**ТРЕБОВАНИЯ К ПРОЕКТУ**

**1 ВВЕДЕНИЕ**

* 1. **Название продукта**

Online Store – веб-приложение для онлайн-магазина, состоящее из серверной части (REST API на C# / ASP.NET) и клиентской части (Blazor).

* 1. **Контекст проекта**

Проект предназначен для реализации полнофункционального интернет-магазина. Серверная часть будет предоставлять REST API для работы с товарами, заказами и пользователями, а клиентская часть на Blazor обеспечит удобный и современный пользовательский интерфейс, взаимодействующий с API. Такой подход позволит разделить логику приложения и интерфейс, обеспечив гибкость и масштабируемость.

**1.3 Границы проекта**

Что продукт будет в себя включать:

– Предоставлять REST API для работы с каталогом товаров, корзиной и заказами.

– Обеспечивать регистрацию и авторизацию пользователей (JWT).

– Реализовывать клиентский интерфейс на Blazor для покупателей и администраторов.

– Поддерживать поиск, фильтрацию и сортировку товаров.

– Обеспечивать административный функционал для управления товарами и заказами.

Что не будет включать:

– Не будет реализовывать интеграцию с платёжными системами (только имитация оплаты).

– Не будет включать складской учёт или интеграцию с ERP.

– Не будет поддерживать офлайн-режим работы.

– Не будет включать мобильное приложение (только веб-версия).

**2 ТРЕБОВАНИЯ ПОЛЬЗОВАТЕЛЯ**

**2.1 Программные интерфейсы**

Продукт Online Store будет взаимодействовать с рядом внешних систем, библиотек и сервисов, обеспечивающих его корректную работу, расширение функционала и интеграцию с другими приложениями. Данный список можно увидеть далее:

1. Веб-фреймворк и серверная платформа

* ASP.NET Core – основной фреймворк для разработки REST API, обеспечивающий маршрутизацию запросов, обработку HTTP-запросов и ответы в формате JSON.
* Kestrel – встроенный веб-сервер для обработки входящих HTTP-запросов.
* IIS / Nginx (опционально) – для деплоя и балансировки нагрузки в продакшн-среде.

2. Библиотеки для работы с данными

* Entity Framework Core – ORM для взаимодействия с базой данных (SQL Server или PostgreSQL), включая миграции и LINQ-запросы.
* System.Text.Json или Newtonsoft.Json – для сериализации и десериализации данных в формате JSON.

3. Аутентификация и безопасность

* ASP.NET Core Identity – управление пользователями, ролями и правами доступа.
* JWT (JSON Web Token) – механизм безопасной аутентификации и авторизации между клиентом и сервером.
* HTTPS / TLS – шифрование передаваемых данных.

4. Клиентская часть (Blazor)

* Blazor WebAssembly – реализация пользовательского интерфейса, взаимодействующего с REST API.
* Blazor Components – готовые и кастомные UI-компоненты для отображения данных и взаимодействия с пользователем.

5. Внешние API и сервисы

* Сервис отправки электронной почты (например, SendGrid или SMTP-сервер) – для отправки писем с подтверждением регистрации, уведомлений о заказах и т.д.
* Сервис загрузки и хранения изображений (например, Azure Blob Storage или локальное файловое хранилище) – для хранения фотографий товаров.
* Сервис геолокации / доставки (опционально) – для расчёта стоимости и сроков доставки (например, интеграция с API почтовых служб).

6. Инструменты логирования и мониторинга

* Serilog или NLog – для централизованного логирования событий и ошибок.
* Application Insights (Azure) или аналог – для мониторинга производительности и анализа работы приложения.

7. Средства тестирования

* xUnit / NUnit – для модульного тестирования серверной логики.
* Postman или Swagger UI – для тестирования и документирования REST API.

**2.2 Интерфейс пользователя**

Система Online Store будет предоставлять пользователю веб-интерфейс, реализованный с помощью Blazor WebAssembly, и взаимодействовать с ним через браузер. Пользовательский интерфейс будет адаптивным, обеспечивая корректное отображение на настольных компьютерах, ноутбуках, планшетах и смартфонах.

Основные принципы взаимодействия:

* Простота и интуитивность – минимальное количество кликов для выполнения ключевых действий.
* Единый стиль – использование согласованной цветовой схемы, шрифтов и элементов управления.
* Обратная связь – система будет отображать уведомления об успешных действиях, ошибках или необходимости подтверждения.
* Доступность – поддержка базовых стандартов доступности (контрастность, масштабирование текста).

Основные пользовательские роли и интерфейсы:

1. Гость

– Просмотр каталога товаров.

– Поиск и фильтрация товаров.

– Просмотр карточки товара.

– Регистрация или вход в систему.

2. Покупатель

– Все возможности гостя.

– Добавление товаров в корзину.

– Оформление заказа.

– Просмотр истории заказов.

– Редактирование профиля.

3. Администратор

– Управление товарами (добавление, редактирование, удаление).

– Управление заказами (изменение статуса, просмотр деталей).

– Управление пользователями (блокировка, изменение ролей).

Таблица «Действие пользователя – реакция системы» представлена далее:

Таблица 2.2.1 – «Действие пользователя – реакция системы»

|  |  |
| --- | --- |
| **Действие пользователя** | **Реакция системы** |
| Пользователь открывает главную страницу | Отображается каталог товаров с возможностью поиска и фильтрации |
| Пользователь вводит запрос в поисковую строку и нажимает «Поиск» | Система отправляет запрос к API и отображает список товаров, соответствующих запросу |

*Продолжение таблицы 2.2.1*

|  |  |
| --- | --- |
| **Действие пользователя** | **Реакция системы** |
| Пользователь кликает на товар | Открывается страница карточки товара с описанием, ценой и фото |
| Пользователь добавляет товар в корзину | Система обновляет корзину и отображает уведомление об успешном добавлении |
| Пользователь оформляет заказ | Система проверяет данные, создаёт заказ и отображает подтверждение |
| Администратор заходит в панель управления | Отображается интерфейс управления товарами, заказами и пользователями |
| Администратор изменяет статус заказа | Система обновляет данные в базе и отображает уведомление об успешном изменении |

**2.3 Характеристики пользователей**

Система Online Store рассчитана на несколько категорий пользователей, каждая из которых имеет свои особенности и уровень подготовки. Наиболее многочисленную группу составляют гости – люди, которые заходят на сайт без регистрации. Это могут быть пользователи разного возраста, от подростков до пожилых людей, с базовыми навыками работы в интернете. Они уверенно пользуются браузером, умеют искать информацию и ориентироваться в простых веб-интерфейсах. Для них важны понятная навигация, наглядное представление товаров и возможность быстро найти нужный продукт.

Зарегистрированные покупатели, как правило, обладают чуть более высоким уровнем технической грамотности. Это взрослые пользователи, привыкшие к онлайн-покупкам и работе с формами ввода данных. Они ожидают от системы стабильной работы, удобного оформления заказа, возможности отслеживать его статус и управлять своим профилем. Для них важны скорость отклика, безопасность передачи данных и предсказуемость интерфейса.

Отдельную группу составляют администраторы – сотрудники компании, которые работают с системой ежедневно. Они имеют высокий уровень компьютерной грамотности, опыт работы с административными панелями и понимание базовых принципов функционирования веб-приложений. Их задачи включают управление каталогом товаров, обработку заказов и взаимодействие с клиентами через встроенные инструменты. Для них критична функциональность, позволяющая быстро вносить изменения и контролировать процессы.

Наконец, в проекте будут задействованы разработчики и технические специалисты, отвечающие за поддержку и развитие системы. Это пользователи с профессиональными навыками в области программирования, администрирования баз данных и веб-технологий.

**2.4 Предположения и зависимости**

При разработке и эксплуатации системы OnlineStore предполагается, что ряд внешних и внутренних факторов может повлиять на требования, изложенные в данном документе. Эти факторы необходимо учитывать на всех этапах жизненного цикла проекта.

Предположения:

– Пользователи будут иметь доступ к современным веб-браузерам, поддерживающим Blazor WebAssembly и работу с JavaScript (Google Chrome, Microsoft Edge, Mozilla Firefox, Safari последних версий).

– Серверная инфраструктура будет обеспечивать стабильное интернет-соединение и достаточные вычислительные ресурсы для обработки запросов в пиковые периоды нагрузки.

– База данных (SQL Server или PostgreSQL) будет доступна и корректно настроена на момент начала интеграции.

– Пользователи (как покупатели, так и администраторы) обладают базовыми навыками работы с веб-приложениями.

– Все интеграции с внешними сервисами (отправка почты, хранение изображений, авторизация) будут предоставлены в виде доступных и документированных API.

Зависимости:

– Доступность и стабильность работы внешних сервисов: Сервис отправки электронной почты (SMTP или SendGrid). Хранилище изображений (локальное или облачное, например Azure Blob Storage).

– Возможные API служб доставки или геолокации.

– Версии и совместимость используемых библиотек и фреймворков (ASP.NET Core, Entity Framework Core, Blazor).

– Политика безопасности и требования к защите данных (в том числе соответствие локальному законодательству о персональных данных).

– Решения по хостингу и инфраструктуре (облачный провайдер, локальный сервер, контейнеризация).

– График и приоритеты разработки, зависящие от доступности команды и ресурсов.

**3 СИСТЕМНЫЕ ТРЕБОВАНИЯ**

Этот раздел должен содержать требования к продукту, описанные на уровне достаточном для однозначного понимания того, как система может быть спроектирована и протестирована на предмет удовлетворения данных требований.

**3.1 Функциональные требования**

FR-1. Система должна предоставлять REST API для получения списка товаров с возможностью фильтрации и сортировки по цене, категории и другим параметрам.

FR-2. Система должна предоставлять REST API для получения детальной информации о товаре по его идентификатору.

FR-3. Система должна обеспечивать регистрацию новых пользователей с сохранением данных в базе и проверкой уникальности e-mail.

FR-4. Система должна обеспечивать авторизацию пользователей с использованием JWT-токенов.

FR-5. Система должна позволять авторизованным пользователям добавлять товары в корзину, изменять количество и удалять позиции.

FR-6. Система должна позволять оформлять заказ, включая выбор адреса доставки и подтверждение состава заказа.

FR-7. Система должна предоставлять пользователю возможность просматривать историю своих заказов.

FR-8. Система должна предоставлять административный интерфейс (через Blazor UI) для добавления, редактирования и удаления товаров.

FR-9. Система должна предоставлять административный интерфейс для управления заказами (просмотр, изменение статуса, удаление).

FR-10. Система должна предоставлять административный интерфейс для управления пользователями (просмотр списка, изменение ролей, блокировка)

FR-11. Система должна отправлять уведомления по электронной почте при регистрации, оформлении заказа и изменении его статуса.

FR-12. Система должна обеспечивать валидацию всех входных данных на стороне сервера.

FR-13. Система должна предоставлять Swagger UI или аналогичный инструмент для документирования и тестирования API.

FR-14. Система должна обеспечивать логирование всех ошибок и ключевых событий.

FR-15. Система должна обеспечивать защиту API от несанкционированного доступа (HTTPS, проверка токенов, ограничения по ролям).

**3.2 Нефункциональные требования**

**3.2.1** Для системы Online Store определены следующие ключевые атрибуты качества, которые напрямую влияют на её успешную эксплуатацию и удовлетворённость пользователей.

1. Надёжность. Система должна обеспечивать стабильную работу без сбоев в течение длительного времени. Это важно, так как сбои могут привести к потере заказов, недовольству клиентов и финансовым потерям.

Как измеряется: коэффициент доступности (uptime) не ниже 99,5% в месяц; количество критических сбоев – не более 1 в месяц. Методы проверки: нагрузочное тестирование, мониторинг доступности, анализ логов ошибок.

2. Безопасность. Система должна защищать персональные данные пользователей и предотвращать несанкционированный доступ. Это критично для соблюдения законодательства о защите данных и поддержания доверия клиентов.

Как измеряется: отсутствие успешных атак при проведении тестов на проникновение; соответствие требованиям HTTPS и шифрования данных; прохождение аудита безопасности. Методы проверки: тестирование на уязвимости (OWASP Top 10), аудит кода, проверка конфигураций сервера.

3. Производительность. Система должна быстро обрабатывать запросы, чтобы пользователи не испытывали задержек при работе.

Как измеряется: время отклика API – не более 500 мс при средней нагрузке; время загрузки основных страниц UI – не более 2 секунд. Методы проверки: нагрузочное тестирование, профилирование кода, мониторинг времени отклика.

4. Масштабируемость. Система должна поддерживать увеличение количества пользователей и объёма данных без значительного снижения производительности.

Как измеряется: способность обслуживать рост нагрузки в 2–3 раза без критического падения скорости работы. Методы проверки: стресс-тестирование, тестирование в условиях увеличенной нагрузки.

5. Удобство использования (Usability). Интерфейс должен быть интуитивно понятным и доступным для пользователей с разным уровнем технической подготовки.

Как измеряется: результаты юзабилити-тестов (не менее 80% пользователей успешно выполняют ключевые сценарии без помощи); количество кликов до выполнения целевого действия – не более 3–4. Методы проверки: тестирование с участием реальных пользователей, опросы, анализ поведения в аналитических системах.

6. Поддерживаемость. Система должна быть легко модифицируема и сопровождаться без значительных затрат времени и ресурсов.

Как измеряется: время, необходимое для внесения изменений в функционал (не более 8 часов для мелких доработок); наличие актуальной документации. Методы проверки: ревью кода, анализ архитектуры, проверка полноты документации.