Министерство образования Республики Беларусь

Учреждение образования  
 БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
 ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

Анализ и оптимизация бизнес-процессов

Отчет

по лабораторной работе №1

на тему:

**РЕФАКТОРИНГ ПРОГРАММНОГО КОДА**

|  |  |  |
| --- | --- | --- |
| Проверил |  | А.В. Шелест |
|  | (подпись) |  |
| зачтено |  |  |
|  | (дата защиты) |  |
|  |  |  |
| Выполнил |  | Ю-Ф.А. Тюленев  гр. 114301 |
|  | (подпись) |  |

Минск, 2025

Ссылка на репозиторий GitHub: <https://github.com/FranGranDev/RSAS>

Программное средство RSAS (*Retail sales analytics system*, *Программное средство анализа продаж и заказов в розничном магазине*) предназначена для упрощения процесса ведения розничной торговли. Оно будет строится на платформе ASP.Net Core и использовать архитектуру MVC Razor Pages. В качестве базы данных была выбрана MS SQL Server в виду хорошей совместимости с ASP.Net, для связи с базой данных будет использоваться фрейворк Entity Framework Core.

Для наглядности проектируемой архитектуры программного средства была построена диаграмма C4.

**Диаграмма C4 в GitHub:** <https://github.com/FranGranDev/RSAS/tree/main/Documentation/Code%20Design/C4>

На рисунке 1 представлена диаграмма C4 на контекстном уровне.

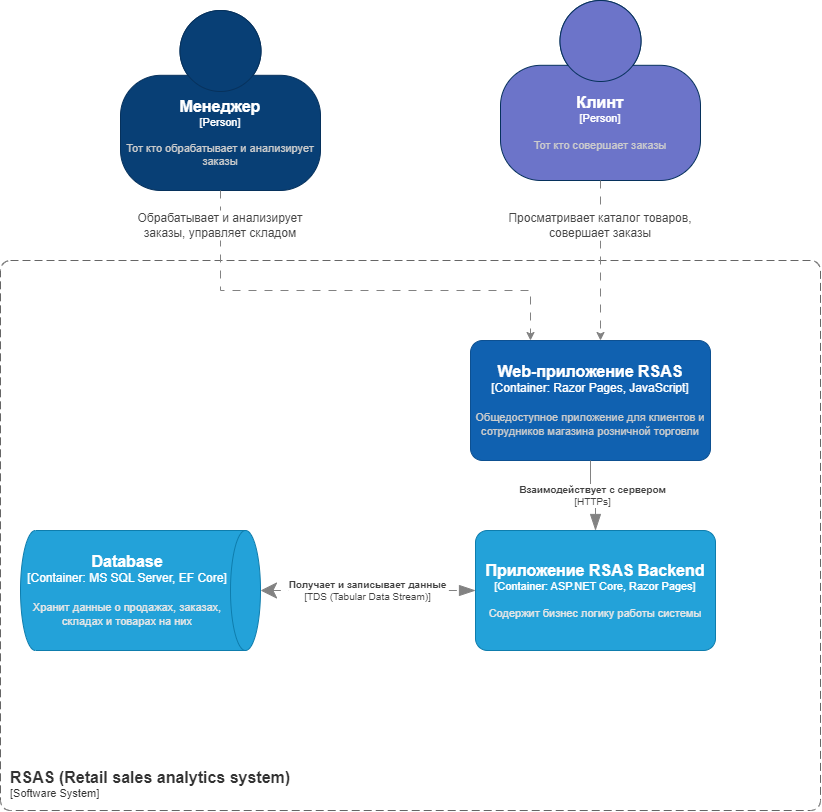


Рисунок 1 - Диаграмма С4 на контекстном уровне

Затем была построена диаграмма С4 на компонентном уровне, представленная на рисунке 2.

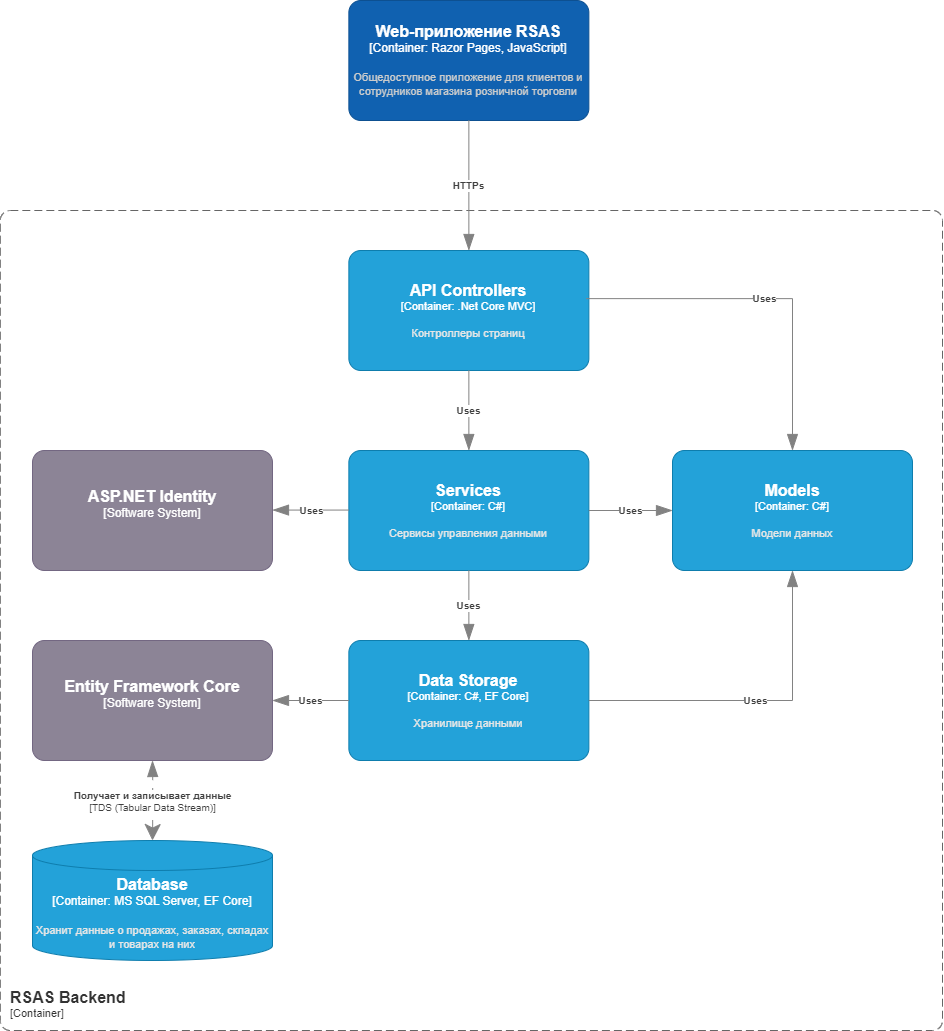


Рисунок 2 - Диаграмма С4 на компонентном уровне

Исходя из диаграммы компонентного уровня, RSAS Backend состоит из следующих контейнеров:

* API Controller: Контроллеры предоставляющие API для взаимодействия с системой.
* Services: Сервисы которые предоставляют простой интерфейс работы с данными для контроллеров.
* Models: Модели данных системы.
* Data Storage: Прослойка для управления базой данных.

Кроме того, RSAS Backend использует следующие внешние системы:

* Entity Framework Core: ORM-инструмент доступа к данным БД
* ASP.Net Identity: Система аутентификации и авторизации

На рисунке 3 представлена диаграмма С4 API Controllers на компонентном уровне.

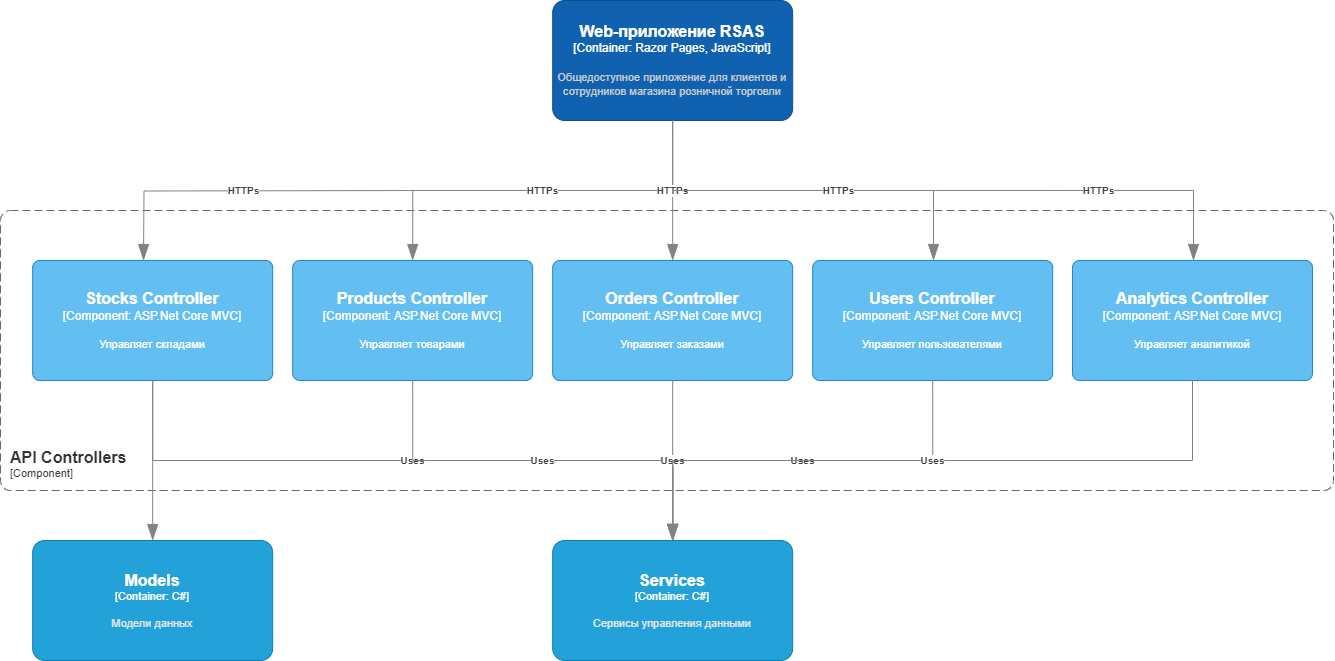


Рисунок 3 – Диаграмма С4 API Controllers на компонентном уровне

На рисунке 4 представлена диаграмма C4 Services на компонентном уровне.

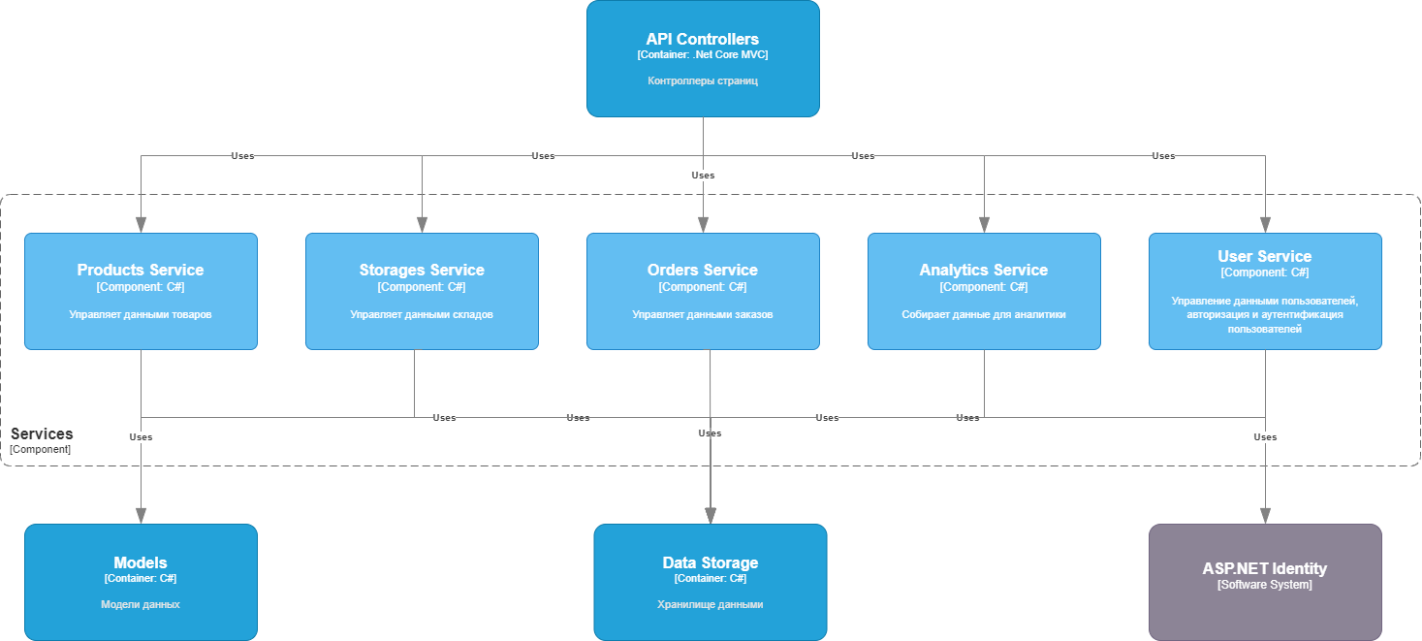


Рисунок 4 – Диаграмма С4 Services на компонентном уровне

На рисунке 5 представлена диаграмма С4 Data Storage на компонентном уровне.

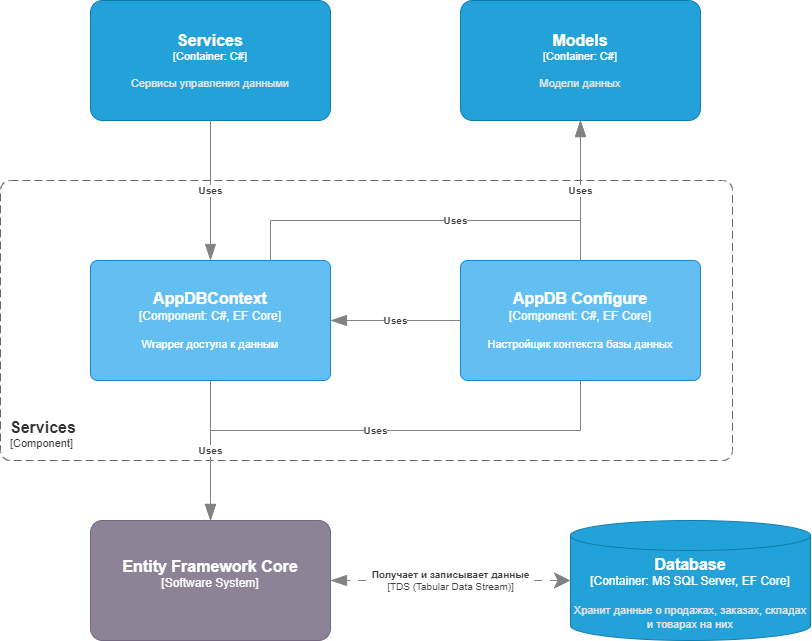


Рисунок 5 – Диаграмма С4 Data Storage на компонентном уровне

Приложения, разработанные с использованием чистой архитектуры, обладают высокой адаптивностью. Они легко подстраиваются под новые требования, сохраняя при этом качество кода и производительность. Такая структура упрощает понимание и сопровождение системы, делая её доступной для других разработчиков.

Чистая архитектура также помогает избежать проблем с зависимостями, разделяя приложение на логические модули. Это облегчает добавление новых функций и изменение существующих, сокращая время разработки и делая приложение более гибким и масштабируемым.

На рисунке 6 представлен шаблон зависимости модулей для платформы ASP.Net Core согласно Clean Architecture.

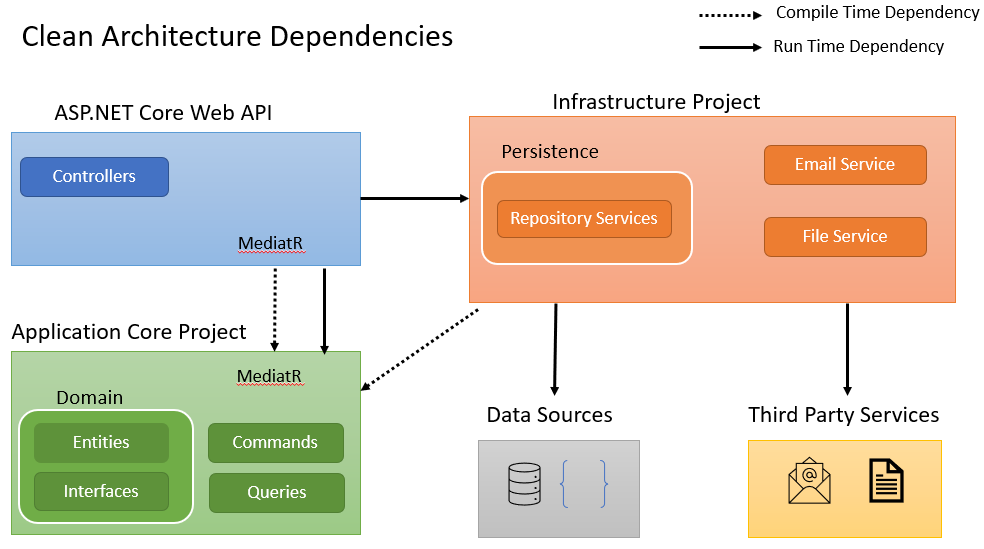


Рисунок 6 – Шаблон зависимости модулей для платформы ASP.Net Core согласно Clean Architecture

Clean Architecture состоит из нескольких принципов, которые помогают разработчикам создавать качественное и устойчивое программное обеспечение.

Ключевой принцип Clean Architecture – разделение приложения на уровни, каждый из которых выполняет свои задачи и управляет своей ответственностью. Обычно такое разделение выглядит следующим образом:

1. Интерфейс пользователя (UI): отвечает за взаимодействие с пользователем.
2. Представление (Presentation): отображает данные и обрабатывает пользовательский ввод.
3. Приложение (Application): содержит бизнес-логику и оркестрирует взаимодействие между различными компонентами системы.
4. Домен (Domain): содержит ядро системы и включает в себя основную бизнес-логику и модели данных.
5. Инфраструктура (Infrastructure): предоставляет реализации абстракций, таких как базы данных, внешние сервисы и т.д.

Уровень представления отвечает за взаимодействие с пользователем и обработку запросов. Уровень приложения выполняет бизнес-логику и координирует работу между уровнями представления и домена. Уровень домена содержит бизнес-логику и компоненты, отвечающие за работу с данными. Уровень инфраструктуры занимается поддержкой структур приложения и связью с внешними системами (например, базами данных, API и т.д.).

**Диаграмму классов в GitHub:** <https://github.com/FranGranDev/RSAS/blob/main/Documentation/Code%20Design/Core/ClassDiagram.png>

Таким образом, данная диаграмма классов иллюстрирует структуру системы, которая обеспечивает эффективное взаимодействие пользователей и бизнес-логики, в рамках концепции чистой архитектуры, способствуя созданию масштабируемого и поддерживаемого приложения розничной торговли.

**UI Kit программного средства:** <https://bootswatch.com/minty/>

Для создания качественного, адаптивного и стильного пользовательского был выбран фреймворк Bootstrap, а в частности тема Minty. Bootstrap — это открытый и бесплатный HTML-, CSS- и JS-фреймворк, который используется для быстрой верстки адаптивных дизайнов сайтов и веб-приложений. Включает в себя CSS- и HTML-шаблоны оформления для веб-форм, меток, типографики, кнопок, блоков навигации и других компонентов веб-интерфейса.

**Макет дизайна программного средства:** <https://github.com/FranGranDev/RSAS/tree/main/Documentation/UI%20Design>

**Вывод:** В системе RSAS реализована интеграция веб-приложений, серверного приложения и базы данных согласно принципам Clean Architecture. Система предназначена для автоматизации процесса розничной торговли и в частности аналитики заказов и продаж. Она включает в себя два модуля: Web-приложение и Backend-приложение.

Web-приложение предназначено для управления товарами, складами и аналитики продаж со стороны менеджера и просмотра каталога и оформления заказа со стороны клиента.

Backend-приложение предназначено для обработки бизнес-логики и управления данными в системе розничной торговли. Оно обеспечивает взаимодействие между Web-приложениями и базами данных, обрабатывает запросы пользователей, управляет аутентификацией и авторизацией, а также выполняет CRUD операции, товаров, складов и заказов.