МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

Ибрагимов Ильшод Дилшадович

Автоматизация работы склада электротоваров

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

По специальность 5В060200 – Информатика

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет
имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

ДОПУЩЕН К ЗАЩИТЕ
Заведующая кафедрой ПИ
канд. физ-мат. наук,
профессор
_____ А.Н. Молдагулова
« 16 » 05 2022 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

На тему: Автоматизация работы склада электротоваров

По специальность 5В060200 – Информатика

Выполнил Ибрагимов И.Д.

Рецензент
Доктор Ph.D
_____ А.А. Досжанова
"____" _____ 2022 г.

Научный руководитель
Ассоциированый профессор
_____ Н.К. Мукажанов
"____" _____ 2022 г.

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

СӘТБАЕВ УНИВЕРСИТЕТІ

Институт автоматики и информационных технологий

Кафедра "Программная инженерия"

5В060200 – Информатика

УТВЕРЖДАЮ
Заведующий кафедрой ПИ
канд. физ-мат. наук,
профессор
_____ Молдагулова А.Н.
"16" _____05_____ 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся: *Ибрагимову Ильшоду Дилшадовичу*
Тема: *Автоматизация работы склада электротоваров.*
Утверждена приказом проректора по академической работе: *N489-П/О от 24.12.21*
Срок сдачи законченного проекта: *27.05.22г.*
Исходные данные к дипломному проекту: *сбор теоретического материала, данные анализа по данному проекту, техническая документация к стеку применяемых технологий, ER-диаграмма базы данных для описания процесса хранения информации.*
Перечень подлежащих разработке в дипломном проекте вопросов:
а) Анализ предметной области и аналоговых систем, исследование актуальности проекта;
б) Разработка модели реляционной БД;
в) Реализация архитектуры приложения;
г) Выбор стека технологий для реализации приложений;
д) Проектирование системы;
г) Реализация системы и пользовательского интерфейса;
д) Разработка и тестирование веб-приложения;
Перечень графического материала (с точным указанием обязательных чертежей): *представлены 18 слайдов презентации.*
Рекомендуемая основная литература: *из 14 наименований.*

ГРАФИК

подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю и консультантам	Примечание
1. Анализ предметной области.	25.01.2022	Выполнено
2. Выбор стека технологий для реализации проекта.	13.02.2022	Выполнено
3. Написание технического задания проекта	21.02.2022	Выполнено
4. Front-end разработка визуального состовляющего проекта на HTML и CSS.	10.03.2022	Выполнено
6. Back-end разработка функциональной составляющей проекта на Python, в фреймворке Django.	20.03.2022	Выполнено
7.Тестирование проекта	24.04.2022	Выполнено

Подписи

консультантов и нормоконтролера на законченный дипломный проект с указанием относящихся к ним разделов проекта

Наименования разделов	Консультанты, И.О.Ф. (уч. степень, звание)	Дата подписания	Подпись
Нормоконтроллер	Жекамбаева М.Н. Доктор Ph.D., ассоциированный профессор	13.05.22	
Программное обеспечение	Марғулан К. Магистр техн.наук, лектор	16.05.22г	

Научный руководитель _____ Н. К. Мукажанов

Задание принял к исполнению обучающийся _____ И.Д. Ибрагимов

Дата «17» ноября 2021 г.

АННОТАЦИЯ

Дипломный проект посвящен автоматизации работы склада электротоваров. Склад электротоваров - это помещение, для хранения товаров, а так же для оказания складских услуг. За частую, предприятия используют неавтоматизированные склада, что в свою приводит очередь к низкой обработки заказов, зависимость от определённых людей работающих на складе, а также к другим проблемам мешающим эффективной работе компании.

Автоматизированная система позволит упростить работу по сбору, хранению и обработки информационных данных. Система будет направленна на автоматизацию поиска товара на складе по базе данных. После авторизации на сайте, менеджер может просмотреть весь товар, находящийся на складе, а также различного рода информацию о товаре. Например: название товара, фото, характеристику, количество товара находящегося на складе, его категорию и дату добавления на склад. Администратор может управлять этими данными, добавлять новый товар в базу, удалять или корректировать ранее добавленный товар.

АҢДАТПА

Дипломдық жоба электротехникалық тауарлар қоймасын автоматтандыруға арналған. Электр тауарларының қоймасы – тауарларды сақтауға, сонымен қатар қойма қызметтерін көрсетуге арналған бөлме. Көбінесе кәсіпорындар қолмен жұмыс істейтін қоймаларды пайдаланады, бұл өз кезегінде тапсырыстың нашар өңделуіне, қоймада жұмыс істейтін белгілі бір адамдарға тәуелділікке, сонымен қатар компанияның тиімді жұмысына кедергі келтіретін басқа да мәселелерге әкеледі.

Автоматтандырылған жүйе ақпаратты жинау, сақтау және өңдеу жұмысын жеңілдетеді. Жүйе қоймадағы тауарларды мәліметтер базасында іздеуді автоматтандыруға бағытталатын болады. Сайтта авторизациядан кейін менеджер қоймадағы барлық тауарларды, сондай-ақ өнім туралы әртүрлі ақпаратты көре алады. Мысалы: тауардың атауы, фотосуреті, сипаттамасы, қоймадағы өнім саны, оның санаты және қоймаға қосылған күні. Әкімші бұл деректерді басқара алады, дерекқорға жаңа өнімді қоса алады, бұрын қосылған өнімді жоя алады немесе түзете алады.

THE ANNOTATION

The diploma project is devoted to the automation of the warehouse of electrical goods. Warehouse of electrical goods is a room for storing goods, as well as for the provision of warehouse services. Often, businesses use manual warehouses, which in turn leads to poor order processing, dependence on certain people working in the warehouse, as well as other problems that interfere with the efficient operation of the company.

An automated system will simplify the collection, storage and processing of information data. The system will be aimed at automating the search for goods in the warehouse in the database. After authorization on the site, the manager can view all the goods in stock, as well as various information about the product. For example: the name of the product, photo, description, quantity of the product in stock, its category and the date it was added to the warehouse. The administrator can manage this data, add a new product to the database, delete or correct a previously added product.

СОДЕРЖАНИЕ

	Введение	9
1	Анализ предметной области	10
1.1	Автоматизация системы	10
1.2	Аналоги автоматизированных систем	10
1.3	Автоматизированная система работы склада электротоваров	12
2	Реализация проекта	14
2.1	Терминология	14
2.2	Среда разработки	15
2.3	Проектирование системы	18
2.4	Описание системы	23
	Заключение	36
	Список используемой литературы	37
	Приложение А. Техническое задание	39
	Приложение Б. Текст программы	41

ВВЕДЕНИЕ

Исследования показывают, что внедрение систем автоматизации в предприятиях, значительно влияет на повышение уровня эффективности компании, а так же на облегчение некоторых процессов работы. Тем самым, увеличивается уровень конкурентоспособности предприятия на рынке.

Данный проект, посвящён автоматизации склада электротоваров. Приложение будет реализовано на Django, это framework такого языка программирования как python. Основная идея Django – это использование шаблона проектирования MVC. Model View Controller – позволяет разделять данные приложения и управления логики в целом, на три отдельных компонента: модель, представление, контроллер. Благодаря этому изменение каждого из компонентов, может быть осуществлена независимо от других представлений. Модель – это данные которые будут реагировать на команды контроллера. Представление – это часть отвечающая за отображение данных модели пользователю за счет реагирования на изменение модели. Контроллер – это взаимосвязь между пользователем и приложением, и о каждом действие пользователя контроллер оповещает модель о необходимости изменений.

Frontend часть проекта будет реализована на языке гипертекстовой разметки и стилизованно с помощью формального языка описания внешнего вида документа. Некоторые визуальные части проекта будут реализованы с помощью JavaScript и фреймворка JQuery.

Так же, в приложении будет присутствовать реляционная база данных о товарах, хранимых на складе. База данных будет храниться в СУБД SQLite. База данных позволит нам хранить и обрабатывать информацию в структурированном виде.

1 Анализ предметной области

1.1 Автоматизация системы

Автоматизированная система представляет собой техническую систему, которая состоит из автоматизированных средств выполняющих деятельность персонала. Данная система осуществляет сбор информации об объекте, а так же преобразует эти данные. Комплекс программных средств позволяет автоматизировать некоторые процессы работы предприятия.

Основная цель автоматизации систем – это увеличение производительности труда персонала, эффективное планирования всех ресурсов компании и улучшение выполнения однотипных операций, по типу обработки объёмного количества информации. Данная система преследует и другие цели, например: ускорение сбора и обработки информации, увеличения уровня контроля, повышение оперативности управления, снижение затрат, увеличение эффективности принимаемых решений.

Основная цель которая преследует автоматизированная система – это повышение эффективности использования средств управления объектами.

Автоматизированная система и её функции напрямую зависят от сферы промышленности, функции такой системы могут включать в себя разные объекты и возможности. Например: создание определенных отчетов о некоторых процессах компании, контроль производственных операций, стратегические решения, различные схема, для помощи в разработке и планировании.

Автоматизированная система имеет свои преимущества:

- повышенная производительность;
- надежность и прочность;
- снижение затрат человеческого труда и расходов;
- улучшенное качество.

Актуальность данного проекта заключается в том, что автоматизация работы склада , является главным фактором успеха предприятия. Склад является важным звеном логистики и его автоматизация повышает эффективность работы предприятия , которая помогает компании увеличить свой уровень конкурентоспособности. Так же, по мере появления новых задач, проект может быть доработан и решать другие проблемы.

1.2 Аналоги автоматизированных систем

EKAM – ПО предназначенное для учета товаров. Приложение работает в

облачном режиме и имеет ряд преимуществ:

- интеграция с кассой;
- поддержка современных движков для интернет магазинов;
- возможность подсчета экономических показателей;
- корректировка цен в зависимости от цен при закупке товара.

EKAM подходит для любого вида бизнеса, софт достаточно легкий в использовании и имеет понятный интерфейс, однако у данного ПО есть свои минусы:

- облачное хранение информации не является гарантией информационной безопасности ваших данных;
- нет поддержки телефонии и колл-центра;
- интеграция с почтой отсутствует;
- дорогая программа за которую придется ежемесячно вовремя вносить оплату, для того чтобы ваша система работала без перебоев.

CloudShop – сервис складского учета, который служит для обработки информации о остатках товара. Данное ПО обладает следующими преимуществами:

- большой объём базы данных;
- аналитика и полная отчётность о деятельности вашего предприятия;
- возможность контролирования работы персонала.

Несмотря на весомые преимущества, программа имеет много недостатков:

- плохой уровень технической поддержки;
- постоянное увеличение оплаты за использование ПО;
- функционал не дорабатывается и не развивается.

SUBTOTAL – приложение для складского учета с поддержкой онлайн касс, ПО обладает следующими преимуществами:

- учет остатков товара на складе;
- в зависимости от закупа, происходит переоценка цен.

У системы также есть свои недостатки:

- не работает функция возврата товара;
- исчезают некоторые данные из базы;
- отсутствие адекватной технической поддержки.

LiteBox – облачное ПО ведущие аналитику о складском учете предприятия. Преимущества LiteBox:

- полная аналитика и отчеты;
- не нужно устанавливать сервис на компьютер и можно пользоваться онлайн.

Недостатки данной системы:

- неудобный интерфейс;
- некоторые браузеры не поддерживают систему;
- система требует дополнительных вложений для технического

сопровождения.

1С Торговля и склад – ПО позволяющее автоматизировать все процессы складского учета. Преимущества приложения:

- полная денежная отчетность;
- отчеты о деятельности предприятия;
- работа производится непосредственно с промаркированными товарами;
- различные операции с разными валютами и банковский учет кредитов.

К минусам софта можно отнести следующее:

- не подходит для маленького бизнеса;
- большая часть функционала просто не используется.

1.3 Автоматизированная система работы склада электротоваров

Цель данного проекта – автоматизировать систему работы склада электротоваров с возможностью:

- хранения данных о товарах;
- поиск имеющихся товаров на складе по базе данных;
- фильтр товаров по категории, производителю, филиалу;
- добавление заказа на закуп товара;
- изменение информации о товарах;
- добавление и удаление товаров;
- добавление сотрудников в базу данных;
- генерация списка имеющегося товара на складе в файле Excel;
- генерация отчета о текущих заказов на закуп товара в Excel.

С помощью данной системы, менеджер сможет посмотреть список имеющихся товаров, без необходимости поиска их на складе. Поиск товаров по базе облегчается специальными фильтрами по которым можно сортировать товар в базе, а также поиском по названию товара. Также менеджер может оставлять заявку на закуп недостающего товара и генерировать Excel файлы со списками имеющихся товаров и списка товаров для закупа. Администратор может добавлять новый товар в базу данных, а так же удалять старый и изменять ранее внесенную информацию. Это упрощает процесс складского учета для предприятия.

Интерфейс приложения будет доступным и включать в себя следующее:

- главная страница с информацией о предприятии;
- вход для менеджера;
- страница с данными о товарах, где можно будет искать товар по названию или же по специальным фильтрам;
- форма для отправки заявки на закуп товара;

- ссылки для скачивания отчетов Excel;
- вход для администратора;
- admin панель с возможностью поиска товара, фильтрации, удаления, добавления, и изменения ранее внесенных данных.

2 Реализация проекта

2.1 Терминология

Таблица 1 – Сокращения, термины и их определения

Сокращение или термин	Определение
ПО	Программное обеспечение
PyChram	Среда для разработки ПО
Python	Язык программирования
MVC	Архитектурный паттерн который предполагает разделение логики приложения на три блока. Данные приложения-Model,Пользовательский интерфейс -View, Управляющая логика – Controller.
HTML	Язык гипертекстовой разметки, для отображения данных в веб-браузере
CSS	Каскадные таблицы стилей, используемые для описания представлений HTML
БД	База данных
СУБД	Система управления базой данных
Фреймворк	Платформа программы с библиотеками и готовыми решениями , которая определяет структуру всей системы
Django	Фреймворк для Веб-приложения написанный на Python, и использующий MVC шаблон проектирования
JavaScript	Объектно-ориентированный язык программирования
jQuery	Набор функций JS
SQL	Язык программирования для управления данными в реляционной базе данных

Продолжение таблицы 1

Сокращение или термин	Определение
UML	Язык моделирования при проектировании программ
CRUD	Функции при работа с базой данных, создание, чтение, изменение, удаление
Use Case	Описание взаимодействие ролей в системе
BPNM	Моделирование бизнес процессов
ORM	Связь базы данных с кодом
URL	Уникальный адрес электронного ресурса
Backend	Логика работы приложения которая не видна пользователю
WEB приложение	Приложение с которым клиент взаимодействует через браузер
Frontend	Пользовательский интерфейс
Open source	Открытое программное обеспечение
Токен	Единица цифрового учета
Токенизатор	Инструмент разбиения текста на токены
Парсер	Скриптовый язык программирования, для генерации HTML страниц
Кэш	Буфер с информацией которая вероятнее всего будет использоваться

2.2 Среда разработки

Среда разработки – это совокупность программных средств, для реализации ПО. В среду разработки, могут входить различные текстовые редакторы, компиляторы, отладчики и средства автоматизации сборки. Каждый программист выбирает программные средства с учетом своих личных требований,

функциональных возможностей персонального компьютера, а так же дополнительного функционала программ.

В данном проекте, будет использоваться среда разработки PyCharm. Данная программа имеет ряд преимуществ, благодаря которым для реализации софта будет использоваться именно это приложение. Pycharm поддерживает множество фреймворков включая Django, на котором будет реализовываться проект. Так же в данной среде можно быстро производить рефакторинг кода. Отдельным плюсом можно выделить удобный графический отладчики большое количество плагинов которые упрощают работу. Ну и главный фактор, это то что приложение доступно в open source и имеет возможность бесплатного пользования.

Проект будет реализован в Django на python. Данный фреймворк имеет обширную базу библиотек, а так же большинство инструментов для создания приложения уже реализованы в Django. Например: ORM, миграция баз данных, панель администратора, формы, миграция баз данных. В Django используется шаблон проектирования MVC, рисунок. Данная модель предполагает разделения данных на три логических компонента, модель, представление и контроллер. За счет этого можно изменять один из компонентов не опасаясь, что изменения повлияют на другие два.

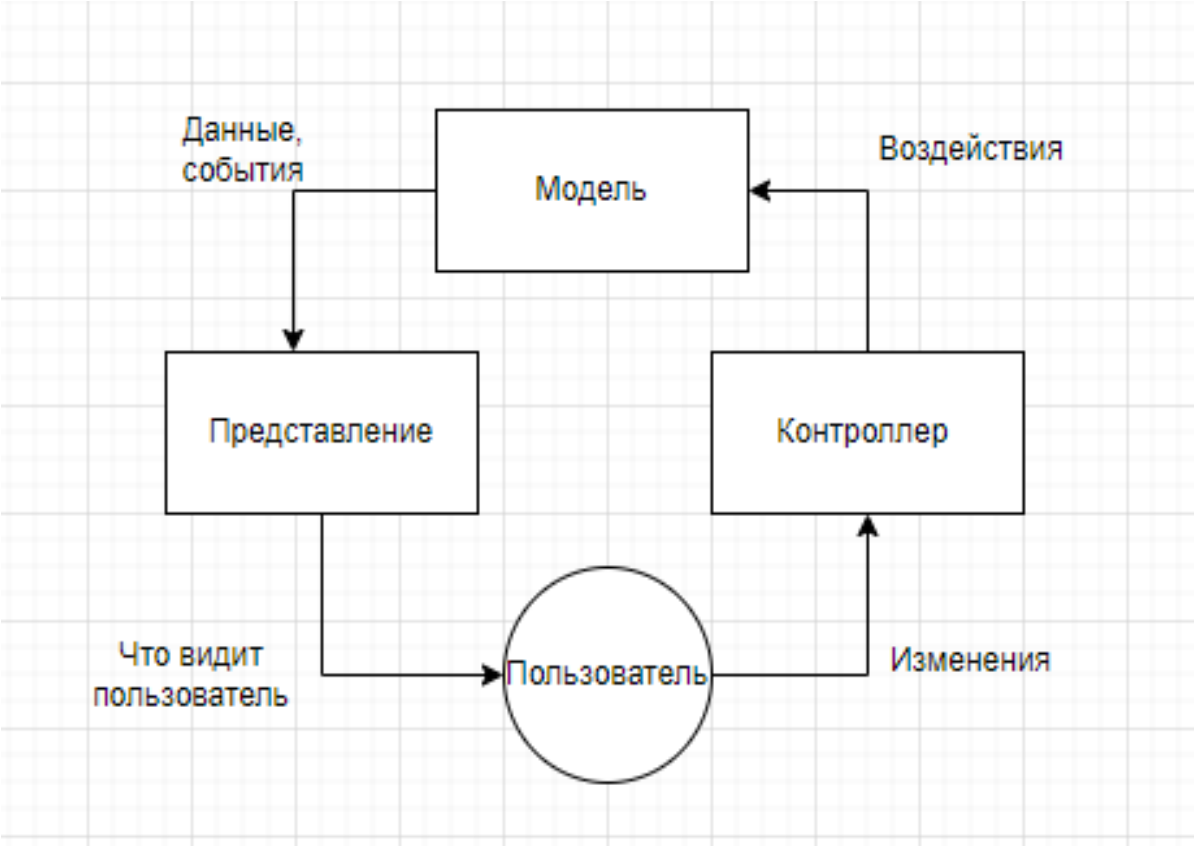


Рисунок-2.1 – Шаблон проектирования MVC

Визуальное составляющее приложения, интерфейс будет написан на HTML и стилизован с помощью CSS. Так же для некоторых элементов будет использоваться Js и jQuery, чтобы сделать слайдер и автоматизировать поиск по базе данных.

База данных , является важным аспектом в данном приложении. Она нужна для того, чтобы получать определенные данные, изменять их, вносить новую информацию о товарах на складе. Большинство СУБД имеют схожий функционал, и основаны на стандарте SQL-92. База данных в данном проекте, будет реализована в SQLite3, так как она является встроенной в фреймворке Django. Так же, данная СУБД является бесплатной, имеет понятный и простой интерфейс и возможность импорта и экспорта данных. Доступ к данным базы для менеджера, будет предоставлен в виде таблицы на странице, однако возможность ее редактирования будет лишь у администратора. Изменение данных, администратор может осуществлять в СУБД или же непосредственно в административной панели.

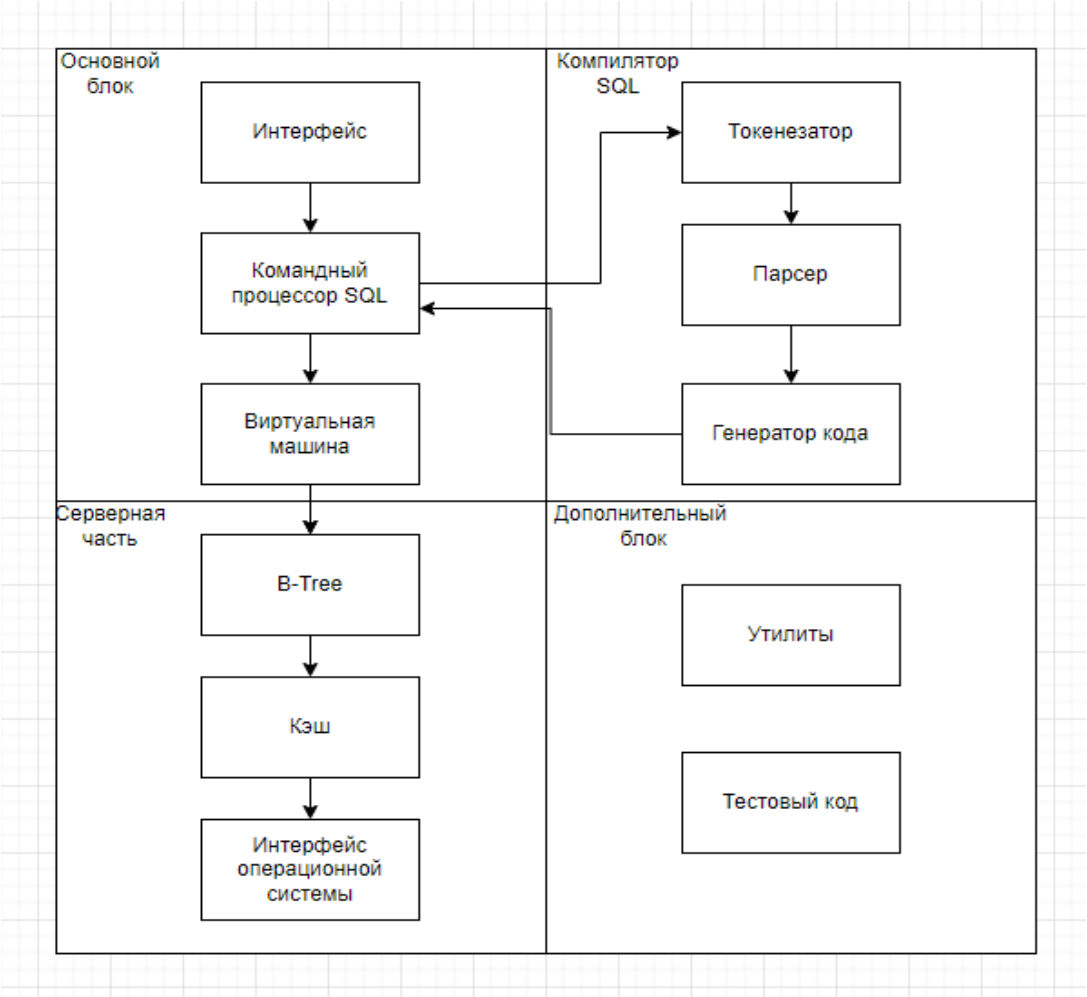


Рисунок-2.2 – Архитектура SQLite и ее взаимодействие БД

На рисунке 2.2 визуализирован принцип взаимодействия БД с backend. С интерфейса передается строка токенизатору, после того, как выполняется строка с инструкцией SQL. Токенизатор разбивает строку на токены и направляет эти токены парсеру. Задача парсера, дать значение каждому символу по их окружениям. После анализа и преобразования символов в SQL оператор, генератор кода запускает код виртуальной машины. Далее этот код будет реализован оператором SQL. Поиск данных осуществляется алгоритмом бинарного дерева. Далее эти данные читаются, записываются и кэшируются и передаются в интерфейс операционной системы. Так же в SQLite есть дополнительные утилиты, предназначенные для тестирования. Тестирование происходит имитацией механизма устранения ошибок.

2.3 Проектирование системы

Процесс проектирования системы, это определенная очередность этапов, которая позволяет найти решения для реализации проекта. Процесс проектирования данного приложения начинается с анализа бизнес процессов и реализации UML диаграмм, для визуального понимания логической составляющей работы системы.

Архитектура приложения - это дизайн всего приложения, который определяет как в программе будут взаимодействовать все компоненты.

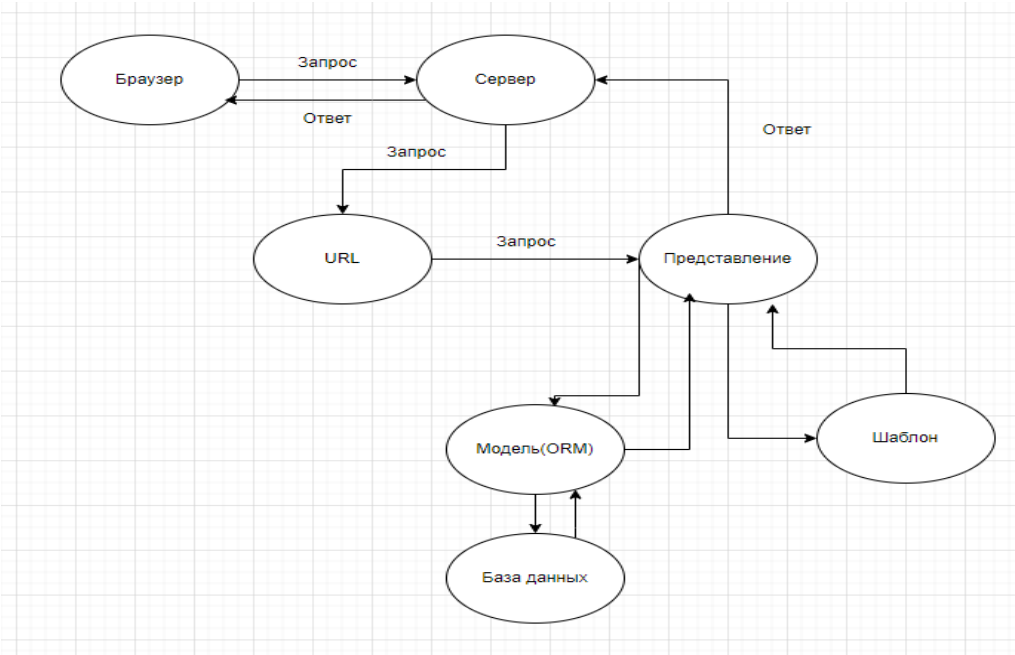


Рисунок-2.3 – Архитектура приложения

На рисунке 2.3 видно как происходит взаимодействие MVC с браузером. Браузер отправляет запрос на сервер, после чего происходит запрос по URl адресу который записан в представлении. Далее из представления передаются изменения в модель, которая в свою очередь берет данные из БД. Затем берется шаблон для данного представления и все передаётся в браузер.

UseCase диаграмма, это диаграмма вариантов использования. Она отображает отношение между классами и элементами самой системы с внешними действующими лицами. С ее помощью, можно увидеть фрагмент поведения системы, функционал приложения, а так же выявить недочеты.

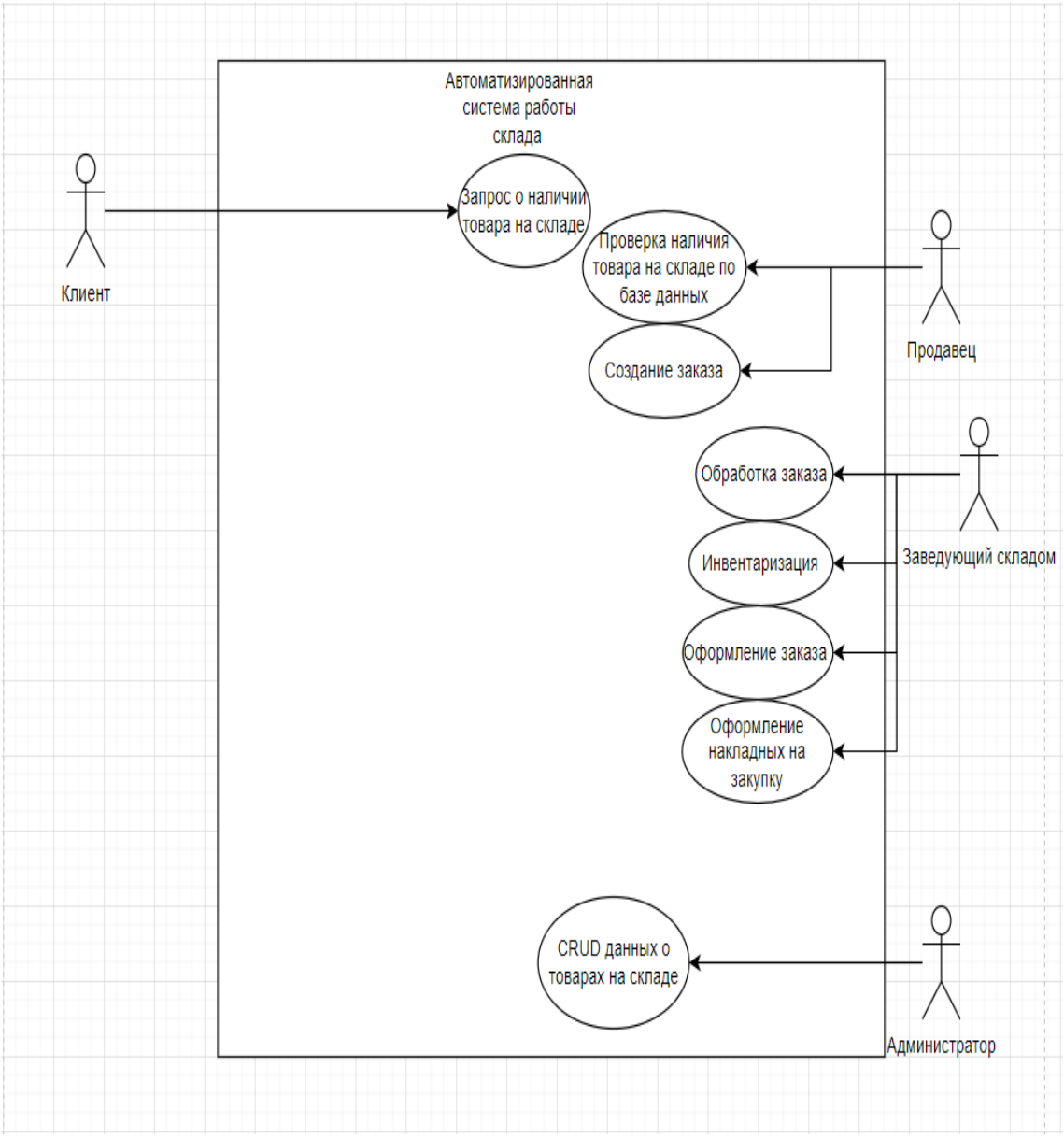


Рисунок-2.4 – Use Case диаграмма

На рисунке 2.4 показана диаграмма фрагмента работы системы. Работа системы начинается с запроса клиента о наличии определенного товара. Далее продавец или менеджер по базе данных проверяет в приложении наличия товара на складе, после чего формирует заказ клиента или, в случае отсутствия товара, создает заказ на закупку товара. Заведующий складом обрабатывает запрос менеджера, проводит инвентаризацию, и оформляет заказ для отправки. Администратор вносит изменения в базу данных о товарах на складе.

BPNM – диаграмма бизнес процессов используемая для описания действий происходящих внутри системы.

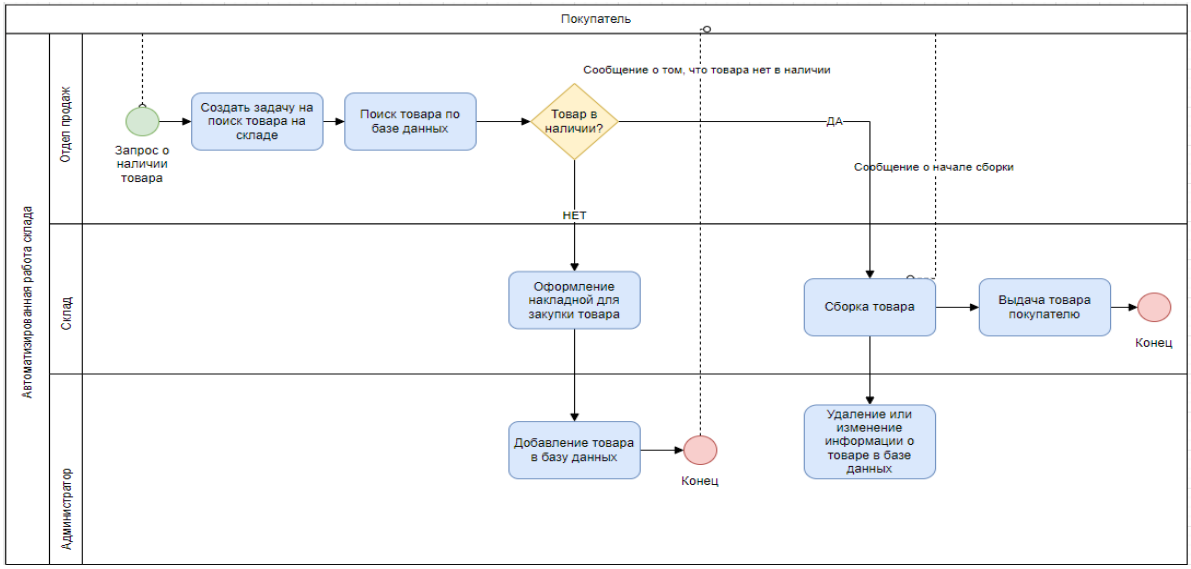


Рисунок-2.5 – BPNM диаграмма

На рисунке 2.5 описаны действия системы. Покупатель делает запрос , этот запрос попадает в отдел продаж. Далее менеджер создает задачу на поиск товара на складе. Затем осуществляется поиск по БД. В случае, если нет товара на складе процесс завершается и оформляется накладная для закупки этого товара, после покупки товар заносится в базу. Когда наше условие выполняется, то есть товар есть на складе, то начинается сборка товара, отправка и выдача его покупателю. В конце администратор заносит все изменения по количеству товара в БД, или же добавляет новый товар.

ER-диаграмма, это схема базы данных, с помощью которой можно определить сущности и их связи. Эта диаграмма нужна, для того чтобы видеть является ли база нормализованной. Понимание как будут связаны таблицы в базе между собой, дает возможность привести БД к минимальной избыточности. Если в базе избыточность будет высокая, то это приведет к различному роду аномалий, а также снизит производительность, и управление данными в БД станет не очень удобным.

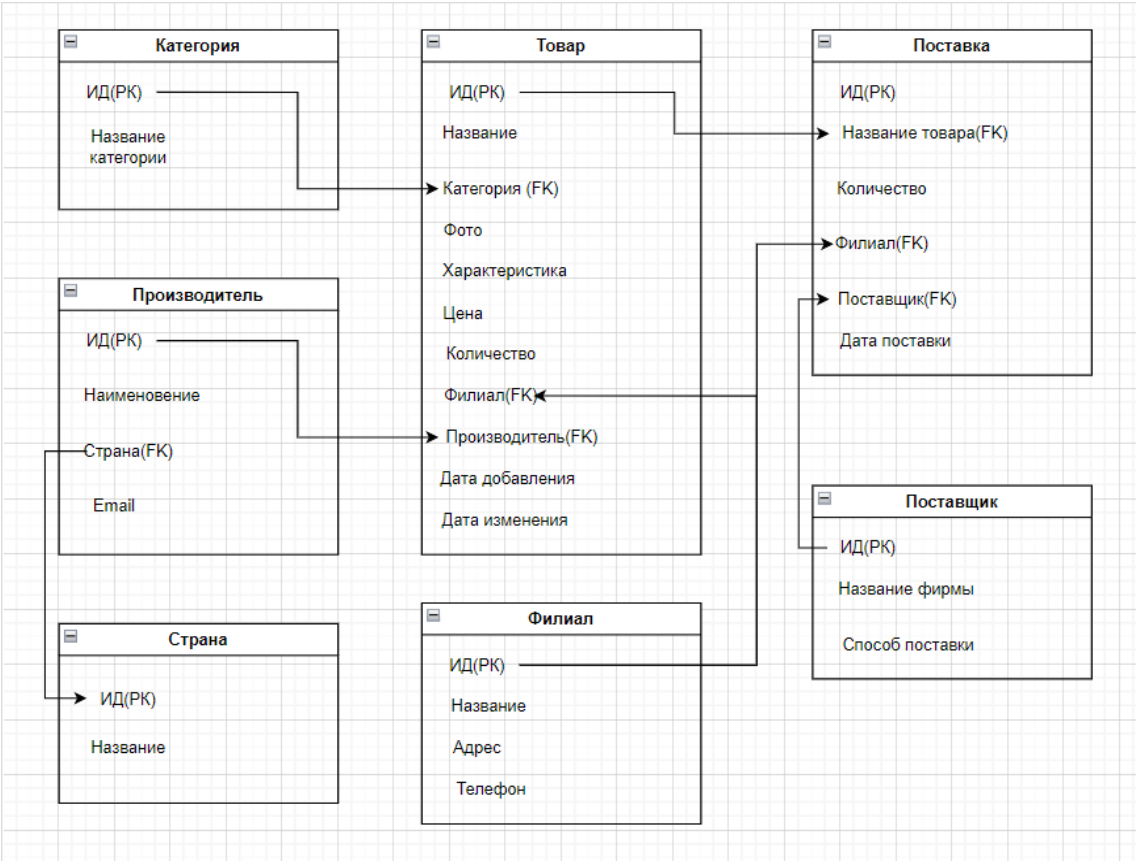


Рисунок-2.6 - ER диаграмма

На рисунке 2.6 изображена реляционная база данных системы. В ней находятся семь логический связанных между собой таблиц. Главная таблица – “Товар”, в ней находится все информация о товарах на складе. Так же в таблицах есть поля под названием ИД, это уникальные идентификаторы для каждой таблицы. Они используются, для того чтобы искать по ним определенные данные, а так же служат внешними ключами для других таблиц. Внешние ключи образуют связи. Таким образом, каждому полю таблицы, будут соответствовать данные другой таблицы. Например, у каждого товара есть страна производитель, и данные об этой стране находятся в таблице “Страна”. Далее каждому производителю, соответствует определенная страна, а в свою очередь у товара есть определенный производитель. Тем самым, мы получаем страну производителя товара. Связи в БД очень важны для структурирования данных. Таблица “Товар” связанна с таблицей “Категория” по ее уникальному идентификатору, и каждому товара соответствует определенная категория. Таблица “Товар”, так же связана с таблицами: “Филиал”, “Производитель”, “Поставка ”. Таблица “Производитель” связана с таблицей “Страна”. Таблица “Поставка” связана с таблицей “Поставщик”.

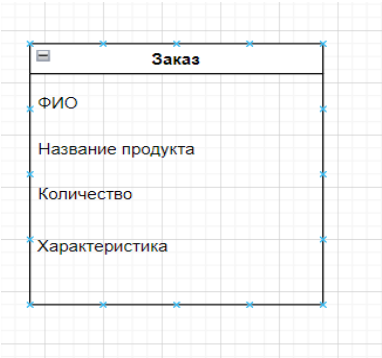


Рисунок-2.7 – Таблица заказов на закуп товара

На рисунке 2.7 изображена таблица в которую будут заноситься данные о заказах на закуп недостающих товаров на складе.

Диаграмма активности отображает поток активностей системы в бизнес процессе.

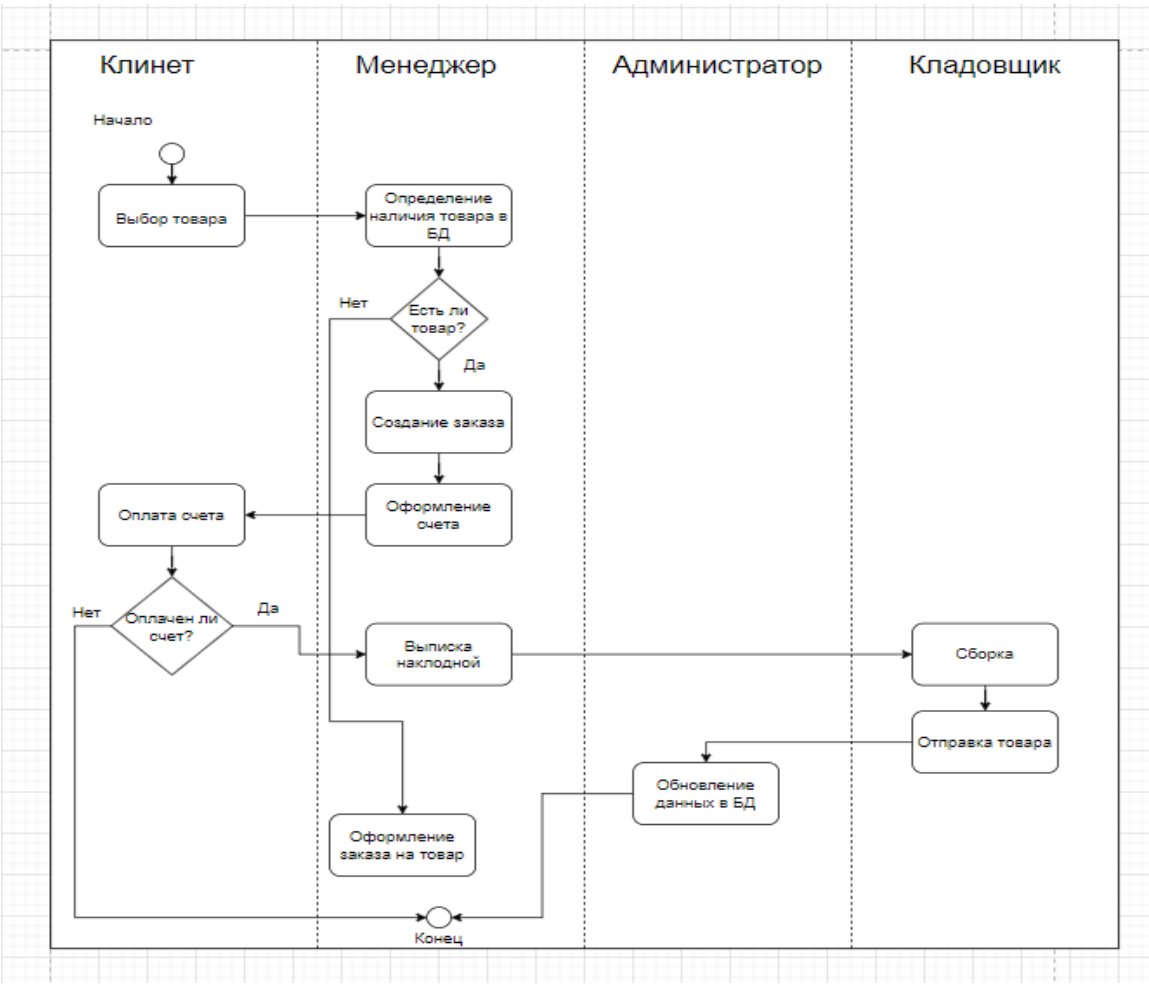


Рисунок-2.8 – Диаграмма активностей

Рисунок 2.8 показывает все активности системы. Клиент выбирает товар, далее менеджером проверяется его наличия и в зависимости от этого выбирается дальнейшее решение. Если товара нет, то оформляется его заказ. Если товар есть, создается заказ, оформляется счет. Далее проверяется выплата клиента этого счета, если счет не оплачивается, то процесс заканчивается. В обратном случае, идет выписка накладной, далее кладовщик начинает сборку заказа и отправляет его. Администратор фиксирует все изменения с товаром в базе данных, после чего процесс завершается.

2.4 Описание системы

Пользовательский интерфейс – это визуальное составляющие, которое обеспечивает обмен информацией между пользователем и ПО.

Запуская WEB приложение пользователь попадает на следующую страницу:

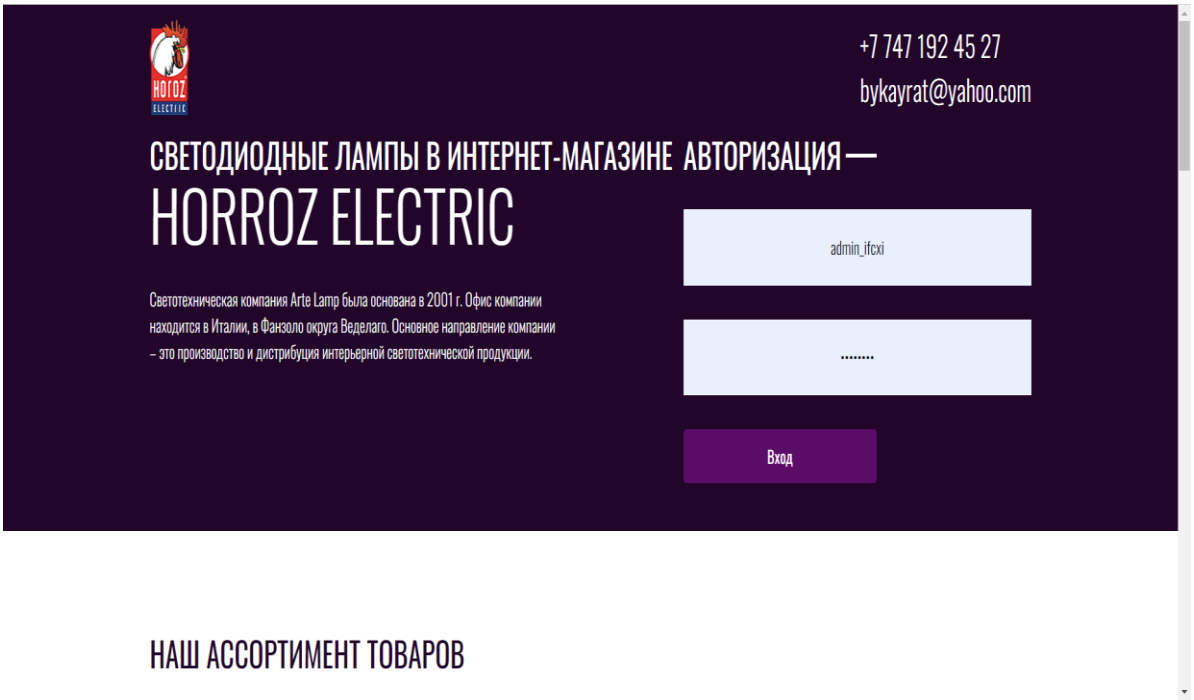


Рисунок-2.9 – Интерфейс первой страницы 1

Рисунок-2.10 – Интерфейс первой страницы 2

Рисунок-2.11 – Интерфейс первой страницы 3

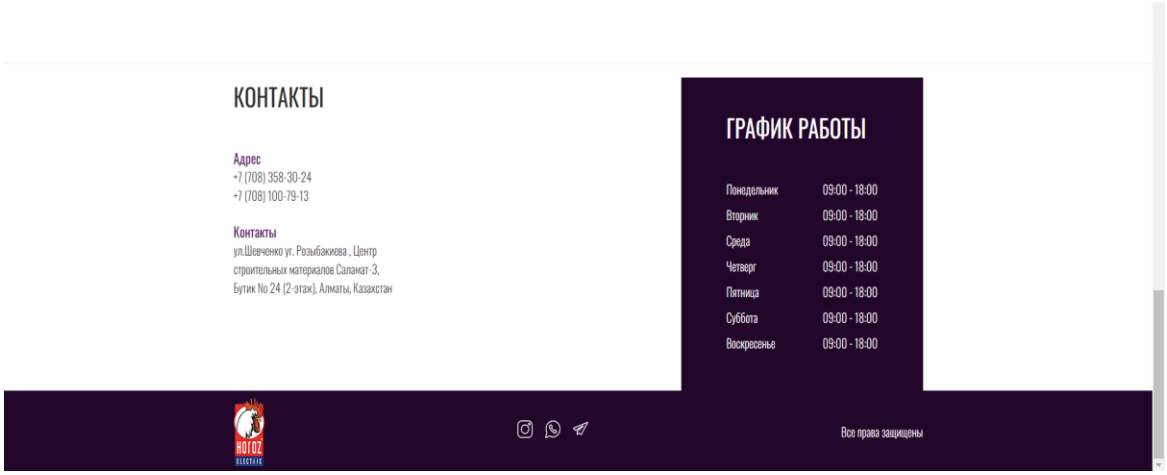


Рисунок-2.12 – Интерфейс первой страницы 4

На главной странице будет реализован вход для менеджера где он вводит логин и пароль для перехода на основную страницу. Также на этой странице есть слайдер с ассортиментом товара предприятия. Далее будет показан список и фотографии работ, а так же информация о компании. В конце будут отображенные контактные данные и график работ. Далее будут переходы для связи с компанией в виде whatsapp, Instagram, почта.

После авторизации менеджер попадает на следующую страницу:

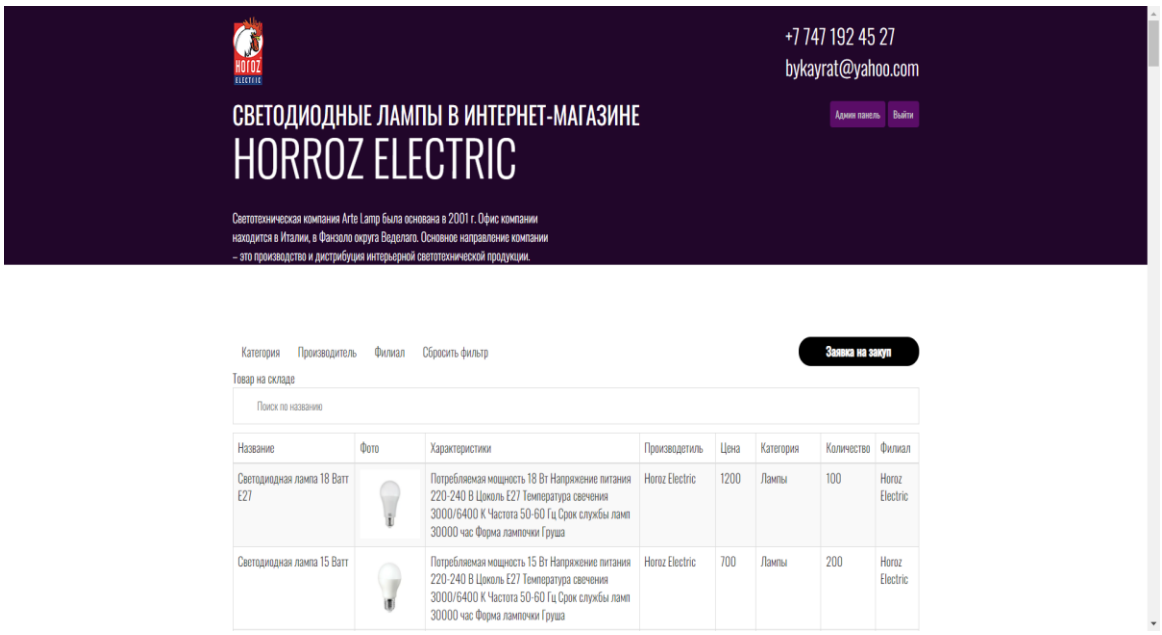


Рисунок-2.13 – Интерфейс страницы менеджера 1

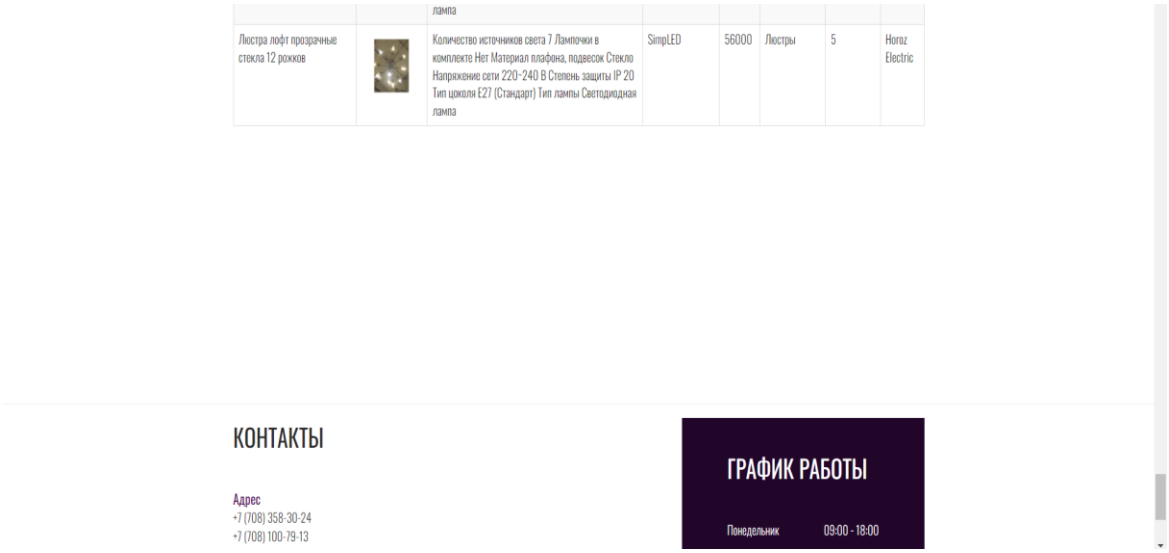


Рисунок-2.14 – Интерфейс страницы менеджера 2

На этой странице реализован переход в админ панель, выход из аккаунта, поиск товара в бд по наименованию. Дополнительная функция упрощающая поиск, это фильтр товара по категориям, производителям, филиалу, а так же можно сбросить заданные фильтры специальной кнопкой. На странице присутствует переход на форму для заполнения заказа на закуп товара, которого не хватает или нет в наличии.

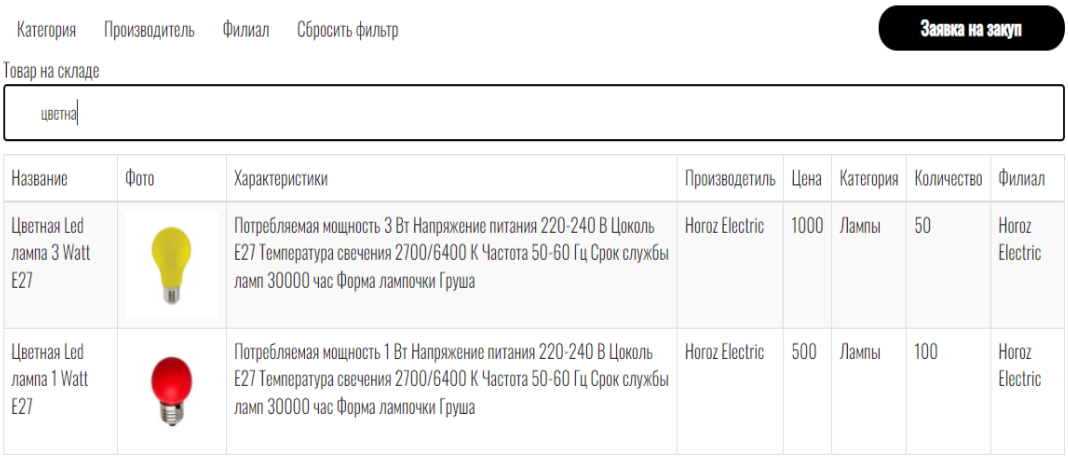


Рисунок-2.15 – Пример поиска товара по названию

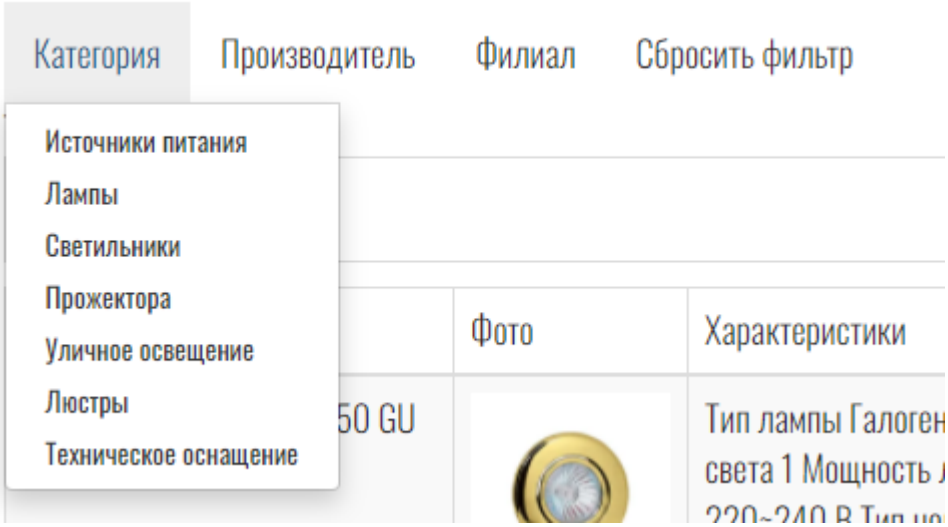
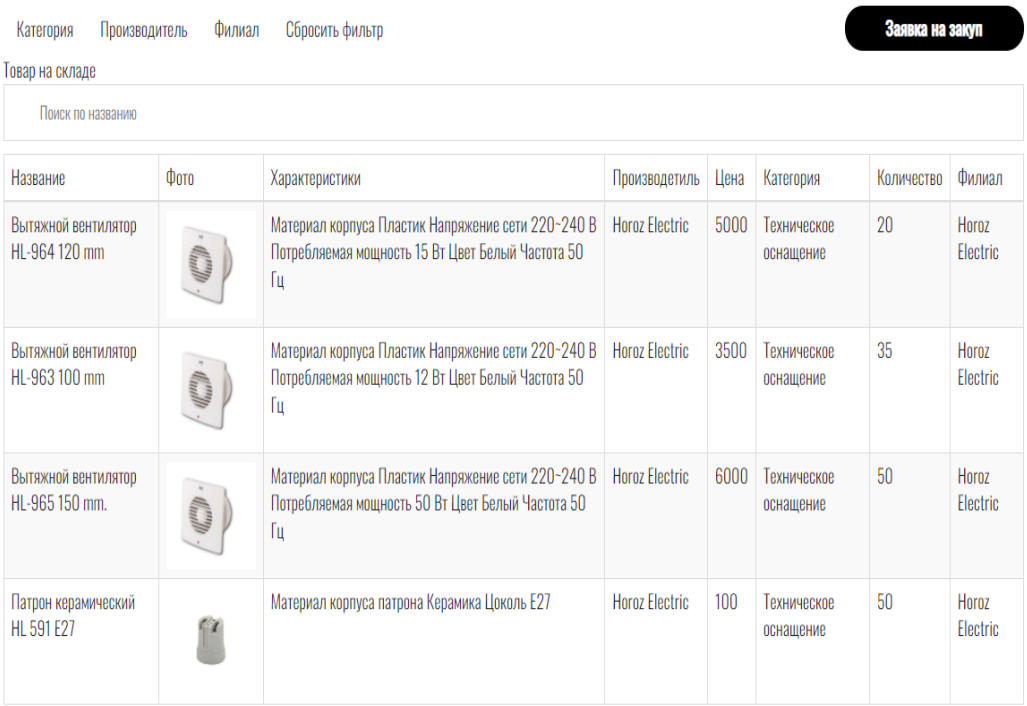


Рисунок-2.16 – Пример фильтрации товара по категории



Название	Фото	Характеристики	Производетиль	Цена	Категория	Количество	Филиал
Вытяжной вентилятор HL-964 120 mm		Материал корпуса Пластик Напряжение сети 220-240 В Потребляемая мощность 15 Вт Цвет Белый Частота 50 Гц	Horoz Electric	5000	Техническое оснащение	20	Horoz Electric
Вытяжной вентилятор HL-963 100 mm		Материал корпуса Пластик Напряжение сети 220-240 В Потребляемая мощность 12 Вт Цвет Белый Частота 50 Гц	Horoz Electric	3500	Техническое оснащение	35	Horoz Electric
Вытяжной вентилятор HL-965 150 mm.		Материал корпуса Пластик Напряжение сети 220-240 В Потребляемая мощность 50 Вт Цвет Белый Частота 50 Гц	Horoz Electric	6000	Техническое оснащение	50	Horoz Electric
Патрон керамический HL 591 E27		Материал корпуса патрона Керамика Цоколь E27	Horoz Electric	100	Техническое оснащение	50	Horoz Electric

Рисунок-2.17 – Пример фильтрации товара по категории “Техническое оснащение”

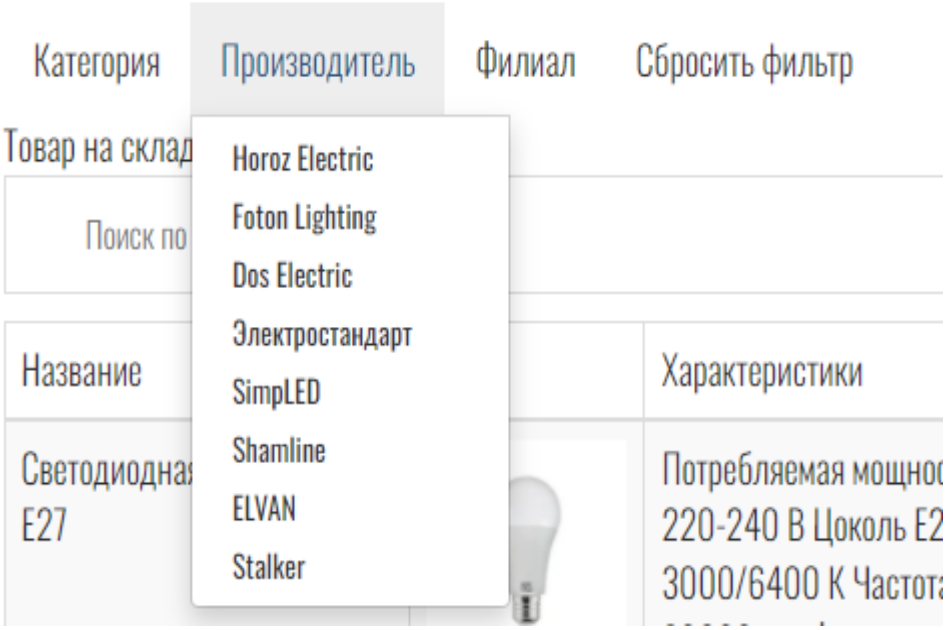


Рисунок-2.18 – Пример фильтрации товара по производителю

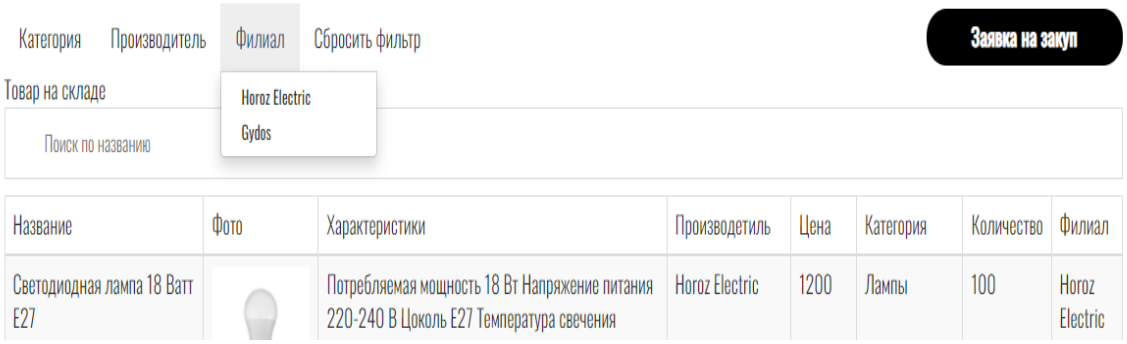


Рисунок-2.19 – Пример фильтрации товара по филиалу

Далее менеджер может оставить заявку на закуп товара на следующей странице:

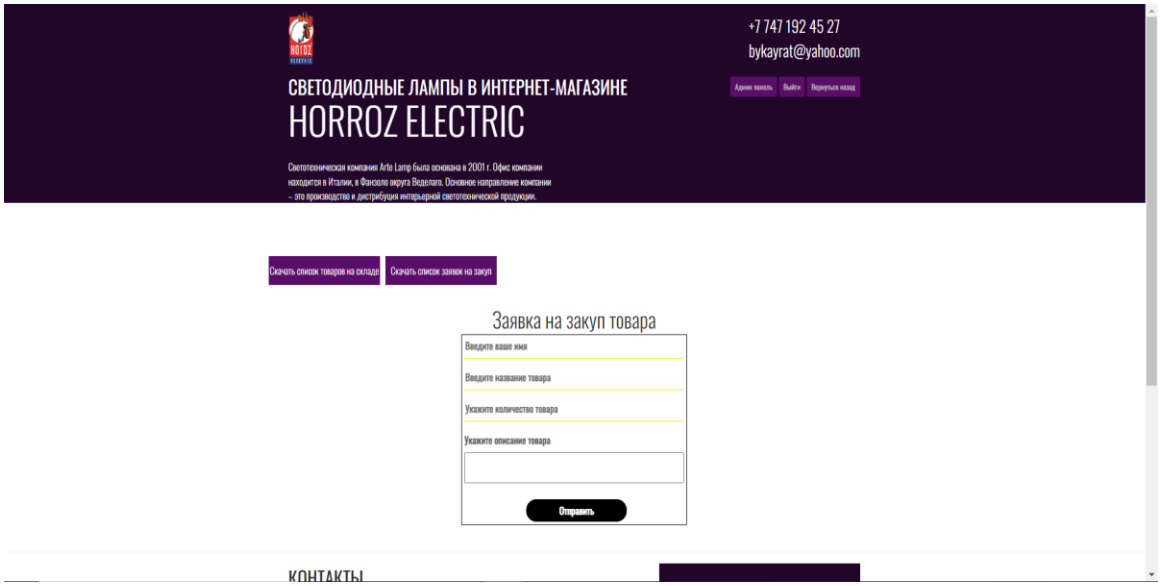
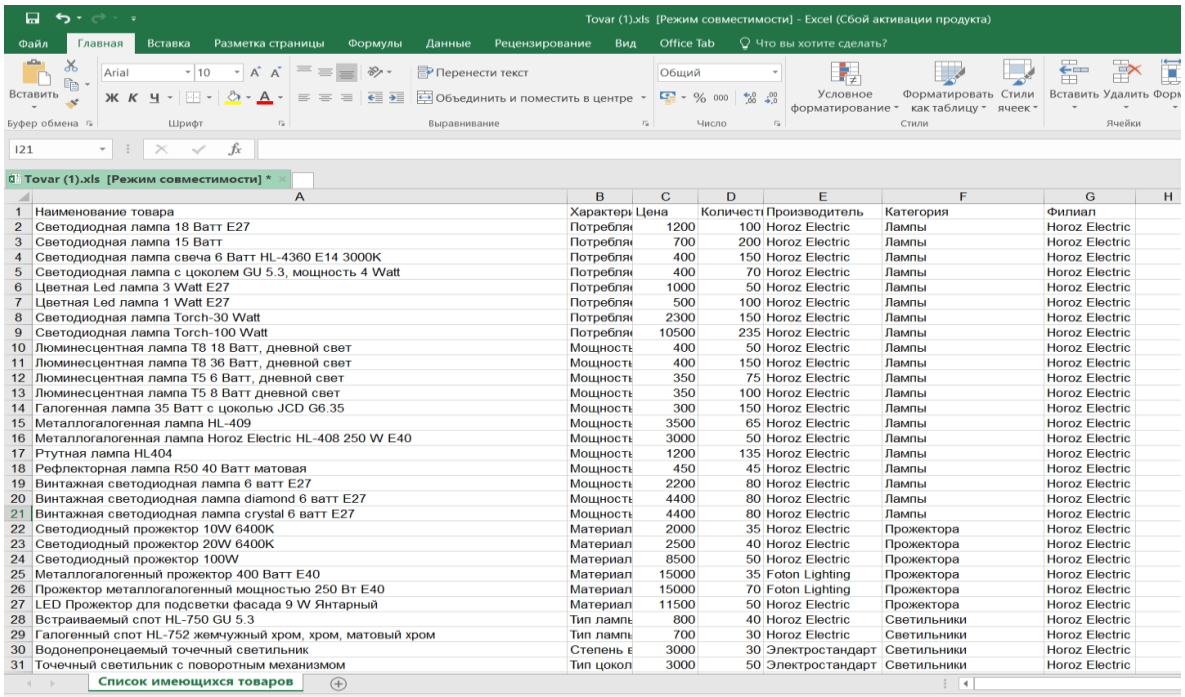


Рисунок-2.20 – Страница для заявки на закуп товара

Здесь менеджер заполняет форму для заявки, прописывает все поля из рисунка 2.20 , а также может скачать Excel файлы по данным о товарах находящихся на складе, список действующих заявок на закуп.



Tovar (1).xls [Режим совместимости] - Excel (Сбой активации продукта)

	A	B	C	D	E	F	G	H
1	Наименование товара	Характери	Цена	Количеств	Производитель	Категория	Филиал	
2	Светодиодная лампа 18 Ватт E27	Потребляе	1200	100	Horoz Electric	Лампы	Horoz Electric	
3	Светодиодная лампа 15 Ватт	Потребляе	700	200	Horoz Electric	Лампы	Horoz Electric	
4	Светодиодная лампа свеча 6 Ватт HL-4360 E14 3000K	Потребляе	400	150	Horoz Electric	Лампы	Horoz Electric	
5	Светодиодная лампа с цоколем GU 5.3, мощность 4 Watt	Потребляе	400	70	Horoz Electric	Лампы	Horoz Electric	
6	Цветная Led лампа 3 Watt E27	Потребляе	1000	50	Horoz Electric	Лампы	Horoz Electric	
7	Цветная Led лампа 1 Watt E27	Потребляе	500	100	Horoz Electric	Лампы	Horoz Electric	
8	Светодиодная лампа Torch-30 Watt	Потребляе	2300	150	Horoz Electric	Лампы	Horoz Electric	
9	Светодиодная лампа Torch-100 Watt	Потребляе	10500	235	Horoz Electric	Лампы	Horoz Electric	
10	Люминесцентная лампа T8 18 Ватт, дневной свет	Мощность	400	50	Horoz Electric	Лампы	Horoz Electric	
11	Люминесцентная лампа T8 36 Ватт, дневной свет	Мощность	400	150	Horoz Electric	Лампы	Horoz Electric	
12	Люминесцентная лампа T5 6 Ватт, дневной свет	Мощность	350	75	Horoz Electric	Лампы	Horoz Electric	
13	Люминесцентная лампа T5 8 Ватт дневной свет	Мощность	350	100	Horoz Electric	Лампы	Horoz Electric	
14	Галогенная лампа 35 Ватт с цоколью JCD G6.35	Мощность	300	150	Horoz Electric	Лампы	Horoz Electric	
15	Металлогалогенная лампа HL-409	Мощность	3500	65	Horoz Electric	Лампы	Horoz Electric	
16	Металлогалогенная лампа Horoz Electric HL-408 250 W E40	Мощность	3000	50	Horoz Electric	Лампы	Horoz Electric	
17	Ртутная лампа HL404	Мощность	1200	135	Horoz Electric	Лампы	Horoz Electric	
18	Рефлекторная лампа R50 40 Ватт матовая	Мощность	450	45	Horoz Electric	Лампы	Horoz Electric	
19	Винтажная светодиодная лампа 6 ватт E27	Мощность	2200	80	Horoz Electric	Лампы	Horoz Electric	
20	Винтажная светодиодная лампа diamond 6 ватт E27	Мощность	4400	80	Horoz Electric	Лампы	Horoz Electric	
21	Винтажная светодиодная лампа crystal 6 ватт E27	Мощность	4400	80	Horoz Electric	Лампы	Horoz Electric	
22	Светодиодный прожектор 10W 6400K	Материал	2000	35	Horoz Electric	Прожектора	Horoz Electric	
23	Светодиодный прожектор 20W 6400K	Материал	2500	40	Horoz Electric	Прожектора	Horoz Electric	
24	Светодиодный прожектор 100W	Материал	8500	50	Horoz Electric	Прожектора	Horoz Electric	
25	Металлогалогенный прожектор 400 Ватт E40	Материал	15000	35	Foton Lighting	Прожектора	Horoz Electric	
26	Прожектор металлогалогенный мощностью 250 Вт E40	Материал	15000	70	Foton Lighting	Прожектора	Horoz Electric	
27	LED Прожектор для подсветки фасада 9 W Янтарный	Материал	11500	50	Horoz Electric	Прожектора	Horoz Electric	
28	Встраиваемый спот HL-750 GU 5.3	Тип лампы	800	40	Horoz Electric	Светильники	Horoz Electric	
29	Галогенный спот HL-752 жемчужный хром, хром, матовый хром	Тип лампы	700	30	Horoz Electric	Светильники	Horoz Electric	
30	Водонепроницаемый точечный светильник	Степень з	3000	30	Электростандарт	Светильники	Horoz Electric	
31	Точечный светильник с поворотным механизмом	Тип цоколя	3000	50	Электростандарт	Светильники	Horoz Electric	

Список имеющихся товаров

Рисунок-2.21 – Excel отчет о товарах на складе

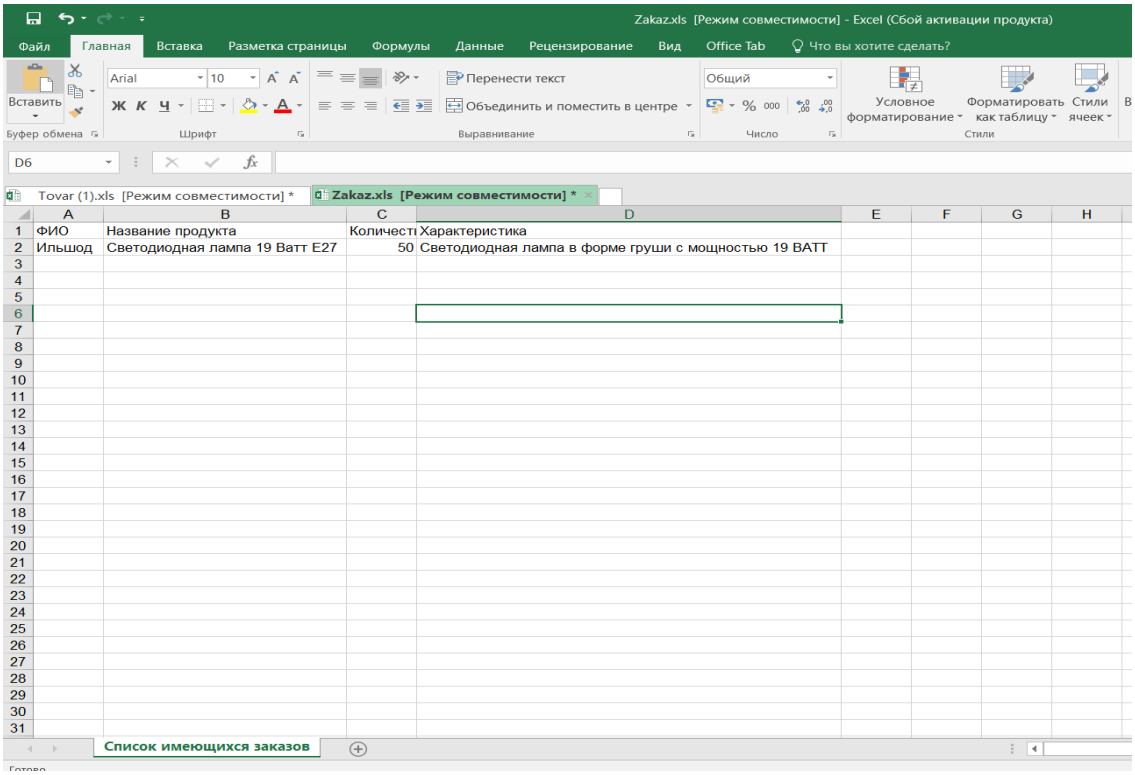


Рисунок-2.22 – Excel отчет о действующих заявок на закуп

Далее можно перейти на административную панель приложения пройдя авторизацию как администратор.

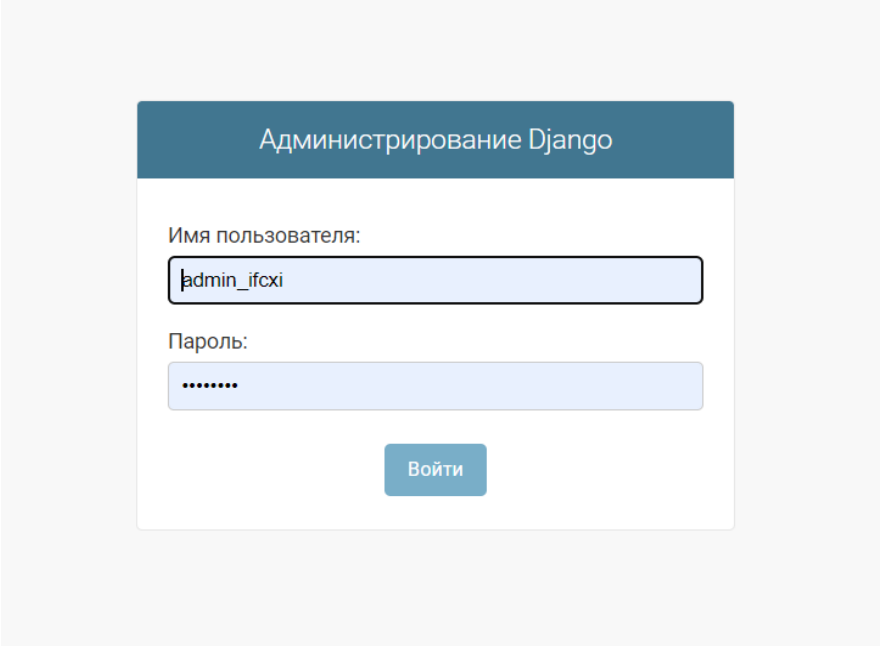


Рисунок-2.23 – Авторизация администратора

После авторизации, открывается следующая административная панель:

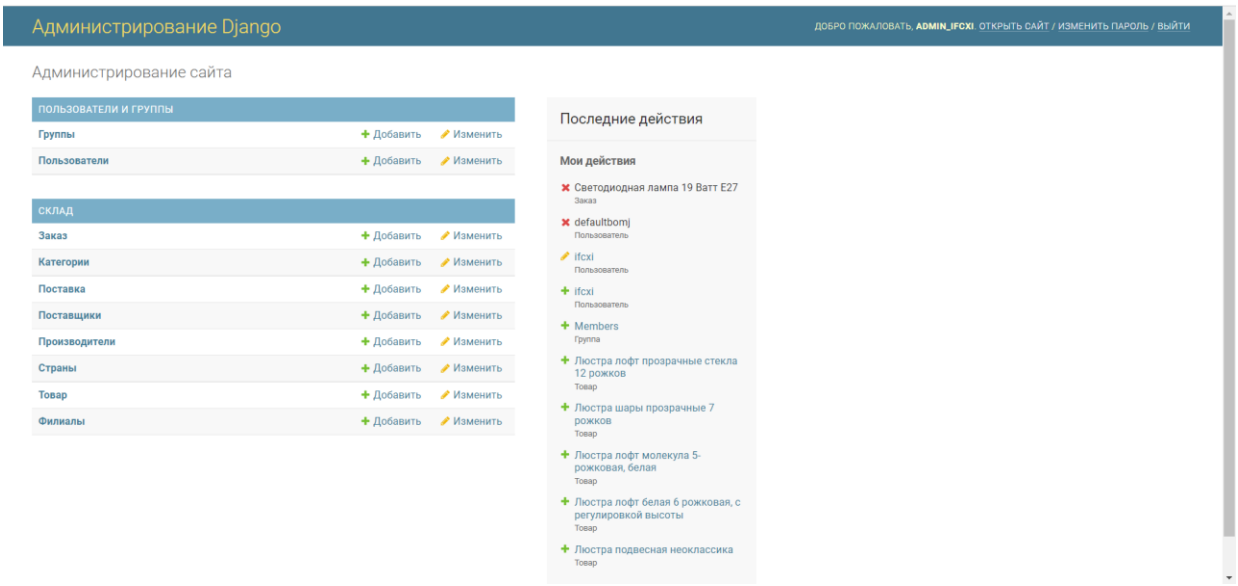


Рисунок-2.24 – Административная панель

На этой странице показаны таблицы из базы данных. Так же изображены данные о пользователях и группы к которым они могут относиться. Справа показаны все последний действия в административной панели. Так же сверху реализована навигация, благодаря которой можно вернуться назад и выйти из административной панели.

После перехода на один из разделов склада, можно увидеть следующий интерфейс:

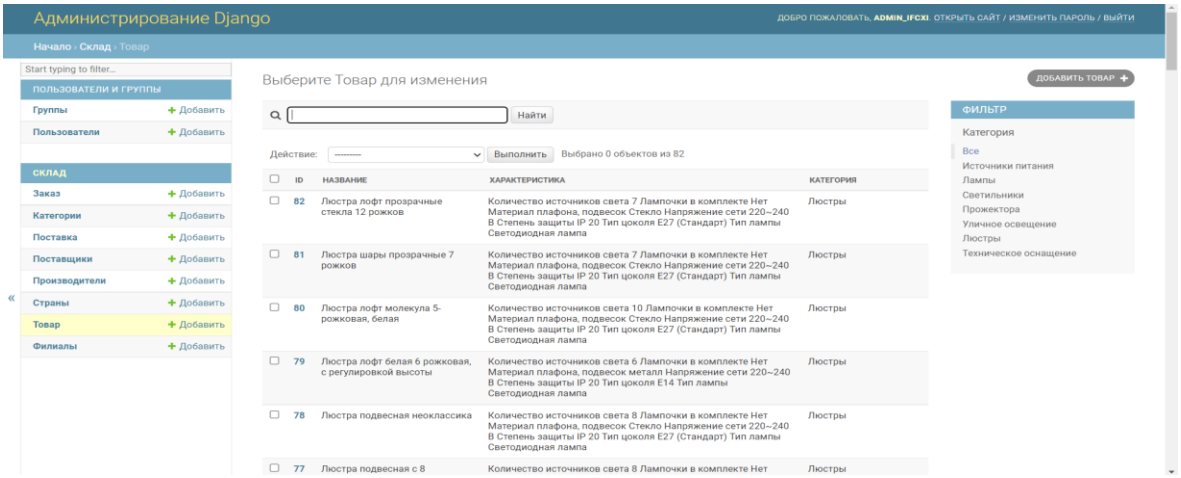


Рисунок-2.25 – Раздел товар

В этом разделе показаны все данные о товаре находящимся на складе. Сверху реализован поиск для быстрой навигации, так же фильтр с допустимыми действиями при выделении товара. Справа есть кнопка для добавления товара, а так же фильтрация по определенным параметрам для автоматизации навигации.

При переходе на добавление товара администратору отображается следующий интерфейс:

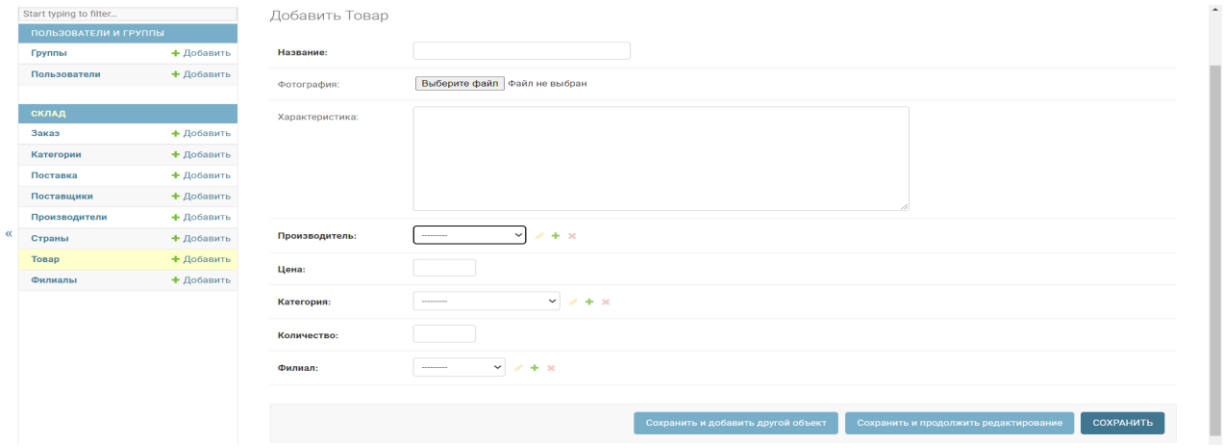


Рисунок-2.26 – Добавление товара в базу

В данной форме администратор заполняет все поля для добавления товара. В таких разделах как: производитель, категория, филиал реализован выпадающий список, где можно выбрать лишь допустимые значения с других связанных между собой таблиц. Так же администратор выбирает фотографию товара который будет отображен на странице менеджера.

Администратор может редактировать ранее добавленный товар:

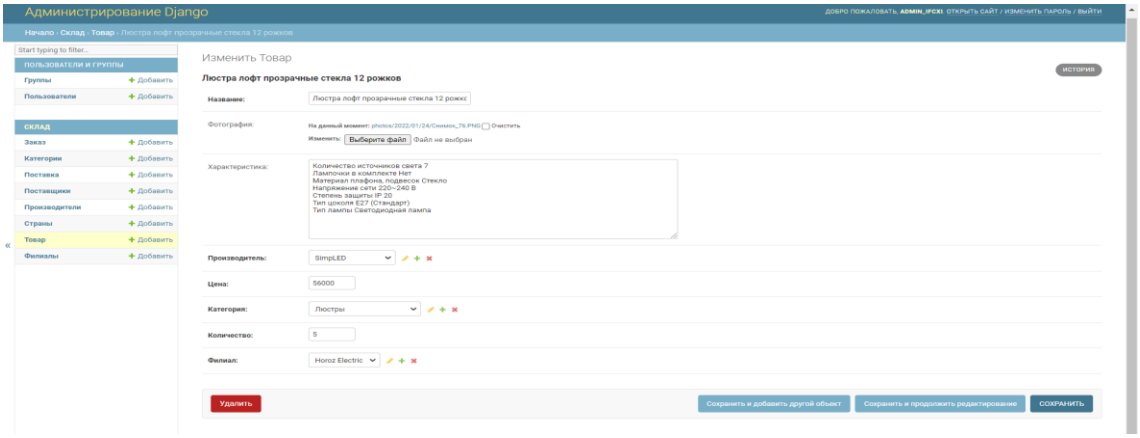


Рисунок-2.27 – Изменение ранее добавленного товара

При изменении каких либо данных о товаре, администратор исправляет их в базе, что показано на рисунке 2.27.

В административной панели, так же показаны действующие заказы на закупку определенного товара:

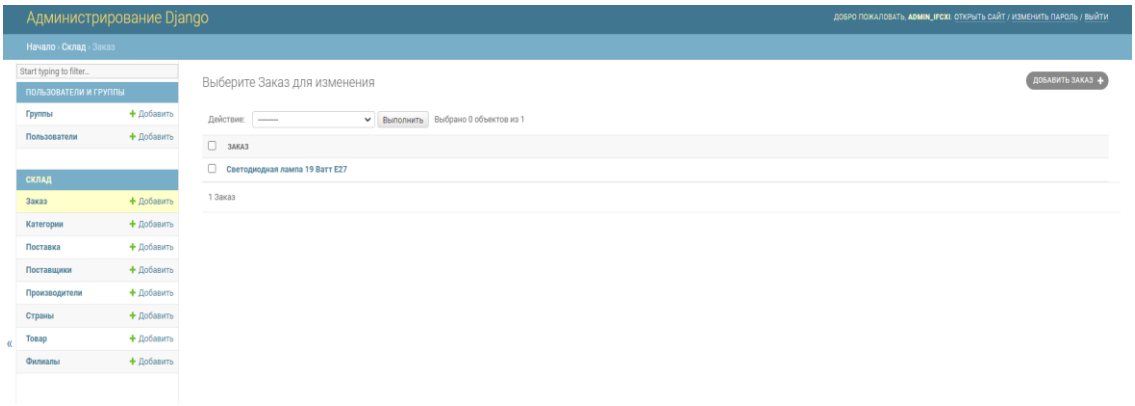


Рисунок-2.28 – Заявка на закуп товара

На рисунке 2.28 показан действующий заказ на закуп светодиодной лампы.

Так же доступ к данным о всей информации можно будет посмотреть в СУБД.

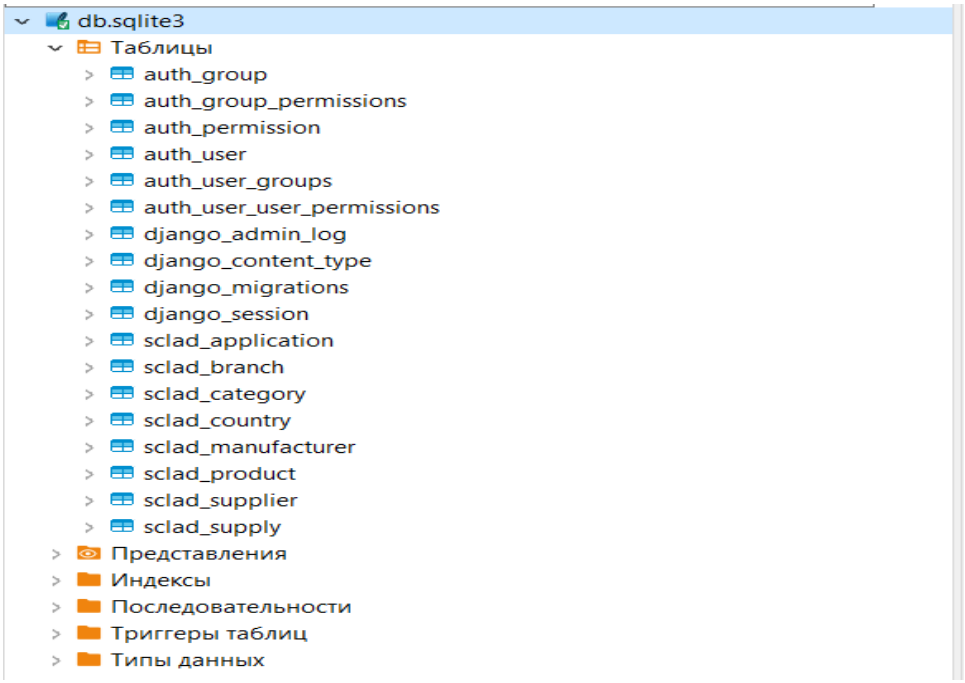


Рисунок-2.29 – Таблицы SQLite

На рисунке 2.29 показаны все таблицы приложения.

sclad_product

Свойства | Данные | Диаграмма | db.sqlite3 | Таблицы

Название: sclad_product Тип Таблицы: TABLE

Описание таблицы:

Колонки | Ключи | Внешние ключи | Индексы | Ссылки | Триггеры | DDL | Virtual

Название	#	Тип Данных	Длина	Not Null	Auto увеличение	По умолчанию	Описание
id	1	INTEGER		[v]	[v]		
title	2	VARCHAR	100	[v]	[ ]		
photo	3	VARCHAR	100	[v]	[ ]		
characteristic	4	TEXT		[v]	[ ]		
price	5	INTEGER		[v]	[ ]		
amount	6	INTEGER		[v]	[ ]		
time_create	7	DATETIME		[v]	[ ]		
time_update	8	DATETIME		[v]	[ ]		
branch_id	9	BIGINT		[v]	[ ]		
category_id	10	BIGINT		[v]	[ ]		
manufacturer...	11	BIGINT		[v]	[ ]		

Рисунок- 2.30 – Таблица Product

На рисунке 2.30 показан пример таблицы product. Так же показаны название полей таблицы и их тип данных. Далее можно посмотреть ограничения в таблицы, внешние ключи куда ссылается эта таблица и какие таблицы связанны с ней.

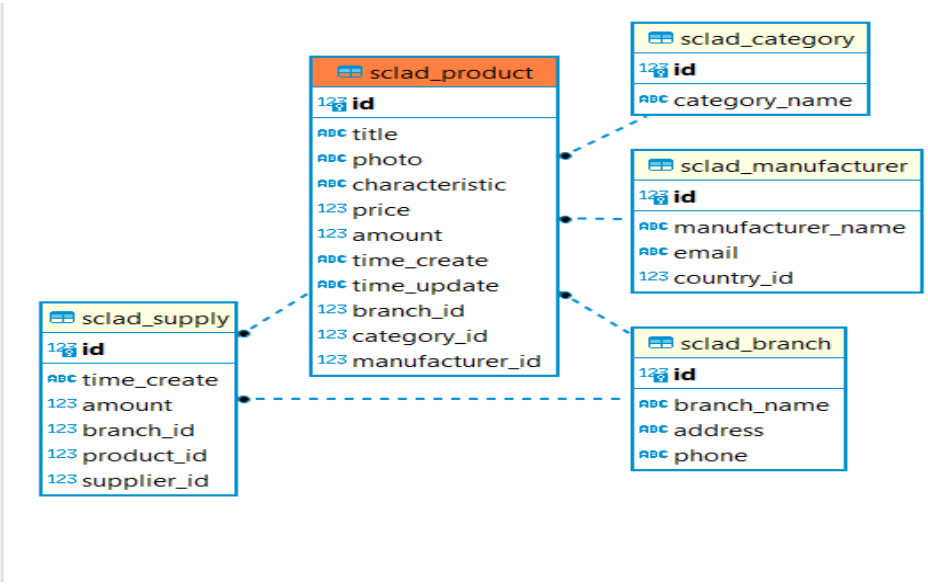
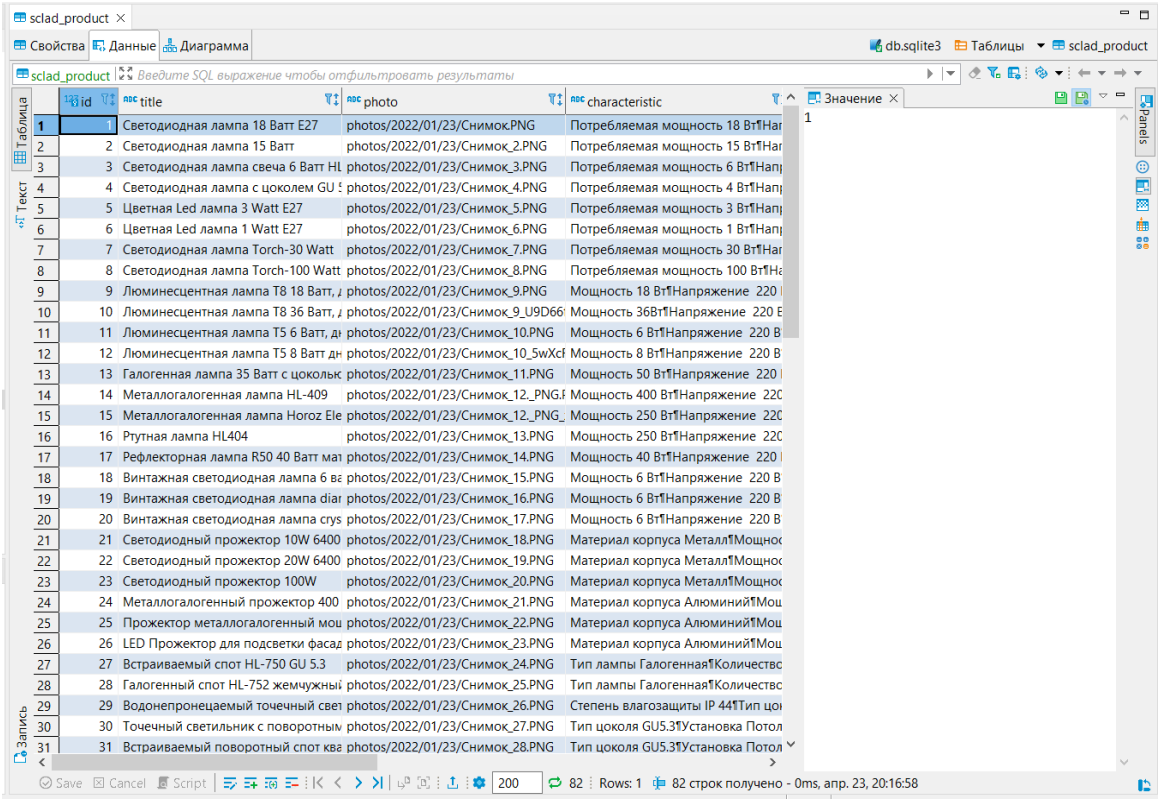


Рисунок-2.31 – Связи таблиц



	id	title	photo	characteristic
1	1	Светодиодная лампа 18 Ватт E27	photos/2022/01/23/Снимок.PNG	Потребляемая мощность 18 Вт¶Нап
2	2	Светодиодная лампа 15 Ватт	photos/2022/01/23/Снимок_2.PNG	Потребляемая мощность 15 Вт¶Нап
3	3	Светодиодная лампа свеча 6 Ватт HL	photos/2022/01/23/Снимок_3.PNG	Потребляемая мощность 6 Вт¶Напр
4	4	Светодиодная лампа с цоколем GU 5	photos/2022/01/23/Снимок_4.PNG	Потребляемая мощность 4 Вт¶Напр
5	5	Цветная Led лампа 3 Watt E27	photos/2022/01/23/Снимок_5.PNG	Потребляемая мощность 3 Вт¶Напр
6	6	Цветная Led лампа 1 Watt E27	photos/2022/01/23/Снимок_6.PNG	Потребляемая мощность 1 Вт¶Напр
7	7	Светодиодная лампа Torch-30 Watt	photos/2022/01/23/Снимок_7.PNG	Потребляемая мощность 30 Вт¶Нап
8	8	Светодиодная лампа Torch-100 Watt	photos/2022/01/23/Снимок_8.PNG	Потребляемая мощность 100 Вт¶На
9	9	Люминесцентная лампа T8 18 Ватт, д	photos/2022/01/23/Снимок_9.PNG	Мощность 18 Вт¶Напряжение 220
10	10	Люминесцентная лампа T8 36 Ватт, д	photos/2022/01/23/Снимок_9_U9D66	Мощность 36Вт¶Напряжение 220 В
11	11	Люминесцентная лампа T5 6 Ватт, дн	photos/2022/01/23/Снимок_10.PNG	Мощность 6 Вт¶Напряжение 220 В
12	12	Люминесцентная лампа T5 8 Ватт дн	photos/2022/01/23/Снимок_10_5wXcf	Мощность 8 Вт¶Напряжение 220 В
13	13	Галогенная лампа 35 Ватт с цоколь	photos/2022/01/23/Снимок_11.PNG	Мощность 50 Вт¶Напряжение 220
14	14	Металлогалогенная лампа HL-409	photos/2022/01/23/Снимок_12._PNG.f	Мощность 400 Вт¶Напряжение 220
15	15	Металлогалогенная лампа Horoz Ele	photos/2022/01/23/Снимок_12._PNG_	Мощность 250 Вт¶Напряжение 220
16	16	Ртутная лампа HL404	photos/2022/01/23/Снимок_13.PNG	Мощность 250 Вт¶Напряжение 220
17	17	Рефлекторная лампа R50 40 Ватт мат	photos/2022/01/23/Снимок_14.PNG	Мощность 40 Вт¶Напряжение 220
18	18	Винтажная светодиодная лампа 6 ва	photos/2022/01/23/Снимок_15.PNG	Мощность 6 Вт¶Напряжение 220 В
19	19	Винтажная светодиодная лампа diar	photos/2022/01/23/Снимок_16.PNG	Мощность 6 Вт¶Напряжение 220 В
20	20	Винтажная светодиодная лампа crys	photos/2022/01/23/Снимок_17.PNG	Мощность 6 Вт¶Напряжение 220 В
21	21	Светодиодный прожектор 10W 6400	photos/2022/01/23/Снимок_18.PNG	Материал корпуса Металл¶Мощнос
22	22	Светодиодный прожектор 20W 6400	photos/2022/01/23/Снимок_19.PNG	Материал корпуса Металл¶Мощнос
23	23	Светодиодный прожектор 100W	photos/2022/01/23/Снимок_20.PNG	Материал корпуса Металл¶Мощнос
24	24	Металлогалогенный прожектор 400	photos/2022/01/23/Снимок_21.PNG	Материал корпуса Алюминий¶Мощ
25	25	Прожектор металлогалогенный мощ	photos/2022/01/23/Снимок_22.PNG	Материал корпуса Алюминий¶Мощ
26	26	LED Прожектор для подсветки фасад	photos/2022/01/23/Снимок_23.PNG	Материал корпуса Алюминий¶Мощ
27	27	Встраиваемый спот HL-750 GU 5.3	photos/2022/01/23/Снимок_24.PNG	Тип лампы Галогенная¶Количество
28	28	Галогенный спот HL-752 жемчужный	photos/2022/01/23/Снимок_25.PNG	Тип лампы Галогенная¶Количество
29	29	Водонепроницаемый точечный свет	photos/2022/01/23/Снимок_26.PNG	Степень влагозащиты IP 44¶Тип цо
30	30	Точечный светильник с поворотным	photos/2022/01/23/Снимок_27.PNG	Тип цоколя GU5.3¶Установка Потол
31	31	Встраиваемый поворотный спот ква	photos/2022/01/23/Снимок_28.PNG	Тип цоколя GU5.3¶Установка Потол

Рисунок-2.32 – Данные в таблице продукт

Через СУБД можно просматривать, изменять и добавлять новые данные. Так же можно писать SQL запросы в зависимости от конечной информации которую хотим получить.

ЗАКЛЮЧЕНИЕ

В ходе данного проекта было разработано Web-приложение предназначенное для автоматизации работа склада электротоваров. Данное ПО полностью соответствует поставленным целям, которые были успешно достигнуты. Учет товара на складе введется непосредственно в приложении, где так же есть поиск товара по базе данных, различная фильтрация и форма для заявки на закуп товара. Различные манипуляции с данными происходят в административной панели, так же в приложении реализована аутентификация пользователя и распечатка отчетов в Excel файлах.

При подготовке к реализации проекта, была изучена предметная область автоматизации складов. В ходе изучения, было выявлено, что наличие автоматизированной системы работы склада, значительно улучшает эффективность работы предприятия, а так же ускоряет все ее процессы.

На этапе разработки и проектирования проекта, были изучены различные технологии для реализации приложения. После чего был выбран наиболее эффективный стек технологий, для того чтобы оптимизировать все процессы разработки.

Web-приложение для автоматизации работы склада электротоваров, является законченным продуктом с понятным и удобным интерфейсом. Приложение способно облегчить многие процессы работы склада, тем самым повысить эффективность предприятия.

Список использованной литературы

1 Автоматизированные складские системы: виды, преимущества, особенности и сферы применения // Электронная версия на сайте https://www.ekam.ru/blogs/pos/avtomatizirovannye-skladskie-sistemy.

2 Автоматизированная система // Электронная версия на сайте https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0.

3 PyCharm // Электронная версия на сайте https://www.jetbrains.com/ru-ru/pycharm/ .

4 Что же представляет из себя Django // Электронная версия на сайте https://tutorial.djangogirls.org/ru/django/.

5 Что такое SQLite и зачем она нужна ? // Электронная версия на сайте https://unetway.com/tutorial/sqlite.

6 MVC для веб: проще некуда // Электронная версия на сайте https://habr.com/ru/post/181772/.

7 Архитектура SQLite // Электронная версия на сайте https://russianblogs.com/article/1781106567/.

8 Архитектура приложения // Электронная версия на сайте https://proglib.io/p/app-architecture-mom#:~:text=%D0%9D%D0%BE%20%D0%B5%D1%81%D0%BB%D0%B8%20%D0%BA%D0%BE%D1%80%D0%BE%D1%82%D0%BA%D0%BE%2C%20%D0%B0%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0%20%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F,%D0%BF%D0%BE%D0%B4%D1%85%D0%BE%D0%B4%D0%B5%20%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D0%B8%D0%B8%20%D0%BA%20%D0%B5%D1%91%20%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B5.

9 Диаграмма вариантов использования // Электронная версия на сайте https://flexberry.github.io/ru/fd_use-case-diagram.html.

10 Краткое описание BPMN // Электронная версия на сайте https://habr.com/ru/company/trinion/blog/331254/.

11 Диаграмма активностей // Электронная версия на сайте https://flexberry.github.io/ru/fd_activity-diagram.html.

12 Документация Django // Электронная версия на сайте https://www.djangoproject.com/.

13 ТОП-10 программ складского учета // Электронная версия на сайте https://mertech.ru/blog/vse-o-markirovke/top-10-programm-dlja-skladskogo-

ucheta/.
14 Основные этапы разработки web-приложений // Электронная версия на сайте http://www.rusnauka.com/16_ADEN_2011/Informatica/3_85389.doc.htm.
15 Дронов В. Django 2.1. Практика создания веб-сайтов на Python // Электронная версия на сайте https://ru.pdfdrive.com/django-21-%D0%9F%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D0%BA%D0%B0-%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D1%8F-%D0%B2%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82%D0%BE%D0%B2-%D0%BD%D0%B0-python-e184062061.html.
16 Нормализация баз данных простыми словами // Электронная версия на сайте https://info-comp.ru/database-normalization.
17 Создание Django API используя Django Rest Framework часть 1 // Электронная версия на сайте https://webdevblog.ru/sozdanie-django-api-ispolzuya-django-rest-framework-apiview/.
18 CRUD. Операции с моделями // Электронная версия на сайте https://metanit.com/python/django/5.3.php.
19 Основные возможности библиотеки Python Imaging Library / Pillow / PIL // Электронная версия на сайте https://pythonru.com/biblioteki/osnovnye-vozmozhnosti-biblioteki-python-imaging-library-pillow-pil.
20 Документация Django // Электронная версия на сайте https://djbook.ru/rel3.0/.
21 Связи Python: Django ORM // Электронная версия на сайте https://ru.hexlet.io/courses/python-django-orm/lessons/relations/theory_unit.

Приложение А
(обязательное)

Техническое задание

А1.1 Техническое задание на разработку автоматизированной системы склада электротоваров

Настоящее техническое задание распространяется на разработку автоматизированной системы работы склада электротоваров. Автоматизированная система позволит улучшить все процессы работы склада, а так же увеличит уровень эффективности предприятия. Система обеспечит быстрое получение данных о товарах хранящихся на складе, тем самым сократив процесс поиска нужного товара.

А1.1.1 Основания для разработки

Система разрабатывается на основании устного распоряжения научного руководителя для создания автоматизированной системы работы склада электротоваров.

А.1.1.2 Назначение

Разрабатываемый проект предназначен для сбора, хранения и обработки информации о товарах находящихся на складе.

А.1.1.3 Требования к функциональным характеристикам

Система должна обеспечить возможность выполнения следующих

функций:
- хранения данных о товарах;
- поиск имеющихся товаров на складе по базе данных;
- фильтр товаров по категории, производителю, филиалу;
- добавление заказа на закуп товара;
- изменение информации о товарах;
- добавление и удаление товаров;
- добавление сотрудников в базу данных;
- генерация списка имеющегося товара на складе в файле Excel;
- генерация отчета о текущих заказов на закуп товара в Excel.

А.1.1.4 Требования к надежности

Обеспечить конфиденциальность данных предприятия о товарах. Предусмотреть блокировку действий пользователей по ролям.

А.1.1.5 Требования к составу и параметрам технических средств

Система должна работать на всех персональных компьютерах и различных браузерах. Минимальные требования: тип процессора – Windows XP и выше, оперативная память – 64 Мб RAM и выше.

А.1.1.6 Требования к информационной и программной совместимости

Система должна работать под управлением операционных систем семейства Microsoft Windows.

Приложение Б
(обязательное)

Текст программы

```
# Модели
from django.db import models
from django.core.validators import MinValueValidator, MaxValueValidator

class Country(models.Model):
    country_name = models.CharField(max_length=100
,verbose_name='Страна')


    class Meta:
        verbose_name = 'Страны'
        verbose_name_plural = 'Страны'

    def __str__(self):
        return self.country_name

class Manufacturer(models.Model):
    manufacturer_name = models.CharField(max_length=100,
verbose_name='Производитель')
    email = models.CharField(max_length=100)
    country = models.ForeignKey(Country , on_delete =
models.PROTECT,verbose_name='Страна производитель')

    class Meta:
        verbose_name = 'Производители'
        verbose_name_plural = 'Производители'

    def __str__(self):
        return self.manufacturer_name

class Category(models.Model):
    category_name = models.CharField(max_length = 100, verbose_name=
'Категория товара')

    class Meta:
```

Продолжение приложения Б

```
            verbose_name = 'Категории'
            verbose_name_plural = 'Категории'

        def __str__(self):
            return self.category_name


    class Branch(models.Model):
        branch_name = models.CharField(max_length = 100,
verbose_name='Филиал')
        address = models.CharField(max_length = 100, verbose_name=' Адрес')
        phone = models.CharField(max_length = 100, verbose_name='Номер
телефона')

        class Meta:
            verbose_name = 'Филиалы'
            verbose_name_plural = 'Филиалы'

        def __str__(self):
            return self.branch_name


    class Supplier(models.Model):
        supplier_name = models.CharField(max_length = 100,
verbose_name='Поставщик')
        way = models.CharField(max_length = 100, verbose_name='Способ
поставки')

        class Meta:
            verbose_name = 'Поставщики'
            verbose_name_plural = 'Поставщики'

        def __str__(self):
            return self.supplier_name


    class Product(models.Model):
        title = models.CharField(max_length=100,verbose_name='Название')
```

Продолжение приложения Б

```
        photo =
models.ImageField(upload_to="photos/%Y/%m/%d/",verbose_name='Фотография',b
lank=True)
        characteristic =
models.TextField(blank=True,verbose_name='Характеристика')
        manufacturer = models.ForeignKey(Manufacturer, on_delete=
models.PROTECT,verbose_name='Производитель')
        price = models.IntegerField(verbose_name='Цена')
        category = models.ForeignKey(Category , on_delete=
models.PROTECT,verbose_name='Категория')
        amount = models.IntegerField(verbose_name='Количество')
        branch = models.ForeignKey(Branch , on_delete=
models.PROTECT,verbose_name='Филиал')
        time_create = models.DateTimeField(auto_now_add=
True,verbose_name='Время создания')
        time_update = models.DateTimeField(auto_now=
True,verbose_name='Время изменения')

        class Meta:
            verbose_name = 'Товар'
            verbose_name_plural = 'Товар'

        def __str__(self):
            return self.title

    class Supply(models.Model):
        branch = models.ForeignKey(Branch,
on_delete=models.PROTECT,verbose_name='Филиал')
        time_create =
models.DateTimeField(auto_now_add=True,verbose_name='Время создания')
        amount = models.IntegerField(verbose_name='Количество')
        supplier = models.ForeignKey(Supplier,
on_delete=models.PROTECT,verbose_name='Поставщик')
        product = models.ForeignKey(Product,
on_delete=models.PROTECT,verbose_name='Товар')

        class Meta:
            verbose_name = 'Поставка'
```

Продолжение приложения Б

```
verbose_name_plural = 'Поставка'


class application(models.Model):
    name_of_manager = models.CharField(max_length=50,
verbose_name='ФИО')
    title_of_product = models.CharField(max_length=100,
verbose_name='Название продукта')
    amount_of_product = models.IntegerField(verbose_name='Количество')
    characteristic_of_product = models.TextField(blank=True,
verbose_name='Характеристика')

    class Meta:
        verbose_name = 'Заказ'
        verbose_name_plural = 'Заказ'

    def __str__(self):
        return self.title_of_product

# URL
from django.urls import path
from .views import *
from django.conf.urls.static import static
from django.conf import settings
from django.contrib import admin

urlpatterns = [
   path('', IndexView.as_view() , name= 'vhod'),
   path('about/', about),
   path('cats/', categories),
   path('main/', main,  name='main'),
   path('register/', register, name='register' ),
   path('logout/', logout_view, name='logout',),
   path('category/<int:id>', category, name='category'),
   path('manufacturer/<int:id>', manufacturer, name='manufacturer'),
   path('branch/<int:id>', branch, name='branch'),
   path('form', Formpost.as_view(), name='form'),
   path('report1/',MakeReport1.as_view(), name= 'report1'),
```

```
  path('report2/',MakeReport2.as_view(), name= 'report2'),

]
urlpatterns += static(settings.MEDIA_URL, document_root =
settings.MEDIA_ROOT)

# Представления
from unicodedata import category
from django.shortcuts import render, redirect
from django.http import HttpResponse
from .models import *
from rest_framework.views import APIView
from django.contrib.auth.models import User
from django.contrib.auth import authenticate, logout
from django.contrib.auth import login as auth_login
import xlwt

class IndexView(APIView):
    def get(self, request, *args, **kwargs):
        return render(request, 'sclad/index.html', {'title':'Вход'})

    def post(self, request, *args, **kwargs):
        username = request.POST.get('username')
        password = request.POST.get('password')
        user = authenticate(request, username=username, password=password)
        print(user if user else 'oshibka')
        print(authenticate(request, username=username, password=password))
        if user is not None:
            auth_login(request, user)
            return redirect('main')
        else:
            return redirect('vhod')


def base(request):
    return render(request, 'sclad/base.html', {'title':'база'})

def categories(request):
```

```
    cats_list = Category.objects.all()
    return render(request, 'sclad/categories.html', {'cats_list': cats_list})

def about(request):
    return render(request, 'sclad/about.html', {'title':'О сайте'})

def main(request):
    if not request.user.is_anonymous:
        products = Product.objects.all()
        categories = Category.objects.all()
        manufacturers = Manufacturer.objects.all()
        branches = Branch.objects.all()
        return render(request, 'sclad/main.html', {'title':'Главная страница',
'products':products, 'categories': categories, 'manufacturers': manufacturers, 'branches':
branches})
    else:
        return redirect('/')


def category(request, id):
    products = Product.objects.filter(category=id)
    categories = Category.objects.all()
    manufacturers = Manufacturer.objects.all()
    branches = Branch.objects.all()
    return render(request, 'sclad/main.html', {'products': products, 'categories':
categories, 'manufacturers': manufacturers, 'branches': branches})


def manufacturer(request, id):
    products = Product.objects.filter(manufacturer=id)
    categories = Category.objects.all()
    manufacturers = Manufacturer.objects.all()
    branches = Branch.objects.all()
    return render(request, 'sclad/main.html', {'products': products, 'categories':
categories, 'manufacturers': manufacturers, 'branches': branches})


def branch(request, id):
```

Продолжение приложения Б

```
 products = Product.objects.filter(branch=id)
    categories = Category.objects.all()
    manufacturers = Manufacturer.objects.all()
    branches = Branch.objects.all()
    return render(request, 'sclad/main.html', {'products': products, 'categories':
categories, 'manufacturers': manufacturers, 'branches': branches})


def register(request):
    return HttpResponse("Авторизация")

def logout_view(request):
    logout(request)
    return render(request, 'sclad/index.html', {'title':'Вход'})

class Formpost(APIView):
    def get(self ,request , *args , **kwargs):
        return render(request, 'sclad/form.html', {'title':'Форма'})

    def post(self, request, *args, **kwargs):
        name_of_manager = request.POST.get('imya_menegera')
        title_of_product = request.POST.get('naz_tovar')
        amount_of_product = request.POST.get('kolvo_tovar')
        characteristic_of_product = request.POST.get('comment')
        new_model = application()
        new_model.name_of_manager = name_of_manager
        new_model.title_of_product = title_of_product
        new_model.amount_of_product = amount_of_product
        new_model.characteristic_of_product = characteristic_of_product
        new_model.save()
        return render(request, 'sclad/form.html', {'title':'Форма'})

class MakeReport1(APIView):
        def get(self ,request , *args , **kwargs):
         products = Product.objects.all()


         wb = xlwt.Workbook(encoding="utf-8")
```

Продолжение приложения Б

```
        ws = wb.add_sheet("Список имеющихся товаров")
            headers = ["Наименование
товара","Характеристика","Цена","Количество","Производитель","Категория","
Филиал"]
            for n, headers in enumerate(headers):
                ws.write(0,n,headers)
            row_num = 1
            for entry in products:
                ws.write(row_num,0,entry.title)
                ws.write(row_num,1,entry.characteristic)
                ws.write(row_num,2,entry.price)
                ws.write(row_num,3,entry.amount)
                ws.write(row_num,4,entry.manufacturer.manufacturer_name)
                ws.write(row_num,5,entry.category.category_name)
                ws.write(row_num,6,entry.branch.branch_name)

                row_num += 1

            response = HttpResponse(
              content_type="application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet",)
            response[
              "Content-Disposition"] = "attachment; filename*=utf-8''Tovar.xls"
            wb.save(response)
            return response
            return render(request, 'sclad/form.html', {'title':'Форма'})

        class MakeReport2(APIView):
           def get(self ,request , *args , **kwargs):
            appl = application.objects.all()


            wb = xlwt.Workbook(encoding="utf-8")
            ws = wb.add_sheet("Список имеющихся заказов")
            headers = ["ФИО","Название
продукта","Количество","Характеристика"]
            for n, headers in enumerate(headers):
                ws.write(0,n,headers)
```

Продолжение приложения Б

```
        row_num = 1
            for entry in appl:
                ws.write(row_num,0,entry.name_of_manager)
                ws.write(row_num,1,entry.title_of_product)
                ws.write(row_num,2,entry.amount_of_product)
                ws.write(row_num,3,entry.characteristic_of_product)

                row_num += 1

            response = HttpResponse(
              content_type="application/vnd.openxmlformats-
officedocument.spreadsheetml.sheet",)
            response[
              "Content-Disposition"] = "attachment; filename*=utf-8''Zakaz.xls"
            wb.save(response)
            return response
            return render(request, 'sclad/form.html', {'title':'Форма'})
```