Министерство образования и науки Российской Федерации

**Муромский институт (филиал)**

федерального государственного бюджетного образовательного учреждения

высшего профессионального образования

**«Владимирский государственный университет**

**имени Александра Григорьевича и Николая Григорьевича Столетовых»**

(МИ (филиал) ВлГУ)

Факультет ИТР

Кафедра ФПМ

*КУРСОВАЯ РАБОТА*

по дисциплине База Данных

Тема Справочная система товаров по торговой фирме

Руководитель

(оценка)  (уч. степень, звание)

Колпаков А.А.

(подпись) (дата)

Члены комиссии Студент ПМИ-122

(группа)

Касаткин Д.В.

(подпись) (дата)

(подпись) (дата) (подпись) (дата)

Муром 2024 г.

В данной работе представлена концепция и разработка справочной системы товаров для торговой фирмы. Приложение разработано с использованием Python и Tkinter, а для хранения данных используется СУБД SQLite. Целью работы является проектирование и реализация системы для эффективного управления информацией о товарах, поставках и взаимодействии с фирмами.

В рамках проекта была разработана база данных, содержащая таблицы для хранения информации о товарах, фирмах, магазинах, а также связи между ними. Программа обеспечивает добавление, просмотр, редактирование и поиск товаров, а также формирование отчетов о поставках и взаимодействии с фирмами. Проведены тестирование и отладка программы.

This work presents the concept and development of a product information system for a trading company. The application was developed using Python and Tkinter, and the SQLite database management system is used to store the data. The aim of this work is to design and implement a system for efficient management of product information, deliveries, and interactions with firms.

Within the project, a database was designed, containing tables for storing information about products, firms, stores, and their relationships. The program provides functionalities for adding, viewing, editing, and searching products, as well as generating reports on deliveries and interactions with firms. Testing and debugging of the program were conducted.

Содержание

[Введение 6](#_Toc187387942)

[1. Анализ технического задания 7](#_Toc187387943)

[2. Разработка моделей данных. 9](#_Toc187387944)

[2.1 Логическая модель 9](#_Toc187387945)

[2.2 Концептуальная модель 10](#_Toc187387946)

[2.3 Физическая модель 11](#_Toc187387947)

[3. Разработка и реализация ИС. 13](#_Toc187387948)

[3.1 Создание базы данных 13](#_Toc187387949)

[3.2 Разработка SQL – запросов для Справочной системы по товарам в торговой фирме. 15](#_Toc187387950)

[3.3 Разработка клиентской части 23](#_Toc187387951)

[3.4 Руководство программисту 28](#_Toc187387952)

[3.5 Руководство пользователя 35](#_Toc187387953)

[4. Тестирование ИС. 40](#_Toc187387954)

[Заключение 42](#_Toc187387955)

[Список используемой литературы 43](#_Toc187387956)

[Приложение 1. Модели данных. 44](#_Toc187387957)

[Приложение 2. Скриншоты приложения и ссылка на репозиторий с кодом 46](#_Toc187387958)

Введение.

В современном бизнесе эффективное управление информацией о товарах является ключевым фактором успеха торговых фирм. Автоматизированные информационные системы (АИС) позволяют оптимизировать процессы хранения, обработки и поиска данных о товарах, тем самым повышая общую эффективность работы предприятия.

Справочная система товаров предназначена для решения задач учета, анализа и контроля товаров торговой фирмы, позволяя менеджерам оперативно получать необходимую информацию, а также формировать отчетность по поставкам и взаимодействию с фирмами-поставщиками.

В рамках данной работы была разработана справочная система товаров, предназначенная для автоматизации процессов учета и управления информацией о товарах торговой фирмы.

Целью курсовой работы является создание информационной системы для добавления, просмотра, редактирования, поиска товаров, а также формирования отчетов о поставках и взаимодействиях с фирмами.

Задачи, работающие для этой цели:

1. Анализ предметной области и выделение ключевых сущностей.

2) Проектирование логической и физической моделей базы данных.

3) Реализация физической модели в СУБД SQLite.

4) Разработка пользовательского интерфейса на Python с использованием Tkinter.

5) Создание функций для добавления, просмотра, редактирования, поиска товаров, а также формирования отчетов.

6) Проведение тестирования и отладки системы.

1. Анализ технического задания.

Предметной областью для данной курсовой работы является торговая фирма-посредник, которая занимается поставками различных товаров от разных фирм в разные магазины. Основные бизнес-процессы включают в себя:

1. Управление каталогом товаров: добавление новых товаров, редактирование информации, поиск, удаление.
2. Управление поставщиками: учет фирм-поставщиков, их контактных данных.
3. Управление магазинами: учет торговых точек, их адресов.
4. Учет поставок: фиксация дат поставок, объемов, цен, связи с фирмами и магазинами.
5. Формирование отчетности: получение отчетов о поставках за периоды, по фирмам, по магазинам. Исходя из анализа предметной области, можно выделить следующие ключевые сущности:
6. **Товар (Product)**: Название, категория, характеристики, цена, количество, единица измерения, дата поступления, магазин, скидка, НДС.
7. **Фирма (Firm)**: Название, контактные данные.
8. **Магазин (Store)**: Название, адрес.
9. **Связи**: Связи между товарами и фирмами, товарами и магазинами. Для разработки базы данных была выбрана СУБД SQLite, которая позволяет хранить данные в файле и не требует настройки сервера. Для разработки пользовательского интерфейса был выбран Python с использованием Tkinter.

1. Разработка моделей данных.

На данном этапе выполняется проектирование логической модели данных, которая определяет сущности, их атрибуты и связи. На основе логической модели создаются функциональные модели системы, диаграммы потоков данных и, в конечном итоге, физическая модель базы данных.

2.1 Логическая модель.

Логическая модель начинается с выделения основных сущностей. Из анализа предметной области выделены следующие сущности: “Товар”, “Фирма” и “Магазин”. Для сущности “Товар” были выделены атрибуты: - Название товара (идентификатор) - Категория - Характеристики - Цена - Количество - Единица измерения - Дата поступления - Магазин - Скидка - НДС Для сущности “Фирма” атрибуты: - Название фирмы (идентификатор) - Контактные данные Для сущности “Магазин” атрибуты: - Название магазина (идентификатор) - Адрес Связи между сущностями: - “Товар” имеет связь “многие ко многим” с “Фирмой” (товар может поставляться разными фирмами, и фирма может поставлять разные товары). - “Товар” имеет связь “многие к одному” с “Магазином”.

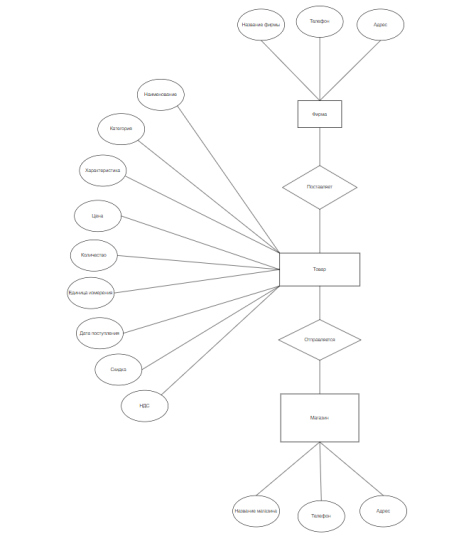


Рис. – Логическая модель

2.2 Концептуальная модель.

**Многие-ко-многим**: Связь между Products и Firms реализована через промежуточную таблицу Product\_firm. Это означает, что для каждого товара может существовать несколько записей в Product\_firm (связанных с разными фирмами), и для каждой фирмы может существовать несколько записей в Product\_firm (связанных с разными товарами).

**Один-ко-многим**: Связь между Stores и Products означает, что каждый магазин может содержать много товаров, но каждый товар связан только с одним конкретным магазином (однако один и тот же товар может присутствовать в разных магазинах, но каждая копия товара в Products связана с одним магазином через store\_name).

Таким образом, мы описали, как каждая сущность (таблица) связана с другими и как эти связи реализованы с использованием первичных и внешних ключей. Это важно для понимания структуры базы данных и для работы с ней.

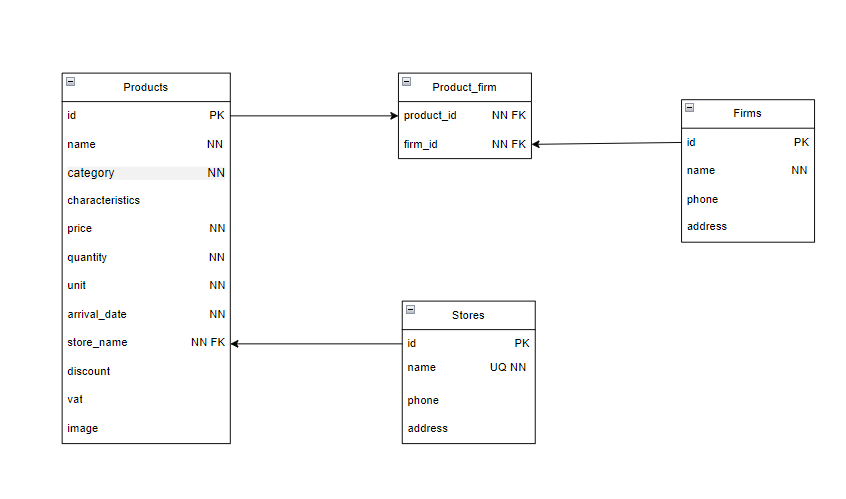


Рис – Концептуальная модель

2.3 Физическая модель.

Процесс разработки физической модели базы данных включает в себя несколько последовательных шагов, направленных на создание структуры для хранения и организации данных. Основные этапы этого процесса следующие:

1. Анализ требований: На этом этапе осуществляется сбор и анализ требований к базе данных, что включает определение сущностей и их атрибутов,

2. Физическая реализация: Здесь создается физическая структура базы данных, включая выбор типов данных, определение размеров полей и индексов, а также оптимизацию производительности.

3. Тестирование и оптимизация: После реализации физической модели базы данных проводится тестирование её функционирования и оптимизация для обеспечения эффективной работы.

* 1. Эти основные этапы помогают разработать физическую модель базы данных, которая будет эффективно хранить и обрабатывать данные.

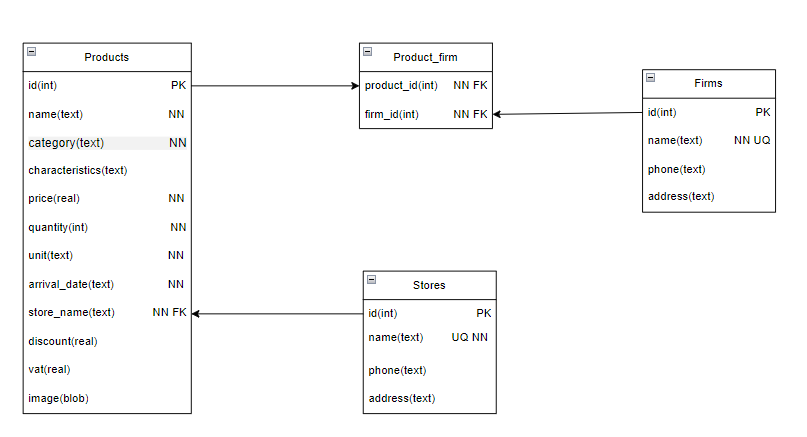


Рис – Физическая модель

В результате разработки этих моделей достигается понимание оптимизации структуры данных и обеспечение эффективного хранения и обработки информации. Каждая модель играет свою роль в процессе создания базы данных, начиная с общего представления данных и заканчивая их конкретной реализацией.

3. Разработка и реализация ИС.

Данный раздел описывает процесс разработки и реализации автоматизированной информационной системы (АИС) для управления товарами в торговой фирме. В рамках проекта используется язык программирования Python с графической библиотекой Tkinter для создания пользовательского интерфейса, а также СУБД SQLite для хранения данных.

3.1 Создание базы данных.

Для хранения данных используется реляционная база данных SQLite, которая позволяет хранить всю информацию в одном файле (trade\_firm.db). Для создания структуры базы данных используются следующие таблицы:

* **products (Товары):** Хранит информацию о товарах, включая их наименование, категорию, характеристики, цену, количество, единицу измерения, дату поступления, название магазина, скидку, НДС и изображение.
* **firms (Фирмы):** Хранит информацию о фирмах-поставщиках, включая их название, телефон и адрес.
* **stores (Магазины):** Хранит информацию о магазинах, включая их название, телефон и адрес.
* **product\_firm (Связь товаров и фирм):** Промежуточная таблица для реализации связи “многие-ко-многим” между товарами и фирмами.

Типы полей для таблиц представлены ниже (Рис. 1 - 4).

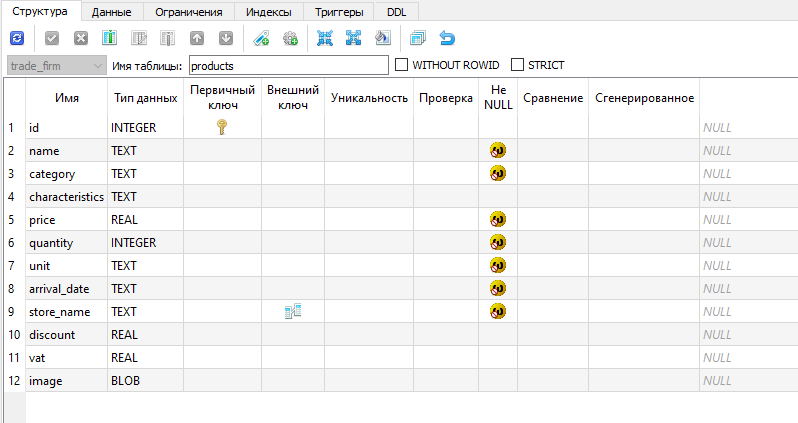


Рис. 1 – Таблица Товары.

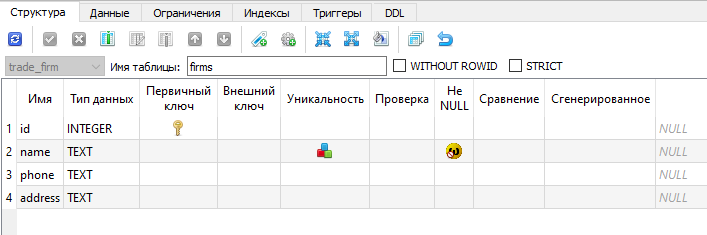


Рис. 2 – Таблица Фирмы.

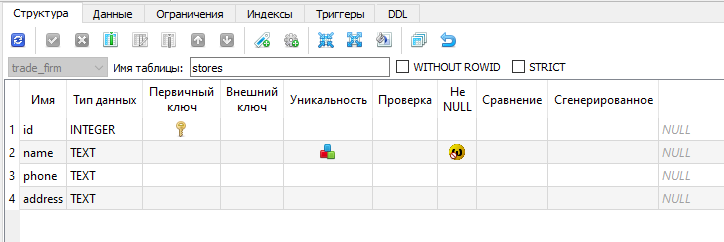


Рис. 3 – Таблица Магазины.

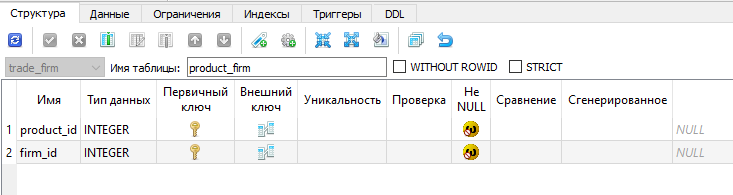


Рис. 4 – Таблица Связь товаров и фирм.

## 3.2 Разработка SQL – запросов для Справочной системы по товарам в торговой фирме.

Для создания сущностей и атрибутов к ним используется запрос.

**Запросы для создания таблиц:**

CREATE TABLE IF NOT EXISTS products (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT NOT NULL,

category TEXT NOT NULL,

characteristics TEXT,

price REAL NOT NULL,

quantity INTEGER NOT NULL,

unit TEXT NOT NULL,

arrival\_date TEXT NOT NULL,

store\_name TEXT NOT NULL,

discount REAL,

vat REAL,

image BLOB,

FOREIGN KEY (store\_name) REFERENCES stores(name)

)

CREATE TABLE IF NOT EXISTS firms (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT NOT NULL UNIQUE,

phone TEXT,

address TEXT

)

CREATE TABLE IF NOT EXISTS stores (

id INTEGER PRIMARY KEY AUTOINCREMENT,

name TEXT NOT NULL UNIQUE,

phone TEXT,

address TEXT

)

**Запросы для вставки данных:**

cursor.executemany("INSERT OR IGNORE INTO firms (name, phone, address) VALUES (?, ?, ?)", firms)

# Вставляет начальные данные в таблицу `firms` (если таких записей нет).

# Параметры: `firms` — список кортежей с данными для каждой фирмы (название, телефон, адрес).

cursor.executemany("INSERT OR IGNORE INTO stores (name, phone, address) VALUES (?, ?, ?)", stores)

# Вставляет начальные данные в таблицу `stores` (если таких записей нет).

# Параметры: `stores` — список кортежей с данными для каждого магазина (название, телефон, адрес).

'''

INSERT INTO products (name, category, characteristics, price, quantity, unit, arrival\_date, store\_name, discount, vat, image)

VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)

''', (name, category, characteristics, price, quantity, unit, arrival\_date, store\_name, discount, vat, image)

)

# Вставляет данные о новом товаре в таблицу `products`.

# Параметры: Данные о товаре.

**Запросы для связи товаров и фирм:**

cursor.execute("SELECT id FROM products WHERE name = ?", (product\_name,))

# Получает `id` товара из таблицы `products` по его имени.

# Параметры: `product\_name` — наименование товара.

cursor.execute("SELECT id FROM firms WHERE name = ?", (firm\_name,))

# Получает `id` фирмы из таблицы `firms` по ее имени.

# Параметры: `firm\_name` — наименование фирмы.

cursor.execute(

'''

INSERT OR IGNORE INTO product\_firm (product\_id, firm\_id)

VALUES (?, ?)

''', (product\_id, firm\_id)

)

# Вставляет запись в таблицу `product\_firm` для создания связи между товаром и фирмой.

# Параметры: `product\_id`, `firm\_id`.

**Запросы выборки данных:**

cursor.execute('SELECT \* FROM products')

# Выбирает все записи (все столбцы) из таблицы `products`.

cursor.execute('SELECT \* FROM firms')

# Выбирает все записи (все столбцы) из таблицы `firms`.

cursor.execute('SELECT \* FROM stores')

# Выбирает все записи (все столбцы) из таблицы `stores`.

**Запросы для отчётов:**

cursor.execute(

'''

SELECT

p.name, p.category, p.quantity, p.price,

p.store\_name, p.arrival\_date, f.name

FROM products AS p

JOIN product\_firm AS pf ON pf.product\_id = p.id

JOIN firms AS f ON f.id = pf.firm\_id

WHERE p.arrival\_date BETWEEN ? AND ?

''',

(start\_date, end\_date)

)

# Формирует отчет об объеме поставок за период.

# Выбирает наименование товара, категорию, количество, цену, название магазина, дату поступления и название фирмы.

# Использует JOIN для связи таблиц `products`, `product\_firm` и `firms`.

# Фильтрует результаты по дате поступления.

# Параметры: `start\_date` и `end\_date`.

cursor.execute(

'''

SELECT

p.name, p.category, p.quantity, p.price,

p.arrival\_date, f.name

FROM products AS p

JOIN product\_firm AS pf ON pf.product\_id = p.id

JOIN firms AS f ON f.id = pf.firm\_id

WHERE p.store\_name = ?

''',

(store\_name,)

)

# Формирует отчет об объеме поставок для заданного магазина.

# Выбирает наименование товара, категорию, количество, цену, дату поступления и название фирмы.

# Использует JOIN для связи таблиц `products`, `product\_firm` и `firms`.

# Фильтрует результаты по названию магазина.

# Параметры: `store\_name`.

cursor.execute(

'''

SELECT

p.name, p.category, p.quantity, p.price,

p.arrival\_date, s.name

FROM products AS p

JOIN product\_firm AS pf ON pf.product\_id = p.id

JOIN firms AS f ON f.id = pf.firm\_id

JOIN stores AS s ON s.name = p.store\_name

WHERE f.name = ?

''',

(firm\_name,)

)

# Формирует отчет об объеме поставок для заданной фирмы.

# Выбирает наименование товара, категорию, количество, цену, дату поступления, и название магазина.

# Использует JOIN для связи таблиц `products`, `product\_firm` и `firms` и `stores`.

# Фильтрует результаты по названию фирмы.

# Параметры: `firm\_name`.

**Запросы для работы с изображениями:**

cursor.execute(

"UPDATE products SET image = ? WHERE name = ?", (image\_data, product\_name)

)

# Обновляет поле `image` у выбранного товара.

# Параметры: `image\_data` - бинарные данные изображения, `product\_name` - имя товара.

cursor.execute("SELECT image FROM products WHERE name = ?", (product\_name,))

# Выбирает данные изображения из таблицы `products` по названию товара.

# Параметры: `product\_name` — наименование товара.

**Запросы для поиска:**

cursor.execute("CALL GetProductsByCategory(?)", (category,))

# Вызывает хранимую процедуру GetProductsByCategory для получения товаров по категории.

# Параметры: `category`.

cursor.execute("SELECT \* FROM products WHERE name LIKE ?", ('%' + name + '%',))

# Выбирает все товары из таблицы products, у которых название похоже на введенное имя

# Параметры: `name`.

**Запросы для создания триггера и хранимой процедуры:**

cursor.execute('''

CREATE TRIGGER IF NOT EXISTS update\_product\_arrival\_date

AFTER UPDATE ON products

BEGIN

UPDATE products

SET arrival\_date = datetime('now')

WHERE id = NEW.id

AND NEW.quantity != OLD.quantity;

END;

''')

# Создает триггер `update\_product\_arrival\_date`, который автоматически обновляет дату поступления товара, при обновлении его количества

# Триггер выполняется после обновления записи в таблице `products`.

cursor.execute('''

CREATE PROCEDURE IF NOT EXISTS GetProductsByCategory (category\_name TEXT)

BEGIN

SELECT \* FROM products WHERE category = category\_name;

END;

''')

# Создает хранимаую процедуру для получения товаров по категории

# Параметры: category\_name

**Запрос для перемещения товара:**

cursor.execute("UPDATE products SET store\_name = ? WHERE name = ?", (new\_store\_name, product\_name))

# Обновляет поле `store\_name` у выбранного товара.

# Параметры: `new\_store\_name`, `product\_name`.

## 3.3 Разработка клиентской части

Интерфейс представляет собой способ взаимодействия между человеком и компьютерной системой, программой или устройством. Он включает в себя набор элементов, позволяющих пользователю взаимодействовать с приложением или устройством, выполнять определённые действия и получать нужную информацию.

Удобный интерфейс имеет решающее значение для пользовательского опыта, так как делает использование приложения более интуитивным и приятным. Важно, чтобы интерфейс был простым, понятным и легким в освоении, чтобы пользователи могли быстро и эффективно достигать своих целей. Хорошо продуманный интерфейс также помогает уменьшить количество ошибок и улучшает общее впечатление от приложения. Поэтому необходимо уделять особое внимание созданию интуитивно понятного и удобного интерфейса для обеспечения положительного опыта пользователей. При этом важно, чтобы интерфейс соответствовал необходимому функционалу, что также играет значительную роль в разработке приложения.

Исходя из всего вышесказанного, был разработан графический интерфейс пользователя:

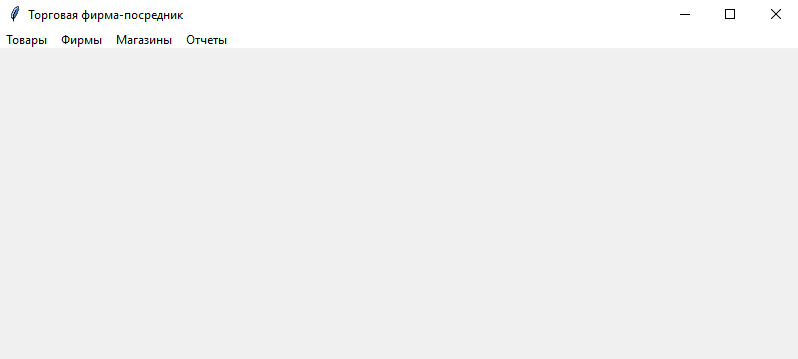


Рис. 1 – главное окно приложения

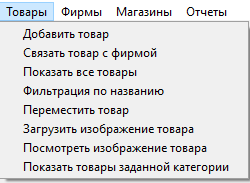


Рис. 2 – интерфейс окна товары в меню



Рис. 3 – интерфейс окна фирмы в меню



Рис. 4 – интерфейс окна магазины в меню

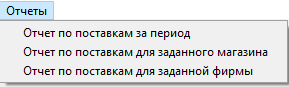


Рис. 5 – интерфейс окна отчёты в меню

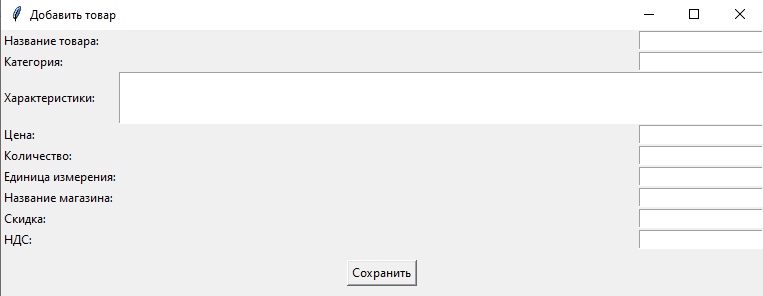


Рис. 6 - интерфейс окна для добавления товара

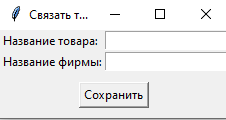


Рис. 7 - интерфейс окна для связи товара с фирмой

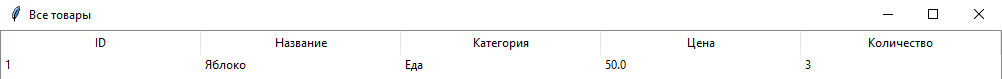


Рис. 8 - интерфейс окна для просмотра всех товаров

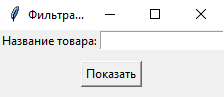


Рис. 9 - интерфейс окна для фильтрации товаров по названию

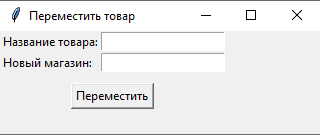


Рис. 10 - интерфейс окна для перемещения товара

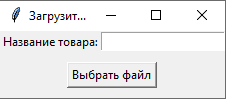


Рис. 11 - интерфейс окна для загрузки изображения товара

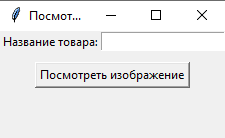


Рис. 12 - интерфейс окна для просмотра изображения товара

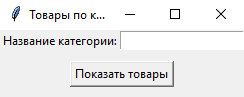


Рис. 13 - интерфейс окна для просмотра товара по заданной категории

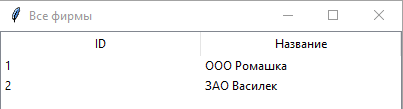


Рис. 14 - интерфейс окна для просмотра всех фирм

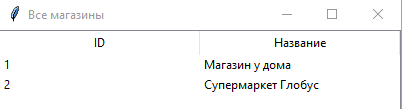


Рис. 15 - интерфейс окна для просмотра всех магазинов

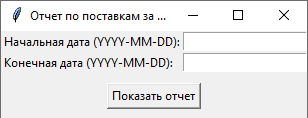


Рис. 16 - интерфейс окна для просмотра отчёта по поставкам за определённый период

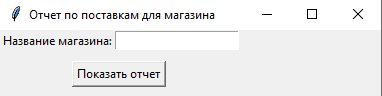


Рис. 17 - интерфейс окна для просмотра отчёта по поставкам для определённого магазина

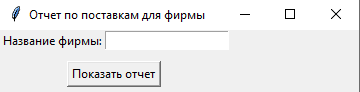


Рис. 18 - интерфейс окна для просмотра отчёта по поставкам для определённой фирмы

## 3.4 Руководство программисту

Модуль db\_operations.py (Работа с базой данных):

create\_tables():

Описание: Функция создает таблицы в базе данных SQLite (trade\_firm.db), если они еще не существуют.

Таблицы: products, firms, stores, product\_firm.

Входные параметры: Нет.

Возвращаемое значение: Нет.

create\_initial\_data():

Описание: Функция заполняет таблицы firms и stores начальными данными, если они пусты.

Входные параметры: Нет.

Возвращаемое значение: Нет.

insert\_product(name, category, characteristics, price, quantity, unit, arrival\_date, store\_name, discount, vat, image):

Описание: Функция добавляет новый товар в таблицу products.

Входные параметры:

name: Название товара (строка, обязательный).

category: Категория товара (строка, обязательный).

characteristics: Характеристики товара (строка, опционально).

price: Цена товара (число с плавающей точкой, обязательный).

quantity: Количество товара (целое число, обязательный).

unit: Единица измерения товара (строка, обязательный).

arrival\_date: Дата поступления товара (строка, обязательный).

store\_name: Название магазина (строка, обязательный).

discount: Скидка на товар (число с плавающей точкой, опционально).

vat: НДС на товар (число с плавающей точкой, опционально).

image: Изображение товара (бинарные данные, опционально).

Возвращаемое значение: Строка: “Товар успешно добавлен” в случае успеха, или сообщение об ошибке.

insert\_product\_firm(product\_name, firm\_name):

Описание: Функция добавляет связь между товаром и фирмой в таблицу product\_firm.

Входные параметры:

product\_name: Название товара (строка, обязательный).

firm\_name: Название фирмы (строка, обязательный).

Возвращаемое значение: Строка: “Связь между товаром и фирмой добавлена” в случае успеха, или сообщение об ошибке.

get\_all\_products():

Описание: Функция возвращает список всех товаров из таблицы products.

Входные параметры: Нет.

Возвращаемое значение: Список кортежей, где каждый кортеж — это запись о товаре, или сообщение об ошибке.

get\_all\_firms():

Описание: Функция возвращает список всех фирм из таблицы firms.

Входные параметры: Нет.

Возвращаемое значение: Список кортежей, где каждый кортеж — это запись о фирме, или сообщение об ошибке.

get\_all\_stores():

Описание: Функция возвращает список всех магазинов из таблицы stores.

Входные параметры: Нет.

Возвращаемое значение: Список кортежей, где каждый кортеж — это запись о магазине, или сообщение об ошибке.

get\_delivery\_report(start\_date, end\_date):

Описание: Функция возвращает отчет о поставках за период (по всем магазинам).

Входные параметры:

start\_date: Начальная дата (строка в формате YYYY-MM-DD, обязательный).

end\_date: Конечная дата (строка в формате YYYY-MM-DD, обязательный).

Возвращаемое значение: Список кортежей (товар, категория, количество, цена, магазин, дата, фирма), или сообщение об ошибке.

get\_delivery\_report\_by\_store(store\_name):

Описание: Функция возвращает отчет о поставках для заданного магазина.

Входные параметры:

store\_name: Название магазина (строка, обязательный).

Возвращаемое значение: Список кортежей (товар, категория, количество, цена, дата, фирма), или сообщение об ошибке.

get\_delivery\_report\_by\_firm(firm\_name):

Описание: Функция возвращает отчет о поставках для заданной фирмы.

Входные параметры:

firm\_name: Название фирмы (строка, обязательный).

Возвращаемое значение: Список кортежей (товар, категория, количество, цена, дата, магазин), или сообщение об ошибке.

load\_image\_to\_db(product\_name, image\_path):

Описание: Функция загружает изображение в поле image товара в таблице products.

Входные параметры:

product\_name: Название товара (строка, обязательный).

image\_path: Путь к файлу изображения (строка, обязательный).

Возвращаемое значение: Строка: “Изображение успешно загружено” в случае успеха, или сообщение об ошибке.

get\_image\_from\_db(product\_name):

Описание: Функция возвращает изображение товара из поля image таблицы products.

Входные параметры:

product\_name: Название товара (строка, обязательный).

Возвращаемое значение: Бинарные данные изображения, или None если не найдено, или сообщение об ошибке.

get\_products\_by\_category(category):

Описание: Функция возвращает список товаров, соответствующих заданной категории

Входные параметры:

category: Название категории (строка, обязательный).

Возвращаемое значение: Список кортежей (товаров), или сообщение об ошибке.

get\_products\_by\_name(name):

Описание: Функция возвращает список товаров, чье название похоже на переданное

Входные параметры: \* name: Название товара (строка, обязательный).

Возвращаемое значение: Список кортежей (товаров), или сообщение об ошибке.

create\_update\_arrival\_date\_trigger():

Описание: Функция создает триггер update\_product\_arrival\_date в базе данных, который автоматически обновляет дату поступления товара при изменении его количества.

Входные параметры: Нет.

Возвращаемое значение: “Триггер для обновления даты поступления товара успешно создан”, или сообщение об ошибке.

create\_get\_products\_by\_category\_proc(): \* Описание: Функция создает хранимую процедуру для выборки товаров по категории. \* Входные параметры: Нет. \* Возвращаемое значение: “Хранимая процедура для получения товаров по категории успешно создана”, или сообщение об ошибке.

move\_product(product\_name, new\_store\_name): \* Описание: Функция перемещает товар в другой магазин, обновляя поле store\_name \* Входные параметры: \* product\_name: Название товара (строка, обязательный). \* new\_store\_name: Название магазина (строка, обязательный). \* Возвращаемое значение: “Товар успешно перемещен”, или сообщение об ошибке.

Модуль main (Пользовательский интерфейс tkinter):

add\_product():

Описание: Функция открывает дополнительное окно для добавления нового товара и обрабатывает сохранения данных.

Входные параметры: Нет.

Возвращаемое значение: Нет.

add\_product\_firm():

Описание: Функция открывает дополнительное окно для связывания товара с фирмой и обрабатывает сохранения данных.

Входные параметры: Нет.

Возвращаемое значение: Нет.

show\_all\_products():

Описание: Функция открывает окно и отображает все товары в виде таблицы.

Входные параметры: Нет.

Возвращаемое значение: Нет.

show\_all\_firms():

Описание: Функция открывает окно и отображает все фирмы в виде таблицы.

Входные параметры: Нет.

Возвращаемое значение: Нет.

show\_all\_stores():

Описание: Функция открывает окно и отображает все магазины в виде таблицы.

Входные параметры: Нет.

Возвращаемое значение: Нет.

show\_delivery\_report():

Описание: Функция открывает окно и отображает отчет по поставкам за период.

Входные параметры: Нет.

Возвращаемое значение: Нет.

show\_delivery\_report\_by\_store():

Описание: Функция открывает окно и отображает отчет по поставкам для заданного магазина.

Входные параметры: Нет.

Возвращаемое значение: Нет.

show\_delivery\_report\_by\_firm():

Описание: Функция открывает окно и отображает отчет по поставкам для заданной фирмы.

Входные параметры: Нет.

Возвращаемое значение: Нет.

load\_product\_image():

Описание: Функция открывает окно загрузки изображения.

Входные параметры: Нет.

Возвращаемое значение: Нет.

view\_product\_image():

Описание: Функция открывает окно для просмотра изображений.

Входные параметры: Нет.

Возвращаемое значение: Нет.

show\_products\_by\_category():

Описание: Функция открывает окно для выбора категории и отображает товары по выбранной категории

Входные параметры: Нет.

Возвращаемое значение: Нет.

filter\_products\_by\_name():

Описание: Функция открывает окно для поиска товаров по имени.

Входные параметры: Нет.

Возвращаемое значение: Нет.

move\_product\_to\_store():

Описание: Функция открывает окно для перемещения товара в другой магазин.

Входные параметры: Нет.

Возвращаемое значение: Нет.

display\_products\_in\_treeview(parent, products):

Описание: Функция выводит список товаров в Treeview

Входные параметры:

parent: Родительский виджет.

products: Список товаров.

Возвращаемое значение: Нет.

## 3.5 Руководство пользователя

**Основные разделы меню**:

**Товары:** Управление информацией о товарах.

**Фирмы:** Просмотр информации о фирмах-поставщиках.

**Магазины:** Просмотр информации о магазинах.

**Отчеты:** Формирование различных отчетов о поставках.

1. **Добавление нового товара:**

В главном меню выберите Товары -> Добавить товар.

Откроется окно “Добавить товар”.

Заполните следующие поля:

Название товара: Полное наименование товара (обязательное поле).

Категория: Категория товара (например, “Электроника”, “Одежда”) (обязательное поле).

Характеристики: Описание характеристик товара (например, “Цвет: красный”, “Размер: L”) (необязательное поле).

Цена: Цена товара (число, например, 1299.99) (обязательное поле).

Количество: Количество товара в наличии (целое число, например, 100) (обязательное поле).

Единица измерения: Единица измерения товара (например, “шт.”, “кг”) (обязательное поле).

Название магазина: Название магазина, где находится товар(обязательное поле).

Скидка: Размер скидки на товар (число с плавающей точкой, например, 0.1 для 10%) (необязательное поле).

НДС: Размер НДС на товар (число с плавающей точкой, например, 0.2 для 20%) (необязательное поле).

Нажмите кнопку “Сохранить” для добавления товара в базу данных.

2. **Связывание товара с фирмой:**

В главном меню выберите Товары -> Связать товар с фирмой.

Откроется окно “Связать товар с фирмой”.

Введите Название товара и Название фирмы.

Нажмите кнопку “Сохранить” для установления связи.

3. **Просмотр списка всех товаров:**

В главном меню выберите Товары -> Показать все товары.

Откроется окно “Все товары” со списком товаров, отсортированных по ID, в формате таблицы.

В таблице вы увидите следующую информацию:

ID: Уникальный идентификатор товара.

Название: Название товара.

Категория: Категория товара.

Цена: Цена товара.

Количество: Количество товара на складе.

4. **Просмотр списка всех фирм:**

В главном меню выберите Фирмы -> Показать все фирмы.

Откроется окно “Все фирмы” со списком всех фирм, отсортированных по ID, в формате таблицы.

В таблице вы увидите следующую информацию:

ID: Уникальный идентификатор фирмы.

Название: Название фирмы.

5. **Просмотр списка всех магазинов**:

В главном меню выберите Магазины -> Показать все магазины.

Откроется окно “Все магазины” со списком всех магазинов, отсортированных по ID, в формате таблицы.

В таблице вы увидите следующую информацию:

ID: Уникальный идентификатор магазина.

Название: Название магазина.

6. **Формирование отчета по поставкам за период:**

В главном меню выберите Отчеты -> Отчет по поставкам за период.

Откроется окно “Отчет по поставкам за период”.

Введите следующие данные:

Начальная дата (YYYY-MM-DD): Дата начала периода (например, 2024-01-01).

Конечная дата (YYYY-MM-DD): Дата окончания периода (например, 2024-01-31).

Нажмите кнопку “Показать отчет”.

Откроется новое окно, содержащее таблицу отчета:

Товар: Название товара.

Категория: Категория товара.

Количество: Количество поступившего товара

Цена: Цена товара.

Магазин: Магазин, куда был поставлен товар.

Дата: Дата поставки товара.

Фирма: Название фирмы-поставщика

7. **Формирование отчета по поставкам для заданного магазина:**

В главном меню выберите Отчеты -> Отчет по поставкам для заданного магазина.

Откроется окно “Отчет по поставкам для магазина”.

Введите “Название магазина”.

Нажмите кнопку “Показать отчет”.

Откроется новое окно, содержащее таблицу отчета:

Товар: Название товара.

Категория: Категория товара.

Количество: Количество поступившего товара.

Цена: Цена товара.

Дата: Дата поставки товара.

Фирма: Название фирмы-поставщика.

8. **Формирование отчета по поставкам для заданной фирмы:**

В главном меню выберите Отчеты -> Отчет по поставкам для заданной фирмы.

Откроется окно “Отчет по поставкам для фирмы”.

Введите “Название фирмы”.

Нажмите кнопку “Показать отчет”.

Откроется новое окно, содержащее таблицу отчета:

Товар: Название товара.

Категория: Категория товара.

Количество: Количество поставленного товара.

Цена: Цена товара.

Дата: Дата поставки товара.

Магазин: Название магазина.

9. **Загрузка изображения товара:**

В главном меню выберите Товары -> Загрузить изображение товара.

В открывшемся окне введите название товара и нажмите кнопку “Выбрать файл”.

Выберите файл с изображением на вашем компьютере.

Нажмите кнопку “Сохранить” для добавления изображения к товару.

10. **Просмотр изображения товара:**

В главном меню выберите Товары -> Посмотреть изображение товара.

В открывшемся окне введите название товара и нажмите “Посмотреть изображение”

Если у товара есть изображение, то оно появится в окне, иначе, вы увидите сообщение о том что изображение не найдено.

11. **Фильтрация товаров по названию**

В главном меню выберите Товары -> Фильтрация по названию

В открывшемся окне введите название товара и нажмите “Показать”

Отобразится окно с товарами, содержащими в названии переданное значение

12. **Фильтрация товаров по категории**

В главном меню выберите Товары -> Показать товары заданной категории

В открывшемся окне введите название категории и нажмите “Показать товары”

Отобразится окно с товарами, относящихся к выбранной категории.

13. **Перемещение товара в другой магазин**

В главном меню выберите Товары -> Переместить товар

В открывшемся окне введите имя товара и название магазина, куда нужно переместить товар.

Нажмите кнопку “Переместить”

Примечание:

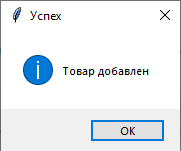
Внимательно вводите все данные, особенно даты.

Если возникла ошибка, то на экране появится соответствующее сообщение.

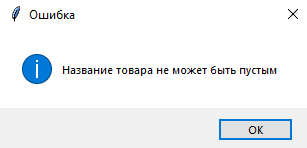
4. Тестирование ИС.

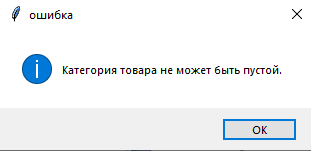
При завершении разработки программы необходимо провести проверку на её работоспособность при различных действиях пользователей. Программа должна корректно отображать и выполнять действия пользователя в любом случае. В исключительных моментах должны выводиться окна с сообщениями, поясняющие ситуацию.

При добавлении записи появляется сообщение об удачном добавлении товара в БД.

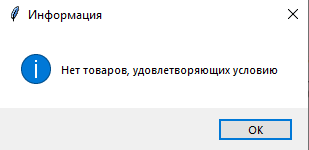


Либо же какая-либо графа в окне добавления товара будет не заполнена, то выдаст ошибку.

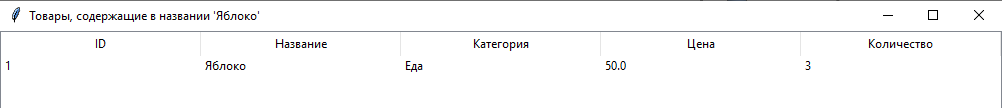


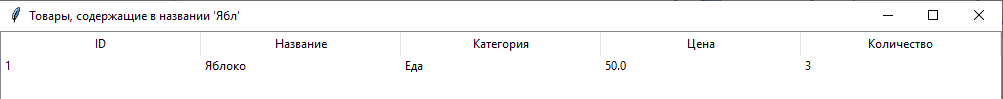


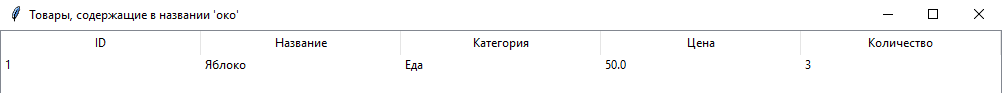
При поиске товара с помощью фильтрации, в случае не нахождения, появляется сообщение информирующее, что таких товаров нет.



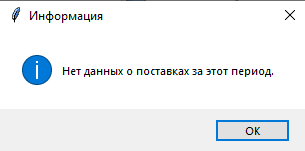
В случае правильного набора наименования товара, выведется окно с данным товаром







В окне отчёты по поставкам за определённый период, для магазина или фирмы, при отсутствии начальной или конечной даты, также периода которого нет в базе, выдаст сообщение.



Заключение.

В данной курсовой работе была разработана справочная система товаров для торговой фирмы, которая представляет собой автоматизированную информационную систему (АИС), предназначенную для эффективного управления данными о товарах, поставках и взаимодействиях с фирмами.

В ходе выполнения проекта были успешно достигнуты поставленные цели и задачи:

1. Проведен анализ предметной области и определены ключевые сущности (товары, фирмы, магазины) и их взаимосвязи.
2. Разработаны логическая, концептуальная и физическая модели базы данных, которые обеспечивают целостность и эффективность хранения информации.
3. Реализована физическая модель в СУБД SQLite, включая создание таблиц и установление связей между ними.
4. Разработан пользовательский интерфейс на языке Python с использованием библиотеки Tkinter, обеспечивающий удобство работы с системой.
5. Реализованы основные функции системы: добавление, просмотр, редактирование и поиск товаров, а также формирование отчетов о поставках и взаимодействиях с фирмами.
6. Проведено тестирование и отладка системы, что позволило обеспечить её стабильную и корректную работу.

Список используемой литературы.

1. Кононов А.В., Лебедев А.В. Реляционные базы данных: теория и практика. [Текст]. – М.: СОЛОН-ПРЕСС, 2017. – 318 с.

2. Карпова И.П. Базы данных. Учебное пособие. – СПб.: Питер, 2013 г. – 240 с. – Электронное издание.

3. Макаров А.И. Базы данных. [Текст]. – СПб. : Питер, 2020. – 416 с.

4. **Использование SQLite в Python-приложениях. [Электронный ресурс].**

URL: [**https://docs.python.org/3/library/sqlite3.html**](https://docs.python.org/3/library/sqlite3.html)

Приложение 1. Модели данных.

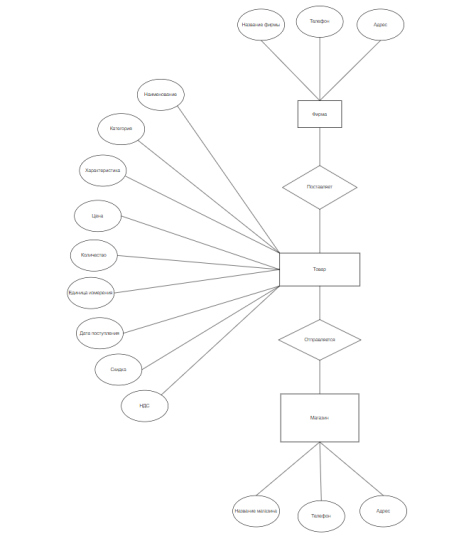


Рис. – Логическая модель

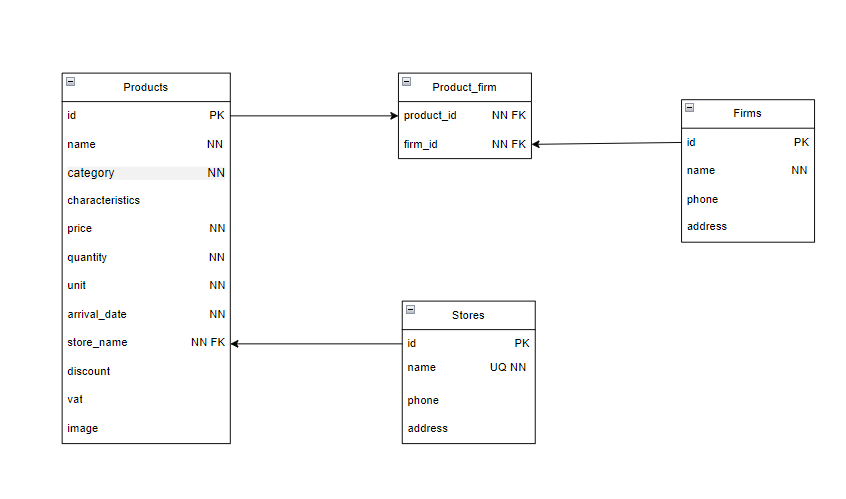


Рис – Концептуальная модель

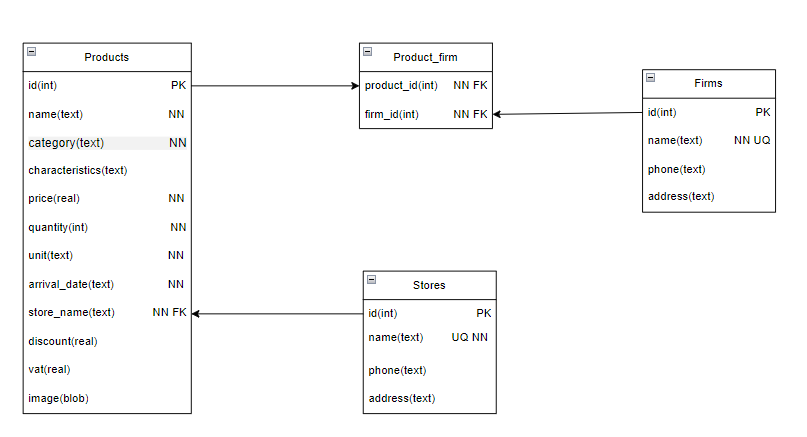


Рис – Физическая модель

# 

# Приложение 2. Скриншоты приложения и ссылка на репозиторий с кодом

https://github.com/pmi-122-student06/trade\_firm\_system.git

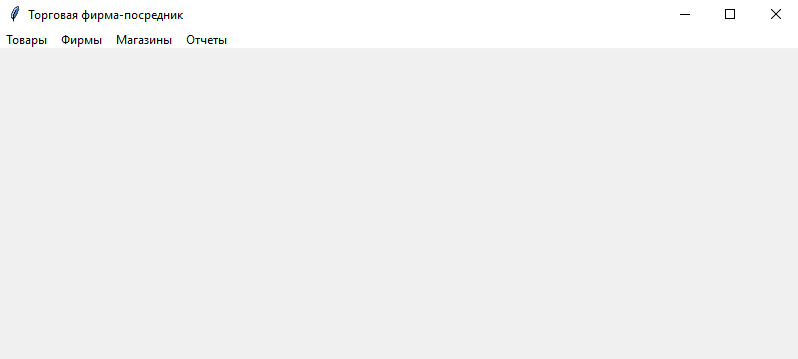


Рис. 1 – главное окно приложения

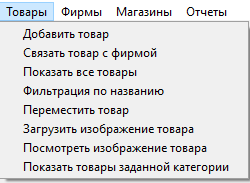


Рис. 2 – интерфейс окна товары в меню



Рис. 3 – интерфейс окна фирмы в меню



Рис. 4 – интерфейс окна магазины в меню

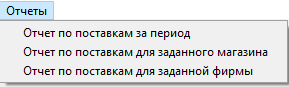


Рис. 5 – интерфейс окна отчёты в меню

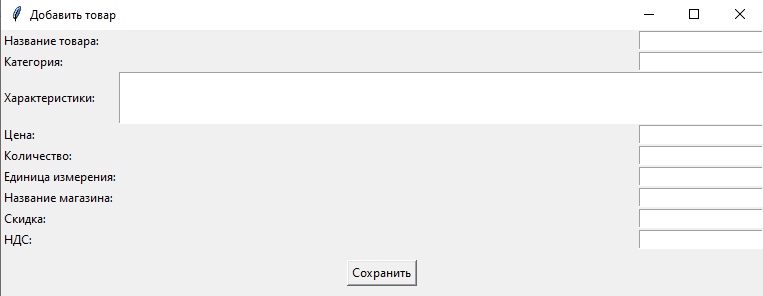


Рис. 6 - интерфейс окна для добавления товара

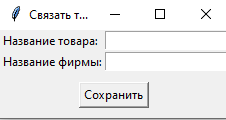


Рис. 7 - интерфейс окна для связи товара с фирмой

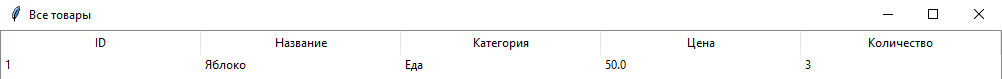


Рис. 8 - интерфейс окна для просмотра всех товаров

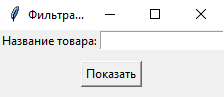


Рис. 9 - интерфейс окна для фильтрации товаров по названию

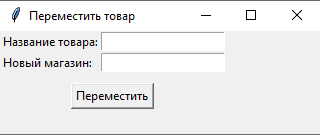


Рис. 10 - интерфейс окна для перемещения товара

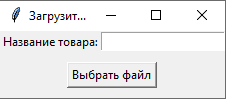


Рис. 11 - интерфейс окна для загрузки изображения товара

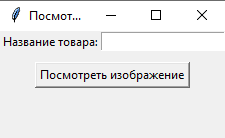


Рис. 12 - интерфейс окна для просмотра изображения товара

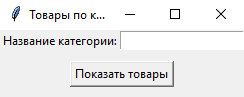


Рис. 13 - интерфейс окна для просмотра товара по заданной категории

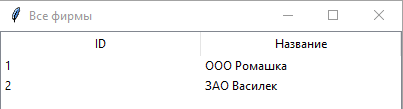


Рис. 14 - интерфейс окна для просмотра всех фирм

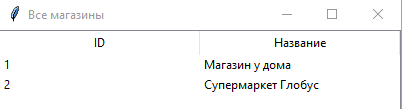


Рис. 15 - интерфейс окна для просмотра всех магазинов

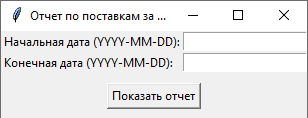


Рис. 16 - интерфейс окна для просмотра отчёта по поставкам за определённый период

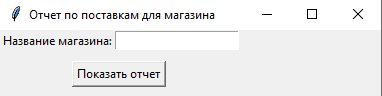


Рис. 17 - интерфейс окна для просмотра отчёта по поставкам для определённого магазина

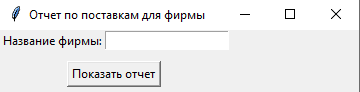


Рис. 18 - интерфейс окна для просмотра отчёта по поставкам для определённой фирмы