**2. Исследовательская часть**

# **2.1. Постановка задачи проектирования**

Целью написания данной курсовой работы является разработка интерфейса web-сервиса «HandyMady». Данный сервис предназначен для облегчения рабочего процесса самозанятых лиц, а также реализации товаров и услуг при помощи поиска и рекомендательной системы.

Задача проектирования такой системы может быть решена при помощи следующих средств разработки: средством реализации пользовательских интерфейсов должно быть реализовано на языках HTML и CSS, системы управления базами данных PostgreSQL, Исходные коды должны быть реализованы на языке Python посредством свободного фреймворка для веб-приложений Django.

Структурное проектирование системы осуществляется с помощью составления концептуальной и логической моделей хранимых данных. Описание этапов проектирования архитектуры приложения сопровождается описанием на естественном языке.

Необходимо разработать интернет-магазин, реализующий товары и услуги самозанятых. Система должна обеспечить удобный интерфейс продавцам для создания своего магазина с возможностью добавления товара в каталог и ведение акций, в то время как пользователь получит удобный интерфейс поиска товаров по категориям и рекомендациям.

# **2.2 Пользовательские сценарии**

В таблицах 1 – 11 представлены основные пользовательские сценарии поведения системы.

Таблица 1 – Сценарий регистрации пользователя

|  |  |
| --- | --- |
| **Название** | Регистрация |
| **Краткое описание** | Пользователь регистрируется в системе для дальнейшего взаимодействия с ней. |
| **Участники** | Пользователь, система |
| 1. Базовый сценарий 2. Пользователь открывает сайт. 3. Пользователь нажимает кнопку «Регистрация» 4. Система отправляет пользователю форму регистрации. 5. Пользователь вводит данные во все поля формы. 6. Система отправляет данные на сервер для занесения пользователя в базу данных. | |
| **Постусловие** | Пользователь зарегистрирован в системе. |
| **Расширения:** | |
| **1** | Пользователь заполнил не все обязательные поля.  Система сообщит об ошибке и попросит их заполнить. |
| **2** | Имя пользователя занято  Система сообщит об ошибке и попросит выбрать другое. |
| **3** | Пароль не соответствует требованиям.  Система сообщит об ошибке и попросит выбрать другой. |

Таблица 2 – Сценарий входа в систему

|  |  |
| --- | --- |
| **Название** | Вход в систему |
| **Краткое описание** | Пользователь входит в систему под своим логином и паролем для дальнейшего взаимодействия с ней |
| **Участники** | Пользователь, система |
| 1. Базовый сценарий 2. Пользователь открывает сайт и нажимает на кнопку «Войти в систему». 3. Система отправляет пользователю форму авторизации. 4. Пользователь вводит имя пользователя и пароль. 5. Система проверяет корректность ввода логина и пароля. 6. Пользователь авторизован в системе. | |
| **Постусловие** | Пользователь авторизовался в системе |
| **Расширения:** | |
| **1** | Неверный логин или пароль  Система уведомит об ошибке и попросит ввести корректные данные. |

Таблица 3 – Сценарий добавления товара в корзину

|  |  |
| --- | --- |
| **Название** | Добавление товара в корзину |
| **Краткое описание** | Пользователь добавляет товар, который он хочет купить в корзину. |
| **Участники** | Пользователь, система |
| 1. Пользователь пишет в строку поиска название товара и при желании изменяет настройки поиска. 2. Система выдает пользователю результат поиска с выбранными настройками 3. Пользователь выбирает в результатах поиска товар, который ему нужен. 4. Ознакомившись с описанием товара, информацией о доставке и о цене пользователь нажимает на кнопку «добавить в корзину». | |
| **Постусловие** | Пользователь добавил товар в корзину |
| **Расширения:** | |
| **1** | Отсутствие товара в наличии  Система уведомляет об этом пользователя об отсутствии товара. Также система уведомит о следующем поступлении если продавец указал информацию. |

Таблица 4 – Сценарий оформления пользовательского заказа

|  |  |
| --- | --- |
| **Название** | Оформление пользовательского заказа |
| **Краткое описание** | Пользователь оформляет заказ и оплачивает его. |
| **Участники** | Пользователь, продавец, система |
| 1. В меню корзины пользователь выбирает кнопку «оформить заказ» 2. Система списывает деньги со счета в профиле сервиса. 3. Система уведомляет продавца об отправке товара. 4. Продавец отправляет товар выбранным способом доставки | |
| **Постусловие** | Пользователь оформил заказ |

Таблица 5 – Сценарий закрытия заказа

|  |  |
| --- | --- |
| **Название** | Закрытие заказа |
| **Краткое описание** | После получения товара пользователь закрывает заказ и ставит оценку. |
| **Участники** | Пользователь, система |
| 1. Пользователь выбирает заказ из списка 2. Пользователь нажимает кнопку «Закрыть заказ» 3. Система помечает заказ в базе данных как завершенный. 4. Система предлагает пользователю оценить товар и услуги магазина. | |
| **Постусловие** | Пользователь закрыл заказ |

# **2.2.2 Сценарии использования.**

После регистрации на сайте пользователь может как делать заказы в магазинах других продавцов, так и создать свои магазины с различными товарами и продавать их. У пользователя есть возможность просмотра каталога товаров по различным категориям. Пользователь как в роли покупателя, так и в роли продавца может создать несколько магазинов с различными категориями товаров. У каждого товара есть свое описание, кратко рассказывающее о том, что представляет из себя продукт, и фотографии, при помощи которых можно построить себе визуальное представление о товаре. Чтобы сделать заказ покупатель должен добавить товары в корзину, указать номер телефона и телефон, если они не написаны в профиле и оплатить заказ. После того как товар был получен покупатель получает право оценить по пятибалльной шкале товар, продавца и магазин. Также пользователь может воспользоваться поиском товаров с различными фильтрами для точности. Для поиска товаров будет реализована поисковая строка. Поиск может проводиться как по названию, так и по различным категориям товаров. Покупатель может зайти на страницу магазина других пользователей, где может подписаться на рассылку новостей о появлении новый товаров и акций. UML диаграмма пользовательских сценариев (рисунок 2) показывает возможные варианты действий покупателя.



Рисунок 2 – UML диаграмма пользовательских сценариев для покупателя

Продавец имеет такие же пользовательские сценарии что и покупатель, но при этом он может заниматься менеджментом своих магазинов. Он может проводить различные акции индивидуально для каждого своего магазина с настройкой описания, даты начала и окончания. В профиле магазина владелец может видоизменить его вид, добавить товары на продажу и купить рекламу для продвижения товаров. UML диаграмма пользовательских сценариев (рисунок 3) показывает возможные варианты действий продавца.

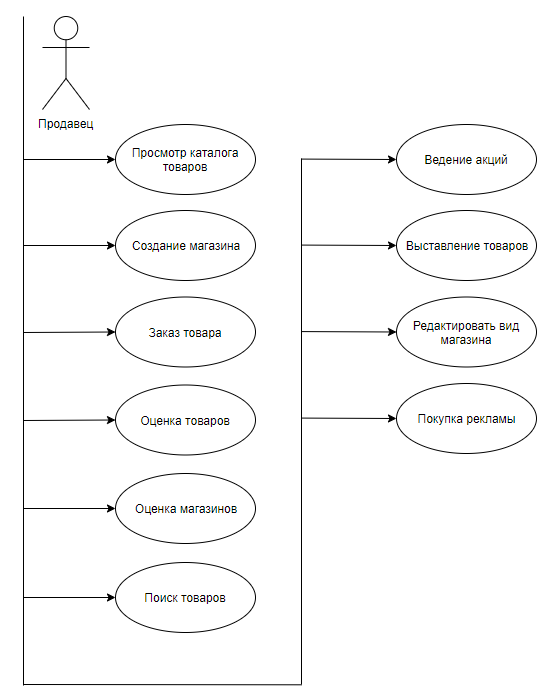


Рисунок 3 – UML диаграмма пользовательских сценариев для продавца

# **2.4. Обоснование выбора инструментов и платформы для разработки**

PostgreSQL - это свободно распространяемая объектно-реляционная система управления базами данных (ORDBMS), наиболее развитая из открытых СУБД в мире и являющаяся реальной альтернативой коммерческим базам данных.

* Надежность PostgreSQL.
* Производительность PostgreSQL основывается на использовании индексов, интеллектуальном планировщике запросов, тонкой системы блокировок, системе управления буферами памяти и кэширования, превосходной масштабируемости при конкурентной работе.
* Поддержка SQL
* PostgreSQL имеет очень богатый набор встроенных функций и операторов для работы с данными, полный список которых можно посмотреть в документации.
* Простота использования всегда являлась важным фактором для разработчиков.
* pgAdmin (GNU Artistic license) предоставляет удобный интерфейс для работы с базами данных PostgreSQL.

HTML (HyperText Markup Language — язык гипертекстовой разметки) — стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

Язык HTML до 5-й версии определялся как приложение SGML (стандартного обобщённого языка разметки по стандарту ISO 8879). Спецификации HTML5 формулируются в терминах DOM (объектной модели документа).

Язык XHTML является более строгим вариантом HTML, он следует синтаксису XML и является приложением языка XML в области разметки гипертекста.

Во всемирной паутине HTML-страницы, как правило, передаются браузерам от сервера по протоколам HTTP или HTTPS, в виде простого текста или с использованием шифрования.

CSS (Cascading Style Sheets) — язык таблиц стилей, который позволяет прикреплять стиль (например, шрифты и цвет) к структурированным документам (например, документам HTML и приложениям XML). Обычно CSS-стили используются для создания и изменения стиля элементов веб-страниц и пользовательских интерфейсов, написанных на языках HTML и XHTML, но также могут быть применены к любому виду XML-документа, в том числе XML, SVG и XUL. Отделяя стиль представления документов от содержимого документов, CSS упрощает создание веб-страниц и обслуживание сайтов.

CSS поддерживает таблицы стилей для конкретных носителей, поэтому авторы могут адаптировать представление своих документов к визуальным браузерам, слуховым устройствам, принтерам, брайлевским устройствам, карманным устройствам и т.д.

Каскадные таблицы стилей описывают правила форматирования элементов с помощью свойств и допустимых значений этих свойств. Для каждого элемента можно использовать ограниченный набор свойств, остальные свойства не будут оказывать на него никакого влияния.

Django — это программный каркас с богатыми возможностями, подходящий для разработки сложных сайтов и веб-приложений, написанный на языке программирования Python.

Django — фреймворк для веб-приложений на языке Python. Один из основных принципов фреймворка — DRY (don't repeat yourself). Веб-системы на Django строятся из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из заметных архитектурных отличий этого фреймворка от некоторых других (например, Ruby on Rails). Также, в отличие от многих других фреймворков, обработчики URL в Django конфигурируются явно (при помощи регулярных выражений), а не автоматически задаются из структуры контроллеров.

Django проектировался для работы под управлением Apache (с модулем mod\_python) и с использованием PostgreSQL в качестве базы данных. В настоящее время, помимо PostgreSQL, Django может работать с другими СУБД: MySQL (MariaDB), SQLite, Microsoft SQL Server, DB2, Firebird, SQL Anywhere и Oracle. Для работы с базой данных Django использует собственный ORM, в котором модель данных описывается классами Python, и по ней генерируется схема базы данных.

Первоначально разработка Django велась для обеспечения более удобной работы с новостными ресурсами, что достаточно сильно отразилось на архитектуре: фреймворк предоставляет ряд средств, которые помогают в быстрой разработке веб-сайтов информационного характера. Например, разработчику не требуется создавать контроллеры и страницы для административной части сайта, в Django есть встроенное приложение для управления содержимым, которое можно включить в любой сайт, сделанный на Django, и которое может управлять сразу несколькими сайтами на одном сервере. Административное приложение позволяет создавать, изменять и удалять любые объекты наполнения сайта, протоколируя все совершённые действия, и предоставляет интерфейс для управления пользователями и группами (с пообъектным назначением прав).