Титульный лист

Лист задания

**Содержание**

Введение

1. Теоретическая часть

2. Технологическая часть

3. Практическая часть

Заключение

Список использованных источников

Приложения

**ВВЕДЕНИЕ**

Темой работы является разработка программной системы для управления аптекой с использованием технологий Blazor и ASP.NET. Целью данного проекта является создание инструмента, с помощью которого директор аптеки сможет управлять информацией о товарах легко и эффективно, а также информацией о поставщиках и других аспектах деятельности аптеки.

Актуальность работы обусловлена постоянной необходимостью оптимизации процессов управления и учета в сфере здравоохранения. Система управления аптеками является важным инструментом для обеспечения пациентов всем необходимым и эффективного ведения бизнеса. Использование современных технологий, таких как Blazor и ASP.NET , позволит создать интуитивно понятный и функциональный инструмент для директора аптеки.

Цель работы — создать программную систему, которая будет предоставлять директору аптеки всю необходимую информацию о складе, поставщиках и аптеках.

Задачи работы:

* провести библиографический обзор источников по теме работы;
* выделить существующие решения, соответствующие тематике разработки, выделить достоинства и недостатки каждого из них;
* составить алгоритм работы программы в виде текстового описания, создать макет приложения и диаграммы;
* создать интерфейс пользователя с использованием Blazor;
* создать базу данных с использованием PostgreSQL;
* написать логику приложения для обработки данных и операций;
* протестировать работу программы;
* составить отчёт по выполненной работе в соответствии с требованиями и защитить его.

**Объект исследования**

Объектом данного исследования является создание программной системы управления аптекой.

**Предмет исследования**

Предметом исследования в данной работе является разработка программной системы для автоматизации управления аптекой.

**Методы исследования**

В качестве основных методов исследования применены анализ существующих программных решений, анализ их функциональности, а также моделирование процессов управления аптекой. Практическая реализация поставленной задачи соответствует основным подходам к разработке программного обеспечения.

**Информационная база исследования**

Информационной базой исследования являются открытые источники, в том числе доступные в сети Интернет, а также руководства и документация по использованию технологий Blazor и ASP.NET.

**Краткая структура**

В данном отчете будет представлен процесс разработки программного продукта, в том числе теоретический обзор области и системы, технологическое проектирование и описание системы, а также непосредственно результаты разработки.

1. **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**
2. **Обзор области разработки**

Область разработки программной системы для управления аптекой является критически важной в контексте оптимизации бизнес-процессов и обеспечения эффективной работы аптечного предприятия. Предметная область включает в себя широкий спектр операций, таких как учет и хранение информации о товарах и поставщиках.

Анализ информационных процессов позволяет выявить ряд сложностей и проблем, с которыми сталкиваются аптеки в повседневной деятельности. Некоторые из них включают:

1. Неэффективное управление запасами: Отсутствие системы учета и контроля за запасами может привести к переизбытку или нехватке товаров, что в свою очередь может привести к финансовым потерям или потере клиентов.

2. Затруднения в заказах у поставщиков: без автоматизированной системы заказов у поставщиков аптеки могут столкнуться с задержками в поставках, ошибками при заказе и неоптимальным использованием финансовых ресурсов.

3. Недостаточный контроль за сроками годности: Отсутствие эффективного механизма отслеживания и контроля за сроками годности лекарств может привести к выдаче устаревших препаратов, что может негативно сказаться на здоровье пациентов и привести к претензиям со стороны контролирующих органов.

Эти проблемы могут иметь серьезные последствия для бизнеса, такие как ухудшение репутации, потеря клиентов и убытки. Разработка программной системы для управления аптекой направлена на решение этих проблем путем автоматизации и оптимизации бизнес-процессов, что повысит эффективность и конкурентоспособность аптечного предприятия.

1. **Обзор существующих продуктов**

С появлением и развитием информационных технологий в современной аптечной отрасли возникает необходимость в эффективном управлении аптечным предприятием. Рассмотрено 4 программных продукта, функционал которых включает в себя управление аптечным предприятием.

1. **1С: Фармацевтика**

1С: Фармацевтика - это программное решение, разработанное на платформе 1С:Предприятие, специально адаптированное для аптечной отрасли. Оно включает в себя модули для учета и управления аптечным предприятием, включая складской учет, учет продаж, финансовый учет, документооборот и аналитику.

Сильные стороны:

* Интегрированное решение, охватывающее все основные аспекты управления аптекой
* Интегрированное решение, охватывающее все основные аспекты управления аптекой.
* Поддержка законодательных требований и стандартов фармацевтической отрасли.

Среди недостатков:

* Высокая стоимость лицензирования и внедрения;
* Требует специализированных знаний для настройки и обслуживания.

1. **СКАУТ-Фарма**

СКАУТ-Фарма - это программный комплекс для автоматизации управления аптечным предприятием. Он включает в себя модули для учета и управления складом, продажами, клиентами, а также аналитики и отчетности.

Достоинствами приложения СКАУТ-Фарма являются [5]:

* Развитый функционал, охватывающий все основные аспекты управления аптекой;
* Интеграция с внешними сервисами и системами;

Из недостатков можно выделить следующие [5]:

* Ограниченный функционал в сравнении с некоторыми другими решениями;
* Возможные проблемы с обновлением и поддержкой;

1. **Медицинская аптека (МедАптека)**

Медицинская Аптека (МедАптека) - это программное решение для автоматизации управления аптекой. Оно предоставляет инструменты для учета товаров, клиентов, продаж и аналитики.

Достоинствами МедАптека являются:

* Простой интерфейс;
* Функционал, ориентированный на основные потребности аптеки;

Среди недостатков данного решения:

* Не столь развитый функционал, как у некоторых других программ.
* Могут возникать проблемы с маштабируемостью для крупных аптечных сетей

1. **Аптекарь (AptekaR)**

Аптекарь (AptekaR) - это комплексное программное решение для управления аптечным предприятием. Он позволяет управлять несколькими аптеками из одной системы, включает в себя учет товаров, продаж, клиентов, а также интегрируется с внешними сервисами.

Сильные стороны:

* Возможность управления несколькими аптеками из одной системы.;
* Интеграция с онлайн-кассами и электронным документооборотом.;

Среди недостатков данного решения:

* Не всегда стабильная работа при большом количестве данных.
* Требует времени на освоение и обучение персонала.

1. **Описание требований к разрабатываемому решению**

В условиях современного рынка аптечных услуг эффективное управление аптекой становится ключевым фактором успешной деятельности. Для обеспечения конкурентоспособности и оптимизации бизнес-процессов необходимо разработать программную систему, способную автоматизировать учет товаров, контроль за поставками и анализ ключевых показателей деятельности аптеки.

Описание требований:

1. Удобный интерфейс пользователя

* Пользовательский интерфейс должен быть интуитивно понятным и простым для директора аптеки
* Меню и элементы управления должны быть организованы логично и удобно для пользователя.

1. Хранение и обработка данных:

* Система должна обеспечивать надежное хранение и обработку информации об аптечном предприятии, товарах и поставщиках

1. Учет товаров:

* Возможность ведения точного учета товаров на складе с учетом основных характеристик, таких как количество, цена, срок годности и т.д

1. Управление поставками:

* Возможность заказа товаров от различных поставщиков.

1. **ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ**
   1. **Выбор инструментов и средств для реализации проекта**
      1. **Фронтенд:**
         1. Blazor:

Преимущества:

* Полностью интегрирован с экосистемой .NET, что делает его привлекательным для разработчиков, знакомых с C# и .NET.
* Возможность повторного использования компонентов как на клиентской стороне (в браузере), так и на серверной.

Недостатки:

* Из-за перевода C# в WebAssembly могут возникнуть проблемы производительности.
* Blazor требует загрузки сборок .NET на стороне клиента, размер загрузки приложения может быть довольно большим, особенно для крупных приложений, что может отрицательно сказаться на скорости загрузки страниц.
  + - 1. React

Преимущества:

* Огромное сообщество разработчиков и множество библиотек и инструментов сторонних разработчиков.
* Высокая производительность благодаря виртуальному DOM.
* Гибкость в выборе структуры проекта и использования других библиотек и фреймворков.

Недостатки:

* Использование JavaScript, что может привести к проблемам с масштабируемостью и поддержкой кода в больших проектах.
* Необходимость использования сторонних библиотек для решения определенных задач.
  + - 1. Angular

Преимущества:

* Полностью интегрирован с экосистемой TypeScript и предоставляет множество инструментов для разработки.
* Встроенные функции для работы с HTTP-запросами, маршрутизацией и формами.
* Статическая типизация TypeScript повышает надежность и облегчает рефакторинг.

Недостатки:

* Большой объем кода, который может потребовать времени для изучения и освоения.
* Сложность в сопровождении проектов из-за большого количества концепций и абстракций.

Почему был выбран Blazor:

1. Blazor позволяет повторно использовать код как на стороне клиента, так и на сервере, что уменьшает объем написанного кода и способствует поддерживаемости проекта.
2. При прочих равных условиях, Blazor обеспечивает высокую производительность и эффективность благодаря использованию языка программирования C# и интеграции с современными веб-технологиями. Это позволяет создавать мощные и отзывчивые веб-приложения, без ущерба для производительности.
3. **Бекенд:**
4. ASP.NET

Преимущества:

* Мощный и гибкий инструментарий для создания различных типов веб-приложений
* Множество готовых компонентов и инструментов для разработки веб-приложений, включая возможности для работы с базами данных, аутентификации, авторизации, маршрутизации и многое другое, что упрощает разработку
* Обладает хорошей производительностью благодаря своей оптимизированной архитектуре и инструментам для управления ресурсами и масштабирования.

Недостатки:

* Тесно связан с операционной системой Windows, что может ограничить выбор хостинг-провайдеров и требовать использования серверов под управлением Windows.
* В некоторых случаях масштабированиеможет быть сложной задачей, особенно при работе с большими объемами данных или высокими нагрузками.

1. Node.js с Express.js:

Преимущества:

* Простота и гибкость в использовании, асинхронность, обширное сообщество и множество доступных плагинов и модулей.

Недостатки:

* Возможные проблемы с производительностью из-за однопоточной природы Node.js и необходимости обработки большого количества одновременных запросов.

1. Java с Spring boot

Преимущества:

* Надежность, масштабируемость, обширная экосистема и поддержка, высокая производительность, множество интегрированных инструментов и библиотек.

Недостатки:

* Большой объем кода и настройки, медленный процесс сборки и развертывания по сравнению с некоторыми другими фреймворками.

Почему был выбран ASP.NET

1. ASP.NET предоставляет множество инструментов и механизмов дл обеспечения безопасности приложений, включая встроенную поддержку аутентификации, авторизации, защиту от атак и управление доступом к данным.
2. ASP.NET тесно интегрируется с другими технологиями и инструментами от Microsoft, такими как Entity Framework для работы с базами данных, Azure для облачного хостинга и многое другое. Это обеспечивает единый стек технологий и упрощает интеграцию с другими сервисами Microsoft.
3. **Разработка проекта решения**

Один из важных этапов проектирования приложения – проведение исследований. Прежде всего необходимо определить, решение каких задач может потребоваться потенциальному пользователю. После этого необходимо выделить задачи и процессы, которые будут реализованы в проекте. Для этого были созданы диаграммы представленные на рисунках 2.1, 2.2 и 2.3



Рисунок 2.1 – Use case диаграмма

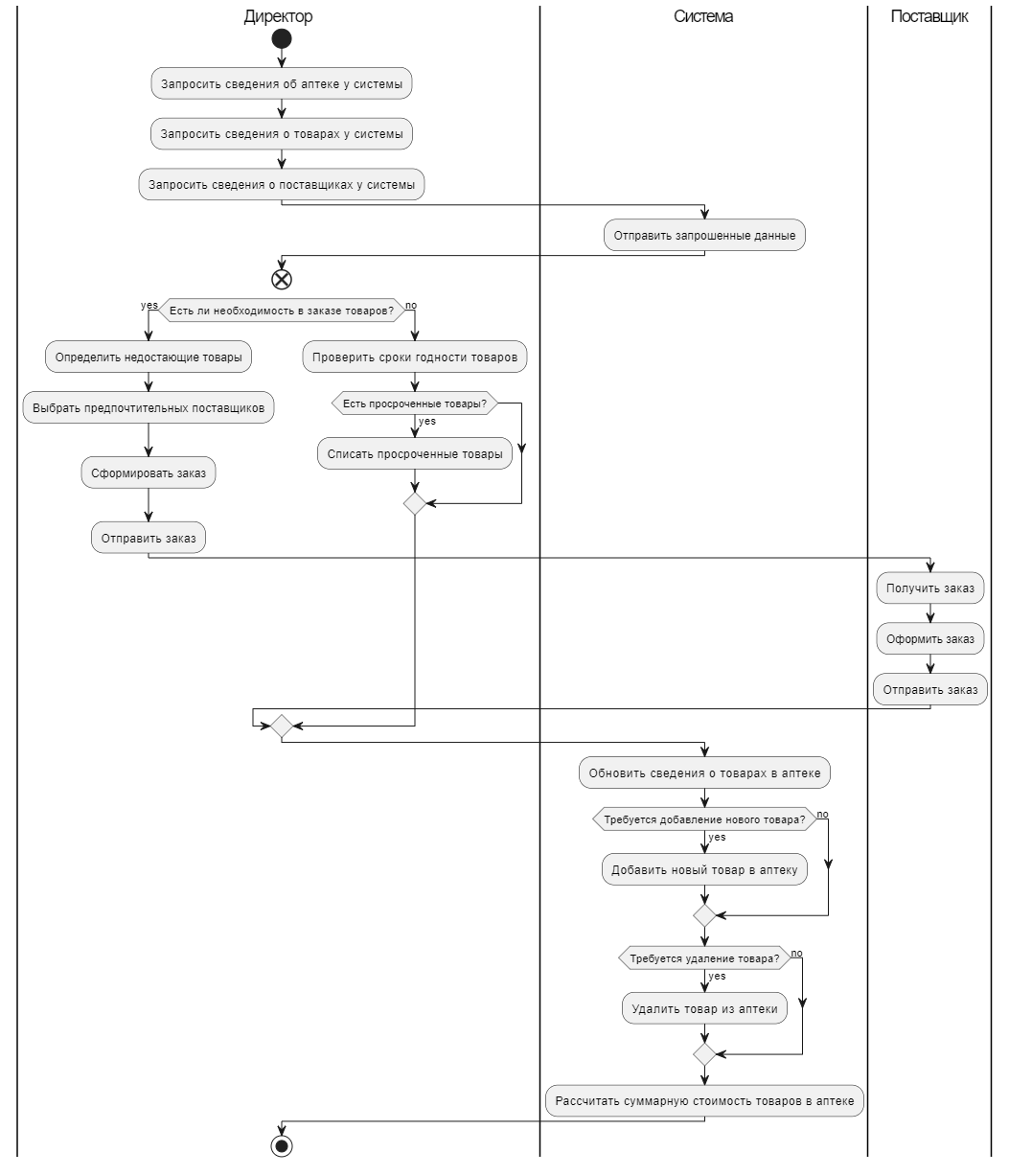


Рисунок 2.2 – Activity диаграмма

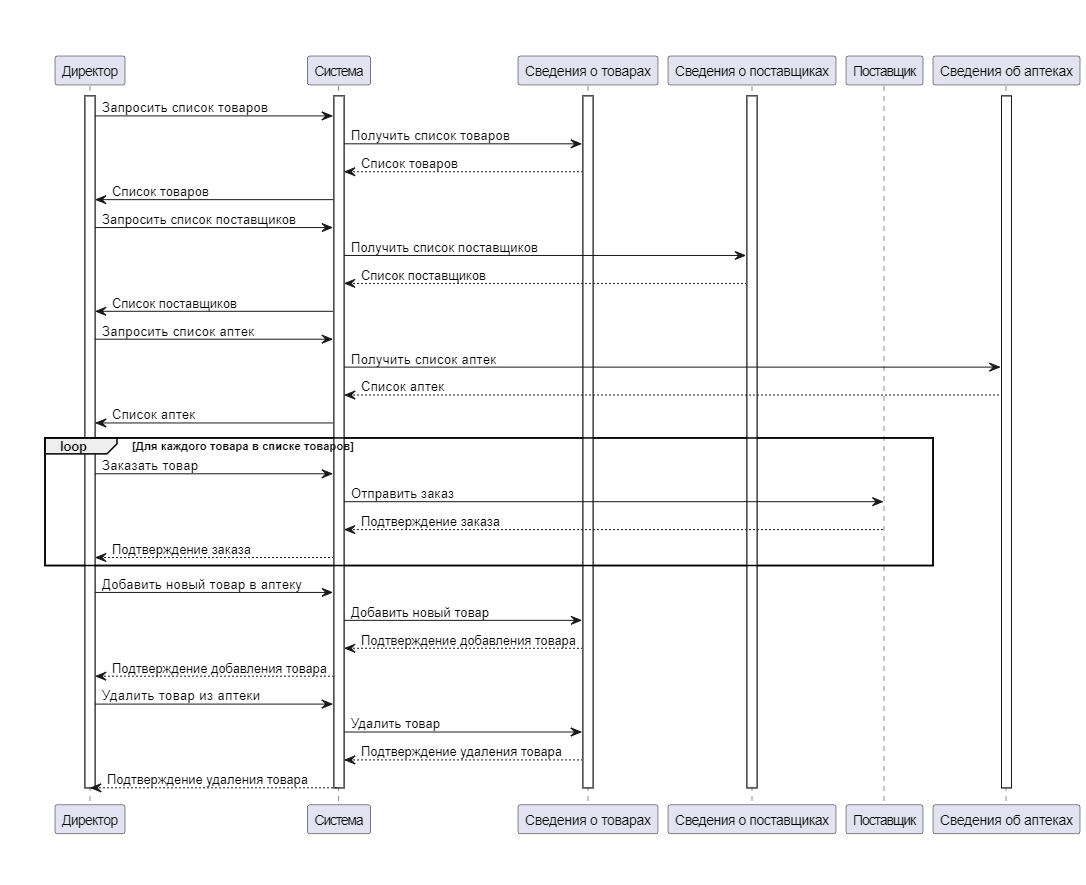


Рисунок 2.3 – Sequences диаграмма

Опираясь на вышеперечисленные диаграммы, был создан Restful api представленный на риунке 2.4, а также само решение будет представлено в виде веб-приложения, которое будет состоять из пяти страниц.

**Restful api:**

{

"Аптека": {

"GET": {

"/pharmacy": "Получить список всех аптек",

"/pharmacy/{id}": "Получить информацию об аптеке по ID"

},

"POST": {

"/pharmacy": "Добавить новую аптеку в базу данных"

},

"PUT": {

"/pharmacy/{id}": "Обновить информацию об аптеке по ID"

}

},

"Товар": {

"GET": {

"/product": "Поиск товаров по названию",

"/product/{id}": "Получить информацию о товаре по ID"

},

"POST": {

"/product": "Добавить новый товар в базу данных"

},

"PUT": {

"/product/{id}": "Обновить информацию о товаре по ID"

},

"DELETE": {

"/product/{id}": "Удалить товар по ID"

}

},

"Поставщик": {

"GET": {

"/supplier": "Получить список всех поставщиков",

"/supplier/{id}": "Получить информацию о поставщике по ID"

},

"POST": {

"/supplier": "Добавить нового поставщика в базу данных"

},

"PUT": {

"/supplier/{id}": "Обновить информацию о поставщике по ID"

}

}

}

**Страницы приложения**

Первая страница — страница с возможностью выбора дальнейших действий (Перейти на страницу с аптеками, перейти на страницу со всеми товарами, перейти на страницу со всеми поставщиками и их товарами). Макет страницы представлен на рисунке 2.4

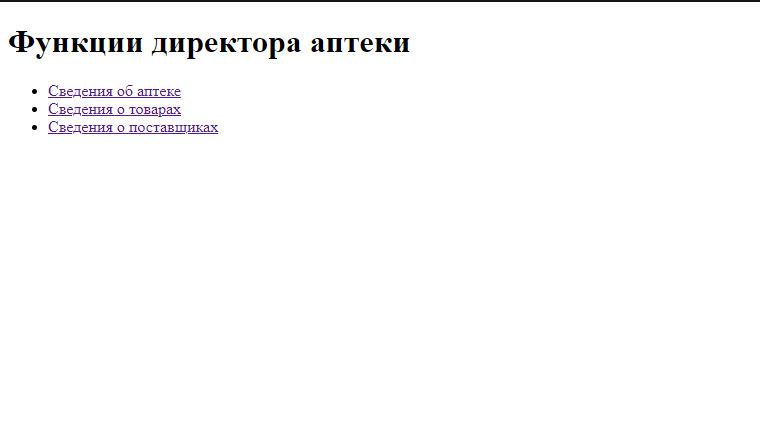


Рисунок 2.4 — Макет главной страницы

Вторая страница — страница со всеми аптеками. Пользователь выбирает из списка доступных аптек и после на странице выводится вся информация об этой аптеке. Также пользователь может добавить новый товар в выбранную аптеку. Выводимая информация представлена на рисунке 2.5.

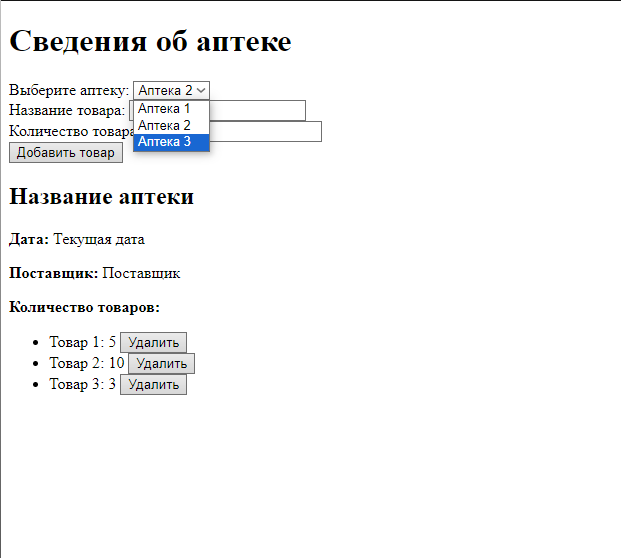


Рисунок 2.5 – Макет страницы “Аптеки”

Третья страница — страница со всеми товарами. Пользователь выбирает интересующий его товар из списка. После выводится вся информация об этом товаре. Также выбранный товар можно заказать, нажав на соответствующую кнопку. Выводимая информация представлена на рисунке 2.6.

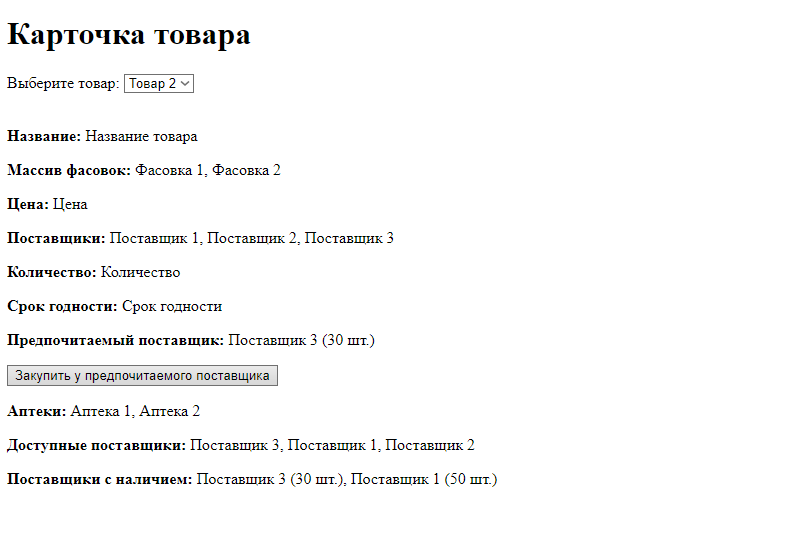


Рисунок 2.6 – Макет страницы “Товары”

Четвертая страница — страница оформления заказа у определенного поставщика. Макет страницы предоставлен на рисунке 2.7

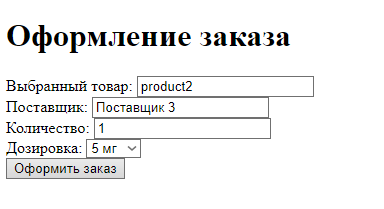


Рисунок 2.7 – Макет страницы “Заказ”

Пятая страница – страница с информацией о всех поставщиках. Пользователь выбирает поставщика, после чего выводится вся информация об этом поставщике. Макет страницы предоставлен на рисунке 2.8

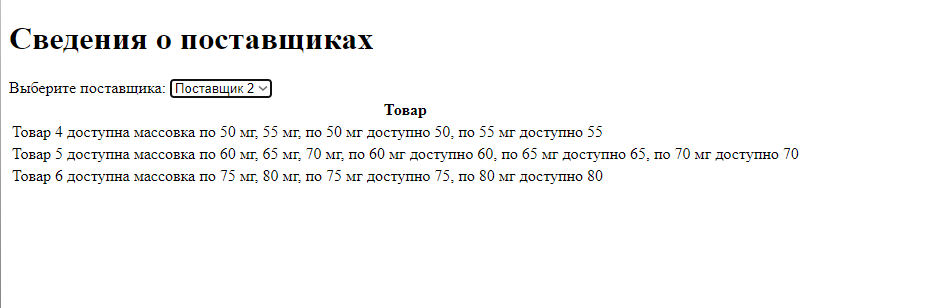


Рисунок 2.8 – Макет страницы “Поставщики”