1. **Научно-исследовательская часть**

# Техническое задание

# Введение

# Наименование программы

Web-сервис для самозанятых “HandyMady”.

# Краткая характеристика области применения

Данное программное обеспечение может быть использовано самозанятыми лицами с целью создавать свои магазины, где он может выставлять свои работы на продажу для заработка и физическими лицами для покупки различных товаров.

# Основания для разработки

Разработка ведется на основании данного технического задания. Этим документом подтверждается процесс разработки программного обеспечения на тему “Разработка web-сервиса для самозанятых “HandyMady”.

# Назначение разработки

# Наименование темы разработки

“Разработка web-сервиса для самозанятых “HandyMady”.

# Функциональное назначение программы

Функциональным назначением разработки указанного программного обеспечения является повышение эффективности работы, облегчения рабочего процесса, помощь в реализации товаров и услуг для самозанятых лиц, а также удобная торговая площадка для покупателей с возможностью поиска.

Критериями повышения эффективности будут служить:

* уменьшение затрат времени на поиск продавцов и товаров;
* удобный интерфейс для выставления товаров и для аукциона;
* возможность общения покупателей с самозанятыми;
* отслеживание доставки;

# Эксплуатационное назначение

Подразумевается эксплуатация системы пользователями и самозанятыми, получающие доход от своей личной трудовой деятельности.

# Требования к программному изделию

# Требования к функциональным характеристикам

Программная система должна представлять собой работоспособный сайт, реализующее следующие функции:

* функция регистрации для пользователей и продавцов;
* функция создания магазина;
* функция выставления товаров;
* функция ведения акций в магазине;
* функция рейтинга для продавцов, основывающаяся на оценках пользователей;
* поиск магазинов и товаров по категориям;
* функция связи между продавцом и покупателем;
* функция отслеживания доставки;

# Требования к надежности

Система должна работать стабильно, не допускается прерываний работы системы, вызванных ее внутренними ошибками. Интерфейс системы не должен содержать элементов, назначение которых неочевидно, которые вводят пользователя в заблуждение или не несут функционального значения.

# Условия эксплуатации

Система предназначена для использования в нормальных условиях. Специальных требований по эксплуатации системе не предъявляется.

# Требования к составу и параметрам технических средств

Специальных требований к составу и параметрам технических средств не предъявляется.

# Требования к информационной и программной совместимости

Описанная и разрабатываемая система должна включать в себя серверный компонент и клиентский интерфейс для взаимодействия с системой. Система должна быть совместима со всеми актуальными браузерами.

В качестве системы управления базой данных будет использоваться PostgreSQL.

Исходные коды (back-end) должны быть реализованы на языке C# посредством свободно-распространяемого кросс-платформенного фреймворка для создания веб-приложений с открытым исходным кодом ASP.NET Core. Визуальное представление (front-end) должно быть реализовано на языках HTML и CSS. В качестве среды разработки программы должна быть использована среда Visual Studio.

# Требования к программной документации

Разрабатываемое программное обеспечение не требует разработки руководства пользователя. Техническое задание должно быть оформлено в соответствии с требованиями ГОСТ.

# Стадии и этапы разработки

Содержимое разделов настоящего технического задания может быть изменено и дополнено по согласованию с руководителем.

# Стадии разработки

Разработка должна быть проведена в пять стадий стадии:

* Утверждение технического задания;
* Эскизный проект;
* Технический проект;
* Рабочий проект;
* Ввод в эксплуатацию

# Этапы разработки

Порядок разработки программной системы разбит на следующие стадии:

**Этап утверждения технического задания**

* Обоснование необходимости разработки ПО;
* Исследование предметной области
* Разработка и утверждение технического задания

Результатом работы на данном этапе является согласованное и утвержденное техническое задание на разработку системы и описание предметной области.

**Этап проектирования:**

* Разработка концептуальной схемы данных, прототипа или эскиза интерфейса;
* Разработка утверждение с заказчиком эскизного проекта;

Результатом работы на данном этапе является оформленная исследовательская часть, часть разработанных и реализованных алгоритмов функционирования приложения, прототип интерфейса, концептуальная и логическая схемы базы данных.

**Этап технического проекта:**

* Оформление проектно-конструкторской части
* Разработка технической версии проекта
* Демонстрация разработанной версии приложения

Результатом работы на данном этапе будет являться оформленная проектно-конструкторская часть и рабочая техническая версия ПО

**Этап рабочего проекта:**

* Разработка оставшегося функционала приложения
* Отладка и тестирование рабочей версии проекта
* Разработка программной документации

Результатом работы на данном этапе является оформленная проекто-технологическая часть, рабочая версия ПО и завершенная программная документация

**Этап ввода в эксплуатацию:**

* Подготовка системы к развертыванию
* Первоначальная настройка системы

Результатом работы на данном этапе является спроектированный базовый функционал разрабатываемой системы.

# Постановка задачи проектирования

Целью написания данной курсовой работы является разработка web-сервиса «HandyMady», 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 C# посредством свободно-распространяемого кросс-платформенного фреймворка для создания веб-приложений с открытым исходным кодом ASP.NET Core.

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

# Описание предметной области

Самозанятость — форма получения вознаграждения за свой труд непосредственно от заказчиков, в отличие от наёмной работы.

Самозанятый — это лицо, у которого нет работодателя и у которого нет наемных работников, то есть это граждане, получающие доход от своей личной трудовой деятельности.

Стать самозанятым могут обычные граждане и индивидуальные предприниматели, которые решили поменять статус. Физические лица и индивидуальные предприниматели, которые перейдут на новый специальный налоговый режим (самозанятые), могут платить с доходов от самостоятельной деятельности только налог по льготной ставке — 4 (при получении денег от физических лиц) или 6% (при получении денег от юридических лиц).

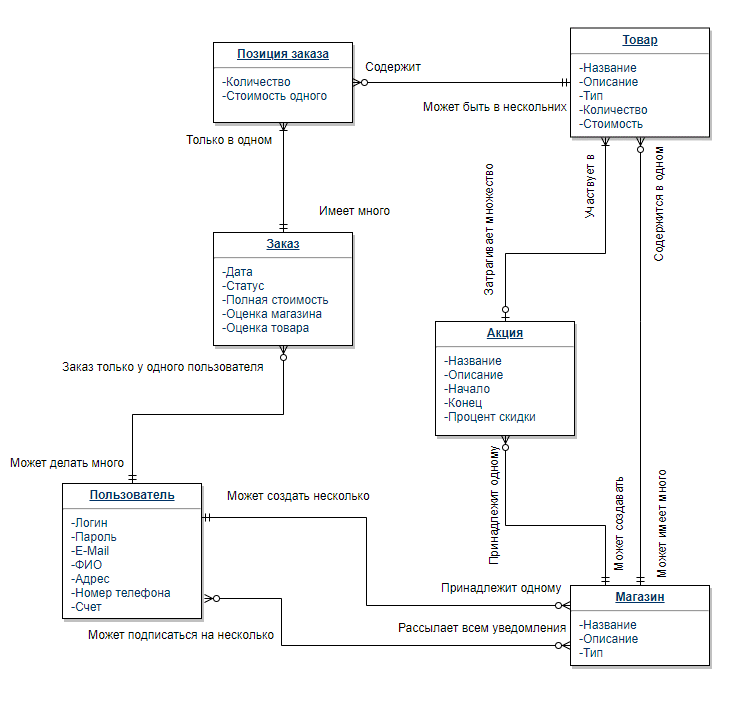
Осуществлять свою рабочую деятельность самозанятые лица могут с помощью следующих интернет площадок:

* Социальные сети — онлайн-платформа, которую люди используют для общения, создания социальных отношений с другими людьми, которые имеют схожие интересы или офлайн-связи. На базе социальных сетей самозанятое лицо создает сообщество с целью продвижения товара и его дальнейшего сбыта. Самозанятое лицо, продавец, ведет каталоги, добавляя новый товар в наличие, ведет переписки с потенциальными покупателями, работает над продвижением своего товара. Оплата в социальных сетях чаще всего производится посредством онлайн переводов при помощи сторонних приложений банков. Это существенно затрудняет ведение финансовой отчетности и порождает проблему мошенничества.
* Интернет-магазин — сайт, торгующий товарами посредством сети Интернет. Позволяет пользователям онлайн, в своём браузере или через мобильное приложение, сформировать заказ на покупку, выбрать способ оплаты и доставки заказа, оплатить заказ. При этом продажа товаров осуществляется дистанционным способом, и она накладывает ограничения на продаваемые товары. В интернет-магазинах оплата осуществляется при помощи посредников.

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.

# Концептуальная схема данных

На основании представленной информации в предметной области можно выделить несколько сущностей (рисунок 1): Пользователь, магазин, товар, акция, заказ и позиция заказа.

Рисунок 1 – Концептуальная модель данных

На рисунке 1 представлена концептуальная схема данных предметной области. Из данной схемы можно выделить следующие связи: существуют пользователи, которые могут делать много заказов, подписываться на чужие магазины и создавать множество своих. Владельцы магазинов могут создавать акции и добавлять в ассортимент новые товары. В одном заказе может быть множество позиций заказов, содержащие в себе товар и его количество.

# 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, изображенная на рисунке 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.

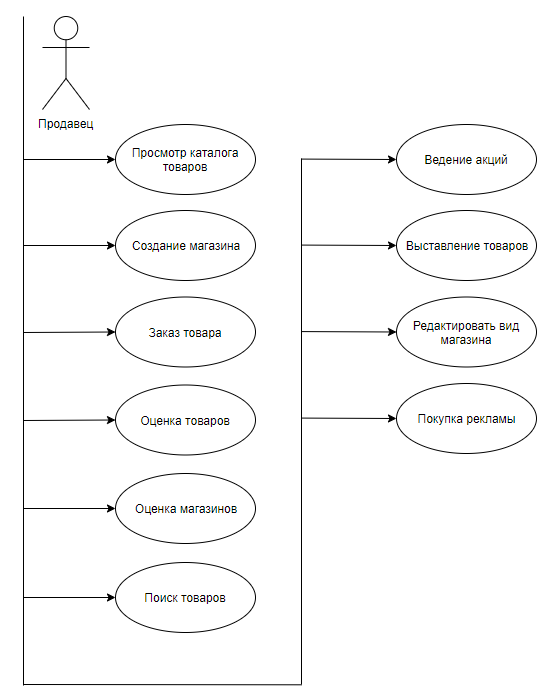


Рисунок 3 – UML диаграмма пользовательских сценариев для продавца

# Перечень задач, подлежащих решению в процессе разработки

Для создания системы необходимо:

* Составить техническое задание.
* Провести исследование предметной области;
* Сформировать концептуальную и логическую модели хранимых данных;
* Подготовить перечень необходимого функционала;
* Оформить описание процессов, протекающих в системе;
* Спроектировать макет сайта;
* Организовать структуру хранения данных и реализовать тестовое наполнение базы данных,
* Реализовать систему:
* Произвести тестирование и отладку;
* Оформить конструкторскую документацию;

Перечень реализуемых функций:

* Реализация поиска товаров по названию, категориям, цене, оценкам и авторам.
* Реализация ведения списков желаемого.
* Система рейтинга, базирующаяся на отзывах пользователей.
* Предоставлять пользователю информацию об выбранном товаре.
* Функция подписок на магазины с возможностью уведомления о поступлении новых товарах и акциях.
* Возможность уведомления о скидках на товары из списка желаемого.

# 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.

Платформа ASP.NET Core представляет технологию от компании Microsoft, предназначенную для создания различного рода веб-приложений: от небольших веб-сайтов до крупных веб-порталов и веб-сервисов.

ASP.NET Core является opensource-фреймворком. Все исходные файлы фреймворка доступны на GitHub.

ASP.NET Core может работать поверх кросс-платформенной среды .NET Core, которая может быть развернута на основных популярных операционных системах: Windows, Mac OS, Linux. И таким образом, с помощью ASP.NET Core мы можем создавать кросс-платформенные приложения. И хотя Windows в качестве среды для разработки и развертывания приложения до сих пор превалирует, но теперь уже мы не ограничены только этой операционной системой. То есть мы можем запускать веб-приложения не только на ОС Windows, но и на Linux и Mac OS. А для развертывания веб-приложения можно использовать традиционный IIS, либо кросс-платформенный веб-сервер Kestrel.

Благодаря модульности фреймворка все необходимые компоненты веб-приложения могут загружаться как отдельные модули через пакетный менеджер Nuget. Кроме того, в отличие от предыдущих версий платформы нет необходимости использовать библиотеку System.Web.dll.

ASP.NET Core включает в себя фреймворк MVC, который объединяет функциональность MVC, Web API и Web Pages. В предыдущих версии платформы данные технологии реализовались отдельно и поэтому содержали много дублирующей функциональности. Сейчас же они объединены в одну программную модель ASP.NET Core MVC. А Web Forms полностью ушли в прошлое.

Кроме объединения вышеупомянутых технологий в одну модель в MVC был добавлен ряд дополнительных функций.

Одной из таких функций являются тэг-хелперы (tag helper), которые позволяют более органично соединять синтаксис html с кодом С#.

ASP.NET Core характеризуется расширяемостью. Фреймворк построен из набора относительно независимых компонентов. И мы можем либо использовать встроенную реализацию этих компонентов, либо расширить их с помощью механизма наследования, либо вовсе создать и применять свои компоненты со своим функционалом.

Также было упрощено управление зависимостями и конфигурирование проекта. Фреймворк теперь имеет свой легковесный контейнер для внедрения зависимостей, и больше нет необходимости применять сторонние контейнеры, такие как Autofac, Ninject. Хотя при желании их также можно продолжать использовать.

В качестве инструментария разработки мы можем использовать последние выпуски Visual Studio, начиная с версии Visual Studio 2015. Кроме того, мы можем создавать приложения в среде Visual Studio Code, которая является кросс-платформенной и может работать как на Windows, так и на Mac OS X и Linux.

Для обработки запросов теперь используется новый конвейер HTTP, который основан на компонентах Katana и спецификации OWIN. А его модульность позволяет легко добавить свои собственные компоненты.

Если суммировать, то можно выделить следующие ключевые отличия ASP.NET Core от предыдущих версий ASP.NET:

* Новый легковесный и модульный конвейер HTTP-запросов
* Возможность развертывать приложение как на IIS, так и в рамках своего собственного процесса
* Использование платформы .NET Core и ее функциональности
* Распространение пакетов платформы через NuGet
* Интегрированная поддержка для создания и использования пакетов NuGet
* Единый стек веб-разработки, сочетающий Web UI и Web API
* Конфигурация для упрощенного использования в облаке
* Встроенная поддержка для внедрения зависимостей
* Расширяемость
* Кроссплатформенность: возможность разработки и развертывания приложений ASP.NET на Windows, Mac и Linux
* Развитие как open source, открытость к изменениям

# 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 |