Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Отчёт**

по предмету «Проектирование программного обеспечения»

Лабораторная работа №6

«Интерфейс поиска и бронирования лекарств»

Студент: Немкович А. В.

ФИТ 3 курс 1 группа

Преподаватель: Якубенко К. Д

Минск 2024

# 1.Постановка задачи

## 1.1. Функциональное назначение программы

Программа предназначена для предоставления пользователям удобного сервиса по поиску, бронированию и покупке лекарственных средств в аптеках города. Функциональное назначение платформы заключается в упрощении процесса поиска необходимых медикаментов и повышении доступности лекарств для населения.

Для пользователя программа предоставляет возможность быстро находить нужные лекарства и медицинские товары, вводя их название или активное вещество, с доступом к информации о наличии в аптеках города и цене, что позволяет сравнивать предложения и выбирать наиболее выгодные. Программа также предоставляет возможность бронирования медикаментов для самовывоза из выбранной аптеки с отображением контактной информации, адреса и времени работы.

Для зарегистрированного пользователя предусмотрен личный кабинет с функциями управления заказами и бронированиями, просмотра истории покупок, настройки уведомлений и создания списков избранных товаров. Интеграция с картами предоставляет возможность просматривать расположение аптек на карте и строить маршруты до них, а также оставлять и просматривать отзывы о товарах и аптеках, что помогает в выборе.

Для партнёрских аптек программа предоставляет возможность обновлять информацию о товарах, их наличии и ценах, управлять заказами и бронированиями через административную панель. Интеграция с базами данных и системами управления запасами обеспечивает актуальность данных и безопасность транзакций.

## 1.2. Требования к составу выполняемых функций

Любой пользователь интернета может подключиться к сервису для поиска лекарств в аптеках города, набрав правильный URI в адресной строке браузера. По умолчанию он соединяется с сервисом как пользователь с ролью «Гость» (пользователь по умолчанию).

В режиме «Гость» пользователю доступен для ознакомления документ «Соглашение об уровне услуг» и следующие функции:

* Производить поиск лекарств по названию, активному веществу или другим параметрам для получения информации о доступных препаратах в городе.
* Просматривать подробную информацию о каждом найденном лекарстве, включая инструкцию по применению, показания, противопоказания и побочные эффекты.
* Узнавать о наличии лекарств в ближайших аптеках и сравнивать цены, чтобы выбрать наиболее выгодное предложение.
* Просматривать информацию о необходимой аптеке: местоположение, время работы, контактные номера (реквизиты, связь с отделами предприятий, адрес для электронных обращений).
* Получать персонализированную информацию о побочных эффектах, взаимодействии с другими лекарствами, необходимости приема на пустой желудок и других особенностях применения конкретного препарата.
* Получить доступ к справочной информация и отзывам.
* Возможность регистрации и аутентификации для получения роли «Пользователь», «Менеджер» или «Представитель аптеки».

Регистрация осуществляется путем заполнения формы, с указанием основной необходимой информации о пользователе.

После регистрации пользователь автоматически переключается в режим «Пользователь», соответствующий пользовательской роли «Пользователь» (зарегистрированный пользователь).

Зарегистрированному пользователю доступны все возможности пользователя роли «Гость» и дополнительные функции, позволяющие ему:

* Оставлять отзывы о лекарствах, делиться своим опытом и помогать другим пользователям принять обоснованное решение при выборе лекарства.
* Осуществлять бронь необходимого лекарства.
* Добавлять необходимые лекарства в «Избранное», что в будущем позволит пользователю осуществить быстрый поиск.

При подключении пользователя в режиме «Менеджер» ему доступна возможность:

* Управлять ресурсом сервиса, включая добавление, редактирование и удаление информации о лекарствах.
* Получать статистику использования сервиса и предоставлять отчеты администратору сервиса.
* Поддержание обратной связи с пользователем интернет-сервиса.

При подключении пользователя в режиме «Представитель аптеки» ему доступна возможность:

* Управлять ресурсом сервиса, включая добавление, редактирование и удаление информации о лекарствах.
* Вносить изменения в базу данных в связи с изменением количества лекарства в хранилище аптеки.
* Осуществлять бронь лекарства (пользователь оставляет на интернет-сервисе запрос, который в последствии будет обработан и внесен в базу данных).

# 2.Описание программных средств

## 2.1. Описание программного средства Draw.io

Название: draw.io (diagrams.net)

Версия: Последняя доступная версия обновляется разработчиком автоматически, так как draw.io является веб-приложением. Пользователи всегда работают с самой актуальной версией, которая обновляется на сервере. Для локального использования доступны десктопные версии, которые необходимо обновлять вручную.

Разработчик: JGraph Ltd.

Адрес загрузки: https://app.diagrams.net

Режим использования: draw.io — это инструмент для создания диаграмм и схем, доступный как в режиме онлайн через браузер, так и оффлайн с помощью настольного приложения. Приложение часто используется для проектирования бизнес-процессов, архитектуры систем, блок-схем, UML-диаграмм и других визуальных моделей. Для работы онлайн не требуется регистрация, а проекты могут сохраняться на локальном устройстве, в облаке (Google Drive, OneDrive, Dropbox), или в корпоративных системах хранения.

Веб-приложение работает на всех устройствах с современными браузерами, такими как Google Chrome, Mozilla Firefox, Safari и Microsoft Edge. Десктопные версии поддерживаются на Windows, macOS и Linux, предоставляя пользователям гибкость для работы оффлайн. Приложения для мобильных платформ отсутствуют, но сайт может быть доступен через мобильный браузер для просмотра и базового редактирования.

Draw.io поддерживает создание различных типов диаграмм и моделей:

* 2D-векторные диаграммы. Позволяет проектировать блок-схемы, организационные структуры, карты процессов, схемы сетевой инфраструктуры и многое другое.
* UML и технические схемы. Поддерживаются готовые шаблоны и библиотеки для моделирования программного обеспечения, инженерных проектов и электронных схем.

Одной из ключевых особенностей draw.io является интеграция с популярными облачными платформами, такими как Google Drive, Microsoft OneDrive и GitHub, что позволяет совместно работать над схемами и сохранять их в привычной среде. Несмотря на отсутствие полноценного многопользовательского режима в реальном времени, файлы можно быстро обменивать между участниками проекта через совместный доступ в облаке.

# 3. Описание практического задания

Диаграмма развертывания для нашего проекта будет выглядеть следующим образом – Рисунок 3.1.

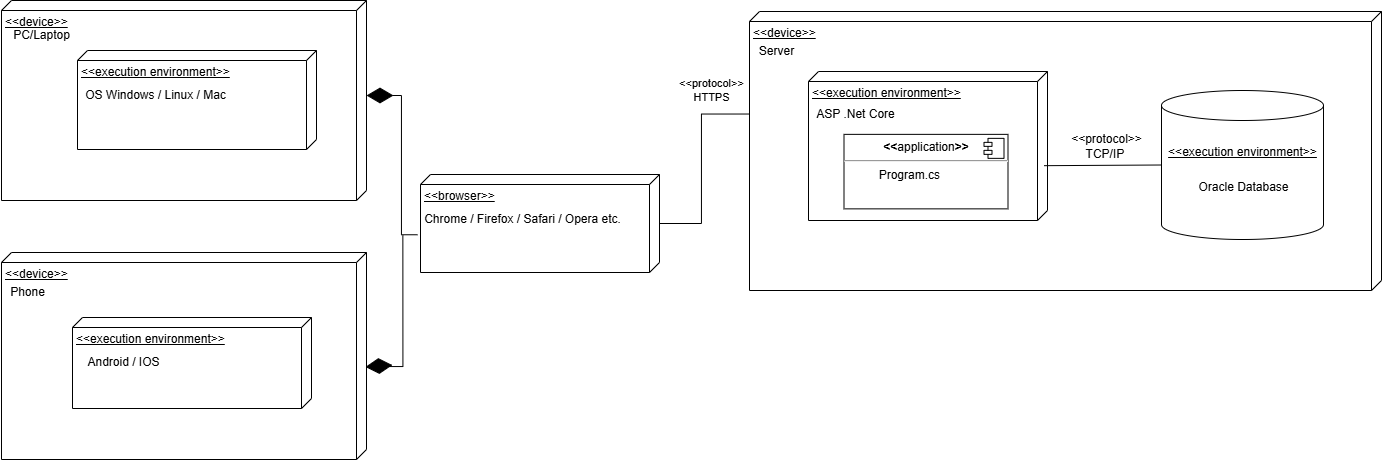


Рисунок 3.1 – Диаграмма развертывания

Для диаграммы развертывания можно выделить следующие элементы:

* Device PC/Laptop.
* Device Phone.
* Browser.
* Server.
* Database.
* Application.

Диаграмма развертывания иллюстрирует физическое размещение компонентов. Клиентские устройства через браузеры взаимодействуют с сервером, который, в свою очередь, обращается к базе данных.

Браузер использует протокол HTTPS для отправки запросов серверу приложения. Сервер, в свою очередь, реализован на базе ASP.NET Core и выполняет обработку запросов, бизнес-логику и взаимодействие с базой данных. Для управления данными используется Oracle Database, которая размещена на сервере приложения. Сервер приложения — это физический сервер поддерживающий выполнение приложений на базе ASP.NET Core и работу базы данных Oracle.

Соединения между элементами реализованы следующим образом: ПК/ноутбук или смартфон взаимодействуют с сервером приложения через безопасный протокол связи HTTPS, обеспечивающий защиту данных во время передачи. Взаимодействие между сервером приложения и базой данных Oracle осуществляется через протокол TCP/IP, что гарантирует надежное и эффективное подключение.

Среды выполнения включают ASP.NET Core, которая используется для выполнения серверной части приложения. Она отвечает за маршрутизацию, обработку запросов и управление данными. Oracle Database представляет среду выполнения для работы с данными, включая сложные SQL-запросы.

Также необходимо было сделать диаграммы компонентов, она представлена на рисунке 3.2.

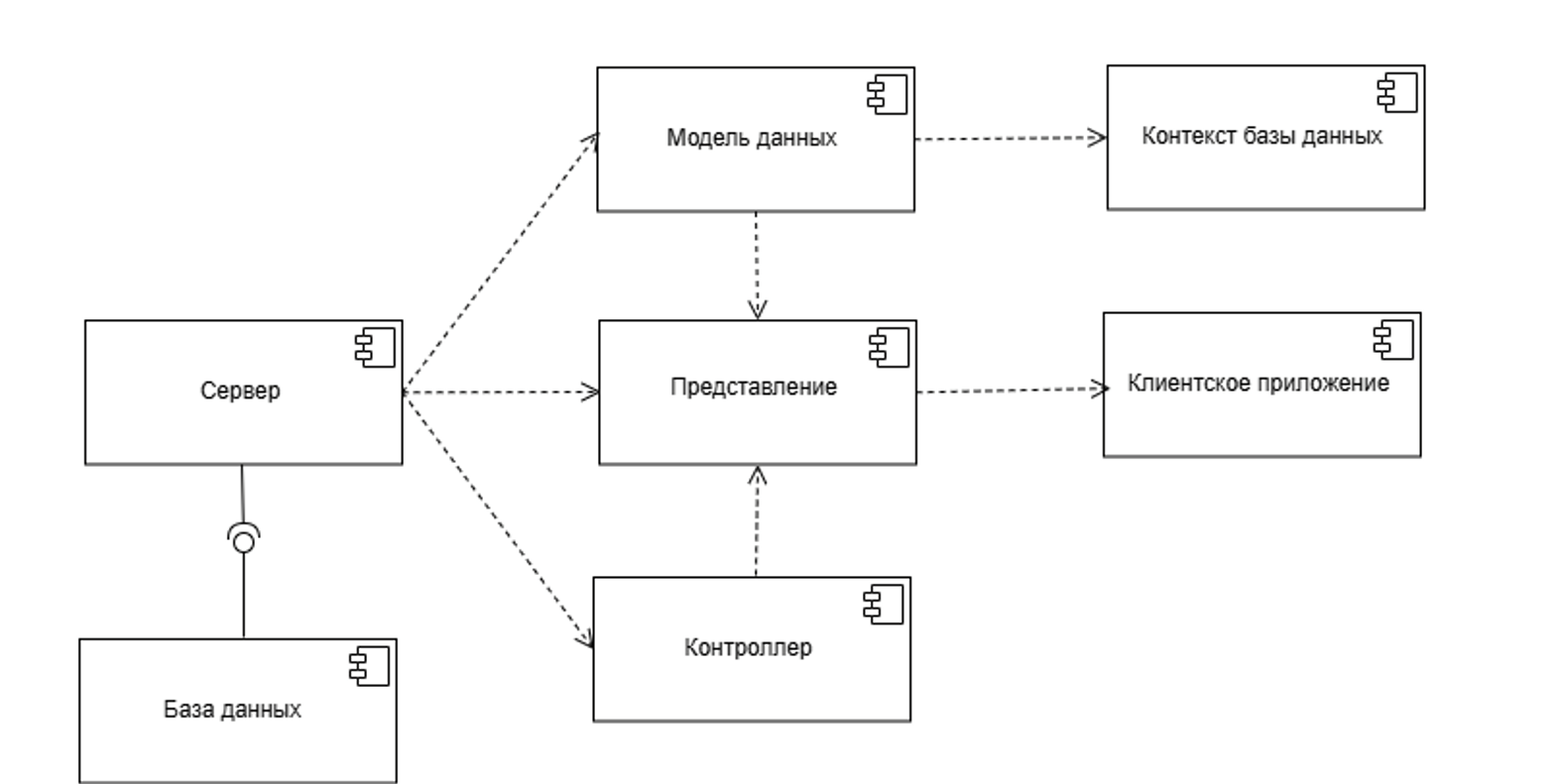


Рисунок 3.2 – Диаграмма компонентов

Опишем используемые компоненты и технологии их реализации в нашей архитектуре:

* База данных: используется СУБД Oracle Database.
* Контекст базы данных: используется Entity Framework.
* Сервер: используется ASP.NET Core.
* Модель данных: используются классы C#.
* Представление: используется Razor.
* Контроллер: используются классы C#.
* Клиентское приложение: используется встроенная в Razor технология представления (без дополнительных фреймворков).

Связи между компонентами представлены в виде зависимости и интерфейсов. Сервер взаимодействует с базой данных через Entity Framework, который обеспечивает доступ к данным.

В данной архитектуре web-приложения используется многоуровневый подход. Серверная часть, реализованная на ASP.NET Core, выполняет обработку запросов и взаимодействие с базой данных Oracle через Entity Framework.

Модель данных представлена классами C#, которые формируют структуру объектов и определяют связь с базой данных. Контроллеры (реализованные на C#) обрабатывают поступающие запросы, управляют бизнес-логикой и передают данные на уровень представления. Представление формируется с использованием Razor, что позволяет генерировать динамические HTML-страницы.

# 4.Ответы на вопросы

## 4.1. Укажите назначение физических диаграмм: компонентов и развертывания.

**Диаграмма компонентов** отображает высокоуровневую структуру системы, показывая, как различные программные компоненты взаимодействуют между собой. Она используется для моделирования физической и функциональной структуры системы, для отображения модулей, библиотек и сервисов, а также их зависимостей.

**Диаграмма развертывания** описывает физическое развертывание системы на аппаратном уровне, отображая, как компоненты системы (например, программные приложения) разворачиваются на физических устройствах или узлах. Она помогает понять, как приложение будет функционировать в реальной инфраструктуре, и показывает взаимодействие между различными узлами.

## 4.2. Дайте определение понятиям: узел, артефакт, интерфейс.

Узел представляет собой физическое или виртуальное устройство, на котором могут выполняться компоненты системы. Он может быть сервером, устройством хранения, компьютером, мобильным устройством и т.д. Узлы отображаются как прямоугольники на диаграммах развертывания.

Артефакт — это результат выполнения процессов разработки или развертывания, например, скомпилированный файл, библиотека, модуль или другой компонент, который имеет физическое существование и может быть развернут на узле. Артефакты отображаются на диаграммах компонентов и развертывания как прямоугольники.

Интерфейс представляет собой точку взаимодействия компонента или узла с другими компонентами или узлами. Он определяет доступные операции и услуги, которые могут быть использованы другими компонентами. Интерфейсы отображаются как маленькие круги, связанные с компонентами или узлами на диаграммах.

## 4.3. Опишите нотации, которые используются для представления компонентов (их вариации).

Стандартный компонент представлен прямоугольником с изображением маленькой "папки" в углу (или стереотипом «component»). Такой элемент показывает единицу программного кода или приложения, что помогает в удобстве разработки.

Вариации компонента могут включать стереотипы для различия между разными типами компонентов (например, "исполняемый компонент", "библиотека").

Также компоненты могут иметь интерфейсы, которые отображаются в виде полуэллипсов (для предоставляемых интерфейсов) или кругов с линиями (для требуемых интерфейсов).

## 4.4. Опишите основные нотации, которые используются для представления архитектуры системы в виде диаграммы развертывания.

* Узел отображается в виде трехмерного куба, представляющего физическое или виртуальное устройство.
* Связи между узлами (например, коммуникации по сети) обычно изображаются в виде линий между узлами.
* Артефакты, развернутые на узлах, могут быть изображены внутри узлов, показывая, где и как выполняются конкретно выбранные программные компоненты.
* Компоненты и артефакты могут быть связаны с узлами через линии ассоциации, что демонстрирует их развертывание на конкретно выбранном устройстве.

## 4.5. Укажите основные виды связей между компонентами и между узлами.

Связи между компонентами:

* **Зависимости**: Стрелки с пунктирной линией, показывающие, что один компонент зависит от другого для выполнения своей функции.
* **Ассоциации**: Связи между компонентами, показывающие, что один компонент использует функциональность другого.
* Реализация (realization) — показывает, что компонент или узел реализует интерфейс, предоставленный другим элементом.
* Композиция (composition) — более сильная связь, показывающая, что один компонент является частью другого.
* **Интерфейсы**: Связи между компонентами и их интерфейсами, обозначающие, как компоненты взаимодействуют через предоставленные интерфейсы.

Связи между узлами:

* **Коммуникации**: Линии, соединяющие узлы, показывающие, что они взаимодействуют друг с другом, например, через сеть или шину данных.
* **Связи с артефактами:** Связи, показывающие, какие артефакты развернуты на каких узлах.
* **Развертывание**: Линии, соединяющие артефакты с узлами, обозначающие, где именно будет развернут компонент или приложение.