

**КУРСОВОЙ ПРОЕКТ**

**по дисциплине: МДК 01.01 Разработка программных модулей**

**Тема: Разработка программного модуля**

**информационной системы «Цех предприятия»**

**Специальность 09.02.07 «Информационные системы и программирование»**

|  |  |  |
| --- | --- | --- |
| **Выполнил студент(ка) группы 311ИС-23** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **В.А.Жуков** |
| **Руководитель** | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Н.И. Кручинкина** |

**Москва 2024**



**УТВЕРЖДАЮ**

**Зам. директора КМПО**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.Ф. Гасанов**

**«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.**

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ**

**по дисциплине: МДК 01.01 Разработка программных модулей**

**Специальность 09.02.07 «Информационные системы и программирование»**

**Студент группы 311ИС-23 Жуков Владимир Александрович**

**ТЕМА: «Разработка программного модуля информационной**

**системы «Цех предприятия»**

Дата выдачи задания «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г.

Срок сдачи проекта «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 г.

**Москва 2024**

**Перечень вопросов, подлежащих разработке:**

Введение

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Предпроектное обследование

1.2

2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА МОДУЛЯ

2.1 Эскизное проектирование

2.1.1

2.1.2

2.2 Проектирование базы данных

2.3 Техническое проектирование

2.3.1 Разработка архитектуры

2.3.2 Защита доступа к данным

3. ОТЛАДКА И ТЕСТИРОВАНИЕ МОДУЛЯ

3.1

3.2.

Заключение

**Исходные данные:**

Перечень разрабатываемых материалов для визуализации: электронная презентация, содержащая в том числе:

Блок-схемы алгоритмов программного модуля

Use-case диаграмма.

ER-диаграмма базы данных.

Архитектура программного модуля

Экранные формы интерфейса.

Задание выдал:

Руководитель курсового проекта Н.И. Кручинкина

Задание принял к исполнению В.А. Жуков

*Рассмотрено*

на заседании предметно-цикловой комиссии

информационных технологий и системного

администрирования

Протокол № \_\_\_*от «\_\_» \_\_\_\_\_\_\_\_\_* 2024 г.

Председатель ПЦК \_\_\_\_\_\_\_\_\_\_\_\_\_

###### СОДЕРЖАНИЕ

[**ВВЕДЕНИЕ** 3](#_Toc185558589)

[1) **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ** 5](#_Toc185558590)

[1.1 Описание предметной области 5](#_Toc185558591)

[1.2 Характеристики системы 6](#_Toc185558592)

[1.3 Анализ имеющихся инструментов для создания системы 7](#_Toc185558593)

[1.3.1 SAP S/4HANA 8](#_Toc185558594)

[1.3.2 Oracle Cloud ERP 8](#_Toc185558595)

[2) **ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРОГРАММЫ** 12](#_Toc185558596)

[2.1 Функциональные требования 12](#_Toc185558597)

[2.2 Основные объекты и их описание 15](#_Toc185558598)

[В базе данных “login.bd” присутствуют данные таблицы 15](#_Toc185558599)

[2.3 План тестирования 17](#_Toc185558600)

[3) **ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ МОДУЛЯ** 19](#_Toc185558601)

[3.1 Описание среды разработки 19](#_Toc185558602)

[3.2 Реализация пользовательского интерфейса приложения 20](#_Toc185558603)

[3.3 Описание кодом функциональности модуля 25](#_Toc185558604)

[3.3.2 Модуль Admin\_menu.py 25](#_Toc185558605)

[3.3.3 Модуль User\_menu.py 27](#_Toc185558606)

[3.3.4 Модуль main.py 27](#_Toc185558607)

[3.3.5 Модуль Userlogin.py 28](#_Toc185558608)

[4) **ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ** 28](#_Toc185558609)

[**ЗАКЛЮЧЕНИЕ** 31](#_Toc185558610)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 33](#_Toc185558611)

# ВВЕДЕНИЕ

Современные предприятия, независимо от их размера и сферы деятельности, сталкиваются с необходимостью автоматизации бизнес-процессов для повышения эффективности работы и снижения затрат. Одним из ключевых элементов производственного процесса является цех предприятия, который отвечает за выполнение технологических операций, контроль качества продукции и управление ресурсами. Однако многие цеха до сих пор функционируют с использованием устаревших методов управления и отсутствием интегрированных информационных систем, что приводит к снижению производительности, увеличению ошибок и сложностям в контроле за производственными процессами.

Актуальность темы курсового проекта "Разработка программного модуля информационной системы «Цех предприятия»" обусловлена необходимостью создания современных инструментов для автоматизации и оптимизации работы цехов. Информационные системы позволяют централизованно управлять данными, контролировать производственные процессы, анализировать показатели эффективности и принимать обоснованные управленческие решения. Разработка такого модуля является важным шагом в повышении конкурентоспособности предприятия и его адаптации к требованиям цифровой экономики.

Целью данного курсового проекта является разработка программного модуля для информационной системы, который обеспечит автоматизацию процессов управления и контроля в цехе предприятия. Основной задачей является создание инструмента, который позволит эффективно управлять производственными ресурсами, контролировать выполнение технологических операций и анализировать данные для повышения производительности.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ существующих подходов к автоматизации управления цехами предприятий.

2. Разработать структуру и функциональные требования к программному модулю.

3. Создать базу данных для хранения информации о производственных процессах, ресурсах и показателях эффективности.

4. Реализовать программный модуль с учетом требований пользователей и современных технологий.

5. Провести тестирование и отладку модуля для обеспечения его стабильной работы.

6. Разработать документацию, включая руководство пользователя и техническую документацию.

Объектом исследования в данном проекте является процесс управления и контроля производственной деятельности цеха предприятия.

Предметом исследования выступают методы и средства автоматизации управления производственными процессами, включая сбор, хранение и анализ данных о выполнении технологических операций, использовании ресурсов и показателях эффективности.

Практическая значимость проекта заключается в возможности внедрения разработанного программного модуля в реальные производственные процессы предприятий. Это позволит повысить эффективность работы цехов, снизить трудозатраты на управление и контроль, а также обеспечить оперативный доступ к важным данным для принятия управленческих решений. Разработанный модуль может стать частью более широкой информационной системы предприятия, что способствует интеграции и оптимизации всех бизнес-процессов.

Таким образом, данный проект направлен на решение актуальной проблемы автоматизации управления цехами предприятий, что является важным шагом в повышении конкурентоспособности и эффективности производственной деятельности.

# 1) АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ

### 1.1 Описание предметной области

В данном курсовом проекте предметной областью для решения поставленных задач является склад цеха предприятия. Это структурное подразделение, отвечающее за хранение, учет и распределение материальных ресурсов, необходимых для производственного процесса.

Склад цеха предприятия обеспечивает бесперебойную работу производственных линий, предоставляя в нужный момент времени необходимые материалы и комплектующие. Основные функции склада включают прием, хранение, учет и выдачу материалов, а также контроль за их остатками и движением.

Ассортимент продукции, хранящейся на складе, достаточно широк. В его состав входят сырье для производства (металлы, пластмассы, химические вещества), комплектующие (детали, узлы, инструменты), готовая продукция (полуфабрикаты, готовые изделия), а также расходные материалы (краски, масла, смазки).

Основные пользователи системы – это работники склада (кладовщики, логисты), производственные подразделения, а также руководство предприятия, которое использует данные для анализа эффективности работы цеха и принятия управленческих решений.

Деятельность склада заключается в последовательном решении следующих вопросов:

приемка материалов и комплектующих от поставщиков;

учет поступивших и выданных материалов;

контроль остатков на складе;

формирование отчетов о движении материалов.

Возникает необходимость в компьютерной обработке вводимых данных о поставках, расходах и остатках материалов, с целью хранения этой информации на носителях, а также для анализа эффективности работы склада и цеха в целом.

Организуется обследование предметной области: оценка объема и цели проекта, определение требований, объектов и функций на высоком уровне. Сбор информации начинается с изучения существующих форм документов, отчетов, имеющихся файлов, баз данных, программ.

Анализ должен заканчиваться подробным описанием информации об объектах предметной области, которая требуется для решения конкретных задач и которая должна храниться в базе данных , формулировкой конкретных задач, которые будут решаться с использованием данной базы данных с кратким описанием алгоритмов их решения, описанием выходных документов, которые должны генерироваться в системе, описанием входных документов, которые служат основанием для заполнения данными базы данных.

### 1.2 Характеристики системы

Характеристики системы описывают особенности функционирования системы, которые не связаны с конкретными функциями, но важны для ее успешной работы.

К ним относятся:

1. Производительность:

Система должна обеспечивать быстрый обход запросов пользователей, включая добавление, редактирование и удаление данных. Время отклика системы на запросы не должно превышать 2 секунды.

1. Безопасность:

Доступ к системе должен быть защищен с помощью аутентификации пользователей (логин и пароль). Необходимо реализовать уровни доступа для различных ролей пользователей (администратор, сотрудник цеха), чтобы ограничить доступ к определенным функциям.

1. Удобство использования:

* интерфейс системы должен быть интуитивно понятным и удобным для пользователей, чтобы минимизировать время на обучение. Система должна поддерживать работу с различными устройствами (компьютеры, смартфоны, планшеты) для обеспечения мобильности пользователей.
* надежность:

Система должна обеспечивать сохранность данных и защиту от потери информации (например, с помощью регулярного резервного копирования). Необходимо реализовать механизмы восстановления системы в случае сбоя или аварии.

4. Адаптивность:

Система должна быть способна адаптироваться к изменениям нагрузки и увеличивать пропускную способность без снижения производительности. Необходимо предусмотреть возможность горизонтального расширения для обеспечения роста системы.

### 1.3 Анализ имеющихся инструментов для создания системы

Существует множество инструментов для создания систем автоматизации. К ним можно отнести программное обеспечение для управления производственными процессами, такое как ERP-системы (SAP, ORACLE) и MES-системы (Manufacturing Execution Systems). Эти системы помогают управлять ресурсами и планированием, включающая CRM-системы (Salesforce, Zoho) и системы управления запасами. Кроме того, важную роль играют роботизированные системы и RPA (Robotic Process Automation), которые позволяют автоматизировать рутинные задачи с помощью платформ, таких как UiPath и Blue Prism. В данном случае будет использоваться ERPсистема, поэтому будет рассматриваться два продукта от двух разных компаний в сфере этой системы.

#### 1.3.1 SAP S/4HANA

SAP S/4HANA является основным продуктом компании SAP. Работает он на базе платформы HANA и использует технологии обработки данных в реальном времени. Это позволяет пользователям получать актуальную информацию для принятия решений, а также обеспечивает высокую скорость обработки транзакций. Модульная архитектура SAP позволяет компаниям внедрять только те модули, которые соответствуют их потребностям. Из таких модулей можно выделить:

- модуль управления производством (PP);

- модуль управления запасами (MM);

- модуль управления финансами (F);

- Однако SAP имеет и свои недостатки:

- сложность внедрения и настройки системы, что требует значительных временных и финансовых затрат;

- необходимость обучения сотрудников для работы с системой, что может занять много времени;

- высокая стоимость лицензий и обслуживания;

#### 1.3.2 Oracle Cloud ERP

Oracle Cloud ERP – это продукт для управления ресурсами предприятия. Он предлагает широкий спектр функциональности, включая в себя управление финансами, проектами, закупками и производственными процессами. Он отличается высокой гибкостью и масштабируемостью, что позволяет предприятиям адаптировать систему под свои специфические требования. Самым главным преимуществом данного продукта является возможность интеграции с другими облачными сервисами Oracle, такими как Oracle Analytics и Oracle SCM Cloud.

Но, как и все, данный продукт имеет и свои недостатки:

- более высокая стоимость внедрения по сравнению с некоторыми конкурентами

- сложность интеграции с существующими системами

- зависимость от облачных решений.

Оба продукта пригодны к использованию в среде компании.

Данная система будет реализована на Python в виде приложения. В нём будет реализована регистрация пользователей. Учет расходов, позволит пользователям данного приложения смотреть, куда уходят средства компании и в каком количестве. Смена языка интерфейса позволит работать в программе в международной среде. Поиск по складу позволит быстро и точно найти необходимый товар. Учет товаров:

- реализована база данных для хранения информации о товарах (название, описание, категория, цена, количество на складе);

- функционал добавления новых товаров, редактирования информации о существующих товарах и удаления товаров из базы данных;

- проверка на ограничение общего количества товаров на складе (не более 100 000 единиц);

1. управление заказами:

- реализован функционал добавления товаров в корзину покупателя;

- оформление заказа с сохранением информации о заказе в базе данных;

- вывод информации о статусе заказа для покупателя;

- возможность удаления товаров из корзины и отмены заказа;

- управление пользователями;

- управление учетной записью пользователя (изменение пароля, удаление аккаунта);

- хранение информации о пользователях в базе данных (имя, пароль, тип доступа);

2. администрирование:

- административный интерфейс для управления товарами и так далее;

- генерация круговой диаграммы для визуализации распределения товаров по количеству;

- возможность смены пользователя и выхода из системы;

3. Техническая реализация:

- используется СУБД SQLite для хранения данных;

- язык программирования Python с использованием библиотеки Tkinter для создания графического интерфейса;

- реализована архитектура системы с разделением на модули (например, Admin\_menu.py, User\_menu.py, Userlogin.py);

- обеспечена безопасность паролей (пароли хранятся в базе данных в верхнем регистре);

4. Структура входной информации:

данные о товарах:

- название, описание, категория, цена, количество на складе;

данные о пользователях:

- имя, логин, пароль, тип доступа (администратор или пользователь);

данные о заказах:

- id пользователя, список товаров в заказе, дата и время заказа;

5. Структура выходной информации:

для покупателя:

- список товаров в ассортименте (название, описание, категория, цена, количество на складе);

- корзина (список товаров, добавленных в корзину, с возможностью изменения количества);

- информация о заказе (список товаров, общая стоимость, дата и время заказа);

Для администратора:

- список товаров в ассортименте (название, описание, категория, цена, количество на складе);

- список заказов (ID заказа, список товаров, общая стоимость, дата и время заказа, статус);

- информация о пользователях (имя, логин, пароль, тип доступа);

6. Дополнительные возможности:

мульти язычность:

- Реализована поддержка русского и английского языков;

- графический интерфейс: используется Tkinter для создания пользовательского интерфейса;

- визуализация данных: Возможность построения круговой диаграммы для анализа количества товаров на складе;

# 2) ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРОГРАММЫ

### 2.1 Функциональные требования

- Модуль для системы «Balumba» должен обеспечивать

- функциональность учета, поиска, обновления данных на складе цеха производства, также данный модуль должен учитывать расходы на товары, которые были взяты со склада.

- Функциональные требования включают в себя следующее:

- регистрация нового пользователя админом.

- хранение данных о зарегистрированном пользователе, включая логин и контактную информацию (адрес электронной почты). Все данные о пользователе и его заказах хранятся в базе данных, которая взаимодействует с системой.

- проверка уникальности имени пользователя при регистрации

- просмотр списка пользователей

- отображение всех зарегистрированных в системе работников цеха.

- поиск по имени пользователя

- поиск товаров по складу

- проверить кол-во товара на складе

- узнать стоимость 1 единицы товара

- добавление товаров на склад

- поиск товаров по складу

- обновление данных пользователя админом

- анализ расходов

2.3Разработка сценария проекта

Проектирование сценариев проекта — это важный этап, который помогает точно определить, как будущая система будет работать с людьми и какие задачи она будет решать. Эти сценарии позволяют представить себе, как система будет использоваться, и служат основой для дальнейшей разработки. Сценарий 1: Добавление материала Участники: Администратор Описание:

Администратор добавляет информацию о новом материале в систему.

Последовательность действий:

- администратор выбирает опцию "Добавить материал";

- система запрашивает ввод информации о материале (название, категория, количество, цена за единицу);

- администратор вводит необходимые данные и подтверждает дополнение;

- система сохраняет информацию о материале и отображает сообщение об успешном добавлении;

- система обновляет список материалов;

Ожидаемый результат:

- новый материал успешно добавлен в систему и отображается в списке доступных материалов;

- информация о материале сохранена в базе данных;

Сценарий 2: Добавление продукции Участники: Сотрудник Описание:

Сотрудник добавляет информацию о произведенной продукции.

Последовательность действий:

- сотрудник выбирает опцию "Добавить продукцию";

- система запрашивает ввод информации о продукции (название, категория, количество, использованные материалы);

- сотрудник вводит необходимые данные и подтверждает добавление;

- система сохраняет информацию о продукции и обновляет остатки материалов;

- система отображает сообщение об успешном добавлении;

Ожидаемый результат:

- новая продукция успешно добавлена в систему и отображается в списке произведенной продукции;

- остатки материалов обновлены в соответствии с использованными ресурсами;

Сценарий 3: Генерация отчетов Участники: Администратор Описание:

Администратор создает детальный анализ производства за указанный период.

Последовательность действий:

- администратор выбирает функцию "Создание отчета";

- система предлагает выбрать временной диапазон для анализа (дата начала и дата окончания);

- администратор вводит дополнительные параметры для уточнения запроса (например, категория продукции, использованные материалы) и подтверждает запрос;

- система формирует отчет и отображает его в удобном формате;

Ожидаемый результат:

- интегрированный отчет о производстве успешно создан и доступен для детального изучения;

- отчет содержит информацию о произведенной продукции, использованных материалах и их остатках;

Нефункциональные требования:

Интерфейс пользователя

Интерфейс пользователя должен обеспечивать удобство использования, предоставляя простой и интуитивно понятный способ взаимодействия с приложением.

Время для отклика интерфейса на действия, производимые пользователем не должно превышать более 1 секунды.

Производительность:

Приложение должно обеспечивать довольно высокую производительность при работе с базой данных, особенно при добавление большого количества товаров.

Время запуска приложения не должно превышать 7 секунд.

Масштабируемость:

Архитектура приложения должна быть довольно гибкой и масштабируемой для возможности добавления новых функциональности в будущем.

Приложение должно поддерживать работу с базой данных, которая содержит большое количество записей.

Документация:

Весь код должен быть задокументирован хорошо, включая все комментарии к нему и содержать инструкции по использованию.

Документация должна быть понятна и доступна для разработчиков, также она должна обеспечивать понимание структуры и функциональности приложения.

### 2.2 Основные объекты и их описание

## В базе данных “login.bd” присутствуют данные таблицы

Таблица 1 – Объекты предметной области

|  |  |  |
| --- | --- | --- |
| № | Наименование объекта | Краткое описание |
| 1 | Пользователи | Пользователи с доступом к программе |
| 2 | Продукты | Основной объект, который представляет товары в системе |
| 3 | Инвентаризация | Процесс отслеживания кол-во продуктов в наличии |
| 4 | Траты | Записи о тратах |
| 5 | Корзина | Временная структура данных хранящая данные о выбранных товарах со склада |
| № | Наименование объекта | Краткое описание |
| 6 | Графики и отчёты | Показывает анализ сколько места на складе занято и сколько осталось |
| 7 | Интерфейс пользователя | Главное окно приложения |

Таблица 2 – Сущность “products”

|  |  |  |
| --- | --- | --- |
| Атрибуты | Тип | Описание |
| “producnt\_id” | Varcar(20) | Идентификатор материала |
| “product\_name” | Varcar(50) | Наименование материала |
| “product\_desc” | Varcar(50) | Описание материала |
| “product\_cat” | Varcar(50) | Категория материала |
| “product\_price” | INTEGER NOT NULL | Цена материалов |
| “stocks” | INTEGER NOT NULL | Кол- во на складе |

Таблица 3 – Сущность “sales”

|  |  |  |
| --- | --- | --- |
| Атрибуты | Тип | Описание |
| “trans\_id” | INTEGER | Номер операции |
| “invoice” | INTEGER NOT NULL | Факс-счёт |
| “product\_id” | varchar(20) | Идентификатор материала |
| “Quantity” | INTEGER NOT NULL | Кол-во материалов |
| “Date” | varchar(20) | Дата заказа |
| “Time” | varchar(20) | Время заказа |
| “username” | varchar(20) | Имя аккаунта, пользователя |

Таблица 4 – Сущность “users”

|  |  |  |
| --- | --- | --- |
| Атрибуты | Тип | Описание |
| “username” | varchar(20) | Имя пользователя |
| “sales” | varchar(20) NOT NULL | № заказа |
| “accont\_type” | varchar(10) NOT NULL | Тип аккаунта |
| “password” | Varchar(10)NOT NULL | Пароль |

ERD модели

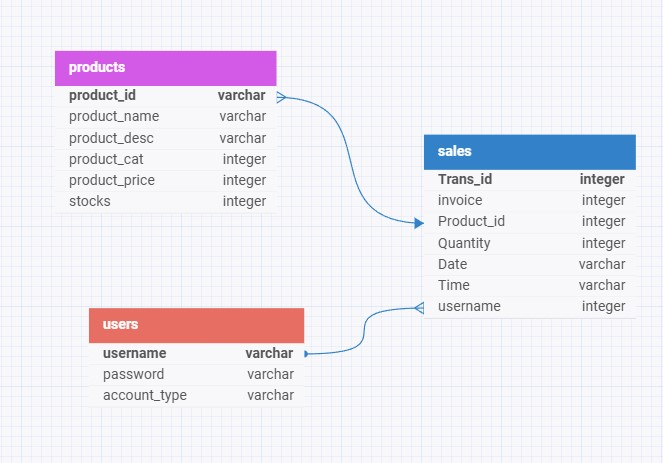


Рисунок 1 – IDEF1X-Диаграмма базы данных складской системы

ER модели

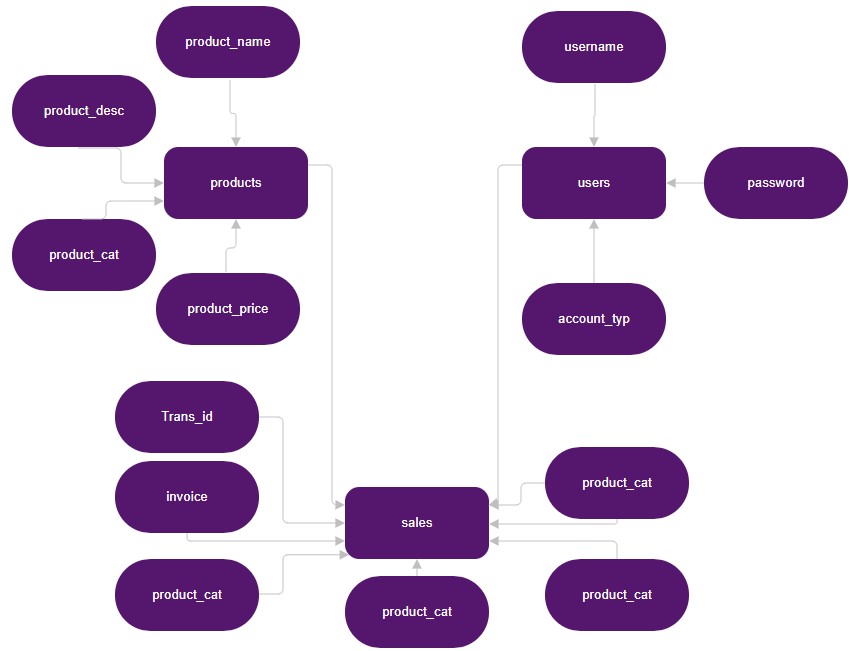


Рисунок 2 – ER-диаграмма базы данных складской системы

### 2.3 План тестирования

Регистрация и управление пользователям:

- проверить возможность регистрации новых пользователей в системе «Balumba»;

-проверить возможность изменять права доступа, пароли и регистрировать новых пользователей;

Управление товарами на складе:

- проверить, что система позволяет добавлять новые товары, менять их количество, менять цену товара, наименование товара, ID товара, категорию;

- проверить, что система позволяет удалить ненужный товар;

Проверка производительности:

- проверить, что система стабильно работает при большом количестве запросов о товарах от двух и более пользователей;

Проверка автоматизации:

- проверить, что система автоматизирует отчеты об затратах;

Проверка пользовательского интерфейса

- проверить, что интерфейс интуитивно понятен для пользователей;

- проверить, что все элементы интерфейса работают корректно и правильно отображаются;

Тестирование проверки анализа расходов:

- проверить, чтобы корректно отображались данные в отчете расходов;

Тестирование процессов удаления пользователя:

- провести тестирование процесса удаления пользователя;

# 3) ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ МОДУЛЯ

### 3.1 Описание среды разработки

Среда разработки:

PyCharm – мощная IDE для Python, предоставляющая интеллектуальную поддержку кода, встроенный отладчик и поддержку тестирования. Язык программирования: Python – универсальный язык с простым синтаксисом, кроссплатформенностью и обширной библиотекой.

Библиотеки:

- tkinter – для создания GUI;

- tkinter.ttk – для современных виджетов;

- Matplotlib – для создания графиков и диаграмм;

- Matplotlib.colors – для работы с цветами;

- atetime – для работы с датой и временем;

- os – для взаимодействия с файловой системой;

- Pillow – для обработки изображений;

База данных:

SQLite3 – встраиваемая СУБД с поддержкой SQL, высокой производительностью и низким потреблением ресурсов.

Преимущества выбранного инструментария:

- PyCharm: обеспечивает интеллектуальную поддержку кода, автоматическое дополнение, встроенный отладчик для выявления ошибок и поддержку тестирования (unittest, pytest, doctest);

- Python: простой и доступный язык с кроссплатформенностью, богатой встроенной библиотекой и огромным количеством внешних библиотек для различных задач;

- Tkinter: встроен в Python, прост в использовании, поддерживает кроссплатформенность и предоставляет гибкость для создания различных типов интерфейсов;

- SQLite3: встраиваемая СУБД без необходимости отдельного сервера,

поддерживает стандартные SQL-команды, обладает высокой производительностью и низким потреблением ресурсов;

### 3.2 Реализация пользовательского интерфейса приложения

Для удобности взаимодействия пользователя с программным модулем для инвентаризации склада цеха производства был разработан пользовательский интерфейс. Реализация пользовательского интерфейса включает в себя размещение кнопок, отображение информации, диаграмм и обработку пользовательских действий.

При первичном запуске приложения пользователь попадает в окно авторизации, где он может пройти авторизацию, если ранее был зарегистрирован в системе администратором. После ввода и подтверждения программа открывает главное окно с основной информацией и функионалом.

Интерфейс окна показан на рисунке 3.

Изображение выглядит как текст, снимок экрана, диаграмма, программное обеспечение

Автоматически созданное описание

Рисунок 3 – Главное окно пользователя

Когда пользователь введет название материала в поиск и кликнет на кнопку добавить, то он отправится в корзину, а снизу высветится сумма добавленных материалов. После как он нажмет на кнопку продолжить, то транзакция завершится.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 4 – Главное окно администратора (инвентаризация)

Когда администратор введет данные о продукте, то он может обновить информацию о нем на складе. Также администратор может найти нужный товар на складе и увидеть о нем всю информацию (остаток, цена, категория, описание). К тому же администратор может удалить товар со склада.

Изображение выглядит как текст, программное обеспечение, Шрифт, число

Автоматически созданное описание

Рисунок 5 – Окно материалов

В данном окне пользователь может посмотреть какие материалы есть в наличие на складе.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 6 – Окно добавить

В данном окне администратор может добавить новый материал на склад, введя все данные о нем (ID, Имя продукта, описание, категория, цена, остаток)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 7 – Окно профили

В данном окне администратор может зарегистрировать нового пользователя, поменять уже имеющимся пользователем логин, пароль и права доступа. Также он может найти нужного пользователя по имени и удалить ненужного.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 8 – Окно трат

Данное окно позволяет увидеть всю информацию по тратам на материалы (имя пользователя, счет-фактура, id продукта, описание, кол-во, общая цена, дата, время)

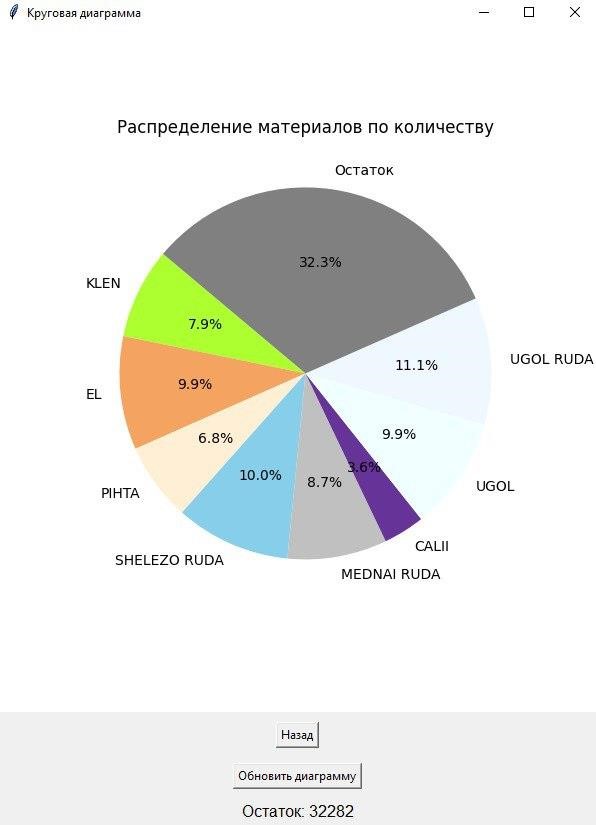


Рисунок 9 – Окно диаграммы

Данное окно показывает процентное соотношение материалов на складе и остаток свободного места под них.

### 3.3 Описание кодом функциональности модуля

3.3.1 Модуль additional\_features.py

Основные методы

Метод -Set\_complection\_list

Устанавливает список для автозаполнения

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 10 - Модуль additional\_features.py

##### 3.3.2 Модуль Admin\_menu.py

Метод -Admin\_mainmenu

Cоздает главное меню администратора с кнопками для управления продуктами, пользователями, инвентаризацией и отчетами.

Изображение выглядит как текст, снимок экрана, документ, Шрифт

Автоматически созданное описание

Рисунок 11 – Метод -Admin\_mainmenu

Изображение выглядит как текст, снимок экрана, документ, Шрифт

Автоматически созданное описание

Рисунок 12 - Метод -Admin\_mainmenu

##### 3.3.3 Модуль User\_menu.py

Метод - User\_mainmenu

Создает главное меню пользователя с кнопками для просмотра продуктов и создания заказов.

Изображение выглядит как текст, снимок экрана, документ, Шрифт

Автоматически созданное описание

Рисунок 13 - Метод - User\_mainmenu

##### 3.3.4 Модуль main.py

Метод - Create\_language\_button

Создает кнопки для смены языка интерфейса.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 14 - Метод - Create\_language\_button

##### 3.3.5 Модуль Userlogin.py

Метод – obj

Создает интерфейс для входа и регистрации.

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рисунок 15 - Метод – obj

## 4) ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ

Тесты, представленные ниже, охватывают основные функциональные возможности системы, включая авторизацию, регистрацию, добавление товаров и проверку ошибок. Каждый тест описывает сценарий, который проверяет корректность работы системы в различных условиях.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID теста** | **Описание теста**  **(тип)** | **Предусловия** | **Шаги для воспроизведения** | **Ожидаемый**  **результат** | **Фактический результат** |
| 1 | Авторизац  ия в системе (позитивн ый) | 1.  Пользователь находится на странице входа. 2.  Пользователь был ранее зарегистриро ван в системе. | 1. Ввести в поля «Логин» и «Пароль» логин и пароль пользователя. 2.  Нажать кнопку «Войти». | Пользователь перенаправляе тся на страницу личного кабинета. | Пользователь перенаправляе тся на страницу личного кабинета. |
| 2 | Авторизац  ия в системе (негативн ый) | 1.  Пользователь находится на странице входа. 2.  Пользователь не зарегистриро ван в системе. | 1. Ввести неверный логин и пароль. 2. Нажать кнопку «Войти». | Появляется сообщение об ошибке: «Неверное имя пользователя или пароль». | Появляется сообщение об ошибке: «Неверное имя пользователя или пароль». |
| 3 | Регистрац ия пользовате ля (позитивн ый) | 1.  Пользователь  находится на странице регистрации.  2.  Пользователь не зарегистриро ван в системе | 1. Ввести в обязательные поля «Имя пользователя», «Пароль»,  «Подтвердите пароль». 2.  Нажать кнопку  «Зарегистрироваться». | Пользователь успешно зарегистриров ан, появляется сообщение: «Пользователь зарегистриров ан». | Пользователь успешно зарегистриров ан, появляется сообщение: «Пользователь зарегистриров ан». |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID теста** | **Описание теста**  **(тип)** | **Предусловия** | **Шаги для воспроизведения** | **Ожидаемый**  **результат** | **Фактический результат** |
| 4 | Регистрац ия пользовате ля (негативн ый) | 1.  Пользователь находится на странице регистрации.  2.  Пользователь уже зарегистриро ван в системе. | 1. Ввести имя пользователя, которое уже существует. 2.  Нажать кнопку  «Зарегистрироват ься». | Появляется сообщение об ошибке: «Имя пользователя уже существует». | Появляется сообщение об ошибке: «Имя пользователя уже существует». |
| 5 | Добавлени е товара  (позитивн ый) | 1.  Пользователь находится на странице добавления товара. 2.  Пользователь имеет права администрат ора. | 1. Ввести в поля «Имя товара»,  «Описание»,  «Категория»,  «Цена»,  «Остаток». 2.  Нажать кнопку  «Добавить товар». | Товар успешно добавлен, появляется сообщение: «Товар успешно добавлен». | Товар успешно добавлен, появляется сообщение: «Товар успешно добавлен». |
| 6 | Добавлени е товара (негативн ый) | 1.  Пользователь находится на странице добавления | 1. Оставить одно из обязательных полей пустым. 2.  Нажать кнопку  «Добавить товар». | Появляется сообщение об ошибке:  «Пожалуйста, | Появляется сообщение об ошибке:  «Пожалуйста, |
| 7 |  | товара. 2.  Пользователь имеет права администрат ора. |  | заполните все поля». | заполните все поля». |

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта была разработана информационная система «Цех предприятия», которая предназначена для автоматизации управления производственными процессами в рамках цеха. Основной целью проекта было создание программного модуля, обеспечивающего управление базой данных производственного процесса, а также взаимодействие между различными подразделениями цеха. Для достижения этой цели были решены следующие задачи: анализ существующих систем автоматизации управления производством, определение требований к функциональным возможностям разрабатываемого модуля, разработка архитектуры программного модуля и базы данных, реализация программного модуля с использованием современных технологий программирования, а также тестирование разработанного модуля и оценка его эффективности.

В главе «анализ предметной области и требования к программному обеспечению» проводится анализ предметной области, связанной с управлением складом цеха производства. Основные пользователи системы — администраторы склада, работники цеха, производственные бригады и бухгалтерия. Система обеспечивает учет поступления материалов, контроль их расхода, расчет остатков, формирование отчетов и анализ эффективности использования ресурсов. Проводится анализ существующих инструментов для создания системы, таких как ERP-системы (SAP S/4HANA, Oracle Cloud ERP) и MES-системы. Рассматриваются преимущества и недостатки этих систем, такие как высокая стоимость, сложность внедрения и необходимость обучения персонала. В качестве альтернативы предлагается разработка собственного программного модуля на языке Python.

В главе «Проектирование и разработка программы» описываются характеристики системы, включая производительность, безопасность, удобство использования, надежность и адаптивность. Определяются функциональные требования к модулю, такие как регистрация пользователей, управление товарами на складе, учет расходов, поиск товаров, обновление данных и анализ расходов. Разрабатываются сценарии использования системы, такие как добавление материала, добавление продукции и генерация отчетов. Описываются основные объекты предметной области, такие как пользователи, продукты, инвентаризация, траты, корзина, графики и отчеты. Разрабатывается структура базы данных с использованием SQLite3 и создаются ER-диаграммы для наглядного представления связей между объектами.

В главе «Практическая реализация модуля» описывается практическая реализация программного модуля. Используется среда разработки PyCharm, язык программирования Python и библиотека Tkinter для создания графического интерфейса. Реализуется пользовательский интерфейс с различными окнами для работы с пользователями, продуктами, инвентаризацией и отчетами. Описывается функциональность модуля, включая регистрацию пользователей, управление товарами, оформление заказов и анализ расходов. Реализуются модули для работы с базой данных, такие как `Admin\_menu.py`, `User\_menu.py`, `Userlogin.py` и `main.py`. Описывается реализация круговой диаграммы для визуализации распределения товаров на складе.

В главе «Тестирование приложения» проводится тестирование разработанного приложения. Описываются тесты для проверки основных функциональных возможностей системы, таких как авторизация, регистрация пользователей, добавление товаров, управление товарами на складе, проверка производительности, автоматизация отчетов, пользовательский интерфейс и анализ расходов. Все тесты успешно пройдены, что подтверждает работоспособность и эффективность разработанного программного модуля.

В заключение можно сказать, что выполненный курсовой проект успешно достиг поставленных целей и задач.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Нормативно-правовые источники

1. ГОСТ 7.32-2001. Межгосударственный стандарт. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научноисследовательской работе. Структура и правила оформления.
2. ГОСТ 7.0.5-2008. Национальный стандарт Российской Федерации. Система стандартов по информации, библиотечному и издательскому делу библиографическая ссылка.
3. ГОСТ 7.1-2003. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления.
4. ГОСТ 7.80-2000. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Заголовок. Общие требования и правила составления.

Учебники, учебные пособия, статьи

1. Laudon, K., Laudon, J. (2020). \*Management Information Systems:

Managing the Digital Firm\*. Pearson.

1. Hammer, M., Champy, J. (2006). \*Reengineering the Corporation: A Manifesto for Business Revolution\*. HarperBusiness.
2. Robey, D., Zmud, R.W. (1999). \*A Behavioral Theory of RD and Innovation\*. In \*Information Systems Research\*, 10(4), 307-335.
3. Lacity, M.C., Willcocks, L.P. (2016). \*Robotic Process Automation and Cognitive Automation: The Next Phase\*. SB Publishing.
4. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A. (2018).

\*Fundamentals of Business Process Management\*. Springer.

1. Turban, E., Sharda, R.,Delen, D. (2018). \*Decision Support and Business Intelligence Systems\*. Pearson.
2. Chaffey, D., Wood, S. (2019). \*Business Information Management:

Improving Performance Using Information Systems\*. Pearson.

1. Womack, J.P., Jones, D.T. (2003). \*Lean Thinking: Banish Waste and Create Wealth in Your Corporation\*. Simon Schuster.
2. Porter, M.E. (1985). \*Competitive Advantage: Creating and Sustaining Superior Performance\*. Free Press.
3. Мельников, А. В. (2020). \*Автоматизация производственных процессов: теория и практика\*. Москва: Издательство "Наука".
4. Кузнецов, И. И., Петрова, О. Н. (2019). \*Современные подходы к автоматизации управления на предприятиях\*. Санкт-Петербург: Издательство

"Бизнес-Пресса".

1. Баранов, Д. С. (2021). \*ERP-системы в управлении производственными процессами\*. Журнал "Управление производством",

12(3), 45-52.

Интернет-источники

1. UiPath. (2022). сайт: https://www.uipath.com
2. SAP SE. (2023). сайт: https://www.sap.com
3. JetBrains.com: сайт: https://www.jetbrains.com/ru-ru/pycharm/
4. WiKi: сайт: https://ru.wikipedia.org/wiki/PyCharm

Монография

GitHub: Курсовой проект по теме разработка программного модуля «цех производства» https://github.com/fossa2006/kursach Приложение 1

Addtional\_features.py

**1. Класс myentry (наследуется от tkinter.Entry)**

class myentry(tkinter.Entry):

def set\_completion\_list(self, completion\_list):

self.\_completion\_list = sorted(completion\_list, key=str.lower)

self.\_hits = []

self.\_hit\_index = 0

self.position = 0

self.bind('<KeyRelease>', self.handle\_keyrelease)

def autocomplete(self, delta=0):

if delta:

self.delete(self.position, tkinter.END)

else:

self.position = len(self.get())

\_hits = []

for element in self.\_completion\_list:

if element.lower().startswith(self.get().lower()):

\_hits.append(element)

if \_hits != self.\_hits:

self.\_hit\_index = 0

self.\_hits=\_hits

if \_hits == self.\_hits and self.\_hits:

self.\_hit\_index = (self.\_hit\_index + delta) % len(self.\_hits)

if self.\_hits:

self.delete(0,tkinter.END)

self.insert(0,self.\_hits[self.\_hit\_index])

self.select\_range(self.position,tkinter.END)

def handle\_keyrelease(self, event):

if event.keysym == "BackSpace":

self.delete(self.index(tkinter.INSERT), tkinter.END)

self.position = self.index(tkinter.END)

if event.keysym == "Left":

if self.position < self.index(tkinter.END):

self.delete(self.position, tkinter.END)

else:

self.position = self.position-1

self.delete(self.position, tkinter.END)

if event.keysym == "Right":

self.position = self.index(tkinter.END)

if event.keysym == "Down":

self.autocomplete(1)

if event.keysym == "Up":

self.autocomplete(-1)

if len(event.keysym) == 1 or event.keysym in tkinter\_umlauts:

self.autocomplete()

**2. Класс mycombobox (наследуется от tkinter.ttk.Combobox)**

class mycombobox(tkinter.ttk.Combobox):

def set\_completion\_list(self, completion\_list):

self.\_completion\_list = sorted(completion\_list, key=str.lower)

self.\_hits = []

self.\_hit\_index = 0

self.position = 0

self.bind('<KeyRelease>', self.handle\_keyrelease)

self['values'] = self.\_completion\_list

def autocomplete(self, delta=0):

if delta:

self.delete(self.position, tkinter.END)

else:

self.position = len(self.get())

\_hits = []

for element in self.\_completion\_list:

if element.lower().startswith(self.get().lower()): # Match case insensitively

\_hits.append(element)

if \_hits != self.\_hits:

self.\_hit\_index = 0

self.\_hits=\_hits

if \_hits == self.\_hits and self.\_hits:

self.\_hit\_index = (self.\_hit\_index + delta) % len(self.\_hits)

if self.\_hits:

self.delete(0,tkinter.END)

self.insert(0,self.\_hits[self.\_hit\_index])

self.select\_range(self.position,tkinter.END)

def handle\_keyrelease(self, event):

if event.keysym == "BackSpace":

self.delete(self.index(tkinter.INSERT), tkinter.END)

self.position = self.index(tkinter.END)

if event.keysym == "Left":

if self.position < self.index(tkinter.END):

self.delete(self.position, tkinter.END)

else:

self.position = self.position-1

self.delete(self.position, tkinter.END)

if event.keysym == "Right":

self.position = self.index(tkinter.END)

if len(event.keysym) == 1:

self.autocomplete()

Admin\_menu.py  
**Класс Admin**

Класс Admin отвечает за управление административным интерфейсом, включая создание меню, таблиц, кнопок и обработку событий. Вот пример кода, связанного с классом Admin:

class Admin:

def \_\_init\_\_(self, mainw):

self.mainw = mainw

self.cur.execute("ALTER TABLE sales ADD COLUMN username VARCHAR(20)") # Добавляем поле username в таблицу sales

self.base.commit()

def admin\_mainmenu(self, a, b):

self.mainframe = LabelFrame(self.mainw, width=1200, height=145, bg="#f7f7f7")

self.mainframe.place(x=100, y=100)

mi = PhotoImage(file="images/accounts.png")

mi = mi.subsample(a, b)

self.accounts = Button(self.mainframe, text="Профили", font="roboto 11 bold", bd=5, image=mi, compound=TOP,

command=self.buildusertable)

self.accounts.image = mi

self.accounts.place(x=655, y=27)

# Другие кнопки и элементы интерфейса...

**2. Функции для работы с базой данных и интерфейсом**

Эти функции отвечают за взаимодействие с базой данных SQLite, создание таблиц, поиск данных, экспорт в Excel и другие операции. Вот пример кода, связанного с этими функциями:

def export\_to\_excel(self):

"""

Экспорт данных из таблицы products в Excel.

"""

try:

# Получаем данные из таблицы products

self.cur.execute("SELECT \* FROM products")

products = self.cur.fetchall()

# Если данных нет, выводим сообщение

if not products:

messagebox.showinfo("Информация", "Нет данных для экспорта")

return

# Создаем DataFrame из данных

columns = ["Product ID", "Product Name", "Description", "Category", "Price", "Stocks"]

df = pd.DataFrame(products, columns=columns)

# Сохраняем DataFrame в Excel

file\_path = "products\_export.xlsx"

df.to\_excel(file\_path, index=False)

messagebox.showinfo("Успешно", f"Данные успешно экспортированы в файл: {file\_path}")

except Exception as e:

messagebox.showerror("Ошибка", f"Ошибка при экспорте данных: {e}")

User\_menu.py

**1. Модуль tkinter**

from tkinter import ttk

from tkinter import \*

from tkinter import messagebox

# Пример создания кнопки

mi = PhotoImage(file="images/items.png")

mi = mi.subsample(a, b)

self.aitems = Button(self.mainframe, text="Материалы", bd=5, font="roboto 11 bold", bg="cyan", image=mi, compound=TOP, command=self.builditemtable)

self.aitems.image = mi

self.aitems.place(x=260, y=17)

# Пример создания Treeview (таблицы)

self.tree = ttk.Treeview(self.tableframe, columns=("Product ID", "Product Name", "Description", "Category", 'Price', 'Stocks'), selectmode="extended", height=18)

self.tree.column('#0', stretch=NO, minwidth=0, width=0)

self.tree.column('#1', stretch=NO, minwidth=0, width=100)

self.tree.heading('Product ID', text="id продукта", anchor=W)

self.tree.grid(row=1, column=0, sticky="W")

**2. Модуль sqlite3**

import sqlite3

# Пример подключения к базе данных

self.base = sqlite3.connect('database.db')

self.cur = self.base.cursor()

# Пример выполнения SQL-запроса

self.cur.execute("SELECT \* FROM products")

productlist = self.cur.fetchall()

# Пример вставки данных в базу данных

self.cur.execute("INSERT INTO sales VALUES (?,?,?,?,?,?)", (int(i[0]), int(self.invoice), int(i[1]), int(i[3]), i[5], i[6]))

self.base.commit()

Userlogin.py

**1. Модуль sqlite3**

import sqlite3

# Подключение к базе данных

self.base = sqlite3.connect("login.db")

self.cur = self.base.cursor()

# Создание таблицы, если она не существует

self.cur.execute("CREATE TABLE if not exists users ( username varchar (20),password varchar (20) NOT NULL,account\_type varchar ( 10 ) NOT NULL,PRIMARY KEY(username));")

# Проверка пользователя в базе данных

self.cur.execute("select \* from users where username=? and password=? ", (s, s1))

l = self.cur.fetchall()

# Добавление нового пользователя в базу данных

self.cur.execute("insert into users values(?,?,?)", (s, s1, 'USER'))

self.base.commit()

**2. Модуль tkinter**

python

Copy

from tkinter import \*

from tkinter import messagebox

from tkinter import ttk

# Создание окна

self.loginw = Tk()

self.loginw.title("Вход")

# Настройка размеров окна

width = 500

height = 600

screen\_width = self.loginw.winfo\_screenwidth()

screen\_height = self.loginw.winfo\_screenheight()

x = (screen\_width / 2) - (width / 2)

y = (screen\_height / 2) - (height / 2)

self.loginw.geometry("%dx%d+%d+%d" % (width, height, x, y))

# Создание виджетов

self.loginframe = LabelFrame(self.loginw, bg="#4267b2", height=400, width=300)

self.toplabel = Label(self.loginframe, fg="white", bg="#4267b2", anchor="center", text="Вход", font="Roboto 40 bold")

self.us = ttk.Entry(self.loginframe, width=20, textvariable=self.username, font="Roboto 14 ")

self.pa = ttk.Entry(self.loginframe, width=20, textvariable=self.password, font="Roboto 14 ")

self.signin = Button(self.loginframe, width=20, text="Войти", bg="lightblue2", fg="dimgray", command=self.checkuser, font="Roboto 14")

# Размещение виджетов

self.loginframe.place(x=103, y=95)

self.toplabel.place(x=75, y=25)

self.us.place(x=35, y=145, height=40)

self.pa.place(x=35, y=185, height=40)

self.signin.place(x=35, y=290)

# Обработка событий

self.loginw.bind('<Return>', self.checkuser)

self.us.bind('<Button-1>', self.onclick)

self.pa.bind('<Button-1>', self.onclick1)

**1. Основное окно приложения (Main класс)**

class Main(Login, Admin, User):

def \_\_init\_\_(self):

Login.\_\_init\_\_(self)

self.loginw.mainloop() # Запускаем окно входа

self.loginw.state('withdraw') # Скрываем окно входа

# Создаем основное окно только после успешного входа

self.mainw = Toplevel(bg="#FFFFFF")

width = 1400

height = 780

screen\_width = self.mainw.winfo\_screenwidth()

screen\_height = self.mainw.winfo\_screenheight()

x = (screen\_width / 2) - (width / 2)

y = (screen\_height / 2) - (height / 2)

self.mainw.geometry("%dx%d+%d+%d" % (width, height, x, y))

self.mainw.title("Инвентаризация")

self.mainw.resizable(0, 0)

self.mainw.protocol('WM\_DELETE\_WINDOW', self.\_\_Main\_del\_\_)

self.getdetails()

# Добавляем кнопки для смены языка

self.current\_language = 'ru' # Устанавливаем язык по умолчанию

self.create\_language\_buttons()

# Инициализация переменной для общей суммы трат

self.total\_sales\_label = None

def \_\_Main\_del\_\_(self):

if messagebox.askyesno("Выход", "Покинуть Инвентаризацию?") == True:

self.loginw.quit()

self.mainw.quit()

exit(0)

else:

pass

def getdetails(self):

self.cur.execute("CREATE TABLE if not exists products(product\_id varchar (20),product\_name varchar (50) NOT NULL,product\_desc varchar (50) NOT NULL,product\_cat varchar (50),product\_price INTEGER NOT NULL,stocks INTEGER NOT NULL,PRIMARY KEY(product\_id));")

self.cur.execute("CREATE TABLE if not exists sales (Trans\_id INTEGER,invoice INTEGER NOT NULL,Product\_id varchar (20),Quantity INTEGER NOT NULL,Date varchar (20),Time varchar (20),username VARCHAR(20),PRIMARY KEY(Trans\_id));")

self.cur.execute("select \* from products ")

self.products = self.cur.fetchall()

capuser = self.username.get()

capuser = capuser.upper()

self.cur.execute("select account\_type from users where username= ? ", (capuser,))

l = self.cur.fetchall()

self.account\_type = l[0][0]

self.buildmain()

def buildmain(self):

if self.account\_type == 'ADMIN':

super(Admin).\_\_init\_\_()

self.admin\_mainmenu(8, 8)

else:

super(User).\_\_init\_\_()

self.user\_mainmenu(8, 8)

self.logout.config(command=self.\_\_Main\_del\_\_)

self.changeuser.config(command=self.change\_user)

self.topframe = LabelFrame(self.mainw, width=1400, height=120, bg="#4267b2")

self.topframe.place(x=0, y=0)

self.store\_name = 'Логиститическая'

self.storelable = Label(self.topframe, text=self.store\_name + " Система цеха", bg="#4267b2", anchor="center")

self.storelable.config(font="Roboto 30 bold", fg="snow")

self.storelable.place(x=360, y=30)

mi = PhotoImage(file="images/myprofile.png")

mi = mi.subsample(4, 4)

self.myprofile = ttk.Label(self.topframe, text=(self.username.get()).capitalize(), image=mi, compound=TOP)

self.myprofile.image = mi

self.myprofile.place(x=1300, y=15)

# Добавляем кнопки для смены языка

self.create\_language\_buttons()

def create\_language\_buttons(self):

self.ru\_button = Button(self.topframe, text="RU", command=lambda: self.change\_language('ru'))

self.ru\_button.place(x=1200, y=10)

self.en\_button = Button(self.topframe, text="EN", command=lambda: self.change\_language('en'))

self.en\_button.place(x=1250, y=10)

# Добавляем кнопку "Экспорт в Excel"

self.export\_excel\_button = Button(self.topframe, text="Экспорт в Excel", command=self.export\_to\_excel)

self.export\_excel\_button.place(x=1050, y=10) # Укажите подходящее место для кнопки

def change\_language(self, lang):

self.current\_language = lang

self.translate\_interface()

def translate\_interface(self):

translations = {

'ru': {

'Инвентаризация': 'Инвентаризация',

'Выйти': 'Выйти',

'Сменить пользователя': 'Сменить пользователя',

'Профили': 'Профили',

'добавить': 'добавить',

'Инвентаризация': 'Инвентаризация',

'Траты': 'Траты',

'Поиск': 'Поиск',

'Сбросить': 'Сбросить',

'Обновить': 'Обновить',

'Удалить': 'Удалить',

'Добавить товар': 'Добавить товар',

'Назад': 'Назад',

'id продукта': 'id продукта',

'Имя продукта': 'Имя продукта',

'Описание': 'Описание',

'Категория': 'Категория',

'Цена': 'Цена',

'Остаток': 'Остаток',

'Добавить': 'Добавить',

'Кол-во': 'Кол-во',

'Сумма к оплате': 'Сумма к оплате',

'Номер счета': 'Номер счета',

'id оплаты': 'id оплаты',

'Название продукта': 'Название продукта',

'Дата': 'Дата',

'Время': 'Время',

'Пользователь с таким именем уже существует!': 'Пользователь с таким именем уже существует!',

'Неизвестный тип аккаунта!': 'Неизвестный тип аккаунта!',

'Данные пользователя успешно обновлены!': 'Данные пользователя успешно обновлены!',

'Вы хотите удалить этот профиль?': 'Вы хотите удалить этот профиль?',

'Вы хотите удалить товар из инвентаризации?': 'Вы хотите удалить товар из инвентаризации?',

'Пожалуйста, заполните все поля': 'Пожалуйста, заполните все поля',

'Неверные данные': 'Неверные данные',

'Товар успешно добавлен': 'Товар успешно добавлен',

'Транзакция успешно завершена!': 'Транзакция успешно завершена!',

'Корзина пуста!': 'Корзина пуста!',

'Вы хотите продолжить?': 'Вы хотите продолжить?',

'Удалить корзину?': 'Удалить корзину?',

'Распечатать эту транзакцию?': 'Распечатать эту транзакцию?',

'Товар с таким описанием уже существует!': 'Товар с таким описанием уже существует!',

'Product ID должен быть уникальным': 'Product ID должен быть уникальным',

'Product ID должен быть числом': 'Product ID должен быть числом',

'Товар с таким количеством недоступен!': 'Товар с таким количеством недоступен!',

'Товар отсутствует в наличии!': 'Товар отсутствует в наличии!',

'Неверное количество!': 'Неверное количество!',

'Не выбрана корзина': 'Не выбрана корзина',

'Вы хотите сменить пользователя?': 'Вы хотите сменить пользователя?',

'Выход': 'Выход',

'Пользователь зарегистрирован': 'Пользователь зарегистрирован',

'Неверное имя пользователя или пароль': 'Неверное имя пользователя или пароль',

'Имя пользователя уже существует': 'Имя пользователя уже существует',

'Выберите имя пользователя': 'Выберите имя пользователя',

'Создайте пароль': 'Создайте пароль',

'Назад': 'Назад',

'ОК': 'ОК',

'Внимание!': 'Внимание!',

'Внимание': 'Внимание',

'Вы хотите удалить текущего пользователя?': 'Вы хотите удалить текущего пользователя?',

'Удалить текущего пользователя?': 'Удалить текущего пользователя?',

'Транзакция успешно завершена': 'Транзакция успешно завершена',

'Диаграмма': 'Диаграмма', # Добавлено для русского языка

},

'en': {

'Инвентаризация': 'Inventory',

'Выйти': 'Logout',

'Сменить пользователя': 'Change User',

'Профили': 'Profiles',

'добавить': 'Add',

'Инвентаризация': 'Inventory',

'Траты': 'Expenses',

'Поиск': 'Search',

'Сбросить': 'Reset',

'Обновить': 'Update',

'Удалить': 'Delete',

'Добавить товар': 'Add Item',

'Назад': 'Back',

'id продукта': 'Product ID',

'Имя продукта': 'Product Name',

'Описание': 'Description',

'Категория': 'Category',

'Цена': 'Price',

'Остаток': 'Stock',

'Добавить': 'Add',

'Кол-во': 'Quantity',

'Сумма к оплате': 'Total Amount',

'Номер счета': 'Invoice Number',

'id оплаты': 'Transaction ID',

'Название продукта': 'Product Name',

'Дата': 'Date',

'Время': 'Time',

'Пользователь с таким именем уже существует!': 'User with this name already exists!',

'Неизвестный тип аккаунта!': 'Unknown account type!',

'Данные пользователя успешно обновлены!': 'User data updated successfully!',

'Вы хотите удалить этот профиль?': 'Do you want to delete this profile?',

'Вы хотите удалить товар из инвентаризации?': 'Do you want to delete the item from inventory?',

'Пожалуйста, заполните все поля': 'Please fill in all fields',

'Неверные данные': 'Invalid data',

'Товар успешно добавлен': 'Item added successfully',

'Транзакция успешно завершена!': 'Transaction completed successfully!',

'Корзина пуста!': 'Cart is empty!',

'Вы хотите продолжить?': 'Do you want to continue?',

'Удалить корзину?': 'Delete cart?',

'Распечатать эту транзакцию?': 'Print this transaction?',

'Товар с таким описанием уже существует!': 'Item with this description already exists!',

'Product ID должен быть уникальным': 'Product ID must be unique',

'Product ID должен быть числом': 'Product ID must be a number',

'Товар с таким количеством недоступен!': 'Item with this quantity is not available!',

'Товар отсутствует в наличии!': 'Item is out of stock!',

'Неверное количество!': 'Invalid quantity!',

'Не выбрана корзина': 'No cart selected',

'Вы хотите сменить пользователя?': 'Do you want to change the user?',

'Выход': 'Exit',

'Пользователь зарегистрирован': 'User registered',

'Неверное имя пользователя или пароль': 'Invalid username or password',

'Имя пользователя уже существует': 'Username already exists',

'Выберите имя пользователя': 'Choose a username',

'Создайте пароль': 'Create a password',

'Назад': 'Back',

'ОК': 'OK',

'Внимание!': 'Attention!',

'Внимание': 'Attention',

'Вы хотите удалить текущего пользователя?': 'Do you want to delete the current user?',

'Удалить текущего пользователя?': 'Delete current user?',

'Транзакция успешно завершена': 'Transaction completed successfully',

'Диаграмма': 'Chart', # Добавлено для английского языка

}

}

# Обновляем текст всех элементов интерфейса

self.storelable.config(text=translations[self.current\_language]['Инвентаризация'])

self.logout.config(text=translations[self.current\_language]['Выйти'])

self.changeuser.config(text=translations[self.current\_language]['Сменить пользователя'])

self.accounts.config(text=translations[self.current\_language]['Профили'])

self.items.config(text=translations[self.current\_language]['добавить'])

self.stocks.config(text=translations[self.current\_language]['Инвентаризация'])

self.sales.config(text=translations[self.current\_language]['Траты'])

self.searchbut.config(text=translations[self.current\_language]['Поиск по Описанию'])

self.resetbut.config(text=translations[self.current\_language]['Сбросить'])

self.pie\_chart\_button.config(text=translations[self.current\_language]['Диаграмма']