**Лабораторная работа №4**

**Оглавление**

[Лабораторная работа № 4. ASP.Net Чистая архитектура 3](#_Toc189509057)

[Содержание пояснительной записки 9](#_Toc189509058)

[Используемое программное обеспечение 10](#_Toc189509059)

[Список литературы 11](#_Toc189509060)

Лабораторная работа № 4.  
ASP.Net Чистая архитектура

**Цель лабораторной работы**

Закрепление теоретических знаний по основам разработки приложений на C# с помощью технологии ASP.NET в соответствии с подходом «Чистая архитектура».

**Постановка задачи**

ASP.NET — это платформа для разработки веб-приложений, созданная компанией Microsoft. Она позволяет разработчикам создавать динамические веб-сайты, веб-приложения и веб-сервисы. ASP.NET поддерживает языки .NET, такие как C#, и предоставляет мощные инструменты и библиотеки для упрощения разработки.

**Основные компоненты ASP.NET**

1. **ASP.NET Web Forms**: Позволяет создавать веб-приложения с использованием событийно-ориентированной модели, аналогичной Windows-приложениям. Подходит для быстрого создания приложений с минимальным количеством кода.
2. **ASP.NET MVC**: Архитектурный паттерн Model-View-Controller, который разделяет приложение на три основных компонента:
   * **Model**: Представляет данные и бизнес-логику.
   * **View**: Отвечает за отображение данных пользователю.
   * **Controller**: Обрабатывает пользовательские запросы, взаимодействует с моделью и выбирает представление для отображения.
3. **ASP.NET Web API**: Позволяет создавать RESTful сервисы, которые могут быть использованы для взаимодействия с клиентскими приложениями, такими как мобильные приложения и одностраничные приложения (SPA).
4. **ASP.NET Core**: Современная версия ASP.NET, которая является кросс-платформенной, высокопроизводительной и более легковесной. Поддерживает создание как веб-приложений, так и API.

В рамках чистой архитектуры центральным элементом приложения являются его бизнес-логика и модель.

В этом случае бизнес-логика не зависит от доступа к данным или другим инфраструктурам, то есть стандартная зависимость инвертируется: инфраструктура и детали реализации зависят от ядра приложения.

Эта функциональность достигается путем определения абстракций или интерфейсов в ядре приложения, которые реализуются типами, определенными в слое инфраструктуры. Такую архитектуру обычно рисуют в виде серии окружностей с общим центром, которая внешне напоминает срез луковицы (рисунок 1).

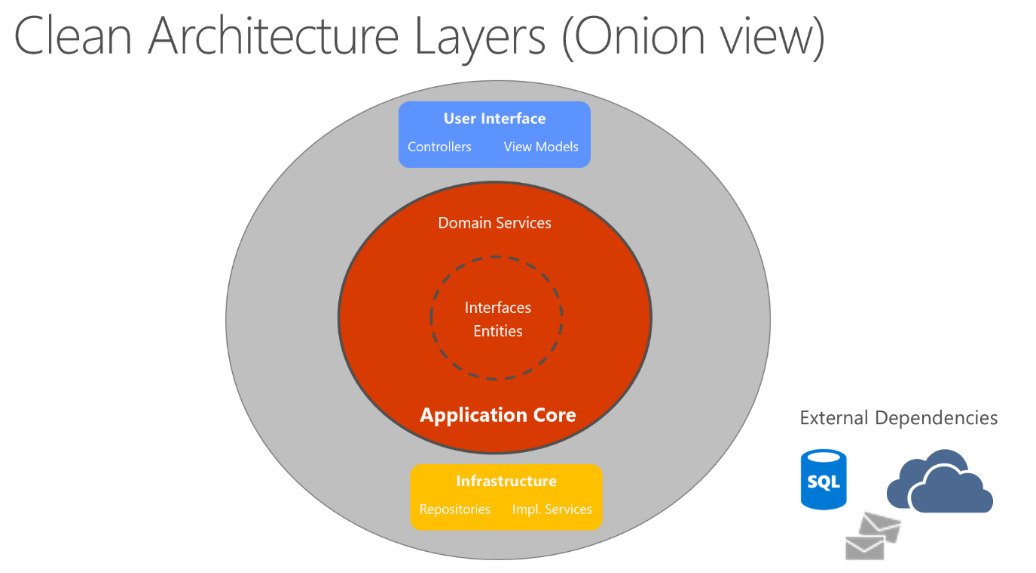
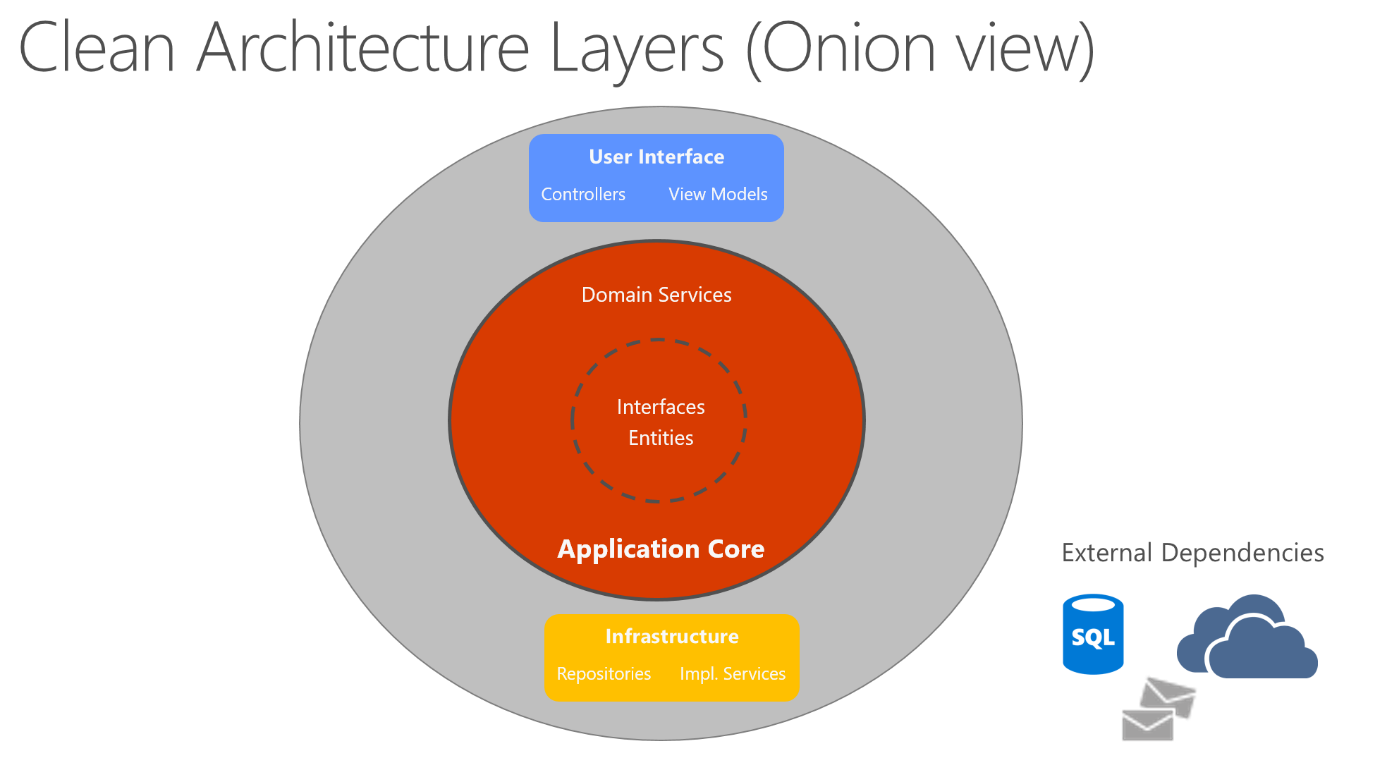
 

Рисунок 1 – Общая схема N-слойной архитектуры

Чистую архитектуру можно описать фразой **- строй приложение вокруг бизнес-логики.** Схема архитектуры представлена на рисунке 2.



Рисунок 2 – Схема «Чистая архитектура»

**Entities**

* Сущности, которые содержат самую важную логику/алгоритмы
* **Пример:** расчет угла броска снаряда

**Use Cases**

* + Наши бизнес-процессы
  + Этот слой содержит прикладную логику - логику нашей предметной области
  + Занимается тем, что манипулирует Entities
  + **Пример:** приложения для расчета угла удара в бильярде и траектории выстрела из пушки

**Контроллеры/Презентеры**

* + Те, с кем внешний мир взаимодействует
  + **Пример:** MVC контроллеры, заглушка для HTTP, модель представления (MVVM)

**UI, БД, Устройства**

* + Это тот самый внешний мир
  + **Пример:** база данных, внешний сервис кеширования или логирования

При правильной реализации поток управления выглядит так:

**Controller -> Use Case Interactor -> Presenter**.

**Пример:**

1. Клиент сделал запрос на контроллер;
2. Контроллер вызвал нужный метод на доменном сервисе;
3. Доменный сервис вызвал метод хранилища, который обновил базу данных.

**Задание на лабораторную работу**

1. Разработать ASP.NET приложение в соответствии с подходом «Чистая архитектура» для предметной области согласно варианту.
2. Для оценки:
3. **удовлетворительно** необходимо использовать 1 основной класс из варианта;
4. **хорошо** – 2 класса из варианта;
5. **отлично** – все классы из варианта;
6. Для успешной защиты:

* программа должна быть задокументирована с помощью комментариев,
* код программы не должен содержать не используемые блоки, должен быть «чистым» и стилистически верно оформлен,
* интерфейс программы должен быть гибким и удобным.

**Варианты заданий на лабораторную работу**

Можно придумать свой **уникальный** вариант.

1. **Пользовательская система**
   * **Класс User**
   * **Класс Role**
   * **Класс Permission**
2. **Интернет-магазин**
   * **Класс Product**
   * **Класс Category**
   * **Класс Order**
   * **Класс ShoppingCart**
3. **Система управления проектами**
   * **Класс Project**
   * **Класс Task**
   * **Класс Comment**
   * **Класс User Assignment**
4. **Система блогов**
   * **Класс Post**
   * **Класс Comment**
   * **Класс Tag**
   * **Класс Author**
5. **Финансовое приложение**
   * **Класс Transaction**
   * **Класс Account**
   * **Класс Category**
   * **Класс Budget**
6. **Система бронирования**
   * **Класс Reservation**
   * **Класс Customer**
   * **Класс Room**
   * **Класс Payment**
7. **Система управления событиями**
   * **Класс Event**
   * **Класс Venue**
   * **Класс Ticket**
   * **Класс Participant**
8. **Система отзывов**
   * **Класс Review**
   * **Класс Product**
   * **Класс User**
   * **Класс Rating**
9. **Система обучения**
   * **Класс Course**
   * **Класс Student**
   * **Класс Instructor**
   * **Класс Enrollment**
10. **Система управления библиотекой**
    * **Класс Book**
    * **Класс Author**
    * **Класс Member**
    * **Класс Loan**
11. **Система управления задачами**
    * **Класс Task**
    * **Класс User**
    * **Класс Status**
    * **Класс Priority**
12. **Система опросов**
    * **Класс Survey**
    * **Класс Question**
    * **Класс Answer**
    * **Класс Respondent**
13. **Система управления контактами**
    * **Класс Contact**
    * **Класс Group**
    * **Класс Address**
    * **Класс PhoneNumber**
14. **Система управления поставками**
    * **Класс Supplier**
    * **Класс Product**
    * **Класс Order**
    * **Класс Shipment**
15. **Система управления задачами по разработке**
    * **Класс Issue**
    * **Класс Project**
    * **Класс User**
    * **Класс Comment**
16. **Система управления здравоохранением**
    * **Класс Patient**
    * **Класс Appointment**
    * **Класс Doctor**
    * **Класс Prescription**
17. **Система управления отзывами о фильмах**
    * **Класс Movie**
    * **Класс Review**
    * **Класс User**
    * **Класс Genre**
18. **Система управления ресторанами**
    * **Класс Restaurant**
    * **Класс MenuItem**
    * **Класс Reservation**
    * **Класс Review**
19. **Система управления недвижимостью**
    * **Класс Property**
    * **Класс Owner**
    * **Класс Tenant**
    * **Класс Lease**
20. **Система управления задачами в команде**
    * **Класс Team**
    * **Класс Task**
    * **Класс User**
    * **Класс Comment**
21. **Система управления спортивными мероприятиями**
    * **Класс Event**
    * **Класс Team**
    * **Класс Player**
    * **Класс Score**
22. **Система управления подписками**
    * **Класс Subscription**
    * **Класс User**
    * **Класс Plan**
    * **Класс Payment**
23. **Система управления курсами валют**
    * **Класс Currency**
    * **Класс ExchangeRate**
    * **Класс Transaction**
    * **Класс User**
24. **Система управления задачами по уборке**
    * **Класс CleaningTask**
    * **Класс User**
    * **Класс Schedule**
    * **Класс Location**
25. **Система управления инвентарем**
    * **Класс Item**
    * **Класс Category**
    * **Класс Supplier**
    * **Класс Stock**
26. **Система управления проектами для фрилансеров**
    * **Класс Project**
    * **Класс Freelancer**
    * **Класс Client**
    * **Класс Proposal**

# Содержание пояснительной записки

1. Постановка задачи. Приводится теоретический материал, использованный при написании приложения.

2. Формулировка задания и вариант. Приводится задание на лабораторную работу и вариант этого задания.

3. Описание выполняемых действий. Необходимо привести описание последовательности разработки программы, реализации используемых методов, алгоритмов, блок-схем.

4. Анализ результатов. Привести анализ входных и выходных данных. Показать результаты выполнения программного кода. Предоставить скриншоты обработки тестовых примеров. Сделать выводы.

5. Листинг программы. Привести листинг разработанного программного кода, содержание файлов входных и выходных данных.

# Используемое программное обеспечение

1. Среда программирования MS Visual Studio Community 2022 (Свободно распространяемое программное обеспечение (в учебных целях));
2. Microsoft Office Standard 2007 (Open License: 42267924);
3. Open Office (Свободно распространяемое программное обеспечение).
4. Браузер (Свободно распространяемое программное обеспечение).

# Список литературы

* + - 1. Мейер Б. Объектно-ориентированное программирование и программная инженерия [Электронный ресурс]/ Мейер Б. – Электрон. текстовые данные. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 285 c.
      2. Биллиг, В. A. Основы объектного программирования на С# (C# 3.0, Visual Studio 2008) [Электронный ресурс] : учебное пособие / В. A. Биллиг. — Электрон. текстовые данные. — Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. — 583 c. — 978-5-4487-0145-0. — Режим доступа: http://www.iprbookshop.ru/72339.html
      3. Павловская, Т. А. Программирование на языке высокого уровня C# [Электронный ресурс] / Т. А. Павловская. — 2-е изд. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 245 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/73713.html
      4. Агапов, В. П. Основы программирования на языке С# [Электронный ресурс] : учебное пособие / В. П. Агапов. — Электрон. текстовые данные. — М. : Московский государственный строительный университет, ЭБС АСВ, 2012. — 128 c. — 978-5-7264-0576-6. — Режим доступа: http://www.iprbookshop.ru/16366.html
      5. Медведев, М. А. Программирование на СИ# [Электронный ресурс] : учебное пособие / М. А. Медведев, А. Н. Медведев ; под ред. А. В. Присяжный. — Электрон. текстовые данные. — Екатеринбург : Уральский федеральный университет, ЭБС АСВ, 2015. — 64 c. — 978-5-7996-1561-1. — Режим доступа: http://www.iprbookshop.ru/69667.html
      6. Казанский А.А. Объектно-ориентированное программирование на языке Microsoft Visual С# в среде разработки Microsoft Visual Studio 2008 и .NET Framework. 4.3 [Электронный ресурс]: учебное пособие и практикум/ Казанский А.А.— Электрон. текстовые данные.— М.: Московский государственный строительный университет, ЭБС АСВ, 2011.— 180 c
      7. Уйманова Н.А. Основы объектно-ориентированного программирования [Электронный ресурс]: практикум/ Уйманова Н.А., Таспаева М.Г.— Электрон. текстовые данные.— Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2017.— 156 c.— Режим доступа: http://www.iprbookshop.ru/78808.html.— ЭБС «IPRbooks»
      8. Новиков П.В. Объектно-ориентированное программирование [Электронный ресурс]: учебно-методическое пособие к лабораторным работам/ Новиков П.В.— Электрон. текстовые данные.— Саратов: Вузовское образование, 2017.— 124 c.— Режим доступа: http://www.iprbookshop.ru/64650.html.— ЭБС «IPRbooks»