|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ |
|  | Руководитель Учебной практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Долженкова М. Л.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |

ТЕХНИЧЕСКий проект

приложения «Информационная система для магазина военной амуниции»

|  |
| --- |
|  |
|  |

2024

**СОДЕРЖАНИЕ**

[1. Описание проекта: 2](#_Toc182777391)

[2. Основные функциональные возможности: 2](#_Toc182777392)

[3. Уточнение структуры данных: 2](#_Toc182777393)

[3.1 Выделение сущностей 3](#_Toc182777394)

[3.2 Построение логической и физической ER-диаграмм 5](#_Toc182777395)

[4. Форма представления данных 6](#_Toc182777396)

[4.1 Контекстные диаграммы 6](#_Toc182777397)

[4.2 Пользовательские интерфейсы 18](#_Toc182777398)

[4.2.1 Пользовательские интерфейсы администратора 18](#_Toc182777399)

[4.2.2 Пользовательские интерфейсы сотрудника 26](#_Toc182777400)

[4.2.3 Пользовательские интерфейсы покупателя 28](#_Toc182777401)

[5. Разработка алгоритмов решения задач: 30](#_Toc182777402)

[6. Определение языка, структуры программы и требований к техническим средствам: 35](#_Toc182777403)

[7. Требования к техническим средствам: 35](#_Toc182777404)

# **Описание проекта:**

Название проекта: Информационная система для магазина по продаже военной амуниции.

Цель проекта: Разработка информационной системы для автоматизации управления магазином военной амуниции.

# **Основные функциональные возможности:**

Система продажи военной амуниции должна обеспечивать следующие возможности:

**Администратору:**

• Добавлять, редактировать и удалять категории амуниции, товары, поставщиков, учётные записи пользователей и заказы для поддержания актуальной информации.

• Добавлять новые категории, товары, поставщиков и учётные записи для внесения новой информации в систему.

• Удалять устаревшие или некорректные данные о категориях, товарах, поставщиках, учётных записях и заказах для поддержания точности и актуальности данных.

**Сотруднику:**

• Просматривать состав заказов, чтобы видеть детали заказанных товаров.

• Изменять статус заказа для отображения актуального состояния заказов.

**Покупателю:**

• Просматривать свои заказы и их состав, чтобы получать информацию о текущих и завершённых заказах.

• Создавать новые заказы для покупки амуниции.

• Редактировать свою учётную запись для актуализации личных данных.

Система будет работать с базой данных, содержащей информацию о товарах, категориях товаров, поставщиках, заказах и учётных записях пользователей.

# **Уточнение структуры данных:**

В ходе уточнения данных были выполнены следующие задачи, показанные в Таблице 2.

Таблица 1 – Задачи для уточнения структуры данных.

|  |  |  |  |
| --- | --- | --- | --- |
| **№ задачи** | **Название** | **Содержание** | **Результат** |
| 1 | Выделение сущностей | Выделение основных сущностей программы, а также их описание | Готовая таблица с сущностями и их описанием |
| 2 | Построение логической ER-диаграммы | Схематическое представление модели базы данных в виде логической ER-диаграммы | Готовая логическая ER-диаграмма |
| 3 | Построение диаграммы потоков данных | Определить, как будут передвигаться данные в программе; построить DFD диаграмму | Готовая диаграмма потоков данных (DFD) |

# **Выделение сущностей**

Таблица 2 – Набор сущностей и их атрибутов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Сущность** | **Свойство** | **Тип** | **Уникальность** | **Обязательность заполнения** | **Огра**  **ниче**  **ния** |
| User | UsersID | Int | Да | Да | РК |
| RoleId | Int |  | Да | FK |
| Phone | Text |  | Да |  |
| FullName | Text |  | Да |  |
| Password | Text | Да | Да |  |
| Email | Text |  | Да |  |
| Role | RolesID | Int | Да | Да | РК |
| Tittle | Text |  | Да |  |
| Order | OrdersID | Int | Да | Да | РК |
| ListID | Int | Да | Да | FK |
| StuffID | Int |  | Да | FK |
| BuyerID | Int |  | Да | FK |
| Date | Date |  | Да |  |
| Status | Text |  | Да |  |
| Sum | Double |  | Да |  |
| Adress | Text |  | Да |  |
| Index | Text |  | Да |  |
| Provider | ProviderID | Int | Да | Да | РК |
| Tittle | Text |  | Да |  |
| Country | Text |  |  |  |
| Category | CategoryID | Int | Да | Да | РК |
| Tittle | Text |  | Да |  |
| Product | ProductID | Int | Да | Да | РК |
| Tittle | Text |  | Да |  |
| CategoryID | Int |  | Да | FK |
| Color | Text |  | Да |  |
| Weight | Double |  | Да |  |
| Size | Text |  | Да |  |
| CountryOfMade | Text |  | Да |  |
| ProciderID | Int |  | Да | FK |
| Cost | Double |  | Да |  |
| ProductList | ListID | Int | Да | Да | РК |
| OrderID | Int |  | Да | FK |
| ProductID | Int |  | Да | FK |
| Count | Int |  | Да |  |

Связи между сущностями:

«Клиент» – «Заказ»: один ко многим, у каждого заказа определен конкретный клиент, но один клиент может оформлять несколько заказов.

«Роль» – «Клиент»: один ко многим, каждый пользователь имеет одну роль, но одна роль, может быть, у нескольких пользователей.

«Категория» – «Продукт»: один ко многим, каждый продукт относится к одной категории, но одна категория может включать несколько продуктов.

«Поставщик» – «Продукт»: один ко многим, каждый продукт поставляется одним поставщиком, но один поставщик может поставлять несколько продуктов.

«Продукт» – «Список продуктов»: один ко многим, каждый элемент списка продуктов относится к одному продукту, но один продукт может находиться в нескольких списках.

«Заказ» – «Список продуктов»: один ко многим, каждый элемент списка продуктов относится к одному заказу, но один заказ может содержать несколько продуктов.

# **Построение логической и физической ER-диаграмм**

Логическая ER-диаграмма показана на рисунке 1.

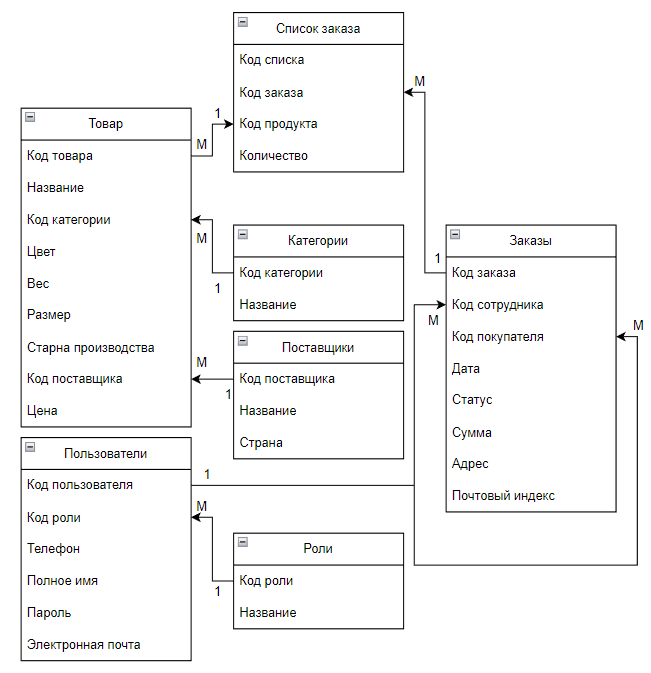


Рисунок 1 – Логическая модель базы данных

Физическая ER-диаграмма показана на рисунке 2.

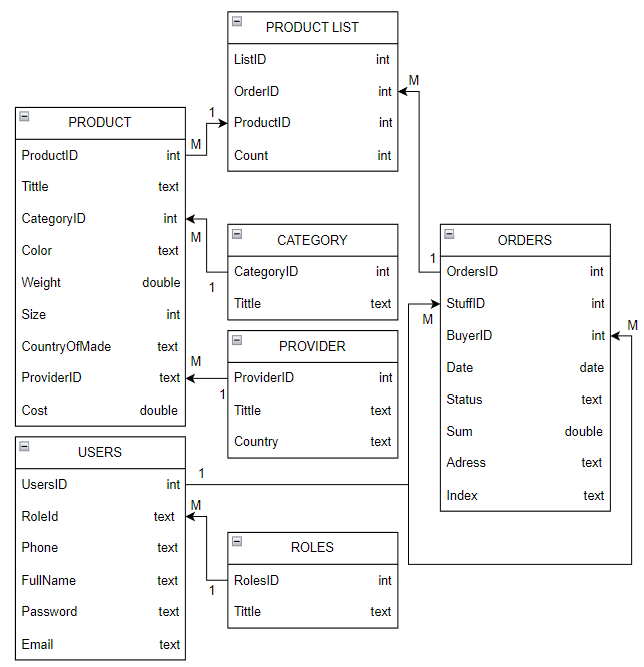


Рисунок 2 – Физическая модель базы данных

# **Форма представления данных**

# **Контекстные диаграммы**

Контекстная диаграмма информационной системы для магазина военной амуниции представляет собой обобщённое изображение взаимодействия приложения с внешней средой.

Контекстная диаграмма представлена на рисунке 3.

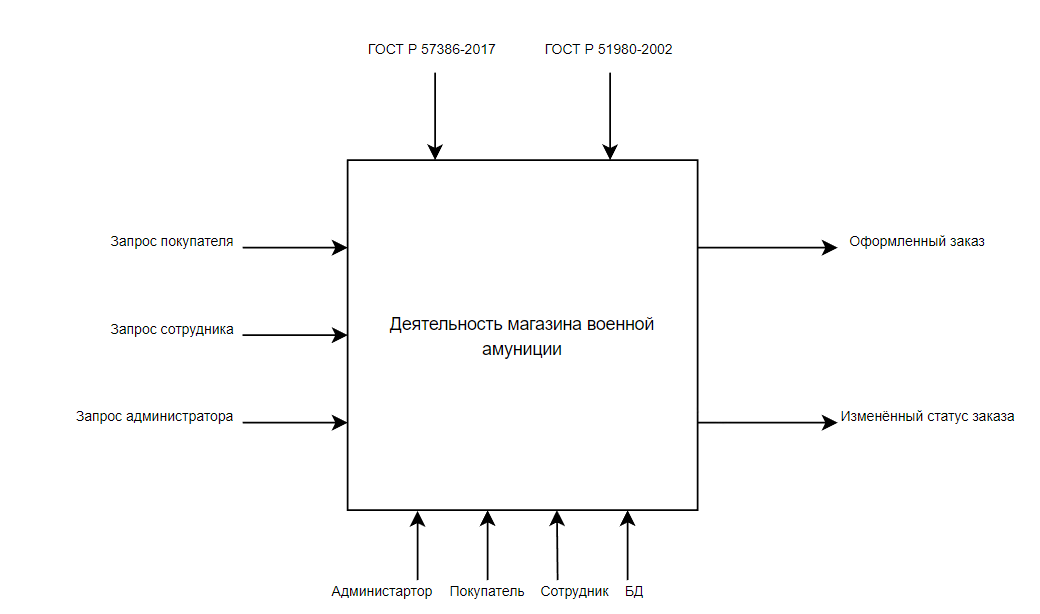


Рисунок 3 – Контекстная диаграмма ИС для магазина военной амуниции

На данной диаграмме изображён блок, обозначающий деятельность магазина военной амуниции.

Внешней входящей информацией:

* Запрос покупателя.
* Запрос сотрудника.
* Запрос администратора.

Управляющим воздействием будут являться:

* ГОСТ Р 57386-2017 Услуги по реализации гражданского и служебного оружия и патронов к нему. Общие требования (дата введения 01.01.2018) – Настоящий стандарт устанавливает требования к качеству услуг по продаже гражданского и служебного оружия и патронов к нему, включая требования к безопасности хранения, учета и передачи данных в торговых точках.
* ГОСТ Р 51980-2002 Система обеспечения качества продукции. Управление качеством. Основные положения и словарь (дата введения 01.07.2002) – Установленные в настоящем стандарте термины и определения формируют единую систему понятий, связанных с управлением качеством продукции, и могут быть применены в деятельности магазинов по продаже военной амуниции.

Механизмом управления является:

* База данных.
* Администратор.
* Покупатель.
* Сотрудник.

Внешней исходящей информацией является:

* Оформленный заказ.
* Изменённый статус заказа.

Далее представлены промежуточные диаграммы первого уровня контекстной диаграммы. Данные диаграммы показывают основные процессы, выполняемые в системе.

На рисунке 4 представлена промежуточная диаграмма «Добавление категории/поставщика/пользователя».

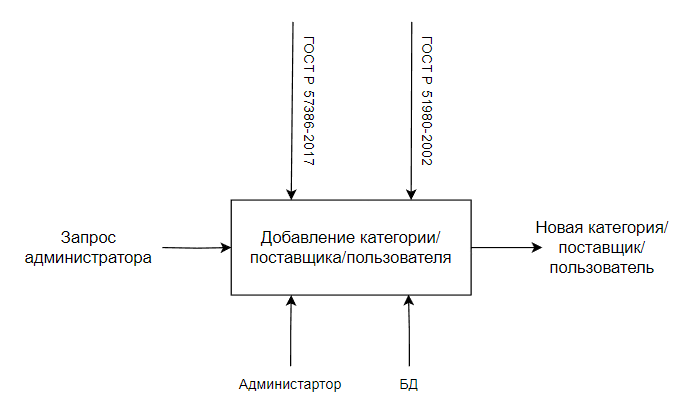


Рисунок 4 – промежуточная диаграмма «Добавление категории/поставщика/пользователя»

В данной диаграмме входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем новую категорию, поставщика или пользователя.

На рисунке 5 представлена промежуточная диаграмма «Удаление категории/поставщика/пользователя».

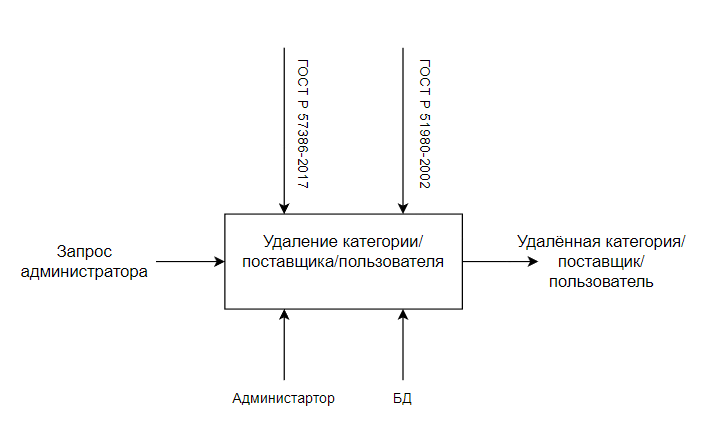


Рисунок 5 – промежуточная диаграмма «Удаление категории/поставщика/пользователя»

В данной диаграмме входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем удалённую из системы категорию, поставщика или пользователя.

На рисунке 6 представлена промежуточная диаграмма «Редактирование категории/поставщика/пользователя».

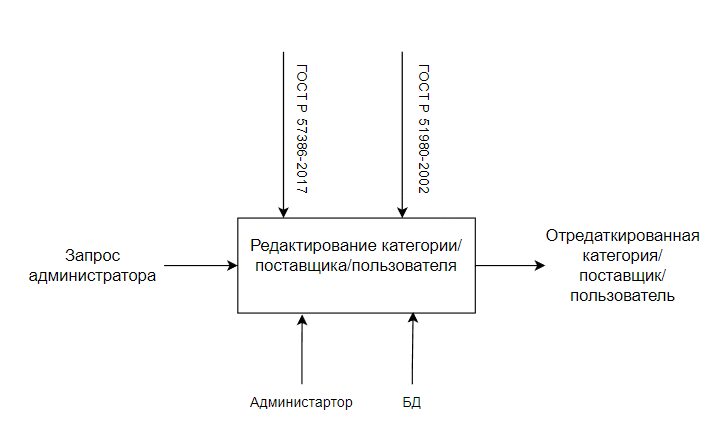


Рисунок 6 – промежуточная диаграмма «Редактирование категории/поставщика/пользователя»

В данной диаграмме входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем отредактированную категорию, поставщика или пользователя в БД.

На рисунке 7 представлена промежуточная диаграмма «Добавление товара».

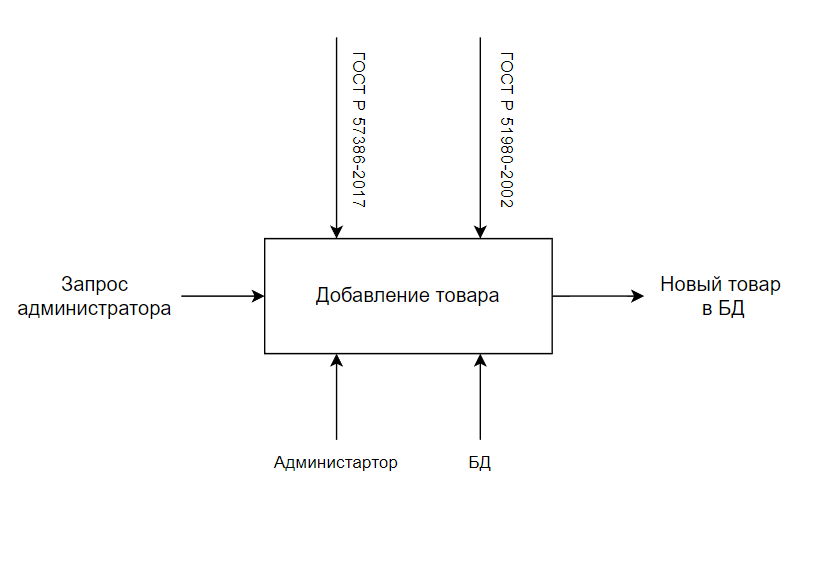


Рисунок 7 – промежуточная диаграмма «Добавление товара»

В данной диаграмме входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем новый товар в БД.

На рисунке 8 представлена промежуточная диаграмма «Удаление заказа».

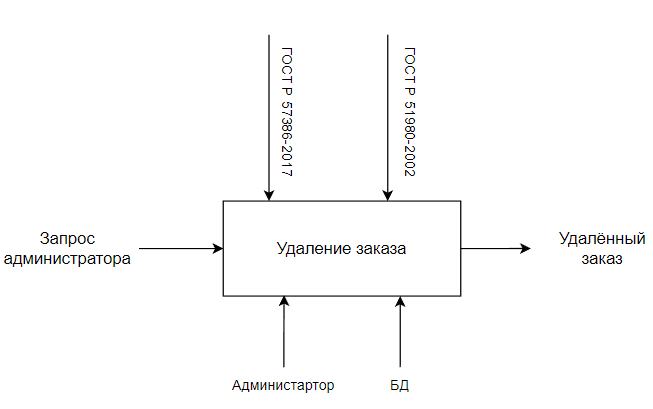


Рисунок 8 – промежуточная диаграмма «Удаление заказа»

В данной диаграмме входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем удалённый заказ из БД.

На рисунке 9 представлена промежуточная диаграмма «Редактирование заказа».

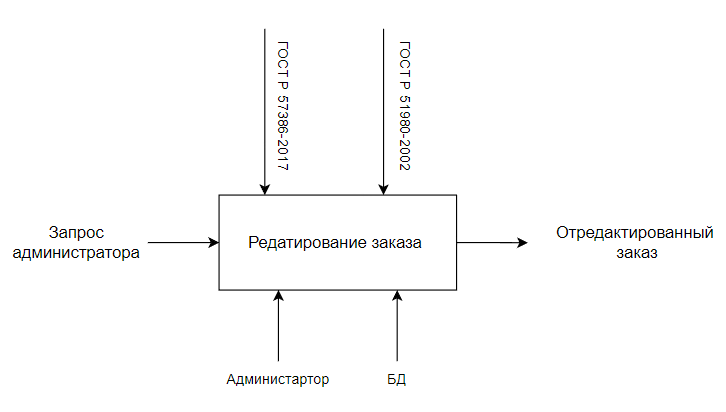


Рисунок 9 – промежуточная диаграмма «Редактирование заказа»

В данной диаграмме входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем заказ с отредактированными данными из БД.

На рисунке 10 представлена промежуточная диаграмма «Создание заказа».

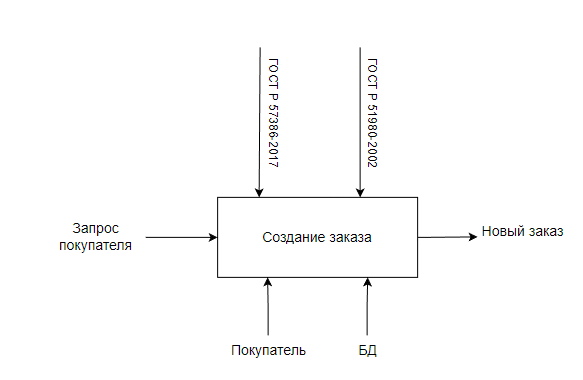


Рисунок 10 – промежуточная диаграмма «Создание заказа»

В данной диаграмме входящей информацией будет запрос покупателя. Механизмом управления являются покупатель и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем новый заказ.

На рисунке 11 представлена промежуточная диаграмма «Изменение статуса заказа».

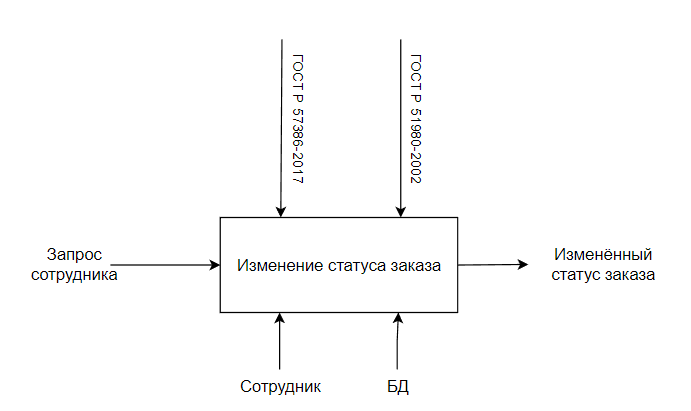


Рисунок 11 – промежуточная диаграмма «Изменение статуса заказа»

В данной диаграмме входящей информацией будет запрос сотрудника. Механизмом управления являются сотрудник и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем изменённый статус заказа в БД.

Далее будут представлены декомпозиции промежуточных диаграмм. Они будут представлять собой разбиение промежуточных диаграмм на более простые процессы и функции.

На рисунке 12 представлена декомпозиция промежуточной диаграммы «Добавление категории/поставщика/пользователя».

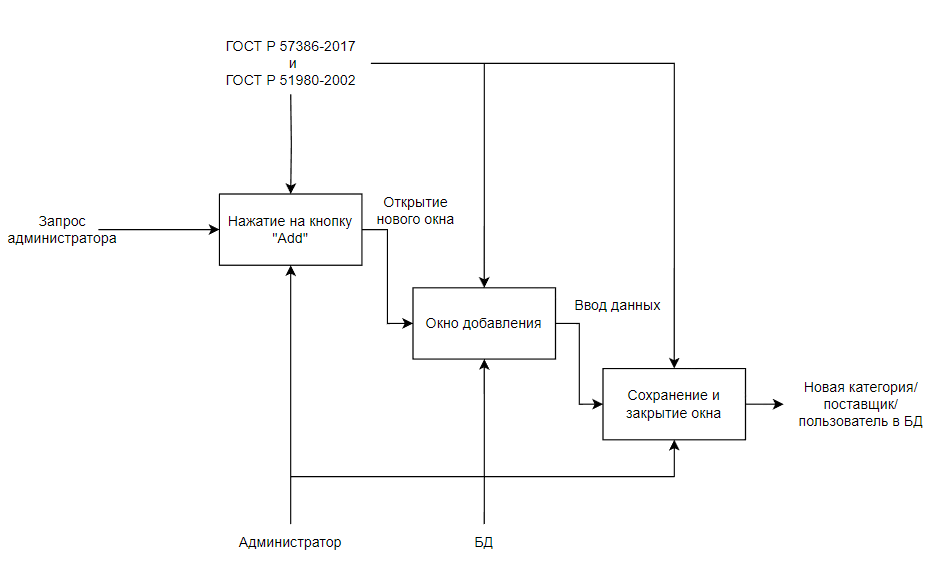


Рисунок 12 – декомпозиция промежуточной диаграммы «Добавление категории/поставщика/пользователя»

В данной декомпозиции диаграммы входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем новую запись в БД. Администратор в соответствующем окне жмёт на кнопку «Add», после чего открывается новое окно – окно добавления новой записи. Далее он должен ввести необходимую информацию в соответствующие поля, после нажатия на кнопку сохранения, программа отправит запрос на добавления информации в БД, после успешного ответа от БД, окно добавления закроется, что будет означать успешное добавление записи.

На рисунке 13 представлена декомпозиция промежуточной диаграммы «Удаление категории/поставщика/пользователя».

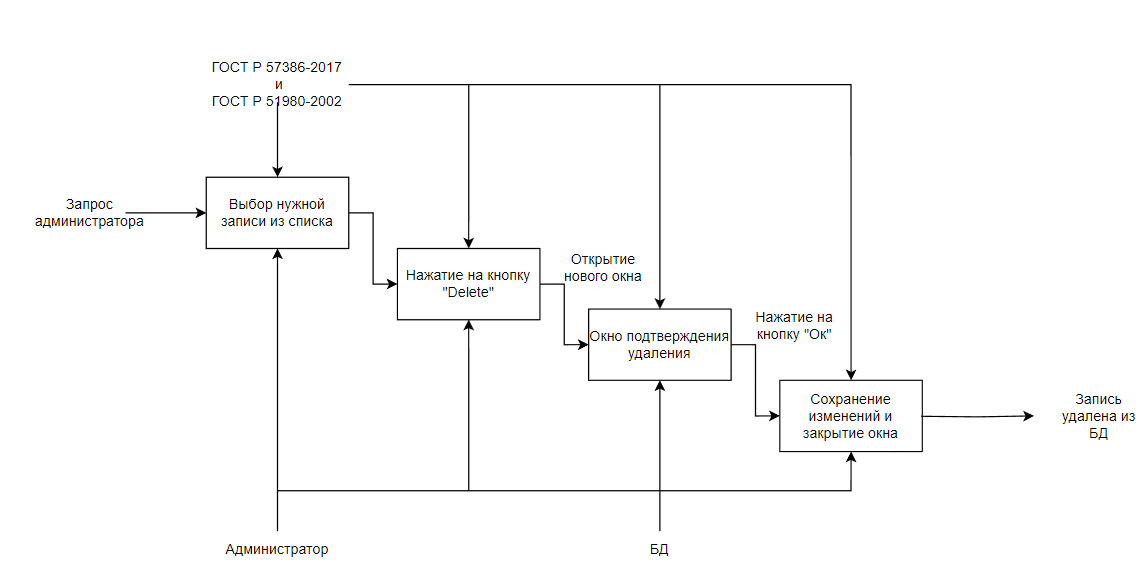


Рисунок 13 – декомпозиция промежуточной диаграммы «Удаление категории/поставщика/пользователя»

В данной декомпозиции диаграммы входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем удалённую запись из БД. Администратор в соответствующем окне выбирает нужную запись, после чего нажимает на кнопку «Delete». Далее открывается окно подтверждения, при утвердительном ответе (нажатие на кнопку «Ок») программа отправит запрос на удаление к БД, при утвердительном ответе окно подтверждения закроется, что будет означать успешное удаление записи.

На рисунке 14 представлена декомпозиция промежуточной диаграммы «Редактирования категории/поставщика/пользователя».

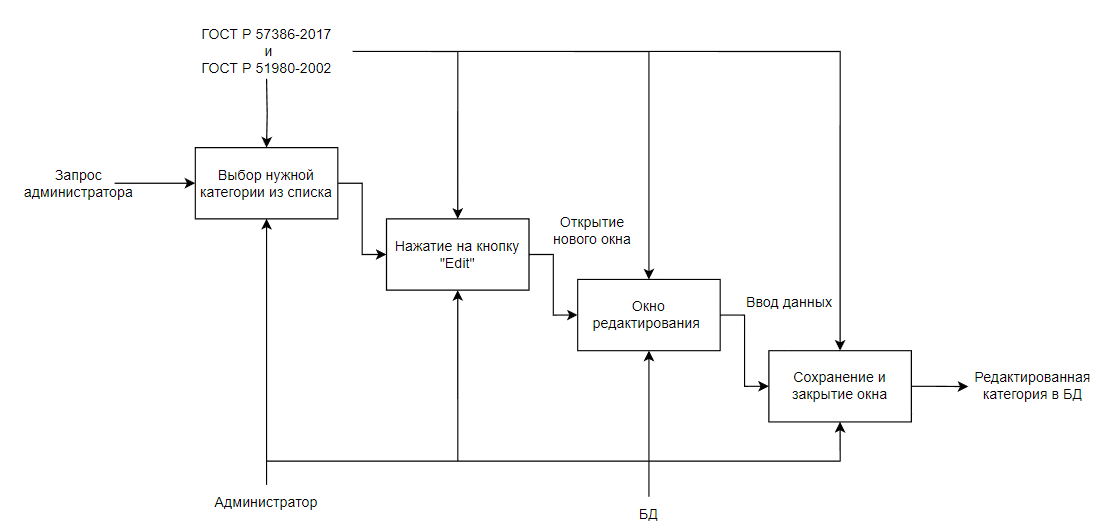


Рисунок 14 – декомпозиция промежуточной диаграммы «Редактирование категории/поставщика/пользователя»

В данной декомпозиции диаграммы, входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем отредактированную запись в БД. Администратор в соответствующем окне выбирает нужную запись, после чего нажимает на кнопку «Edit», далее открывается окно редактирования, в котором можно внести необходимые правки в запись, далее нужно нажать на кнопку сохранения, далее отправится запрос к БД на изменение записи, после успешного ответа, окно добавления закроется, что будет означать успешное изменение записи.

На рисунке 15 представлена декомпозиция промежуточной диаграммы «Добавление товара».

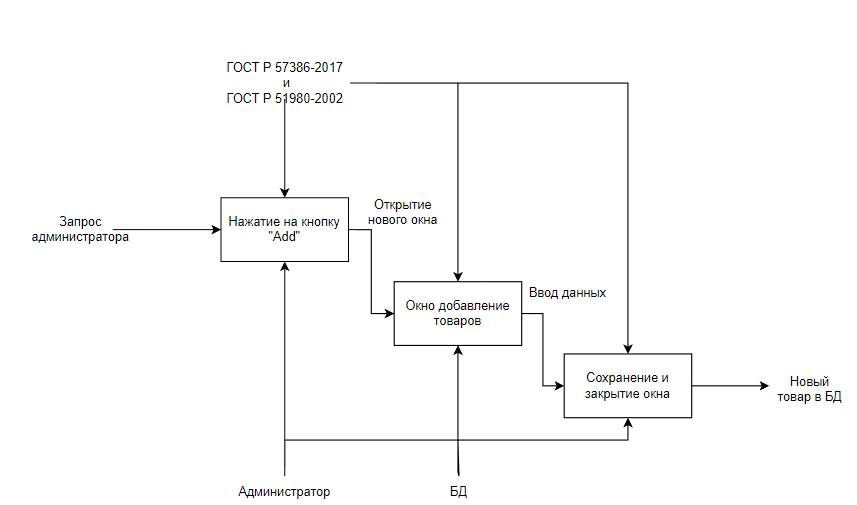


Рисунок 15 – декомпозиция промежуточной диаграммы «Добавление товара»

В данной декомпозиции диаграммы, входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем отредактированную запись в БД. Администратор в окне просмотра товаров жмёт на кнопку «Add», после чего открывается новое окно – окно добавления нового товара. Далее он должен ввести необходимую информацию в соответствующие поля, после нажатия на кнопку сохранения, программа отправит запрос на добавления информации в БД, после успешного ответа от БД, окно добавления закроется, что будет означать успешное добавление записи.

На рисунке 16 представлена декомпозиция промежуточной диаграммы «Удаление заказа».

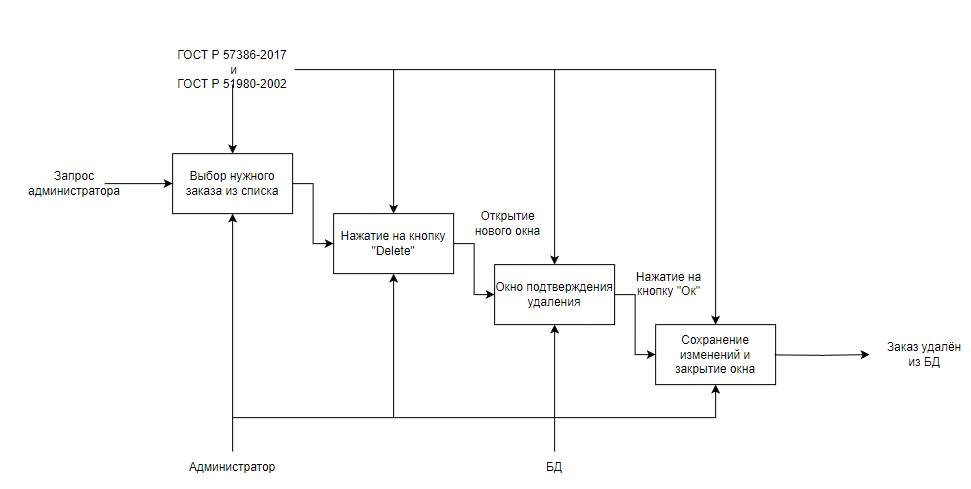


Рисунок 16 - декомпозиция промежуточной диаграммы «Удаление заказа»

В данной декомпозиции диаграммы, входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем удалённый заказ в БД. Администратор в окне просмотра заказов выбирает нужную запись, после чего нажимает на кнопку «Delete», далее открывается окно подтверждения, при утвердительном ответе (нажатие на кнопку «Ок») программа отправит запрос на удаление к БД, при утвердительном ответе окно подтверждения закроется, что будет означать успешное удаление записи.

На рисунке 17 представлена декомпозиция промежуточной диаграммы «Редактирование заказа».

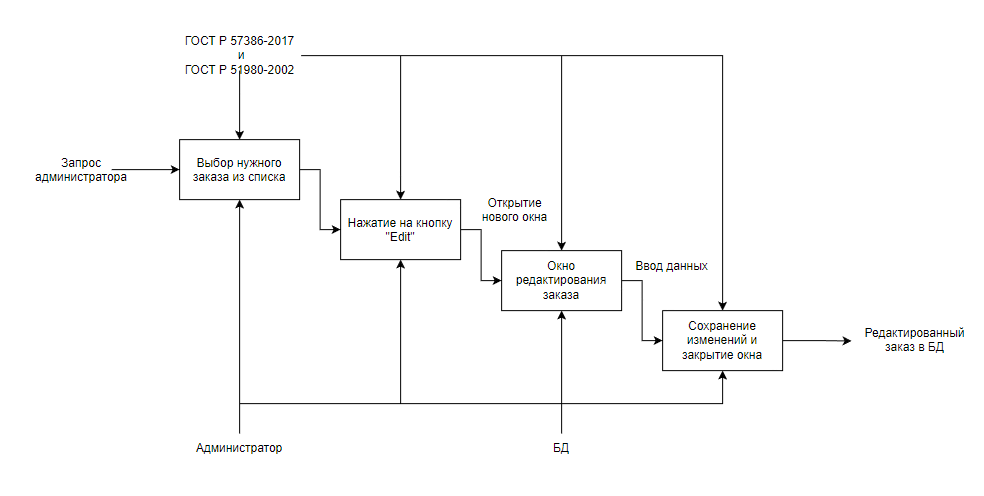


Рисунок 17 – декомпозиция промежуточной диаграммы «Редактирование заказа»

В данной декомпозиции диаграммы, входящей информацией будет запрос администратора. Механизмом управления являются администратор и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем отредактированный заказ в БД. Администратор в окне «View orders» выбирает нужную запись, после чего нажимает на кнопку «Edit», далее открывается окно редактирования, в котором можно внести необходимые правки в запись, далее нужно нажать на кнопку сохранения, далее отправится запрос к БД на изменение записи, после успешного ответа, окно добавления закроется, что будет означать успешное изменение записи.

На рисунке 18 представлена декомпозиция промежуточной диаграммы «Создание заказа».

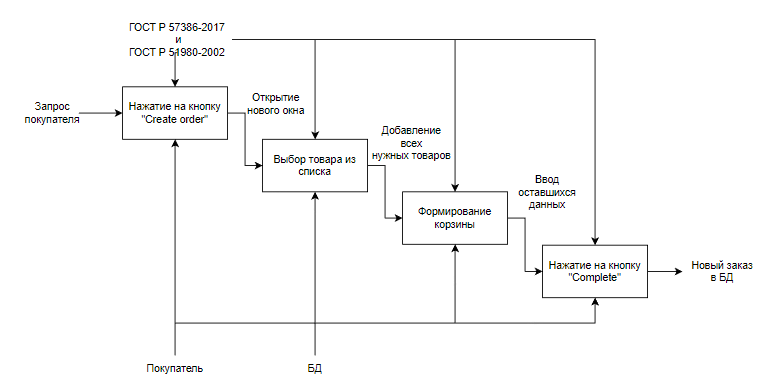


Рисунок 18 – декомпозиция промежуточной диаграммы «Создание заказа»

В данной декомпозиции диаграммы, входящей информацией будет запрос покупателя. Механизмом управления являются покупатель и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем отредактированный заказ в БД. Покупатель в главном меню нажимает на кнопку «Create order», после чего открывается окно, предназначенное для создания заказа. В нём покупатель из списка товаров выбирает нужные и формирует корзину, после чего ему остаётся ввести лишь оставшиеся данные (Адрес, почтовый индекс). Далее ему нужно завершить формирование заказа, нажатием на кнопку «Complete», после этого программа направит БД запрос на создание нового заказа, после успешного создания заказа у пользователя будет соответствующее уведомление.

На рисунке 19 представлена декомпозиция промежуточной диаграммы «Изменение статуса заказа».

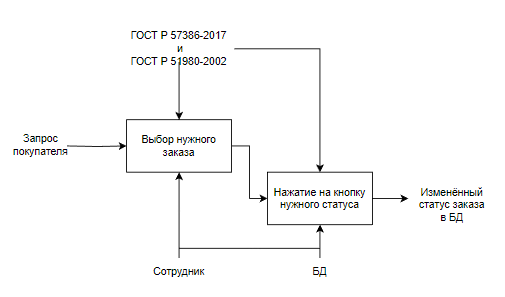


Рисунок 19 – декомпозиция промежуточной диаграммы «Изменение статуса заказа»

В данной декомпозиции диаграммы, входящей информацией будет запрос сотрудника. Механизмом управления являются сотрудник и база данных. Управляющим воздействием будут являться ГОСТ Р 57386-2017 и ГОСТ Р 51980-2002. На выходе получаем изменённый статус заказа в БД. Сотрудник в главном меню выбирает нужный заказ из списка, после чего он нажимает на соответствующую кнопку со статусом заказа, далее программа отправляет БД запрос на изменение записи, после положительного ответа изменённый статус отразиться в общем списке заказов, что считается успешной сменой статуса заказа.

# **Пользовательские интерфейсы**

На рисунках ниже предоставлены прототипы ИС для магазина военной амуниции.

При входе в приложение пользователя встречает окно авторизации, прототип (см. рис. 20) показывает, что в окне должны быть поля ввода логина и пароля, также должна быть информация о полях. Внизу окна должна быть кнопка «Войти».

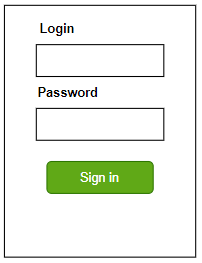


Рисунок 20 – Прототип окна авторизации

После входа пользователя встречает окно той роли, какая у его учётной записи, то есть после окна авторизации программа должна сразу понимать, что за пользователь перед ней, это либо администратор, сотрудник или покупатель.

# **Пользовательские интерфейсы администратора**

В главном меню администратора должны быть кнопки просмотра «Продуктов», «Категорий», «Поставщиков», «Пользователей» и «Заказов». При нажатии каждой из кнопок должно открываться окно для просмотра соответствующей информации. В самом низу окна должна быть кнопка выхода из учётной записи (см. рис. 21). Вверху окна должна быть текстовая надпись, в которой указано то, что это администратор и его имя.

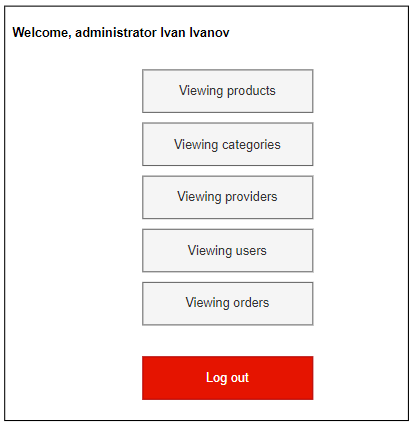


Рисунок 21 – Прототип главного окна администратора

При нажатии кнопки «Viewing products» должно открыться окно, в котором храниться табличная информация по всем продуктам, а также с возможностями добавления, редактирования, обновления и удаления данной информации (см. рис. 22). На прототипе представлены 5 кнопок, при нажатии на «Add» должно открыться окно добавления товара (см. рис. 23), при нажатии на «Edit» должно открываться окно редактирования товара (см. рис. 24), при нажатии на «Delete» должно появляться окно подтверждения, после утвердительного ответа по поводу удаления записи, запись должна быть удалена. При нажатии на кнопку «Refresh» должны обновляться данные. Ниже всех кнопок должна располагаться кнопка выхода из данного окна.

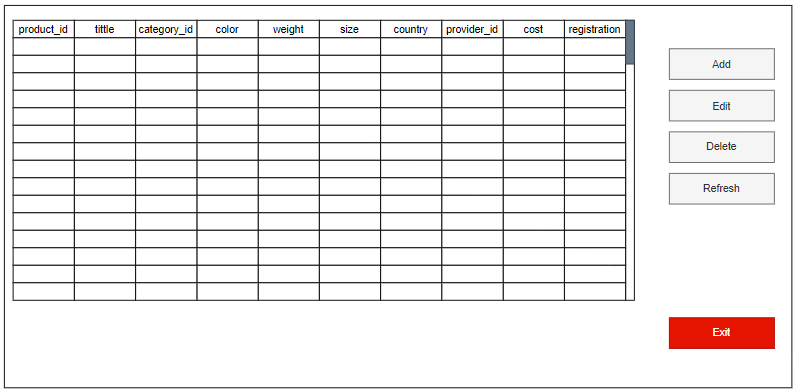


Рисунок 22 – Прототип окна просмотра товаров

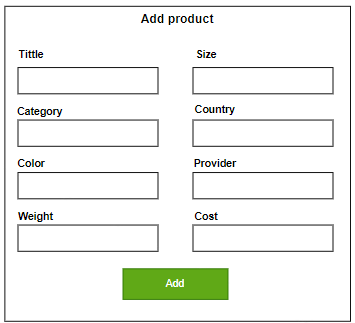


Рисунок 23 – Прототип окно добавления товара

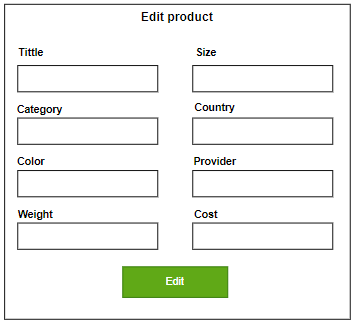


Рисунок 24 – Прототип окна редактирования товара

При нажатии кнопки «Viewing categories» должно открыться окно, в котором храниться табличная информация по всем категориям, а также с возможностями добавления, редактирования, обновления и удаления данной информации (см. рис. 25). На прототипе представлены 5 кнопок, при нажатии на «Add» должно открыться окно добавления категории (см. рис. 26), при нажатии на «Edit» должно открываться окно редактирования категории (см. рис. 27), при нажатии на «Delete» должно появляться окно подтверждения, после утвердительного ответа по поводу удаления записи, она должна быть удалена. При нажатии на кнопку «Refresh» должны обновляться данные. Ниже всех кнопок должна располагаться кнопка выхода из данного окна.

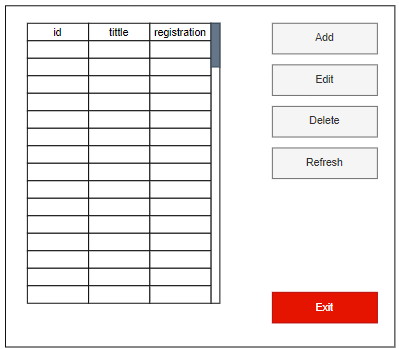


Рисунок 25 – Прототип окна просмотра данных о категориях



Рисунок 26 – Прототип окна добавления категории

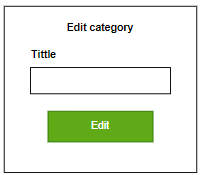


Рисунок 27 – Прототип окна редактирования категории

При нажатии кнопки «Viewing providers» должно открыться окно, в котором храниться табличная информация по всем поставщикам, а также с возможностями добавления, редактирования, обновления и удаления данной информации (см. рис. 28). На прототипе представлены 5 кнопок, при нажатии на «Add» должно открыться окно добавления поставщика (см. рис. 29), при нажатии на «Edit» должно открываться окно редактирования поставщика (см. рис. 30), при нажатии на «Delete» должно появляться окно подтверждения, после утвердительного ответа по поводу удаления записи, она должна быть удалена. При нажатии на кнопку «Refresh» должны обновляться данные. Ниже всех кнопок должна располагаться кнопка выхода из данного окна.

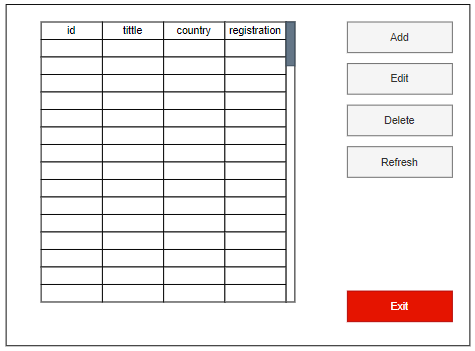


Рисунок 28 – Прототип окна просмотра данных о поставщиках

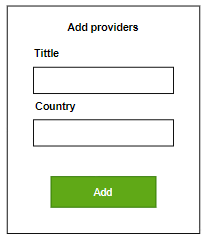


Рисунок 29 – Прототип окна добавления поставщика

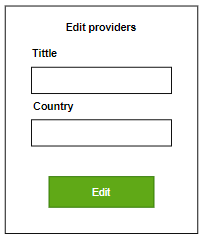


Рисунок 30 – Прототип окна редактирования поставщика

При нажатии кнопки «Viewing users» должно открыться окно, в котором храниться табличная информация по всем пользователям, а также с возможностями добавления, редактирования, обновления и удаления данной информации (см. рис. 31). На прототипе представлены 5 кнопок, при нажатии на «Add» должно открыться окно добавления пользователя (см. рис. 32), при нажатии на «Edit» должно открываться окно редактирования пользователя (см. рис. 33), при нажатии на «Delete» должно появляться окно подтверждения, после утвердительного ответа по поводу удаления записи, она должна быть удалена. При нажатии на кнопку «Refresh» должны обновляться данные. Ниже всех кнопок должна располагаться кнопка выхода из данного окна.

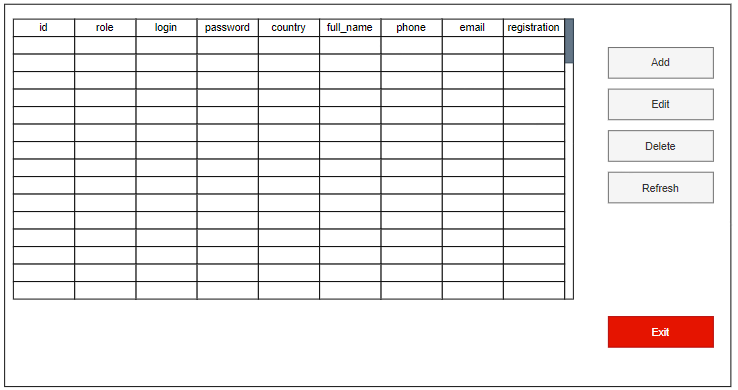


Рисунок 31 – Прототип окна просмотра данных о пользователях

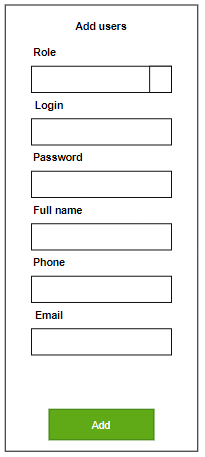


Рисунок 32 – Прототип окна добавления пользователя

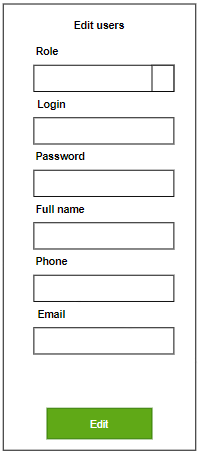


Рисунок 33 – Прототип окна редактирования пользователя

При нажатии кнопки «Viewing orders» должно открыться окно, в котором храниться табличная информация по всем заказам, а также с возможностями редактирования, обновления и удаления данной информации (см. рис. 34). На прототипе представлены 4 кнопки, при нажатии на «Edit» должно открываться окно редактирования заказа, но только общей суммы, адреса и почтового индекса (см. рис. 35), при нажатии на «Delete» должно появляться окно подтверждения, после утвердительного ответа по поводу удаления записи, она должна быть удалена. При нажатии на кнопку «Refresh» должны обновляться данные. Ниже всех кнопок должна располагаться кнопка выхода из данного окна.

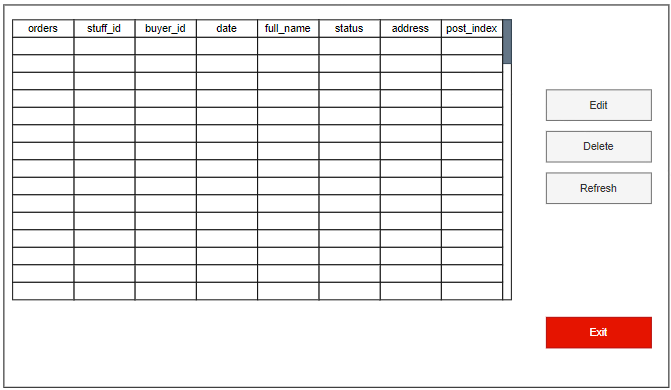


Рисунок 34 – Прототип окна просмотра данных о заказах

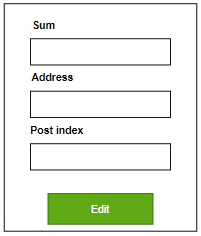


Рисунок 35 – Прототип окна редактирования данных заказа

# **Пользовательские интерфейсы сотрудника**

В главном меню сотрудника должна быть таблица с заказами, в ней должны быть названия столбцов (см. рис. 36). Также в этой таблице должна быть возможность выбирать строку с заказом, ещё нужно, чтобы была возможность просматривать все данные таблицы, можно использовать слайдер, как указанно на рисунке. В окне должны присутствовать кнопки для смены статуса заказа «Взять в обработку», «Собран», «Отправлен». Также кнопки «Обновить», «Подробнее» и «Выйти из учётной записи». Также должна присутствовать информативная надпись о пользователе, о том, что его роль – сотрудник и его имя.

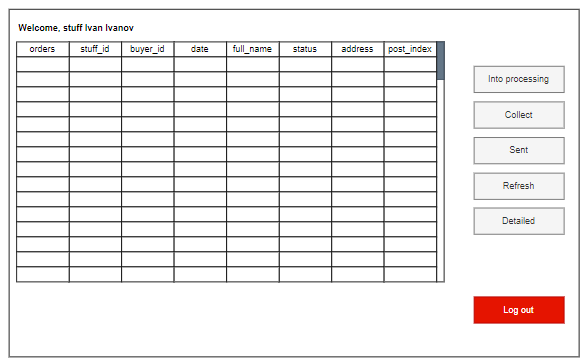


Рисунок 36 – Прототип главного окна сотрудника

При нажатии кнопки «Detailed» должно открыться окно, в котором храниться табличная информация по составу выбранного заказа (см. рис. 37). На прототипе представлена лишь одна кнопка и она позволяет закрыть окно и вернуться к главному окну сотрудника.

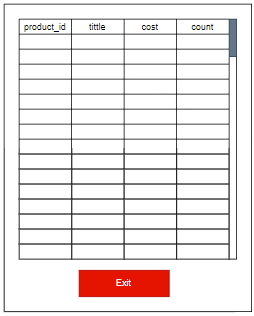


Рисунок 37 – Прототип окна просмотра состава заказа

# **Пользовательские интерфейсы покупателя**

В главном меню покупателя должны быть кнопки «Просмотр заказов», «Создать заказ», «Редактировать учётную запись» и «Выйти из учётной записи» (см. рис. 38). Вверху окна должна быть текстовая надпись, в которой указано то, что это покупатель и его имя.

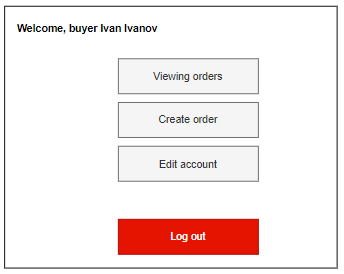


Рисунок 38 – Прототип главного окна покупателя

При нажатии кнопки «Viewing orders» должно открыться окно, в котором храниться табличная информация по всем заказам пользователя (см. рис. 39). На прототипе представлены лишь две кнопки первая «Detailed» позволяет посмотреть подробную информацию по выбранному заказу (см. рис. 40), вторая должна закрывать окно и возвращаться на главное меню покупателя.

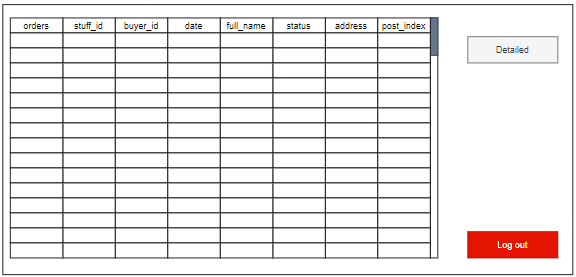


Рисунок 39 – Прототип окна просмотра данных о заказах пользователя

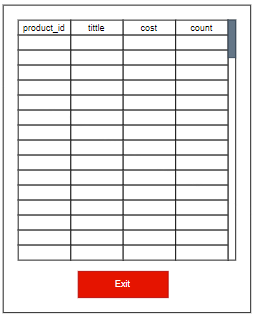


Рисунок 40 – Прототип окна просмотра состава заказа

При нажатии кнопки «Create order» должно открыться окно, в котором пользователь может создать новый заказ (см. рис. 41). В окне должна быть табличная информация по всем заказам, поле (корзина) в которое пользователь добавляет товары, также должны присутствовать поля для ввода адреса и почтового индекса. Ещё 5 кнопок «Add in order» позволяет добавить выбранный товар в корзину, «Delete» позволяет удалить выбранный товар из корзины, «Refresh» позволяет обновить информацию по списку товаров, «Complete» позволяет создать заказ и пустить его в обработку.

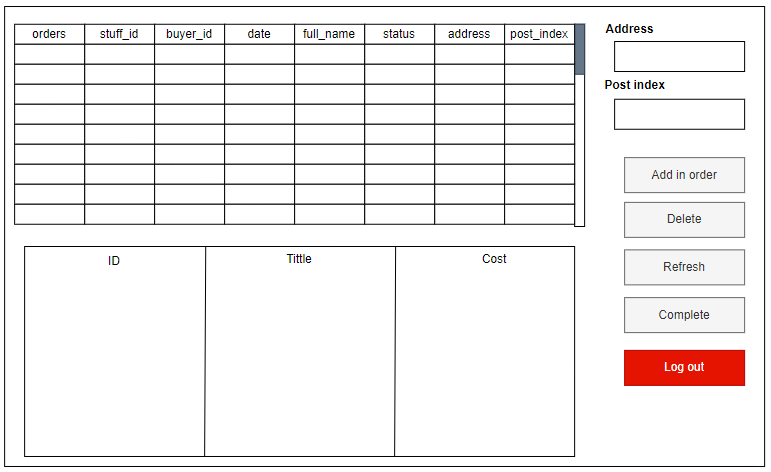


Рисунок 41 – Прототип окна создания заказа

При нажатии кнопки «Edit account» должно открыться окно, в котором пользователь сможет изменить данные своего аккаунта. Он может изменить логин, пароль, имя, телефон и электронную почту (см. рис. 42).

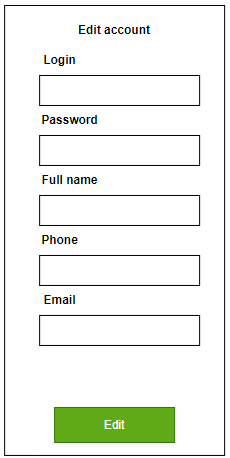


Рисунок 42 – Прототип окна редактирования аккаунта

# **Разработка алгоритмов решения задач**

# **Описание алгоритмов**

# **Добавление категории, поставщика, пользователя**

Алгоритм проверяет корректность введенных данных и прекращает работу, если они некорректны. Затем данные, введенные пользователем, извлекаются и используется для формирования запроса на добавление новой записи в базу данных. Этот запрос вместе с параметрами преобразуется в JSON-формат и отправляется на сервер с помощью POST-запроса. Если запрос выполняется успешно, метод закрывает окно и завершает выполнение. В случае возникновения ошибки выводится сообщение с описанием ошибки. Аналогичным образом происходит с алгоритмами поставщика и пользователя.

# **Удаление категории, поставщика, пользователя**

Метод проверяет, выбрана ли строка в таблице. Если строка не выбрана, выводится сообщение, и выполнение прекращается. Если строка выбрана, извлекается идентификатор категории. На его основе формируется запрос на удаление записи из базы данных, который затем упаковывается в JSON-формат и отправляется на сервер с помощью POST-запроса. После успешного выполнения запроса проверяется статус ответа. Если операция завершилась успешно, выводится сообщение об успешном удалении, и данные обновляются. В случае ошибки или проблем с выполнением запроса пользователю отображается сообщение с описанием проблемы. был введён неверно отображается соответствующее сообщение. Аналогичным образом происходит с алгоритмами поставщика и пользователя.

# **Редактирование категории, поставщика, пользователя**

Алгоритм извлекает введенные пользователем данные для обновления категории. На их основе создается объект с обновленными значениями, который затем используется для формирования SQL-запроса на изменение записи в базе данных. Запрос упаковывается в JSON-формат и отправляется на сервер через POST-запрос. После выполнения запроса проверяется статус ответа. Если операция прошла успешно, выводится сообщение об успешном обновлении, и окно закрывается. В случае ошибки пользователю отображается сообщение с описанием проблемы. Аналогичным образом происходит с алгоритмами поставщика и пользователя.

# **Добавление заказа**

Алгоритм проверяет, выбрана ли строка в таблице товаров. Если товар не выбран, отображается сообщение, и выполнение прекращается. Если строка выбрана, извлекаются данные о товаре, такие как идентификатор, название и стоимость. На их основе создается элемент для отображения в списке товаров заказа и добавляется в визуальный компонент списка. Затем информация о товаре записывается в временный файл, чтобы сохранить данные для дальнейшей обработки.

# **Удаление заказа**

Алгоритм проверяет, выбрана ли строка в таблице заказов. Если заказ не выбран, выводится сообщение, и выполнение прекращается. При выборе заказа извлекается его идентификатор, после чего пользователю предлагается подтвердить удаление. Если пользователь соглашается, формируется запрос на удаление записи с указанным идентификатором из базы данных. Этот запрос упаковывается в JSON-формат и отправляется на сервер через POST-запрос. После успешного выполнения запроса отображается сообщение об успешном удалении, и список заказов обновляется. В случае ошибки пользователю выводится сообщение с описанием проблемы.

# **Редактирование заказа**

Метод выполняет обновление данных заказа. На основе введенных пользователем значений формируется запрос для изменения записи в базе данных, включая обновление суммы, адреса и почтового индекса заказа по его идентификатору. Этот запрос упаковывается в JSON-формат и отправляется на сервер через POST-запрос. Если запрос выполняется успешно, выводится сообщение об успешном обновлении данных, и окно закрывается. В случае ошибки пользователю отображается сообщение с описанием проблемы.

# **Создание заказа**

Метод проверяет наличие добавленных товаров в заказе и корректность введенных данных, таких как адрес и почтовый индекс. Если проверки не проходят, выводится соответствующее сообщение, и выполнение прекращается. Если данные корректны, вычисляется сумма заказа на основе содержимого временного файла, где хранятся данные о товарах.

Далее формируется запрос на создание нового заказа в базе данных, который упаковывается в JSON-формат и отправляется на сервер через POST-запрос. После успешного выполнения запроса выполняется запрос для получения идентификатора созданного заказа.

С помощью идентификатора заказа данные о каждом товаре из временного файла добавляются в связанную таблицу, представляющую список товаров заказа. Если все операции выполняются успешно, выводится сообщение об успешном завершении заказа, временный файл удаляется, и окно закрывается. В случае ошибок на любом этапе пользователю отображается сообщение с описанием проблемы.

# **Изменение статуса заказа**

Алгоритм проверяет, корректно ли изменение статуса заказа. Для этого определяется порядок возможных статусов в виде списка. Затем вычисляются индексы текущего и нового статуса в этом списке. Изменение статуса считается допустимым, если новый статус следует за текущим по заданной последовательности.

# **Пользовательские сценарии**

Пользовательские сценарии представлены в таблице 3.

Таблица 3 – Пользовательские сценарии

|  |  |  |
| --- | --- | --- |
| **№** | **Пользователь** | **Система** |
| **1** | Администратор заполнил поля для ввода новой категории корректными данными и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Сохраняет данные в БД.  3. Переходит на страницу «Категории», на которой отображены все ранние занесённые категории. |
| **2** | Администратор заполнил поля для ввода нового товара корректными данными и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Сохраняет данные в БД.  3. Переходит на страницу «Товары», на которой отображены все ранние занесённые товары. |
| **3** | Администратор заполнил поля для ввода нового поставщика корректными данными и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Сохраняет данные в БД.  3. Переходит на страницу «Поставщики», на которой отображены все ранние занесённые поставщики. |
| **4** | Администратор заполнил поля для ввода новой учетной записи пользователя корректными данными и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Сохраняет данные в БД.  3. Переходит на страницу «Пользователи», на которой отображены все ранние занесённые пользователи. |
| **5** | Администратор заполнил поля для редактирования категории и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Изменяет данные в БД.  3. Переходит на страницу «Категории», на которой отображены все ранние занесённые категории. |
| **6** | Администратор заполнил поля для редактирования товара и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Изменяет данные в БД.  3. Переходит на страницу «Товары», на которой отображены все ранние занесённые товары. |
| **7** | Администратор заполнил поля для редактирования поставщика и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Изменяет данные в БД.  3. Переходит на страницу «Поставщики», на которой отображены все ранние занесённые поставщики. |
| **8** | Администратор заполнил поля для редактирования учетной записи пользователя и нажал на кнопку «Сохранить» | 1. Система получает введённые данные.  2. Изменяет данные в БД.  3. Переходит на страницу «Пользователи», на которой отображены все ранние занесённые пользователи. |
| **9** | Администратор нажимает на кнопку «Удалить категорию» | 1. Система отправляет подтверждающее сообщение.  1.1. Если ответ «Да»:     А) Получает ID удаляемой категории.     Б) Находит по полученному ID запись в БД.     В) Удаляет найденную запись.  1.2. Если ответ «Нет»:     А) См. пункт 2.  2. Переходит на страницу «Категории». |
| **10** | Администратор нажимает на кнопку «Удалить товар» | 1. Система отправляет подтверждающее сообщение.  1.1. Если ответ «Да»:     А) Получает ID удаляемого товара.     Б) Находит по полученному ID запись в БД.     В) Удаляет найденную запись.  1.2. Если ответ «Нет»:     А) См. пункт 2.  2. Переходит на страницу «Товары». |
| **11** | Администратор нажимает на кнопку «Удалить поставщика» | 1. Система отправляет подтверждающее сообщение.  1.1. Если ответ «Да»:     А) Получает ID удаляемого поставщика.     Б) Находит по полученному ID запись в БД.     В) Удаляет найденную запись.  1.2. Если ответ «Нет»:     А) См. пункт 2.  2. Переходит на страницу «Поставщики». |
| **12** | Администратор нажимает на кнопку «Удалить пользователя» | 1. Система отправляет подтверждающее сообщение.  1.1. Если ответ «Да»:     А) Получает ID удаляемого пользователя.     Б) Находит по полученному ID запись в БД.     В) Удаляет найденную запись.  1.2. Если ответ «Нет»:     А) См. пункт 2.  2. Переходит на страницу «Пользователи». |
| **13** | Сотрудник выбирает конкретный заказ для просмотра его состава | 1. Система получает запрос на просмотр заказа.  2. Извлекает данные о составе заказа из БД.  3. Отображает информацию о составе заказа на экране. |
| **14** | Сотрудник выбирает заказ и изменяет его статус на нужный | 1. Система получает запрос на изменение статуса заказа.  2. Изменяет статус заказа в БД на нужный.  3. Отображает обновлённый статус на странице заказов. |
| **15** | Покупатель заходит на страницу «Мои заказы» и выбирает конкретный заказ для просмотра | 1. Система получает запрос на просмотр заказов.  2. Извлекает данные о заказах из БД.  3. Отображает список заказов покупателя и их состав на экране. |
| **16** | Покупатель заполнил все необходимые поля для создания заказа и нажал на кнопку «Создать заказ» | 1. Система получает введённые данные.  2. Автоматически генерирует уникальный ID для нового заказа.  3. Сохраняет данные заказа в БД.  4. Отображает уведомление об успешном создании заказа. |
| **17** | Покупатель заходит на страницу своей учётной записи и вносит изменения, после чего нажимает кнопку «Сохранить» | 1. Система получает введённые данные.  2. Изменяет данные пользователя в БД.  3. Отображает уведомление об успешном обновлении информации. |

# **Определение языка, структуры программы и требований к техническим средствам:**

**Сервер:** Python используется для создания сервера, так как он обеспечивает гибкость, обширную экосистему библиотек и легкость работы с API, что позволяет эффективно обрабатывать данные и взаимодействовать с базой данных.

**Фреймворк и графическая библиотека**: основное приложение разрабатывается на языке C# с использованием Windows Forms (WinForms) для построения интуитивно понятного и удобного графического интерфейса, подходящего для настольных систем.

**Хранение данных**: для хранения данных используется SQLite, поскольку она является легковесной и легко интегрируется с Python и C#. SQLite позволяет надёжно хранить данные и обеспечивает высокую производительность при работе с умеренными объемами данных, что идеально подходит для целей данной информационной системы.

# **Требования к техническим средствам:**

Минимальные требования для информационной системы Military Store:

* Операционная система: Windows 7 и выше;
* Оперативная память (ОЗУ): 4 ГБ;
* Жесткий диск: 5 ГБ;
* Процессор: Intel Core i3 или аналогичный;
* Разрешение экрана: минимум 1024x768 пикселей для удобного отображения интерфейса;
* Сетевое подключение: стабильное соединение с высокой скоростью передачи данных для корректного подключения и работы базы данных.