# Глава 3. Разработка информационной системы e-agrologistic.uz для оптимизации логистики в сельском хозяйстве

# 3.1. Выбор инструментов и технологий

Требуется разработать CRM-систему для турагентства и личный кабинет с помощью веб-фреймворка Django, где CRM-менеджеры могут добавлять новые туры, отслеживать потенциальных заказчиков и вести заказы. Заказчики, в свою очередь, могут заказывать туры на сайте и просматривать свои заказы в личном кабинете.

Программное обеспечение должно обеспечивать следующие функциональные возможности:

1. Добавление данных в базу данных; 2. Считывание данных из базы данных;

3. Стилизация шаблонов с помощью CSS-фреймворка Bootstrap;

4. Динамическое изменение количества добавляемых заказов, количества человек в заказе и цены заказа, а также динамическое удаление заказа с помощью JQuery;

5. Отображение данных о заказе в личном кабинете;

6. Отображение данных о клиентах и потенциальных клиентах в административной части.

На сегодняшний день существует множество способов реализации CRM-системы с помощью веб-фреймворка, например, AllcountJS, Zend, Yii, Django и т. д. Для реализации своего проекта я выбрал именно веб-фреймворк Django версии 1.11. Django это свободный фреймворк для веб-приложений написанный на языке Python.

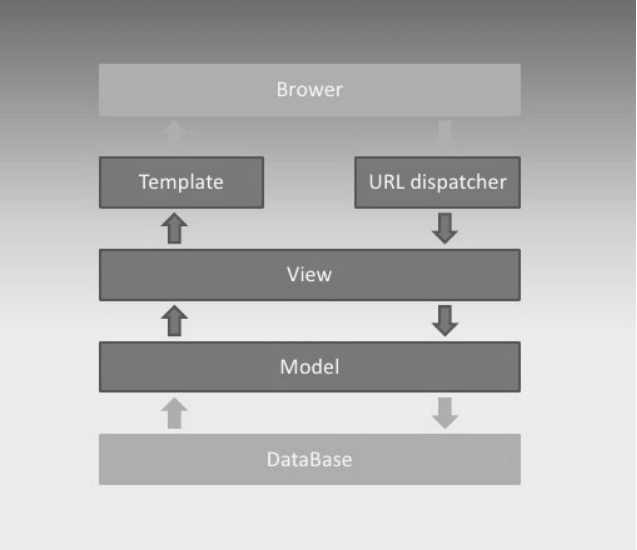
Python представляет популярный высокоуровневый язык программирования, который предназначен для создания приложений различных типов. Это и веб-приложения, и игры, и настольные программы, и работа с базами данных. Довольно большое распространение Python получил в области машинного обучения и исследований искусственного интеллекта. В своей работе я выбрал версию языка Python 2.7.9.

Изначально Django был создан в 2005 году для управления новостными сайтами LJWorld.com, lawrence.com и KUsports.com, принадлежащими компании The World Company. В 2008 году вышел публичный первый релиз фреймворка. На сегодняшний день он продолжает развиваться. Текущей версией фреймворка является версия 2.2 вышедшая 1 апреля 2019 года. Также постоянно выходят подверсии, который дополняют фреймворк и исправляют малочисленные ошибки [4].

Django довольно популярен. Он используется на многих сайтах, таких как Instagram, Disqus, Mozilla, The Washington Times, Pinterest, YouTube, Google и др.

Также Django используется в качестве веб-компонента в различных проектах, таких как Graphite - система построения графиков и наблюдения, FreeNAS - свободная реализация системы хранения и обмена файлами и др.

Фреймворк Django реализует архитектурный паттерн Model-View-Template или сокращенно MVT, который по факту является модификацией распростаненного в веб-программировании паттерна MVC (Model-View-Controller, Модель-Представление-Контроллер).

Схематично можно представить архитектуру MVT в Django следующим образом (рисунок 2): 

*Архитетура MVT*

Основные элементы паттерна:

· URL dispatcher: при получении запроса на основании запрошенного адреса URL определяет, какой ресурс должен обрабатывать данный запрос;

· View: получает запрос, обрабатывает его и отправляет в ответ пользователю некоторый ответ. Если для обработки запроса необходимо обращение к модели и базе данных, то View взаимодействует с ними. Для создания ответа может применять Template или шаблоны. В архитектуре MVC этому компоненту соответствуют контроллеры (но не представления);

· Model: описывает данные, используемые в приложении. Отдельные классы, как правило, соответствуют таблицам в базе данных;

· Template: представляет логику представления в виде сгенерированной разметки html. В MVC этому компоненту соответствует View, то есть представления.

Когда к приложению приходит запрос, то URL dispatcher определяет, с каким ресурсом сопоставляется данный запрос и передает этот запрос выбранному ресурсу. Ресурс фактически представляет функцию или View, который получает запрос и определенным образом обрабатывает его. В

процессе обработки View может обращаться к моделям и базе данных, получать из нее данные, или, наоборот, сохранять в нее данные. Результат обработки запроса отправляется обратно, и этот результат пользователь видит в своем браузере. Как правило, результат обработки запроса представляет сгенерированный html-код, для генерации которого применяются шаблоны (Template).

Из плюсов данного веб-фреймворка можно выделить:

· Быстрота: Django был разработан, чтобы помочь разработчикам создать приложение быстро настолько, на сколько это возможно. Это включает в себя формирование идеи, разработку и выпуск проекта, где Django экономит время и ресурсы на каждом из этих этапов. Таким образом, его можно назвать идеальным решением для разработчиков, для которых вопрос дедлайна стоит в приоритете;

· Полная комплектация: Django работает с десятками дополнительных функций, которые заметно помогают с аутентификацией пользователя, картами сайта, администрированием содержимого и многим другим. Данные аспекты помогают осуществить каждый этап веб разработки;

· Безопасность: Работая в Django, вы получаете защиту от ошибок, связанных с безопасностью и ставящих под угрозу проект. Я имею ввиду такие распространенные ошибки, как инъекции SQL, кросс-сайт подлоги, clickjacking и кросс-сайтовый скриптинг. Для эффективного использования логинов и паролей, система пользовательской аутентификации является ключом;

· Масштабируемость: фреймворк Django наилучшим образом подходит для работы с самыми высокими трафиками. Следовательно, логично, что великое множество загруженных сайтов используют Django для удовлетворения требований, связанных с трафиком;

· Разносторонность: менеджмент контента, научные вычислительные платформы, даже крупные организации – со всем этим можно эффективно справляться при помощи Django.

Также стоит отметить, что для работы с данным веб-фреймворком нет необходимости устанавливать IDE редакторы. Достаточно установить интерпретатор Python и установить в него Django.

Для того чтобы установить Django, потребуется пакетный менеджер pip. pip – система управления пакетами, которая используется для установки и

управления программными пакетами, написанными на Python.

Начиная с версии Python 2.7.9 и Python 3.4, они содержат пакет pip (или pip3 для Python 3) по умолчанию.

Для стилизации шаблонов в проекте мной был использован CSS-фреймворк Bootstrap версии 3.3.7.

Bootstrap – это CSS фреймворк, который изначально создавался для внутреннего использования компанией «Twitter» с рабочим названием «Twitter Blueprint», но в итоге был опубликован в открытый доступ и стал хорошим набором инструментов для front-end разработки под названием «Bootstrap» [8].

Преимущества Bootstrap:

· Высокая скорость разработки макетов страниц сайта. Bootstrap содержит огромный набор готовых решений и элементов;

· Кроссбраузерность и адаптивность сайта. Все элементы фреймворка адаптивны под все устройства и корректно отображаются во всех современных браузерах;

· Легкость в использовании. Даже человек, имеющий базовые знания о HTML и CSS, может свободно создавать web-страницы с использованием фреймворка;

· Простота в обучении. У Bootstrap очень хорошая документация с большим количеством примеров готового кода.

Основные инструменты Bootstrap:

· Сетки – заранее заданные размеры колонок, которые можно сразу же использовать, например, ширина колонки 140 px относится к классу .span2 (.col-md-2 в третьей версии фреймворка), который можно использовать в CSS-описании документа;

· Шаблоны – фиксированный или резиновый шаблон документа;

· Типографика – описания шрифтов, определение некоторых классов для шрифтов, таких как код, цитаты и т. п.;

· Медиа – предоставляет некоторое управление изображениями и видео.

· Таблицы – средства оформления таблиц, вплоть до добавления функциональности сортировки;

· Формы – классы для оформления форм и некоторых событий, происходящих с ними;

· Навигация – классы оформления для панелей, вкладок, перехода по страницам, меню и панели инструментов;

· Алерты – оформление диалоговых окон, подсказок и всплывающих окон.

Для динамического изменение количества добавляемых заказов, количества человек в заказе и цены заказа, а также динамического удаление заказа в проекте был использован jQuery версии 1.11.1.

jQuery – библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX. Разработка jQuery ведѐтся командой добровольцев на пожертвования [5].

Возможности jQuery:

· Движок кросс-браузерных CSS-селекторов Sizzle, выделившийся в отдельный проект;

· Переход по дереву DOM, включая поддержку XPath как плагина; · События;

· Визуальные эффекты; · AJAX-дополнения;

· JavaScript-плагины.

Для хранения данных как правило необходима база данных.

База данных – совокупность данных, организованных в соответствии с концептуальной структурой, описывающей характеристики этих данных и взаимоотношения между ними.

Существует огромное количество разновидностей баз данных, отличающихся по различным критериям.

Классификация по модели данных: · Иерархическая;

· Объектная и объектно-ориентированная; · Объектно-реляционная;

· Реляционная; · Сетевая;

· Функциональная.

Django проектировался для работы под управлением Apache с модулем mod python и с использованием PostgreSQL в качестве базы данных.

В настоящее время, помимо базы данных PostgreSQL, Django может работать с другими СУБД: MySQL, SQLite и Oracle.

В данной работе используется база данных SQLite. SQLite — это база данных, чем-то похожая на MySQL. Принципиальное отличие SQLite от других БД в том, что вся база представляет собой один файл. Если в MySQL база хранится где-то в дебрях сервера и недоступна для переноса, то в SQLite просто: один файл — одна база.

# 3.2. Реализация пользовательского интерфейса

Для оптимизации логистических процессов в аграрной сфере была разработана информационная система **e-agrologistic.uz**. Пользовательский интерфейс реализован в виде веб-приложения, чтобы аграрии, представители логистических компаний и экспортеры могли легко взаимодействовать с системой через браузер.

Архитектура проекта

Проект реализован как веб-платформа с использованием фреймворка Django. Система позволяет пользователям (например, сотрудникам фермерских хозяйств или логистических центров) авторизоваться на сайте и вводить информацию о перевозках, наличии сельхозпродукции, направлениях доставки и складах. Эти данные обрабатываются системой и сопоставляются с оптимальными логистическими маршрутами, рассчитанными на основе заданных критериев, таких как расстояние, стоимость и наличие транспорта. Форма, которую заполняет пользователь, напрямую связана с реляционной базой данных (в нашем случае используется SQLite или PostgreSQL). После того как пользователь укажет информацию о товаре и пункте назначения, система анализирует логистические возможности и выдает рекомендации по оптимальному маршруту или партнерской компании для транспортировки.

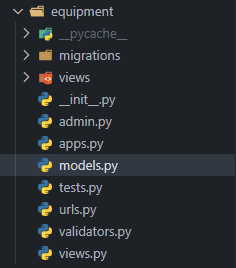
Таким образом, все действия пользователей и результаты логистического анализа сохраняются в базе данных, что позволяет в дальнейшем вести учет и проводить аналитику по логистическим операциям.

Структура кода проекта

В рамках реализации используется фреймворк Django, основанный на архитектурной модели MTV (Model-Template-View), аналогичной широко известной MVC-модели.

* **Model** — описывает структуру базы данных, в том числе модели для продукции, складов, транспорта и маршрутов;
* **Template** — отвечает за отображение информации в виде HTML-шаблонов, адаптированных для конечного пользователя;
* **View** — содержит бизнес-логику, обрабатывает запросы пользователей, взаимодействует с моделью и передает данные в шаблоны.

Интерфейс интуитивно понятен, адаптирован под мобильные устройства и позволяет легко вводить и получать необходимую информацию по логистическим операциям.



*Основные файлы в структуре проекта*

В файле models.py содержатся классы, описывающие структуру данных для системы **e-agrologistic.uz**. Здесь определены модели, отражающие основные сущности системы — такие как сельскохозяйственная продукция, склады, транспортные средства, маршруты и логистические заявки. Каждое поле модели описывает конкретные атрибуты объекта (например, вес продукции, точка отправки, статус заявки и т.д.), которые в дальнейшем используются при работе с базой данных.

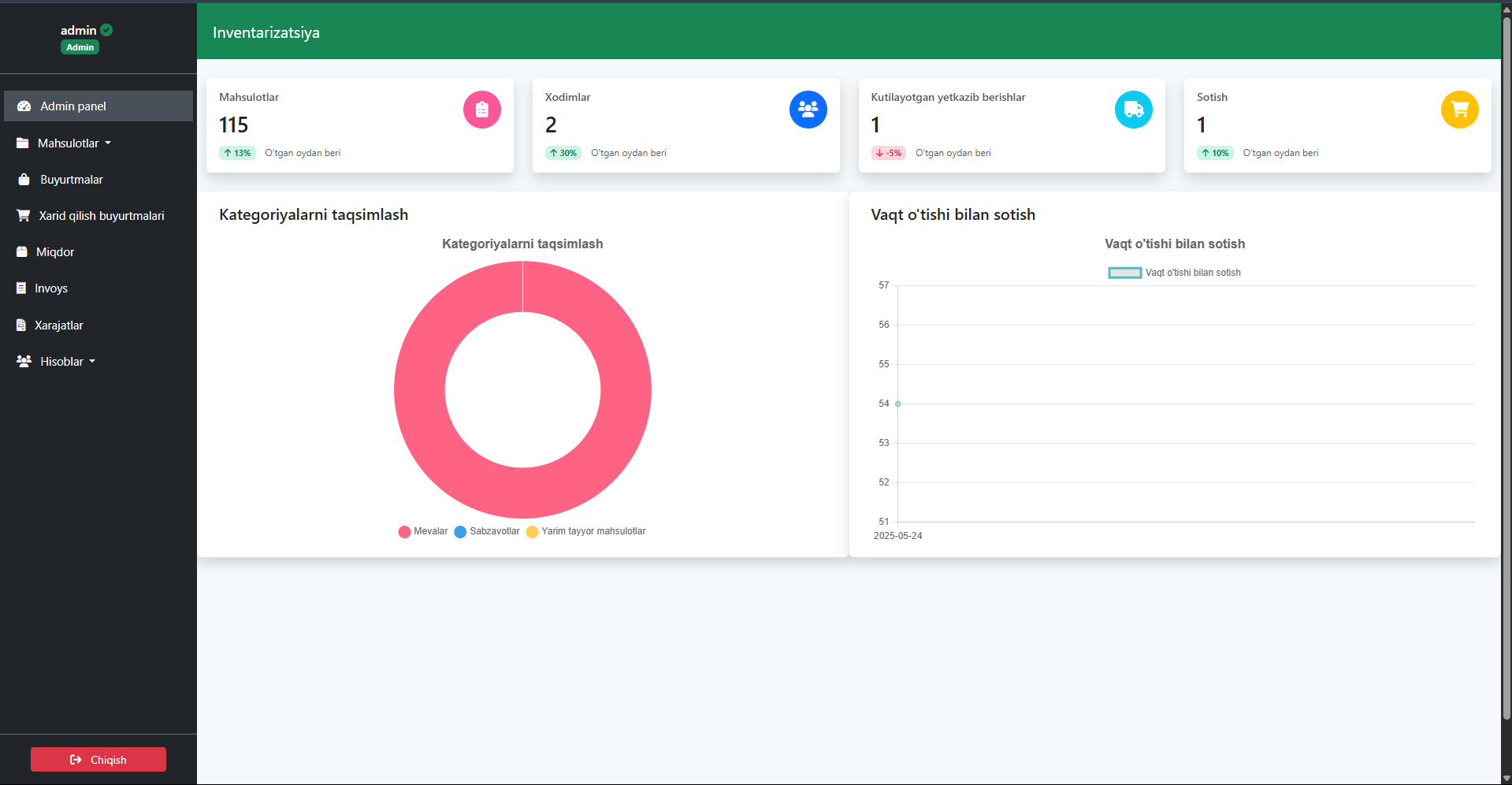
Файл urls.py отвечает за маршрутизацию запросов внутри веб-приложения. Здесь определены URL-адреса, по которым пользователи могут получить доступ к различным функциональным модулям системы. Это могут быть маршруты для добавления новой заявки, получения оптимального логистического маршрута, просмотра статистики по поставкам и т.д.

Также через маршруты осуществляется доступ к модулю анализа и оптимизации. Например, при обращении к определённому URL система инициирует обработку данных, включая анализ загруженности транспортных путей, расстояний и предложений от логистических партнеров. На основе этих данных система выдает рекомендации по наилучшему маршруту доставки или выбору склада.

Таким образом, urls.py связывает пользовательский интерфейс с бизнес-логикой и обеспечиваeт удобную и структурированную навигацию по системе.

*Сбор данных*

Для первоначального сбора данных от пользователей я создал форму в Html файле при помощи встроенных инструментов Django для работы с формами. Также, для создания внешнего вида сайта, я использовал файл CSS и прописал там соответствующие атрибуты дизайна для классов и отдельных типов элементов. Чтобы пользователям было удобнее заполнять форму, перед каждым элементом типа Input есть надпись, указывающая на данные, которые следует ввести. Данные поля используются на первой странице формы, для заполнения имени и фамилии пользователя.

**

*Главная страница панели инвентаризации в системе e-agrologistic.uz*

*На данном изображении представлена главная страница пользовательской панели администратора в системе* ***e-agrologistic.uz****, предназначенной для управления логистикой в аграрном секторе.*

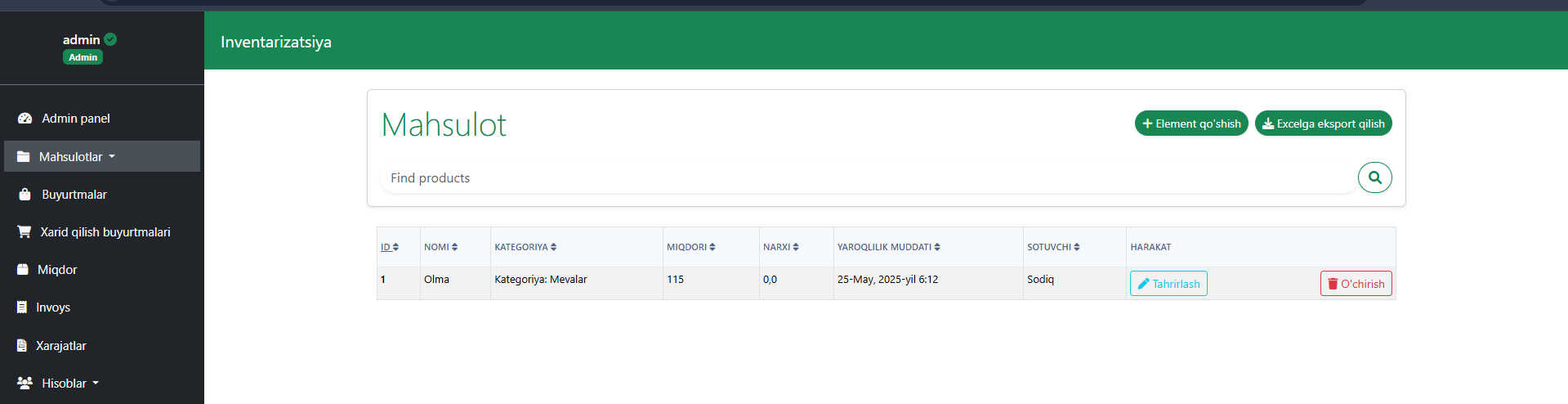
*В центральной части страницы расположены аналитические графики:*

* *Слева – диаграмма распределения товаров по категориям (фрукты, овощи, полуфабрикаты),*
* *Справа – график продаж по времени, позволяющий отслеживать изменение активности сбыта в разрезе дат.*

*Слева размещено навигационное меню, с помощью которого можно перейти к управлению следующими разделами:*

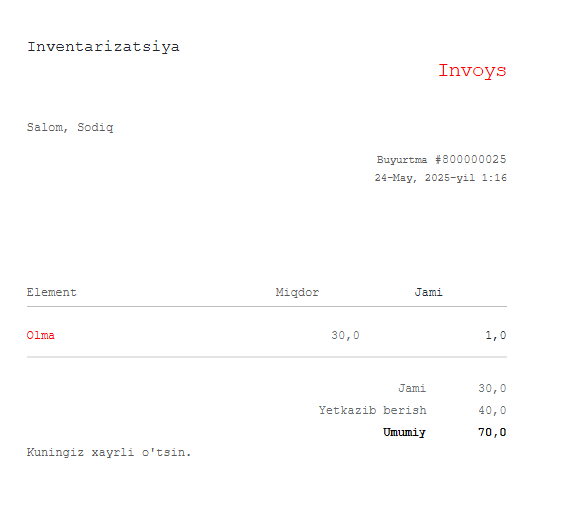
* ***Махсулотлар*** *(Продукты),*
* ***Буюртмалар*** *(Заказы),*
* ***Харид қилиш буюртмалари*** *(Закупки),*
* ***Миқдор*** *(Количество),*
* ***Инвойс*** *(Счета),*
* ***Харажатлар*** *(Расходы),*
* ***Ҳисоблар*** *(Учётные записи).*

*Кнопка* ***"Чиқиш"*** *в нижней части панели позволяет выйти из административного интерфейса. Интерфейс разработан таким образом, чтобы упростить отслеживание и управление логистическими потоками и товарными остатками на складе.*

**

*Управление товарами в системе*

*На странице отображается список товаров, добавленных в систему. Администратор может искать продукты, добавлять новые элементы и экспортировать данные в Excel. В таблице указаны: название продукта, категория, количество, цена, срок годности и продавец. Доступны действия — редактирование и удаление товара.Интерфейс упрощает управление складскими запасами и ускоряет работу с данными.*

**

*Страница формирования и печати инвойса в системе e-agrologistic.uz*

*На данной странице реализован функционал генерации инвойсов (счётов-фактур) для заказов, оформленных через платформу* ***e-agrologistic.uz****. Интерфейс предназначен для отображения полной информации по одному конкретному заказу и последующей печати документа. В верхней части экрана находится кнопка* ***"Chop etish"*** *(Распечатать), при нажатии на которую открывается диалог печати и формируется документ, готовый для передачи клиенту или внутреннего учёта. Этот модуль особенно полезен для логистов, бухгалтеров и менеджеров агропредприятий, так как позволяет быстро формировать официальные финансовые документы на основании данных из системы. Автоматизация этого процесса минимизирует количество ошибок и ускоряет документооборот.*