть может создавать тексты и изображения по запросам пользователей.

## Создание и заполнение базы данных

Исходя из предоставленного в приложении Е скрипта базы данных была восстановлена база данных, представленная на рисунке 30.

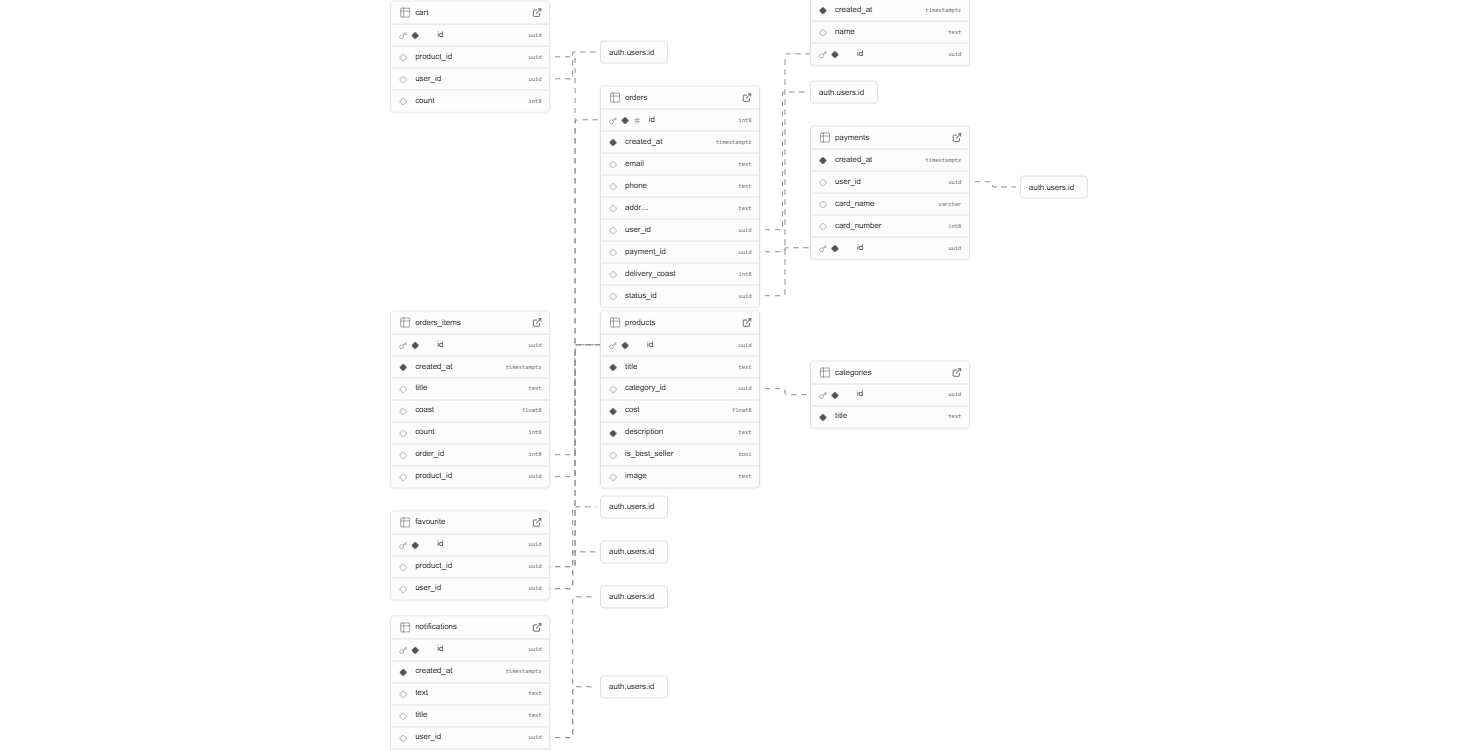


Рисунок 30 – Восстановленная база данных

## Реализация информационной системы

Согласно проведенному анализу технического задания и возможностей программного обеспечения, было разработано мобильное приложение, а именно разработан вывод списка продукции, хранящихся в БД, регистрация и авторизация пользователя, представленные на рисунках   
31-33 соответственно.

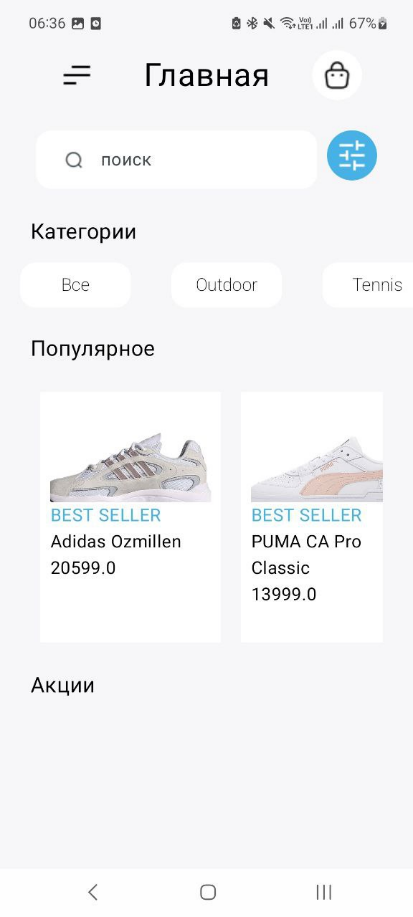


Рисунок 32 – Главная страница приложения

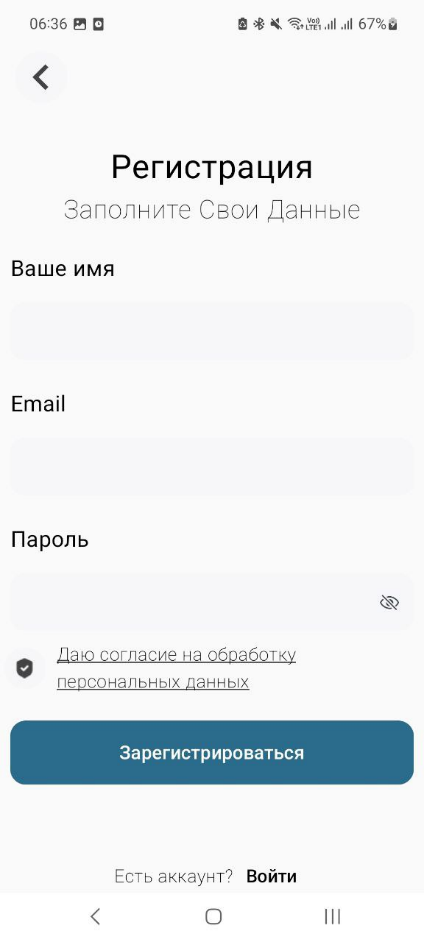


Рисунок 32 – Страница регистрации

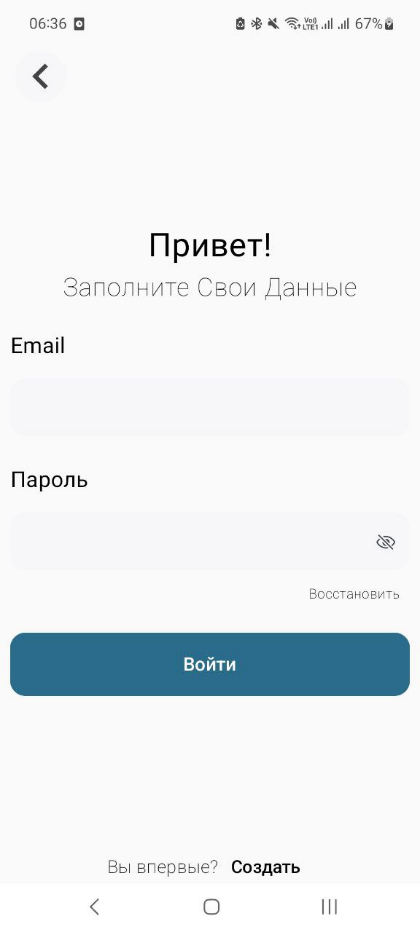


Рисунок 33 – Страница авторизации

# ЗАКЛЮЧЕНИЕ

В результате прохождения учебной практики по ПМ.01 «Разработка модулей программного обеспечения для компьютерных систем» в ГБПОУ «Нижегородский Губернский колледж» были успешно выполнены все поставленные задачи. В результате практики были закреплены необходимые профессиональные компетенции, такие как формирование алгоритмов разработки программных модулей, их разработка и отладка, тестирование, рефакторинг и оптимизация программного кода, а также разработка модулей для мобильных платформ.

Практическая работа включала в себя изучение технического задания, разработку базы данных, создание пользовательского интерфейса, реализацию функционала приложения, проведение отладки и тестирования, а также рефакторинг и оптимизацию кода. Все эти этапы были выполнены в соответствии с требованиями технического задания.

Практическим результатом прохождения практики стал отчёт, содержащий подробные описания выполненных заданий, а также иллюстрации в виде рисунков, таблиц и диаграмм, которые наглядно демонстрируют процесс разработки и результаты работы.

# ПРИЛОЖЕНИЕ А

Программный код А.1 – Создание таблиц

-- public.material\_type определение

-- Drop table

-- DROP TABLE public.material\_type;

**CREATE** **TABLE** public.material\_type (

id serial4 **NOT** **NULL**,

"name" **varchar** **NOT** **NULL**,

procent **float8** **NOT** **NULL**,

**CONSTRAINT** material\_type\_pk **PRIMARY** **KEY** (id)

);

-- public.partners\_type определение

-- Drop table

-- DROP TABLE public.partners\_type;

**CREATE** **TABLE** public.partners\_type (

id serial4 **NOT** **NULL**,

"name" **varchar** **NOT** **NULL**,

**CONSTRAINT** partners\_type\_pk **PRIMARY** **KEY** (id)

);

-- public.partner определение

-- Drop table

-- DROP TABLE public.partner;

**CREATE** **TABLE** public.partner (

id serial4 **NOT** **NULL**,

partner\_type **int4** **NOT** **NULL**,

"name" **varchar** **NOT** **NULL**,

address **varchar** **NOT** **NULL**,

inn **varchar** **NOT** **NULL**,

director **varchar** **NOT** **NULL**,

telephone **varchar** **NOT** **NULL**,

mail **varchar** **NOT** **NULL**,

logo **varchar** **NULL**,

rating **int4** **NOT** **NULL**,

shops **varchar** **NULL**,

**CONSTRAINT** partner\_pk **PRIMARY** **KEY** (id)

);

-- public.partner внешние включи

**ALTER** **TABLE** public.partner **ADD** **CONSTRAINT** partner\_type\_fk **FOREIGN** **KEY** (partner\_type) **REFERENCES** public.partners\_type(id) **ON** **DELETE** **CASCADE**;

-- public.product\_type определение

-- Drop table

-- DROP TABLE public.product\_type;

**CREATE** **TABLE** public.product\_type (

id serial4 **NOT** **NULL**,

"name" **varchar** **NOT** **NULL**,

coef **float8** **NOT** **NULL**,

**CONSTRAINT** product\_type\_pk **PRIMARY** **KEY** (id)

);

-- public.product определение

-- Drop table

-- DROP TABLE public.product;

**CREATE** **TABLE** public.product (

id serial4 **NOT** **NULL**,

article **varchar** **NOT** **NULL**,

product\_type **int4** **NOT** **NULL**,

"name" **varchar** **NOT** **NULL**,

description **varchar** **NULL**,

image **varchar** **NULL**,

min\_cost **float8** **NOT** **NULL**,

"size" **float8** **NULL**,

weight\_with **float8** **NULL**,

weight\_without **float8** **NULL**,

certificate **varchar** **NULL**,

standard\_number **int4** **NULL**,

history\_of\_changes **varchar** **NULL**,

production\_time **date** **NULL**,

cost\_price **varchar** **NULL**,

workshop **int4** **NULL**,

quantity\_of\_people **int4** **NULL**,

materials **varchar** **NULL**,

**CONSTRAINT** products\_pk **PRIMARY** **KEY** (id)

);

-- public.product внешние включи

**ALTER** **TABLE** public.product **ADD** **CONSTRAINT** product\_type\_fk **FOREIGN** **KEY** (product\_type) **REFERENCES** public.product\_type(id) **ON** **DELETE** **CASCADE**;

-- public.product\_partner определение

-- Drop table

-- DROP TABLE public.product\_partner;

**CREATE** **TABLE** public.product\_partner (

id serial4 **NOT** **NULL**,

id\_product **int4** **NOT** **NULL**,

id\_partner **int4** **NOT** **NULL**,

quantity **int4** **NOT** **NULL**,

date\_of\_sale **date** **NOT** **NULL**,

**CONSTRAINT** product\_partner\_pk **PRIMARY** **KEY** (id)

);

-- public.product\_partner внешние включи

**ALTER** **TABLE** public.product\_partner **ADD** **CONSTRAINT** partner\_fk **FOREIGN** **KEY** (id\_partner) **REFERENCES** public.partner(id) **ON** **DELETE** **CASCADE**;

**ALTER** **TABLE** public.product\_partner **ADD** **CONSTRAINT** product\_fk **FOREIGN** **KEY** (id\_product) **REFERENCES** public.product(id) **ON** **DELETE** **CASCADE**;

Программный код А.2 – Заполнение таблиц

**INSERT** **INTO** public.material\_type ("name",procent) **VALUES**

('Тип материала 1',0.001),

('Тип материала 2',0.0095),

('Тип материала 3',0.0028),

('Тип материала 4',0.0055),

('Тип материала 5',0.0034);

**INSERT** **INTO** public.partner (partner\_type,"name",address,inn,director,telephone,mail,logo,rating,shops) **VALUES**

(2,'Паркет 29','164500, Архангельская область, город Северодвинск, ул. Строителей, 18','3333888520','Петров Василий Петрович','987 123 56 78','vppetrov@vl.ru','','7',''),

(3,'Стройсервис','188910, Ленинградская область, город Приморск, ул. Парковая, 21','4440391035','Соловьев Андрей Николаевич','812 223 32 00','ansolovev@st.ru','','7',''),

(4,'Ремонт и отделка','143960, Московская область, город Реутов, ул. Свободы, 51','1111520857','Воробьева Екатерина Валерьевна','444 222 33 11','ekaterina.vorobeva@ml.ru','','5',''),

(1,'МонтажПро','309500, Белгородская область, город Старый Оскол, ул. Рабочая, 122','5552431140','Степанов Степан Сергеевич','912 888 33 33','stepanov@stepan.ru','','10',**NULL**),

(2,'рррррр','ррррррррррррррррр','ррррррррррррррр','рррррррррррррррррр','ррррррррррррррррр','рррррррррррррррр',**NULL**,'ррррррррррр',**NULL**),

(1,'База Строитель','652050, Кемеровская область, город Юрга, ул. Лесная, 15','2222455179','Иванова Александра Ивановна','493 123 45 67','aleksandraivanova@ml.ru','','6',''),

(1,'1','3','2','4','5','5',**NULL**,'1',**NULL**);

**INSERT** **INTO** public.partners\_type ("name") **VALUES**

('ЗАО'),

('ООО'),

('ПАО'),

('ОАО');

**INSERT** **INTO** public.product (article,product\_type,"name",description,image,min\_cost,"size",weight\_with,weight\_without,certificate,standard\_number,history\_of\_changes,production\_time,cost\_price,workshop,quantity\_of\_people,materials) **VALUES**

('8758385',3,'Паркетная доска Ясень темный однополосная 14 мм','','',4456.9,**NULL**,**NULL**,**NULL**,'',**NULL**,'',**NULL**,'',**NULL**,**NULL**,''),

('8858958',3,'Инженерная доска Дуб Французская елка однополосная 12 мм','','',7330.99,**NULL**,**NULL**,**NULL**,'',**NULL**,'',**NULL**,'',**NULL**,**NULL**,''),

('7750282',1,'Ламинат Дуб дымчато-белый 33 класс 12 мм','','',1799.33,**NULL**,**NULL**,**NULL**,'',**NULL**,'',**NULL**,'',**NULL**,**NULL**,''),

('7028748',1,'Ламинат Дуб серый 32 класс 8 мм с фаской','','',3890.41,**NULL**,**NULL**,**NULL**,'',**NULL**,'',**NULL**,'',**NULL**,**NULL**,''),

('5012543',4,'Пробковое напольное клеевое покрытие 32 класс 4 мм','','',5450.59,**NULL**,**NULL**,**NULL**,'',**NULL**,'',**NULL**,'',**NULL**,**NULL**,'');

**INSERT** **INTO** public.product\_partner (id\_product,id\_partner,quantity,date\_of\_sale) **VALUES**

(1,5,15500,'2023-03-23'),

(3,5,12350,'2023-12-18'),

(4,5,37400,'2024-06-07'),

(2,6,35000,'2022-12-02'),

(5,6,1250,'2023-05-17'),

(3,6,1000,'2024-06-07'),

(1,6,7550,'2024-07-01'),

(1,7,7250,'2023-01-22'),

(2,7,2500,'2024-07-05'),

(4,8,59050,'2023-03-20');

**INSERT** **INTO** public.product\_partner (id\_product,id\_partner,quantity,date\_of\_sale) **VALUES**

(3,8,37200,'2024-03-12'),

(5,8,4500,'2024-05-14'),

(3,9,50000,'2023-09-19'),

(4,9,670000,'2023-11-10'),

(1,9,35000,'2024-04-15'),

(2,9,25000,'2024-06-12');

**INSERT** **INTO** public.product\_type ("name",coef) **VALUES**

('Ламинат',2.35),

('Массивная доска',5.15),

('Паркетная доска',4.34),

('Пробковое покрытие',1.5);

# ПРИЛОЖЕНИЕ Б

Программный код Б.1 – Метод расчета количества продукции

namespace ClassLibrary1

{

public class Class1

{

int count;

float width;

float length;

float[] product\_type = new float[] { 0.1F, 0.5F, 0.75F };

float[] material\_type = new float[] { 0.01F, 0.02F };

public int GetQuantityForProduct(int count, float width, float length, int product\_type, int material\_type)

{

if (product\_type <= this.product\_type.Length && product\_type >= 1)

{

if (material\_type <= this.material\_type.Length && material\_type >= 1)

{

float result = ((width \* length) \* this.product\_type[product\_type - 1]) \* Convert.ToSingle(count);

result = result + result \* this.material\_type[material\_type - 1];

return Convert.ToInt32(Math.Ceiling(result));

}

else

{

return -1;

}

}

else

{

return -1;

}

}

}

}

Программный код Б.1 – Автоматизированное тестирование

using ClassLibrary1;

namespace TestProject1

{

[TestClass]

public sealed class Test1

{

[TestMethod]

public void GetQuantityForProduct\_NonExistentProductType()

{

int count = 15;

float width = 20;

float length = 45;

int product\_type = 4;

int material\_type = 1;

int except = -1;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.AreEqual(except, actual);

}

[TestMethod]

public void GetQuantityForProduct\_NonExistentMaterialType()

{

int count = 15;

float width = 20;

float length = 45;

int product\_type = 3;

int material\_type = 3;

int except = -1;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.AreEqual(except, actual);

}

[TestMethod]

public void GetQuantityForProduct\_CalculationsAreEqual()

{

int count = 15;

float width = 20;

float length = 45;

int product\_type = 3;

int material\_type = 1;

int except = 10227;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.AreEqual(except, actual);

}

[TestMethod]

public void GetQuantityForProduct\_BoundaryValues()

{

int count = 123;

float width = 987.654f;

float length = 321.098f;

int product\_type = 3;

int material\_type = 2;

int expected = 29840696;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void GetQuantityForProduct\_LargeDimensions()

{

int count = 500;

float width = 10000f;

float length = 15000f;

int product\_type = 3;

int material\_type = 2;

int expected = 37500000;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void GetQuantityForProduct\_CalculationsIsNotNull()

{

int count = Convert.ToInt32(null);

float width = 20;

float length = 45;

int product\_type = 3;

int material\_type = 1;

int except = 114147;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.IsNotNull(actual);

}

[TestMethod]

public void GetQuantityForProduct\_CalculationsIsInt()

{

int count = 15;

float width = 20;

float length = 45;

int product\_type = 3;

int material\_type = 1;

int except;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.IsInstanceOfType(actual, typeof(int));

}

[TestMethod]

public void GetQuantityForProduct\_CalculationsIsNotDouble()

{

int count = 15;

float width = 20;

float length = 45;

int product\_type = 3;

int material\_type = 1;

int except;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.IsNotInstanceOfType(actual, typeof(double));

}

[TestMethod]

public void GetQuantityForProduct\_DecimalDimensions()

{

int count = 12;

float width = 11.23f;

float length = 17.89f;

int product\_type = 2;

int material\_type = 1;

int expected = 1218;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void GetQuantityForProduct\_MinWidthLength()

{

int count = 5;

float width = 0.001f;

float length = 0.002f;

int product\_type = 1;

int material\_type = 1;

int expected = 1;

Class1 class1 = new Class1();

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.AreEqual(expected, actual);

}

[TestMethod]

public void GetQuantityForProduct\_OverflowInResult()

{

int count = int.MaxValue;

float width = float.MaxValue;

float length = float.MaxValue;

int product\_type = 3;

int material\_type = 2;

Class1 class1 = new Class1();

try

{

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.Fail("Expected exception was not thrown.");

}

catch (OverflowException ex)

{

StringAssert.Contains(ex.Message, "Value was either too large or too small for an Int32.");

}

}

[TestMethod]

public void GetQuantityForProduct\_AllParametersAreNegative()

{

int count = -5;

float width = -20f;

float length = -30f;

int product\_type = -2;

int material\_type = -1;

Class1 class1 = new Class1();

try

{

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.Fail("Expected exception was not thrown.");

}

catch (ArgumentOutOfRangeException ex)

{

StringAssert.Contains(ex.Message, "All parameters must be non-negative");

}

}

[TestMethod]

public void GetQuantityForProduct\_StringInsteadOfNumber()

{

string count = "abc";

float width = 20f;

float length = 30f;

int product\_type = 2;

int material\_type = 1;

Class1 class1 = new Class1();

try

{

int actual = class1.GetQuantityForProduct(Convert.ToInt32(count), width, length, product\_type, material\_type);

Assert.Fail("Expected exception was not thrown.");

}

catch (FormatException ex)

{

StringAssert.Contains(ex.Message, "The input string 'abc' was not in a correct format.");

}

}

[TestMethod]

public void GetQuantityForProduct\_NullValueForCount()

{

int? count = null;

float width = 20f;

float length = 30f;

int product\_type = 2;

int material\_type = 1;

Class1 class1 = new Class1();

try

{

int actual = class1.GetQuantityForProduct((int)count, width, length, product\_type, material\_type);

Assert.Fail("Expected exception was not thrown.");

}

catch (NullReferenceException ex)

{

StringAssert.Contains(ex.Message, "System.InvalidOperationException: Nullable object must have a value");

}

}

[TestMethod]

public void GetQuantityForProduct\_DivideByZero()

{

int count = 10;

float width = 0f;

float length = 0f;

int product\_type = 1;

int material\_type = 1;

Class1 class1 = new Class1();

try

{

int actual = class1.GetQuantityForProduct(count, width, length, product\_type, material\_type);

Assert.Fail("Expected exception was not thrown.");

}

catch (DivideByZeroException ex)

{

StringAssert.Contains(ex.Message, "Attempted to divide by zero");

}

}

}

}

# ПРИЛОЖЕНИЕ В

Таблица В.1 - Аннотация теста

|  |  |
| --- | --- |
| **Название проекта** | Мастер пол |
| **Рабочая версия** | 1.0 |
| **Имя тестирующего** | Батракова Валерия |
| **Дата(ы) теста** | **06.03.2025** |

Таблица В.2 – Тест-кейс 1

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_UI\_1 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Редактирование наименования партнера |
| **Краткое изложение теста** | Проверка возможности изменения наименования и типа партнера |
| **Этапы теста** | 1. Выбрать из списка партнеров партнера ООО «Тестовый пример» 2. В открывшемся окне изменить поле наименование на «Тест» 3. Изменить в выпадающем списке поле тип с ООО на ЗАО 4. Нажать кнопку «сохранить» 5. Подтвердить сохранение в всплывшем предупреждении 6. Проверить партнера в списке партнеров |
| **Тестовые данные** | Изначальное наименование: «Тестовый пример»  Изначальный тип: ООО  Измененное наименование: «Тест»  Измененный тип: ЗАО |
| **Ожидаемый результат** | Успешное обновление наименования и типа партнера |
| **Фактический результат** | Успешное обновление наименования и типа партнера |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.3 – Тест-кейс 2

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_UI\_2 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Отклонение сохранения изменений партнера |
| **Краткое изложение теста** | Проверка возможности отказа от изменения наименования и типа партнера |
| **Этапы теста** | 1. Выбрать из списка партнеров партнера ООО «Тестовый пример» 2. В открывшемся окне изменить поле наименование на «Тест» 3. Изменить в выпадающем списке поле тип с ООО на ЗАО 4. Нажать кнопку «сохранить» 5. Во всплывшем предупреждении отклонить сохранение изменений 6. При помощи кнопки «назад» в правом верхнем углу экрана вернуться к списку партнеров 7. Проверить партнера в списке партнеров |
| **Тестовые данные** | Изначальное наименование: «Тестовый пример»  Изначальный тип: ООО  Измененное наименование: «Тест»  Измененный тип: ЗАО |
| **Ожидаемый результат** | Изменение партнера не происходит |
| **Фактический результат** | Изменение партнера не происходит |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.4 – Тест-кейс 3

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_UI\_3 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Проверка обязательной заполненности полей |
| **Краткое изложение теста** | Проверка обязательности заполнения всех необходимых полей при редактировании партнера |
| **Этапы теста** | 1. Выбрать из списка партнеров партнера ЗАО «Тестовый пример» 2. В открывшемся окне изменить все поля на пустые 3. Нажать кнопку «сохранить» |
| **Тестовые данные** | Изначальное наименование: «Тестовый пример»  Изначальный тип: ЗАО  После изменения: тип – ЗАО, рейтинг – 0, все остальные поля пусты |
| **Ожидаемый результат** | Изменение партнера не происходит и выводится предупреждение о том, что обязательные поля должны быть заполнены |
| **Фактический результат** | Успешное обновление наименования и типа партнера |
| **Статус** | Незачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.5 – Тест-кейс 4

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_UI\_4 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Проверка ограничения поля рейтинга |
| **Краткое изложение теста** | Проверка невозможности ввода отрицательного рейтинга партнера |
| **Этапы теста** | 1. Выбрать из списка партнеров партнера ООО «Тестовый пример» 2. В открывшемся окне изменить поле рейтинг на -1 3. Нажать кнопку «сохранить» |
| **Тестовые данные** | Изначальный рейтинг: 5  После изменения: -1 |
| **Ожидаемый результат** | Изменение партнера не происходит и выводится предупреждение о том, что введенный рейтинг не соответствует требованиям системы |
| **Фактический результат** | Изменение партнера не происходит и выводится предупреждение о том, что введенный рейтинг не соответствует требованиям системы |

Продолжение таблицы В.5

|  |  |
| --- | --- |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.6 – Тест-кейс 5

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_UI\_5 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Проверка сохранения без внесения изменений |
| **Краткое изложение теста** | Проверка поведения системы при попытке сохранить форму редактирования партнера без внесения каких-либо изменений |
| **Этапы теста** | 1. Выбрать из списка партнеров партнера ООО «Тестовый пример» 2. В открывшемся окне оставить все поля без изменений 3. Нажать кнопку «сохранить» |
| **Тестовые данные** | Изначальное название: «Тестовый пример» |
| **Ожидаемый результат** | Система должна обработать нажатие кнопки "Сохранить" без ошибок и уведомлений, так как никаких изменений не было сделано. Страница должна перезагрузиться с текущими данными партнера. |
| **Фактический результат** | Система обработывает нажатие кнопки "Сохранить", а страница с выводом списка открывается с текущими данными партнера. |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

# ПРИЛОЖЕНИЕ Г

Программный код Г.1 – Создание таблиц

-- Создание таблицы Role

**CREATE** **TABLE** role (

role\_id SERIAL **PRIMARY** **KEY**,

role\_name **VARCHAR** **NOT** **NULL**

);

-- Создание таблицы User

**CREATE** **TABLE** "user" (

user\_id SERIAL **PRIMARY** **KEY**,

user\_surname **VARCHAR** **NOT** **NULL**,

user\_name **VARCHAR** **NOT** **NULL**,

user\_patronymic **VARCHAR**,

user\_login **TEXT** **NOT** **NULL**,

user\_password **TEXT** **NOT** **NULL**,

user\_role **INT** **NOT** **NULL** **REFERENCES** role(role\_id)

);

-- Создание таблицы Order

**CREATE** **TABLE** "order" (

order\_id SERIAL **PRIMARY** **KEY**,

order\_date **DATE** **NOT** **NULL**,

order\_delivery\_date **DATE** **NOT** **NULL**,

order\_pickup\_point **int** **REFERENCES** pickup\_point(pickup\_point\_id),

order\_code **int** **not** **null**,

order\_status **int** **REFERENCES** status(status\_id)

);

**ALTER** **TABLE** "order" **ADD** order\_user **int**

**ALTER** **TABLE** **"order"** **ADD** **CONSTRAINT** order\_user\_fk **FOREIGN** **KEY** (order\_user) **REFERENCES** **"user"**(user\_id) **ON** **DELETE** **CASCADE**;

**CREATE** **TABLE** manufacturer (

manufacturer\_id SERIAL **PRIMARY** **KEY**,

name **TEXT** **NOT** **NULL**

);

**CREATE** **TABLE** status (

status\_id SERIAL **PRIMARY** **KEY**,

name **TEXT** **NOT** **NULL**

);

**CREATE** **TABLE** provider (

provider\_id SERIAL **PRIMARY** **KEY**,

name **TEXT** **NOT** **NULL**

);

**CREATE** **TABLE** category (

category\_id SERIAL **PRIMARY** **KEY**,

name **TEXT** **NOT** **NULL**

);

**CREATE** **TABLE** pickup\_point (

pickup\_point\_id SERIAL **PRIMARY** **KEY**,

name **TEXT** **NOT** **NULL**

);

-- Создание таблицы Product

**CREATE** **TABLE** product (

product\_article\_number **VARCHAR** **PRIMARY** **KEY**,

product\_name **TEXT** **NOT** **NULL**,

product\_description **TEXT**,

product\_category **INT** **REFERENCES** category(category\_id),

product\_photo **text**,

product\_manufacturer **INT** **REFERENCES** manufacturer(manufacturer\_id),

product\_provider **INT** **REFERENCES** provider(provider\_id),

product\_cost **DECIMAL**(19,4) **NOT** **NULL**,

product\_discount\_max\_amount **FLOAT**,

unit **VARCHAR**,

product\_discount\_amount **FLOAT**,

product\_quantity\_in\_stock **INT** **NOT** **NULL**

);

-- Создание таблицы OrderProduct

**CREATE** **TABLE** order\_product (

id SERIAL **PRIMARY** **KEY**,

order\_id **INT** **NOT** **NULL** **REFERENCES** "order"(order\_id),

product\_article\_number **VARCHAR**(100) **NOT** **NULL** **REFERENCES** product(product\_article\_number),

product\_quantity **INT**

);

Программный код А.2 – Заполнение таблиц

**INSERT** **INTO** public.manufacturer ("name") **VALUES**

('Pilot'),

('Attache'),

('Unimax'),

('FLEXOFFICE CANDEE'),

('Svetocopy'),

('Hatber'),

('Erich Krause'),

('Комус'),

('Pilot'),

('GoodMark');

**INSERT** **INTO** public.manufacturer ("name") **VALUES**

(' GoodMark');

**INSERT** **INTO** public.category ("name") **VALUES**

('Школьные принадлежности'),

('Для офиса'),

('Бумага офисная'),

('Тетради школьные');

**INSERT** **INTO** public."user" (user\_surname,user\_name,user\_patronymic,user\_login,user\_password,user\_role) **VALUES**

('Никифоров ','Всеволод','Иванович','loginDEjrm2018','Cpb+Im',2),

('Воронов ','Донат','Никитевич','loginDEpxl2018','P6h4Jq',1),

('Игнатьева ','Евгения','Валентиновна','loginDEwgk2018','&mfI9l',2),

('Буров ','Федот','Егорович','loginDEpou2018','gX3f5Z',1),

('Петров','Иван','Семёновна','loginDEjwl2018','D4ZYHt',3),

('Денисов ','Дамир','Филатович','loginDEabf2018','\*Tasm+',2),

('Ершов ','Максим','Геласьевич','loginDEwjm2018','k}DJKo',2),

('Копылов ','Куприян','Пётрович','loginDEjvz2018','&|bGTy',3),

('Носов ','Валерьян','Дмитрьевич','loginDEuyv2018','8hhrZ}',3),

('Силин ','Игорь','Авдеевич','loginDExdm2018','DH68L9',1);

**INSERT** **INTO** public."user" (user\_surname,user\_name,user\_patronymic,user\_login,user\_password,user\_role) **VALUES**

('Дроздова ','Александра','Мартыновна','loginDEeiv2018','H\*BxlS',3),

('Дроздов ','Аркадий','Геласьевич','loginDEfuc2018','VuM+QT',2),

('Боброва ','Варвара','Евсеевна','loginDEoot2018','usi{aT',3),

('Чернова ','Агата','Данииловна','loginDElhk2018','Okk0jY',3),

('Лыткина ','Ульяна','Станиславовна','loginDEazg2018','s3bb|V',2),

('Лаврентьев ','Леонид','Игнатьевич','loginDEaba2018','#ИМЯ?',1),

('Кулаков ','Юрий','Владленович','loginDEtco2018','tTKDJB',2),

('Соловьёв ','Андрей','Александрович','loginDEsyq2018','2QbpBN',3),

('Корнилова ','Марфа','Макаровна','loginDEpxi2018','+5X&hy',2),

('Белоусова ','Любовь','Георгьевна','loginDEicr2018','3+|Sn{',2);

**INSERT** **INTO** public."user" (user\_surname,user\_name,user\_patronymic,user\_login,user\_password,user\_role) **VALUES**

('Анисимов ','Никита','Гордеевич','loginDEcui2018','Zi1Tth',3),

('Стрелкова ','Фаина','Федосеевна','loginDEpxc2018','G+nFsv',2),

('Осипов ','Евгений','Иванович','loginDEqrd2018','sApUbt',1),

('Владимирова ','Иванна','Павловна','loginDEsso2018','#ИМЯ?',3),

('Кудрявцева ','Жанна','Демьяновна','loginDErsy2018','{Aa6nS',3),

('Матвиенко ','Яков','Брониславович','loginDEvpz2018','mS0UxK',3),

('Селезнёв ','Егор','Артёмович','loginDEfog2018','glICay',2),

('Брагин ','Куприян','Митрофанович','loginDEpii2018','Ob}RZB',2),

('Гордеев ','Виктор','Эдуардович','loginDEhyk2018','\*gN}Tc',1),

('Мартынов ','Онисим','Брониславович','loginDEdxi2018','ywLUbA',3);

**INSERT** **INTO** public."user" (user\_surname,user\_name,user\_patronymic,user\_login,user\_password,user\_role) **VALUES**

('Никонова ','Евгения','Павловна','loginDEzro2018','B24s6o',3),

('Полякова ','Анна','Денисовна','loginDEuxg2018','K8jui7',2),

('Макарова ','Пелагея','Антониновна','loginDEllw2018','jNtNUr',2),

('Андреева ','Анна','Вячеславовна','loginDEddg2018','gGGhvD',1),

('Кудрявцева ','Кира','Ефимовна','loginDEpdz2018','#ИМЯ?',2),

('Шилова ','Кира','Егоровна','loginDEyiw2018','cnj3QR',3),

('Ситников ','Игорь','Борисович','loginDEqup2018','95AU|R',1),

('Русаков ','Борис','Христофорович','loginDExil2018','w+++Ht',1),

('Капустина ','Ульяна','Игоревна','loginDEkuv2018','Ade++|',3),

('Беляков ','Семён','Германнович','loginDEmox2018','Je}9e7',3);

**INSERT** **INTO** public.**"user"** (user\_surname,user\_name,user\_patronymic,user\_login,user\_password,user\_role) **VALUES**

('Гурьев ','Ириней','Игнатьевич','loginDEvug2018','lEa{Cn',2),

('Мишин ','Христофор','Леонидович','loginDEzre2018','N\*VX+G',2),

('Лазарева ','Антонина','Христофоровна','loginDEbes2018','NaVtyH',3),

('Маркова ','Ираида','Сергеевна','loginDEkfg2018','r1060q',2),

('Носкова ','Пелагея','Валерьевна','loginDEyek2018','KY2BL4',2),

('Баранов ','Станислав','Дмитрьевич','loginDEloq2018','NZV5WR',1),

('Ефремов ','Демьян','Артёмович','loginDEjfb2018','TNT+}h',3),

('Константинов ','Всеволод','Мэлсович','loginDEueq2018','GqAUZ6',3),

('Ситникова ','Ираида','Андреевна','loginDEpqz2018','F0Bp7F',3),

('Матвеев ','Кондрат','Иванович','loginDEovk2018','JyJM{A',1);

**INSERT** **INTO** public."role" (role\_name) **VALUES**

('Администратор'),

('Клиент'),

('Менеджер');

**INSERT** **INTO** public."order" (order\_date,order\_delivery\_date,order\_pickup\_point,order\_code,order\_status,order\_user) **VALUES**

('2022-05-18','2022-05-24',3,311,1,**NULL**),

('2022-05-19','2022-05-25',4,312,2,**NULL**),

('2022-05-20','2022-05-26',5,313,1,**NULL**),

('2022-05-21','2022-05-27',6,314,1,**NULL**),

('2022-05-22','2022-05-28',7,315,2,**NULL**),

('2022-05-23','2022-05-29',10,316,1,**NULL**),

('2022-05-24','2022-05-30',11,317,1,**NULL**),

('2022-05-25','2022-05-31',20,318,1,**NULL**),

('2022-05-26','2022-06-01',30,319,1,**NULL**),

('2022-05-27','2022-06-02',33,320,1,**NULL**);

**INSERT** **INTO** public.status ("name") **VALUES**

('Новый '),

('Завершен');

**INSERT** **INTO** public.provider ("name") **VALUES**

('Комус'),

('Буквоед');

**INSERT** **INTO** public.product (product\_article\_number,product\_name,product\_description,product\_category,product\_photo,product\_manufacturer,product\_provider,product\_cost,product\_discount\_max\_amount,unit,product\_discount\_amount,product\_quantity\_in\_stock) **VALUES**

('А112Т4','Ручка шариковая','Ручка шариковая с синими чернилами,толщина стержня 7 мм',1,'А112Т4.jpg',1,1,12.0000,30.0,'шт.',2.3,6),

('A346R4','Ручка шариковая автоматическая','Ручка шариковая автоматическая с синими чернилами, диаметр шарика 0,9 мм',1,'A346R4.jpg',2,1,35.0000,20.0,'шт.',3.0,23),

('T564P5','Набор шариковых ручек одноразовых','Набор шариковых ручек одноразовых Attache Economy Spinner 10 цветов (толщина линии 0.5 мм)',1,'T564P5.jpg',2,1,50.0000,15.0,'уп.',9.0,5),

('B730E2','Ручка шариковая','Ручка шариковая одноразовая автоматическая Unimax Fab GP синяя (толщина линии 0.5 мм)',1,'B730E2.jpg',3,1,41.0000,10.0,'шт.',3.0,45),

('G278R6','Ручка шариковая','Ручка шариковая FLEXOFFICE CANDEE 0,6 мм, синяя',2,'G278R6.png',4,2,15.0000,30.0,'шт.',7.0,23),

('R259E6','Бумага офисная','Бумага офисная Svetocopy NEW A4 80г 500л',3,'R259E6.jpg',5,2,299.0000,25.0,'уп.',4.0,32),

('H452A3','Тетрадь','Тетрадь, 24 листа, Зелёная обложка Hatber/Хатбер, офсет, клетка с полями',4,'H452A3.png',6,2,10.0000,8.0,'шт.',3.0,25),

('A543T6','Ручка шариковая','Ручка шариковая Erich Krause, R-301 ORANGE 0.7 Stick, синий',2,'A543T6.jpg',7,2,13.0000,30.0,'шт.',6.0,12),

('F719R5','Папка-скоросшиватель','Папка-скоросшиватель, А4 Hatber/Хатбер 140/180мкм АССОРТИ, пластиковая с перфорацией прозрачный верх',2,'F719R5.jpg',6,2,18.0000,20.0,'шт.',3.0,8),

('D419T7','Клей-карандаш','Клей-карандаш Erich Krause 15 гр.',2,'D419T7.png',7,2,61.0000,18.0,'шт.',4.0,26);

**INSERT** **INTO** public.product (product\_article\_number,product\_name,product\_description,product\_category,product\_photo,product\_manufacturer,product\_provider,product\_cost,product\_discount\_max\_amount,unit,product\_discount\_amount,product\_quantity\_in\_stock) **VALUES**

('N592T4','Стикеры','Стикеры Attache 76x76 мм пастельные желтые (1 блок, 100 листов)',2,'',2,1,34.0000,15.0,'уп.',2.0,17),

('N459R6','Стикеры','Стикеры Attache Selection 51х51 мм неоновые 5 цветов (1 блок, 250 листов)',2,'',2,1,194.0000,25.0,'уп.',3.0,9),

('S276E6','Скрепки','Скрепки Комус металлические никелированные 33 мм (100 штук в упаковке)',2,'',8,1,46.0000,30.0,'уп.',2.0,14),

('S453G7','Скрепки','Скрепки 28 мм Attache металлические (100 штук в упаковке)',2,'',2,1,21.0000,15.0,'уп.',4.0,20),

('J539R3','Кнопки','Кнопки канцелярские Комус металлические цветные (50 штук в упаковке)',2,'',8,1,96.0000,20.0,'уп.',3.0,24),

('Z539E3','Лента клейкая','Лента клейкая 12мм\*33м прозрачная, Hatber/Хатбер',2,'',6,2,16.0000,15.0,'шт.',2.0,14),

('A297U6','Ручка гелевая','Ручка гелевая ErichKrause® G-Cube®, цвет чернил черный',1,'',7,2,52.0000,20.0,'шт.',4.0,18),

('A567R4','Ручка шариковая','Шариковая ручка PILOT SuperGrip 0,7 мм синяя BPGP-10R-F-L',1,'',1,2,64.0000,30.0,'шт.',2.0,32),

('K932R4','Корректирующая лента','Корректор лента 5мм\*4м, блистер, GoodMark',2,'',10,2,70.0000,25.0,'шт.',3.0,16),

('K345R5','Корректирующая лента','Корректирующая лента Attache Economy 5 мм x 5 м',2,'',2,1,87.0000,20.0,'шт.',3.0,12);

**INSERT** **INTO** public.product (product\_article\_number,product\_name,product\_description,product\_category,product\_photo,product\_manufacturer,product\_provider,product\_cost,product\_discount\_max\_amount,unit,product\_discount\_amount,product\_quantity\_in\_stock) **VALUES**

('K753R3','Корректирующая жидкость','Корректирующая жидкость (штрих) Attache быстросохнущая 20 мл',1,'',2,1,50.0000,30.0,'шт.',2.0,5),

('S563T6','Степлер','Степлер Attache 8215 до 25 листов черный',2,'',2,1,231.0000,25.0,'шт.',4.0,17),

('S425T6','Скобы','Скобы для степлера №24/6 Attache оцинкованные (1000 штук в упаковке)',2,'',2,1,25.0000,20.0,'уп.',2.0,16),

('A340R5','Маркер','Маркер перманентный GoodMark 2-х сторонний.для СD/DVD черный',2,'',10,2,66.0000,15.0,'шт.',4.0,27),

('A384T5','Тетрадь','Тетрадь, 18 листов, А5 линейка Hatber/Хатбер Серия Зеленая 10шт в блистере',1,'',6,2,87.0000,20.0,'уп.',3.0,23),

('K502T9','Карандаш-корректор','Карандаш-корректор GoodMark, морозостойкий, 8мл, металлический наконечник',2,'',10,2,70.0000,25.0,'шт.',2.0,7),

('S512T7','Скобы','Скобы №10 1000шт, к/к, GoodMark',2,'',11,2,25.0000,15.0,'уп.',3.0,32),

('D367R4','Клей ПВА','Клей ПВА 85г Hatber/Хатбер',1,'',6,2,26.0000,20.0,'шт.',4.0,16),

('M892R4','Ножницы','Ножницы 195 мм Attache с пластиковыми прорезиненными анатомическими ручками бирюзового/черного цвета',1,'',2,1,209.0000,15.0,'шт.',5.0,13),

('Z390R4','Лента клейкая','Клейкая лента упаковочная Комус 50 мм x 100 м 50 мкм прозрачная',2,'',8,1,195.0000,20.0,'шт.',2.0,9);

**INSERT** **INTO** public.pickup\_point (address) **VALUES**

('344288, г. Тихорецк, ул. Чехова, 1'),

('614164, г.Тихорецк, ул. Степная, 30'),

('394242, г. Тихорецк, ул. Коммунистическая, 43'),

('660540, г. Тихорецк, ул. Солнечная, 25'),

('125837, г. Тихорецк, ул. Шоссейная, 40'),

('125703, г. Тихорецк, ул. Партизанская, 49'),

('625283, г. Тихорецк, ул. Победы, 46'),

('614611, г. Тихорецк, ул. Молодежная, 50'),

('454311, г.Тихорецк, ул. Новая, 19'),

('660007, г.Тихорецк, ул. Октябрьская, 19');

**INSERT** **INTO** public.pickup\_point (address) **VALUES**

('603036, г. Тихорецк, ул. Садовая, 4'),

('450983, г.Тихорецк, ул. Комсомольская, 26'),

('394782, г. Тихорецк, ул. Чехова, 3'),

('603002, г. Тихорецк, ул. Дзержинского, 28'),

('450558, г. Тихорецк, ул. Набережная, 30'),

('394060, г.Тихорецк, ул. Фрунзе, 43'),

('410661, г. Тихорецк, ул. Школьная, 50'),

('625590, г. Тихорецк, ул. Коммунистическая, 20'),

('625683, г. Тихорецк, ул. 8 Марта'),

('400562, г. Тихорецк, ул. Зеленая, 32');

**INSERT** **INTO** public.pickup\_point (address) **VALUES**

('614510, г. Тихорецк, ул. Маяковского, 47'),

('410542, г. Тихорецк, ул. Светлая, 46'),

('620839, г. Тихорецк, ул. Цветочная, 8'),

('443890, г. Тихорецк, ул. Коммунистическая, 1'),

('603379, г. Тихорецк, ул. Спортивная, 46'),

('603721, г. Тихорецк, ул. Гоголя, 41'),

('410172, г. Тихорецк, ул. Северная, 13'),

('420151, г. Тихорецк, ул. Вишневая, 32'),

('125061, г. Тихорецк, ул. Подгорная, 8'),

('630370, г. Тихорецк, ул. Шоссейная, 24');

**INSERT** **INTO** public.pickup\_point (address) **VALUES**

('614753, г. Тихорецк, ул. Полевая, 35'),

('426030, г. Тихорецк, ул. Маяковского, 44'),

('450375, г. Тихорецк ул. Клубная, 44'),

('625560, г. Тихорецк, ул. Некрасова, 12'),

('630201, г. Тихорецк, ул. Комсомольская, 17'),

('190949, г. Тихорецк, ул. Мичурина, 26');

**INSERT** **INTO** public.order\_product (order\_id,product\_article\_number,product\_quantity) **VALUES**

(1,'А112Т4',10),

(2,'A346R4',10),

(3,'B730E2',10),

(4,'H452A3',10),

(5,'F719R5',1),

(6,'N459R6',2),

(7,'J539R3',20),

(8,'A567R4',5),

(9,'K753R3',1),

(10,'S425T6',5);

**INSERT** **INTO** public.order\_product (order\_id,product\_article\_number,product\_quantity) **VALUES**

(1,'R259E6',2),

(2,'T564P5',1),

(3,'G278R6',10),

(4,'A543T6',10),

(5,'D419T7',1),

(6,'S276E6',15),

(7,'Z539E3',1),

(8,'K932R4',1),

(9,'S563T6',1),

(10,'A340R5',3);

# ПРИЛОЖЕНИЕ Д

Программный код Д.1 – Метод расчета

namespace SF2022User\_NN\_Lib.dll

{

public class Calculations

{

// Метод для поиска свободных интервалов времени

public static string[] AvailablePeriods(TimeSpan[] startTimes, int[] durations, TimeSpan beginWorkingTime, TimeSpan endWorkingTime, int consultationTime)

{

List<string> freeIntervals = new List<string>();

// Преобразуем начальные времена и длительности в пары

List<(TimeSpan Start, TimeSpan End)> busyIntervals = new List<(TimeSpan Start, TimeSpan End)>();

for (int i = 0; i < startTimes.Length; ++i)

{

TimeSpan start = startTimes[i];

TimeSpan end = start.Add(new TimeSpan(0, durations[i], 0)); // Добавляем длительность к началу

busyIntervals.Add((Start: start, End: end));

}

// Сортируем занятные интервалы по времени начала

busyIntervals.Sort((a, b) => a.Start.CompareTo(b.Start));

// Начальное свободное время - от начала рабочего дня до первого занятого интервала

TimeSpan currentFreeStart = beginWorkingTime;

foreach (var interval in busyIntervals)

{

TimeSpan currentFreeEnd = interval.Start;

// Если промежуток между текущим свободным временем и началом следующего занятого интервала достаточен для консультации

if (currentFreeEnd - currentFreeStart >= new TimeSpan(0, consultationTime, 0))

{

freeIntervals.Add(FormatInterval(currentFreeStart, currentFreeEnd));

}

// Обновляем текущее свободное время после занятого интервала

currentFreeStart = interval.End;

}

// Последнее свободное время - от конца последнего занятого интервала до окончания рабочего дня

TimeSpan lastFreeEnd = endWorkingTime;

if (lastFreeEnd - currentFreeStart >= new TimeSpan(0, consultationTime, 0))

{

freeIntervals.Add(FormatInterval(currentFreeStart, lastFreeEnd));

}

return freeIntervals.ToArray();

}

// Вспомогательная функция для форматирования временного интервала

private static string FormatInterval(TimeSpan start, TimeSpan end)

{

return $"{start.Hours:D2}:{start.Minutes:D2}-{end.Hours:D2}:{end.Minutes:D2}";

}

}

}

Программный код Д.2 – Автоматизированное тестирование

using SF2022User\_NN\_Lib.dll;

namespace TestProject

{

[TestClass]

public sealed class Test1

{

[TestMethod]

public void TestNoConsultations()

{

TimeSpan[] startTimes = Array.Empty<TimeSpan>();

int[] durations = Array.Empty<int>();

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0); // Начало рабочего дня в 09:00

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0); // Конец рабочего дня в 18:00

int consultationTime = 30; // Время одной консультации — 30 минут

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:00-18:00", result[0]);

}

[TestMethod]

public void TestOneShortConsultationInTheMiddleOfDay()

{

TimeSpan[] startTimes = { new TimeSpan(12, 0, 0) };

int[] durations = { 60 }; // Консультация длится 1 час

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:00-12:00", result[0]);

Assert.AreEqual("13:00-18:00", result[1]);

}

[TestMethod]

public void TestFirstConsultationStartsImmediatelyAfterWorkBegins()

{

TimeSpan[] startTimes = { new TimeSpan(9, 0, 0) };

int[] durations = { 120 }; // Консультация длиной 2 часа

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("11:00-18:00", result[0]);

}

[TestMethod]

public void TestLastConsultationEndsJustBeforeEndOfWorkday()

{

TimeSpan[] startTimes = { new TimeSpan(16, 0, 0) };

int[] durations = { 90 }; // Консультация занимает 1,5 часа

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:00-16:00", result[0]); // Остается всего полчаса

}

[TestMethod]

public void TestInsufficientTimeBetweenConsultations()

{

TimeSpan[] startTimes = { new TimeSpan(9, 0, 0), new TimeSpan(9, 40, 0) };

int[] durations = { 30, 30 };

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("10:10-18:00", result[0]); // Промежуток между 9:30 и 9:40 слишком мал

}

[TestMethod]

public void TestLongConsultationLeavesSmallFreePeriod()

{

TimeSpan[] startTimes = { new TimeSpan(10, 0, 0) };

int[] durations = { 300 }; // Очень долгая консультация — 5 часов

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:00-10:00", result[0]); // Оставшиеся три часа после консультации

}

[TestMethod]

public void TestSingleConsultationAtMidpointOfWorkday()

{

TimeSpan[] startTimes = { new TimeSpan(12, 0, 0) };

int[] durations = { 60 };

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:00-12:00", result[0]);

Assert.AreEqual("13:00-18:00", result[1]);

}

[TestMethod]

public void TestTwoCloseConsultationsWithSpaceInBetween()

{

TimeSpan[] startTimes = { new TimeSpan(10, 0, 0), new TimeSpan(11, 30, 0) };

int[] durations = { 30, 30 };

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:00-10:00", result[0]);

Assert.AreEqual("10:30-11:30", result[1]);

Assert.AreEqual("12:00-18:00", result[2]);

}

[TestMethod]

public void TestFreeTimeOnlyAtBeginningOfWorkday()

{

TimeSpan[] startTimes = { new TimeSpan(9, 30, 0) };

int[] durations = { 60 };

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:00-09:30", result[0]);

Assert.AreEqual("10:30-18:00", result[1]);

}

[TestMethod]

public void TestMinimumDurationConsultation()

{

TimeSpan[] startTimes = { new TimeSpan(9, 0, 0) };

int[] durations = { 1 };

TimeSpan beginWorkingTime = new TimeSpan(9, 0, 0);

TimeSpan endWorkingTime = new TimeSpan(18, 0, 0);

int consultationTime = 30;

var result = Calculations.AvailablePeriods(startTimes, durations, beginWorkingTime, endWorkingTime, consultationTime);

Assert.AreEqual("09:01-18:00", result[0]);

}

}

}

Таблица Д.1 - Аннотация теста

|  |  |
| --- | --- |
| **Название проекта** | UP2 |
| **Рабочая версия** | 1.0 |
| **Имя тестирующего** | Батракова Валерия |
| **Дата(ы) теста** | **12.03.2025** |

Таблица В.2 – Тест-кейс 1

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_UI\_1 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Добавление нового уникального товара |
| **Краткое изложение теста** | Администратор добавляет новый товар с уникальными характеристиками. |
| **Этапы теста** | 1. Открыть форму добавления товара. 2. Заполнить поля: название товара, цена, категория, описание, количество на складе. 3. Нажать кнопку "Добавить". |
| **Тестовые данные** | Уникальные значения наименования и артикула, заполеннные поля количества на складе, стоимости, описания, категории и т.д. |
| **Ожидаемый результат** | Товар успешно добавляется в базу данных. Пользователь видит сообщение об успехе. |
| **Фактический результат** | Товар успешно добавляется в базу данных. Пользователь видит сообщение об успехе. |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.3 – Тест-кейс 2

|  |  |
| --- | --- |
| **Тестовый пример #** | TC\_UI\_2 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Добавление товара с уже существующим названием |
| **Краткое изложение теста** | Администратор пытается добавить товар с названием, которое уже существует в базе данных. |
| **Этапы теста** | 1. Открыть форму добавления товара. 2. Заполнить поля: ввести название товара, которое уже используется другим товаром. 3. Нажать кнопку "Добавить". |
| **Тестовые данные** | Неуникальное значение наименования и уникальное значение артикула, заполеннные поля количества на складе, стоимости, описания, категории и т.д. |
| **Ожидаемый результат** | Система выдает ошибку, сообщающую, что товар с таким названием уже существует. Пользователь не может продолжить процесс добавления. |
| **Фактический результат** |  |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.4 – Тест-кейс 3

|  |  |
| --- | --- |
| Тестовый пример **#** | TC\_UI\_3 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Добавление товара с отрицательной ценой |
| **Краткое изложение теста** | Администратор вводит отрицательную цену товара |
| **Этапы теста** | 1. Открыть форму добавления товара. 2. Заполнить поля: указать отрицательное число в поле цены. 3. Нажать кнопку "Добавить". |
| **Тестовые данные** | Уникальные значения наименования и артикула, заполеннные поля количества на складе, описания, категории и т.д. с отрицательной стоимостью |
| **Ожидаемый результат** | Система блокирует операцию и выводит предупреждение о недопустимости ввода отрицательных значений. |
| **Фактический результат** | Система блокирует операцию и выводит предупреждение о недопустимости ввода отрицательных значений. |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.5 – Тест-кейс 4

|  |  |
| --- | --- |
| Тестовый пример **#** | TC\_UI\_4 |
| **Приоритет тестирования** | Средний |
| **Заголовок/название теста** | Добавление товара с нулевым количеством на складе |
| **Краткое изложение теста** | Администратор пытается добавить товар с количеством равным нулю. |
| **Этапы теста** | 1. Открыть форму добавления товара. 2. Указать количество товара равное нулю. 3. Нажать кнопку "Добавить". |
| **Тестовые данные** | Уникальные значения наименования и артикула, заполеннные поля стоимости, описания, категории и т.д. с нулевым количеством на складе |
| **Ожидаемый результат** | Система предупреждает администратора о том, что количество товара должно быть больше нуля. Операция не выполняется. |
| **Фактический результат** | Система предупреждает администратора о том, что количество товара должно быть больше нуля. Операция не выполняется. |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

Таблица В.6 – Тест-кейс 5

|  |  |
| --- | --- |
| **Test Case #** | TC\_UI\_5 |
| **Приоритет тестирования** | Высокий |
| **Заголовок/название теста** | Добавление товара без категории |
| **Краткое изложение теста** | Администратор не выбирает категорию товара. |
| **Этапы теста** | 1. Открыть форму добавления товара. 2. Пропустить выбор категории. 3. Нажать кнопку "Добавить". |
| **Тестовые данные** | Уникальные значения значения наименования и артикула, заполеннные поля количества на складе, стоимости, описания, без указания категории |
| **Ожидаемый результат** | Система требует обязательного выбора категории. Операция добавления невозможна без заполнения всех обязательных полей. |
| **Фактический результат** | Система требует обязательного выбора категории. Операция добавления невозможна без заполнения всех обязательных полей. |
| **Статус** | Зачёт |
| **Предварительное условие** | Перед выполнением программы IDE не выдаёт сообщений об ошибках, соблюдена вся логика проектирования библиотеки. |
| **Постусловие** | После завершения тестирования система не должна выдавать сообщений об ошибках или исключений. При следующих запусках сохраняется корректная работа программы. |
| **Примечания/комментарии** | - |

# ПРИЛОЖЕНИЕ Е

Программный код Е.1 – скрипт базы данных

-- DROP SCHEMA public;

-- CREATE SCHEMA public AUTHORIZATION pg\_database\_owner;

-- DROP SEQUENCE public."Orders\_id\_seq";

CREATE SEQUENCE public."Orders\_id\_seq"

  INCREMENT BY 1

  MINVALUE 1

  MAXVALUE 9223372036854775807

  START 1

  CACHE 1

  NO CYCLE;

-- public.actions definition

-- Drop table

-- DROP TABLE public.actions;

CREATE TABLE public.actions (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  created\_at timestamptz NOT NULL DEFAULT now(),

  photo text NULL,

  CONSTRAINT actions\_pkey PRIMARY KEY (id)

);

-- public.categories definition

-- Drop table

-- DROP TABLE public.categories;

CREATE TABLE public.categories (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  title text NOT NULL,

  CONSTRAINT "Categories\_pkey" PRIMARY KEY (id)

);

-- public.deliveries definition

-- Drop table

-- DROP TABLE public.deliveries;

CREATE TABLE public.deliveries (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  created\_at timestamptz NOT NULL DEFAULT now(),

  city text NULL,

  coast int8 NULL,

  CONSTRAINT deliveries\_pkey PRIMARY KEY (id)

);

-- public.order\_status definition

-- Drop table

-- DROP TABLE public.order\_status;

CREATE TABLE public.order\_status (

  created\_at timestamptz NOT NULL DEFAULT now(),

  "name" text NULL,

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  CONSTRAINT order\_status\_pkey PRIMARY KEY (id)

);

-- public.products definition

-- Drop table

-- DROP TABLE public.products;

CREATE TABLE public.products (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  title text NOT NULL,

  category\_id uuid NULL,

  "cost" float8 NOT NULL,

  description text NOT NULL,

  is\_best\_seller bool NULL DEFAULT false,

  CONSTRAINT "Products\_pkey" PRIMARY KEY (id),

  CONSTRAINT "Products\_categoryId\_fkey" FOREIGN KEY (category\_id) REFERENCES public.categories(id) ON DELETE SET NULL ON UPDATE CASCADE

);

-- public.cart definition

-- Drop table

-- DROP TABLE public.cart;

CREATE TABLE public.cart (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  product\_id uuid NULL,

  user\_id uuid NULL,

  count int8 NULL,

  CONSTRAINT "Basket\_pkey" PRIMARY KEY (id)

);

-- public.favourite definition

-- Drop table

-- DROP TABLE public.favourite;

CREATE TABLE public.favourite (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  product\_id uuid NULL,

  user\_id uuid NULL,

  CONSTRAINT "Favoutite\_pkey" PRIMARY KEY (id)

);

-- public.notifications definition

-- Drop table

-- DROP TABLE public.notifications;

CREATE TABLE public.notifications (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  created\_at timestamptz NOT NULL DEFAULT now(),

  "text" text NULL,

  title text NULL,

  user\_id uuid NULL,

  is\_read bool NULL DEFAULT false,

  CONSTRAINT messages\_pkey PRIMARY KEY (id)

);

-- public.orders definition

-- Drop table

-- DROP TABLE public.orders;

CREATE TABLE public.orders (

  id int8 NOT NULL GENERATED BY DEFAULT AS IDENTITY,

  created\_at timestamptz NOT NULL DEFAULT now(),

  email text NULL,

  phone text NULL,

  address text NULL,

  user\_id uuid NULL,

  payment\_id uuid NULL,

  delivery\_coast int8 NULL,

  status\_id uuid NULL,

  CONSTRAINT "Orders\_pkey" PRIMARY KEY (id)

);

-- public.orders\_items definition

-- Drop table

-- DROP TABLE public.orders\_items;

CREATE TABLE public.orders\_items (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  created\_at timestamptz NOT NULL DEFAULT now(),

  title text NULL,

  coast float8 NULL,

  count int8 NULL,

  order\_id int8 NULL,

  product\_id uuid NULL,

  CONSTRAINT "OrdersItems\_pkey" PRIMARY KEY (id)

);

-- public.payments definition

-- Drop table

-- DROP TABLE public.payments;

CREATE TABLE public.payments (

  created\_at timestamptz NOT NULL DEFAULT now(),

  user\_id uuid NULL,

  card\_name varchar NULL DEFAULT '16'::character varying,

  card\_number int8 NULL,

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  CONSTRAINT payments\_pkey PRIMARY KEY (id)

);

-- public.profiles definition

-- Drop table

-- DROP TABLE public.profiles;

CREATE TABLE public.profiles (

  id uuid NOT NULL DEFAULT gen\_random\_uuid(),

  created\_at timestamptz NOT NULL DEFAULT now(),

  user\_id uuid NOT NULL,

  photo text NULL,

  firstname text NULL,

  lastname text NULL,

  address text NULL,

  phone text NULL,

  CONSTRAINT profiles\_pkey PRIMARY KEY (id)

);

-- public.cart foreign keys

ALTER TABLE public.cart ADD CONSTRAINT "Basket\_id\_product\_fkey" FOREIGN KEY (product\_id) REFERENCES public.products(id);

ALTER TABLE public.cart ADD CONSTRAINT "Basket\_id\_user\_fkey" FOREIGN KEY (user\_id) REFERENCES auth.users(id);

-- public.favourite foreign keys

ALTER TABLE public.favourite ADD CONSTRAINT "Favourite\_id\_product\_fkey" FOREIGN KEY (product\_id) REFERENCES public.products(id);

ALTER TABLE public.favourite ADD CONSTRAINT "Favoutite\_id\_user\_fkey" FOREIGN KEY (user\_id) REFERENCES auth.users(id);

ALTER TABLE public.favourite ADD CONSTRAINT favourite\_product\_id\_fkey FOREIGN KEY (product\_id) REFERENCES public.products(id);

ALTER TABLE public.favourite ADD CONSTRAINT favourite\_user\_id\_fkey FOREIGN KEY (user\_id) REFERENCES auth.users(id);

-- public.notifications foreign keys

ALTER TABLE public.notifications ADD CONSTRAINT "Messages\_id\_user\_fkey" FOREIGN KEY (user\_id) REFERENCES auth.users(id);

ALTER TABLE public.notifications ADD CONSTRAINT notifications\_user\_id\_fkey FOREIGN KEY (user\_id) REFERENCES auth.users(id);

-- public.orders foreign keys

ALTER TABLE public.orders ADD CONSTRAINT "Orders\_id\_user\_fkey" FOREIGN KEY (user\_id) REFERENCES auth.users(id);

ALTER TABLE public.orders ADD CONSTRAINT orders\_payment\_id\_fkey FOREIGN KEY (payment\_id) REFERENCES public.payments(id);

ALTER TABLE public.orders ADD CONSTRAINT orders\_status\_id\_fkey FOREIGN KEY (status\_id) REFERENCES public.order\_status(id);

-- public.orders\_items foreign keys

ALTER TABLE public.orders\_items ADD CONSTRAINT "OrdersItems\_id\_order\_fkey" FOREIGN KEY (order\_id) REFERENCES public.orders(id);

ALTER TABLE public.orders\_items ADD CONSTRAINT "OrdersItems\_uuid\_product\_fkey" FOREIGN KEY (product\_id) REFERENCES public.products(id);

-- public.payments foreign keys

ALTER TABLE public.payments ADD CONSTRAINT payments\_user\_id\_fkey FOREIGN KEY (user\_id) REFERENCES auth.users(id);

-- public.profiles foreign keys

ALTER TABLE public.profiles ADD CONSTRAINT profiles\_user\_id\_fkey FOREIGN KEY (user\_id) REFERENCES auth.users(id);

INSERT INTO public.categories (id,title) VALUES

   ('ea4ed603-8cbe-4d57-a359-b6b843a645bc','Outdoor'),

   ('4f3a690b-41bf-4fca-8ffc-67cc385c6637','Tennis'),

   ('76ab9d74-7d5b-4dee-9c67-6ed4019fa202','Men'),

   ('8143b506-d70a-41ec-a5eb-3cf09627da9e','Women');

INSERT INTO public.order\_status (created\_at,"name",id) VALUES

   ('2024-05-12 19:16:36.039641+00','Собираем','970aed1e-549c-499b-a649-4bf3f9f93a01'),

   ('2024-05-12 19:17:04.920148+00','Отменен','8ac05d2f-8371-42f3-b2a9-7beac2fb2c75'),

   ('2024-05-12 19:16:26.245791+00','Доставлен','a513ae92-701c-49c8-b6ba-c1eda3e79a93');

INSERT INTO public.products (id,title,category\_id,"cost",description,is\_best\_seller) VALUES

   ('6478da8e-87e6-4a7a-821c-ac38dd861cec','PUMA CA Pro Classic','ea4ed603-8cbe-4d57-a359-b6b843a645bc',13999.0,'Ретро-кроссовки в стиле баскетбола отличаются массивной подошвой и спортивным силуэтом. Прочный нейлоновый верх с замшевыми вставками обеспечивает поддержку и воздухопроницаемость. Амортизирующая подошва гарантирует мягкость и отзывчивость при каждом шаге. Прочная резиновая подошва обеспечивает сцепление и долговечность для бега или активного образа жизни.',true),

   ('64229908-3713-4147-bedc-68ddaef9c67a','STREETBEAT Plimsoll Retro','76ab9d74-7d5b-4dee-9c67-6ed4019fa202',8499.0,'Ностальгические кеды в стиле ретро перенесут вас в прошлое с классическим холщовым верхом для воздухопроницаемости и комфорта. Контрастная резиновая подошва добавляет винтажный штрих. Удобный круглый носок и гибкая конструкция делают их идеальными для повседневных прогулок и неформальных мероприятий.',false),

   ('60f2ced6-7b81-465d-bc0f-9c3395b78c56','STREETBEAT Wave','8143b506-d70a-41ec-a5eb-3cf09627da9e',4899.0,'Эти кроссовки сочетают в себе ретро-вдохновение с современными технологиями. Многослойный верх из различных материалов создает уникальный и привлекательный внешний вид. Эксклюзивная подошва обеспечивает амортизацию и комфорт, а выглядывающий механизм шнуровки добавляет спортивный штрих. Идеально подходят для тех, кто хочет выделиться из толпы стильной и функциональной обувью.',false),

   ('815aa749-975e-4285-bc49-6921556fb9ad','Adidas Niteball','76ab9d74-7d5b-4dee-9c67-6ed4019fa202',19499.0,'Беговые кроссовки с инновационной амортизацией обеспечат мягкий и плавный бег. Трио-резиновая подошва с пенным сердечником поглощает удары и обеспечивает отзывчивость. Верх из сетки с поддерживающими накладками обеспечивает воздухопроницаемость и поддержку. Идеальный выбор для бегунов, ищущих максимальный комфорт и амортизацию.',false),

   ('21c2d7c0-a2ea-49a2-9198-52bafafc6958','PUMA Velophasis Phased','4f3a690b-41bf-4fca-8ffc-67cc385c6637',16999.0,'Минималистичные повседневные кроссовки сочетают в себе стиль и комфорт. Прочный кожаный верх с тиснением придает изысканный вид. Гибкая подошва из полиуретана обеспечивает комфорт и поддержку во время ходьбы и повседневных занятий. Универсальная и стильная обувь, которую можно сочетать с различными образами.',false),

   ('6bbdb878-4b52-433c-adb4-a742759939fa','HUGO Leon Run','76ab9d74-7d5b-4dee-9c67-6ed4019fa202',19999.0,'Кроссовки в уличном стиле сочетают в себе смелый дизайн и функциональность. Верх из различных материалов обеспечивает воздухопроницаемость, поддержку и долговечность. Промежуточная подошва поглощает удары и обеспечивает амортизацию, а резиновая подошва гарантирует сцепление и долговечность. Отличный выбор для тех, кто хочет выделиться из толпы стильной и комфортной обувью.',false),

   ('3fb5c391-a841-45fb-9c7a-2f74d11a4bfa','Adidas Ozmillen','8143b506-d70a-41ec-a5eb-3cf09627da9e',20599.0,'Легкие и дышащие беговые кроссовки созданы для плавных и комфортных пробежек. Верх из сетки и синтетических материалов обеспечивает вентиляцию и поддержку. Амортизирующая подошва поглощает удары и обеспечивает стабильность для преодоления любых расстояний с легкостью. Гибкая резиновая подошва гарантирует сцепление и долговечность.',true),

   ('74bc6394-6695-4134-992b-1d9972aeb639','HUGO Go1st','8143b506-d70a-41ec-a5eb-3cf09627da9e',21999.0,'Стильные беговые кроссовки сочетают в себе функциональность и привлекательный дизайн. Верх из сетки и синтетических материалов обеспечивает воздухопроницаемость и поддержку. Амортизирующая промежуточная подошва из EVA поглощает удары и возвращает энергию, повышая эффективность бега. Прочная резиновая подошва обеспечивает сцепление и долговечность на различных поверхностях. Отличный выбор для бегунов, желающих хорошо выглядеть и чувствовать себя во время пробежек.',false),

   ('9b56e21d-9e2c-4cce-b48d-497397f0ba47','Reebok Floatzig 1','76ab9d74-7d5b-4dee-9c67-6ed4019fa202',14999.0,'Гибридные кроссовки, сочетающие элементы баскетбола и беговых кроссовок, создают уникальный и функциональный дизайн. Верх из сетки с накладками из синтетической кожи обеспечивает поддержку и воздухопроницаемость. Амортизация в подошве поглощает удары и обеспечивает отзывчивость, а изогнутая резиновая подошва обеспечивает динамичное движение. Идеально подходят для игроков в баскетбол, которым нужна обувь, обеспечивающая поддержку, комфорт и стиль.',false),

   ('63d56e11-2769-4980-96df-dc11ea70148d','PUMA Orkid II','ea4ed603-8cbe-4d57-a359-b6b843a645bc',14999.0,'Женские кроссовки для тренировок сочетают в себе стиль и функциональность. Дышащий сетчатый верх с поддерживающими накладками обеспечивает комфорт и воздухопроницаемость во время тренировок. Промежуточная подошва из пены поглощает удары и обеспечивает амортизацию, а резиновая подошва гарантирует сцепление и долговечность. Универсальная и стильная обувь для фитнеса и повседневной носки.',true);

INSERT INTO public.products (id,title,category\_id,"cost",description,is\_best\_seller) VALUES

   ('9e9c9924-7000-4e7b-8332-33bd8d0bd9b6','PUMA Blktop Rider','76ab9d74-7d5b-4dee-9c67-6ed4019fa202',13999.0,'Кроссовки PUMA Blktop Rider в уличном стиле сочетают в себе смелый дизайн и функциональность. Верх из сетки, замши и кожи обеспечивает воздухопроницаемость, поддержку и долговечность. Промежуточная подошва IMEVA поглощает удары и обеспечивает амортизацию, а резиновая подошва гарантирует сцепление и долговечность.',false),

   ('4f311959-5ce2-4b3c-94b9-e41dd790e571','STREETBEAT Wave Runner','ea4ed603-8cbe-4d57-a359-b6b843a645bc',4899.0,'Беговые кроссовки STREETBEAT Wave Runner сочетают в себе классический дизайн с современными технологиями. Верх из сетки с замшевыми накладками обеспечивает воздухопроницаемость и поддержку. Волнистая амортизирующая подошва поглощает удары и обеспечивает плавный бег. Гибкая резиновая подошва гарантирует сцепление и долговечность.',false),

   ('b12a26f0-31ff-4c54-b6e8-9701a25030fe','PUMA All-Pro Nitro','4f3a690b-41bf-4fca-8ffc-67cc385c6637',14499.0,'Футбольные кроссовки премиум-класса разработаны для профессиональных игроков. Верх из синтетической кожи с текстурированными накладками обеспечивает превосходный контроль над мячом и поддержку. Подошва из пены обеспечивает амортизацию и отзывчивость, а резиновые шипы гарантируют превосходное сцепление на поле. Отличный выбор для футболистов, которым нужна высококачественная обувь для соревнований и тренировок.',false),

   ('afa022ec-f15b-425e-b9d9-fef6c7da654e','PUMA Genetics','ea4ed603-8cbe-4d57-a359-b6b843a645bc',11499.0,'Беговые кроссовки в ретро-стиле сочетают в себе классический дизайн с современными технологиями. Верх из сетки с замшевыми накладками обеспечивает воздухопроницаемость и поддержку. Волнистая амортизирующая подошва поглощает удары и обеспечивает плавный бег. Отличный выбор для бегунов, которые хотят выглядеть стильно и чувствовать себя комфортно во время пробежек.',false);