**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**Факультет: Информационные технологии и управление**

**Кафедра: Компьютерная инженерия**

**Предмет: «Веб-системы и технологии»**

Курсовая работа

**Тема** «Разработка онлайн информационной системы для продуктовых магазинов»

**Группа:** 680.22

**Курc:** 3

**Специальность: 050616** Информационные технологии

**Студент**: **Алиев Эмиль**

**Руководитель:** **асс. Халилов М. Э.**

**Зав.кафедрой: доц.Рагимова Н.А.**

**Баку – 2023**

**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**ЗАДАНИЕ ПО КУРСОВОЙ РАБОТЕ**

**ФАКУЛЬТЕТ**: Информационные технологии и управление

**КАФЕДРА**: Компьютерная инженерия

**Группа** 680.22 **курс** III

**Специальность** **050616** Информационные технологии

**Студент** Алиев Эмиль

**Зачетная книжка**

**Руководитель курсовой работы**  **асс. Халилов М. Э.**

**Срок выдачи**

**Дата сдачи**

**Тема курсовой работы** «Разработка онлайн информационно-справочной

системы для аптек»

**Отзыв руководителя курсовой**:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись студента**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись рук.курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись зав.кафедрой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дата защиты курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Оценка**\_\_\_\_\_\_\_\_\_\_

**Пред.комиссии**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**Члены комиссии**: 1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**СОДЕРЖАНИЕ**

**ВВЕДЕНИЕ** .................................................................................................................... 5

**1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**1.1 Web-программирование** .................................................................................... 6  
**1.2 HTML** .................................................................................................... 7  
**1.3 Dynamic HTML** ................................................................................................... 8

**1.3.1 Каскадные таблицы стилей** ......................................................................... 8  
**1.3.2 JavaScript** .......................................................................................................... 10

**2. ПРАКТИЧЕСКАЯ ЧАСТЬ**

**2.1 Описание предметной области** ....................................................................... 12  
**2.2 Создание окружения для проекта** .................................................................. 13  
**2.3 Верстка HTML-страницы** .................................................................................. 14

**2.3.1 Главная страница**  
**2.3.2 Страница каталога**  
**2.3.3 Страница деталей продукта**

**2.4 Использование стилей CSS** ............................................................................... 29

**2.4.1 Главная страница**  
**2.4.2 Страница каталога**  
**2.4.3 Страница деталей продукта**

**2.5 Использование JavaScript** ................................................................................. 52

**2.5.1 JavaScript-функции на главной странице**  
**2.5.2 JavaScript-функции на странице каталогов**

**3. РЕЗУЛЬТАТ ................................................................................................................. 56**

**4. ЛИТЕРАТУРА ............................................................................................................... 57**

**5. ПРИЛОЖЕНИЯ ............................................................................................................. 57**

**ВВЕДЕНИЕ**

Веб-системы и технологии представляют собой разнообразные инструменты, языки программирования и платформы, предназначенные для создания, размещения и поддержки веб-приложений. Эти технологии развивались на протяжении времени, соответствуя растущим запросам на онлайн-сервисы, что способствовало появлению множества языков, библиотек, фреймворков и систем управления базами данных. С момента появления Web-технологий пользоваться компьютерами начали совершенно новые группы пользователей. Число социальных слоев населения, подключающихся к интернету для поиска информации в сети, постоянно растёт за счёт тех, кто не является ИТ-специалистом. Web-технологии кардинально изменили представления об обработке информации и взаимодействии с вычислительной техникой.  
Оказалось, что традиционные параметры оценки развития вычислительной техники — производительность, скорость передачи данных и объём памяти — практически не учитывали главный системный фактор, называемый пользовательским интерфейсом. Старая модель взаимодействия между человеком и компьютером мешала развитию технологий и сводила на нет их преимущества. Лишь когда интерфейс сделали понятным для обычного пользователя, произошёл резкий рост интереса к компьютерным возможностям.  
Web-технологии охватывают всё множество средств, необходимых для функционирования WWW — Всемирной паутины. Поскольку каждый сеанс представляет собой взаимодействие клиента и сервера, Web-технологии делятся на два типа:

1. Серверные технологии (server-side).
2. Клиентские технологии (client-side).  
   Клиентская сторона охватывает технологии создания веб-страниц (HTML, JavaScript, DHTML и др.), а серверная — доступ к базам данных через интернет (CGI, PHP). Серверные программы обрабатывают запросы от клиентов и передают им необходимые ресурсы. Клиент при необходимости формирует запрос и отправляет его на сервер. Серверная программа отвечает запрошенной информацией или уведомлением об ошибке, если данные недоступны. Такая схема организации систем называется архитектурой «клиент-сервер» или двухуровневой архитектурой. Именно на ней базируются практически все интернет-сервисы, включая [WWW](http://WWW).  
   Данная курсовая работа направлена на создание информационной системы для продовольственных магазинов.  
   Создание онлайн-системы включает следующие этапы:
3. Сбор требований: На начальной стадии проводится сбор пожеланий клиента, включая ассортимент, целевую аудиторию и желаемые функции сайта.
4. Дизайн: Далее следует проектирование сайта — разработка макетов, подбор шрифтов, цветов и структуры пользовательского интерфейса.
5. Разработка: На этом этапе сайт создаётся с помощью HTML, CSS, JavaScript и других технологий.
6. Тестирование: После завершения разработки сайт проверяется на корректную работу в различных браузерах и на разных устройствах.  
   WEB-сайт продуктового магазина должен быть ориентирован на пользователя, предоставляя сервисы поиска, просмотра и оформления заказов.  
   Таким образом, целью курсовой работы является разработка интернет-магазина, отличающегося удобным, безопасным и интуитивно понятным интерфейсом для пользователя.

Хочешь, чтобы я ещё и стилистически улучшил текст или вставил абзацы для оформления документа?

1. **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**
   1. **Web-программирование**

Web-программирование (или Web-разработка) — это активно развивающееся направление программирования, основная цель которого заключается в создании динамических интернет-приложений. Языки Web-программирования делятся на две ключевые категории: клиентские и серверные. Клиентские языки выполняются на устройстве пользователя, как правило, в браузере. Поэтому успешность их исполнения напрямую зависит от возможностей и настроек браузера, который может не поддерживать определённые языки или версии. Некоторые пользователи даже могут полностью отключить обработку скриптов. Однако в современных браузерах такие проблемы возникают крайне редко, ведь кардинальные обновления языков происходят нечасто, а ключевые технологии уже давно стали стандартом.

Код клиентских скриптов полностью открыт — любой пользователь может просмотреть его, открыв исходный код страницы в своём браузере. Главное достоинство клиентских языков — возможность обработки данных без необходимости обращения к серверу. Это, например, позволяет сразу проверять корректность заполнения форм и мгновенно выдавать ошибки, не отправляя данные на сервер. Однако такие языки не имеют доступа к серверу и не могут сохранять или обрабатывать данные на его стороне. Самым популярным клиентским языком является JavaScript, разработанный компаниями Netscape и Sun Microsystems. Среди других клиентских технологий можно отметить VBScript, а также AJAX, Adobe Flash, Microsoft Silverlight, которые в своё время получили широкое распространение.

Серверные языки программирования предоставляют значительно больше возможностей. Когда пользователь обращается к определённой странице — переходит по ссылке или вводит адрес в браузере — сначала запрос обрабатывается на сервере. Все программы, связанные с этой страницей, выполняются на серверной стороне, и только после этого результат передаётся пользователю в виде готового HTML-файла. Эти файлы могут иметь различные расширения: HTML, PHP, ASP, Perl, SSI, XML, DHTML, XHTML и другие.

* 1. **HTML**

**HTML (HyperText Markup Language)** — это язык разметки, используемый для создания веб-страниц и их отображения в браузере. Он является основой любого сайта и отвечает за структуру, содержание и оформление страницы. Почти все страницы в Интернете написаны с использованием HTML, поэтому знание этого языка обязательно для любого веб-разработчика.

HTML не является языком программирования в полном смысле, так как он не содержит логики, условий или циклов. Это именно *язык разметки*, предназначенный для описания структуры документа с помощью специальных обозначений — тегов. Каждый HTML-документ состоит из набора элементов, каждый из которых начинается и, как правило, заканчивается тегом. Теги выделяются угловыми скобками < >. Например, тег <p> обозначает абзац, а <h1> — заголовок первого уровня. Большинство тегов парные: <tagname>контент</tagname>. Однако есть и непарные (самозакрывающиеся) теги, например, <br> (перенос строки) или <img> (вставка изображения).

**Основная структура HTML-документа** включает следующие элементы:

**<html>**

**<head>**

**<title>Заголовок страницы</title>**

**</head>**

**<body>**

**<!-- Контент страницы -->**

**</body>**

**</html>**

* <html> обозначает начало и конец HTML-документа.
* Внутри него находятся теги <head> и <body>.
* <head> содержит метаинформацию о документе, заголовок (<title>), подключение стилей, скриптов и т.д.
* <body> содержит видимую пользователю часть страницы: текст, изображения, таблицы, формы и другие элементы.

**HTML позволяет:**

* **Структурировать текст**: разбивать его на абзацы (<p>), заголовки (<h1>–<h6>), списки (<ul>, <ol>, <li>).
* **Форматировать содержимое**: выделять курсивом (<i>), жирным (<b>), подчёркиванием (<u>), использовать надстрочные (<sup>) и подстрочные (<sub>) символы, изменять размер шрифта (<big>, <small>), задавать цвет и стиль шрифта с помощью тега <font>.
* **Создавать гиперссылки** с помощью тега <a>. Атрибут href указывает адрес ссылки, а target — поведение при открытии (в этом же окне, в новом и т.д.).
* **Вставлять изображения** с помощью тега <img>, у которого есть атрибуты src (путь к изображению), alt (альтернативный текст), width, height, title и другие.
* **Создавать таблицы**: тег <table> задаёт таблицу, <tr> — строку, <td> — ячейку, <th> — заголовок столбца. Таблицы можно стилизовать через атрибуты: рамки (border), ширина, высота, отступы между ячейками (cellspacing) и т.д.
* **Создавать формы ввода данных**, которые взаимодействуют с сервером. Используются теги <form>, <input>, <textarea>, <select> и <option>. Это позволяет реализовать поля для ввода текста, кнопки, чекбоксы, выпадающие списки и т.д.
* **Работать с мультимедиа**: через теги <embed>, <object> и <video> можно вставлять видео, аудио и интерактивные элементы. Также HTML позволяет встраивать скрипты с помощью тега <script>.

**HTML также поддерживает специальные символы**, которые нельзя вставить напрямую. Например, знак «<» записывается как &lt;, «>» — &gt;, амперсанд — &amp;. Это важно, поскольку эти символы используются в синтаксисе языка.

Язык HTML не чувствителен к регистру — теги можно писать как строчными, так и заглавными буквами. Однако в современной практике принято писать всё в нижнем регистре для соблюдения единообразия и лучшей читаемости кода.

Благодаря HTML разработчик может задать удобную и логичную структуру сайта, правильно оформить информацию и сделать страницу доступной для отображения в любом браузере. Хотя сам HTML отвечает только за структуру и содержание, в связке с CSS и JavaScript он становится мощным инструментом для создания полноценных, динамичных и привлекательных сайтов.

* 1. **Dynamic HTML**

Dynamic HTML (или DHTML) — это технология, объединяющая язык разметки HTML, каскадные таблицы стилей (CSS), скриптовые языки (чаще всего JavaScript) и объектную модель документа (DOM). Эта связка позволяет создавать интерактивные и динамически изменяющиеся веб-страницы прямо в браузере пользователя. Хотя в качестве скриптового языка может использоваться и Visual Basic, наибольшее распространение получил именно JavaScript благодаря своей универсальности и поддержке во всех современных браузерах.

С помощью DHTML можно реализовывать гибкие пользовательские интерфейсы, реагирующие на действия пользователя в реальном времени — например, перемещение элементов мышью (Drag'n'Drop), динамическое обновление содержимого страницы без её перезагрузки, визуальные эффекты и анимации. Такие возможности делают DHTML популярным при создании веб-приложений, игр и других интерактивных сервисов.

Важно, что использование DHTML не требует подключения к серверу или базам данных — всё взаимодействие происходит на стороне клиента, то есть внутри браузера. Это делает разработку быстрее и упрощает реализацию многих функций.

* + 1. **Каскадные таблицы стилей**

**CSS (Cascading Style Sheets)** — это каскадные таблицы стилей, предназначенные для описания внешнего вида HTML-документов. Вместе с HTML и JavaScript, CSS является важнейшим компонентом технологии **DHTML (Dynamic HTML)**, позволяющей создавать динамичные и визуально привлекательные веб-страницы. Если HTML отвечает за структуру контента, то CSS — за его оформление: цвета, шрифты, отступы, положение на экране, анимации и многие другие параметры.

Название «каскадные» происходит от механизма **каскадирования** — способа, по которому браузер определяет, какие именно стили применить к элементу, если на него влияют несколько правил одновременно. Это даёт гибкость и позволяет разработчику точно управлять внешним видом элементов при разных условиях.

CSS используется для определения:

* Цветов текста, фона и границ;
* Типа, размера и начертания шрифтов;
* Отступов (margin), полей (padding), рамок (border);
* Позиционирования элементов на странице (static, relative, absolute, fixed, sticky);
* Отображения элементов в виде блоков, строк или сеток (с помощью display: block, inline, flex, grid);
* Анимаций, переходов, эффектов наведения и состояний взаимодействия (например, :hover, :focus);
* И даже адаптивной верстки с помощью **медиазапросов** (media queries), позволяющей подстраивать стиль страницы под разные экраны и устройства.

Стили можно подключать к HTML-документу **тремя способами**:

1. **Встроенные (inline)** — применяются прямо к элементу с помощью атрибута style.  
   Пример:

**<p style="color: red; font-size: 18px;">Пример текста</p>**

1. **Внутренние (internal)** — задаются внутри тега <style> в секции <head> страницы.  
   Пример:

**<style>**

**body {**

**background-color: lightgray;**

**font-family: Arial;**

**}**

**</style>**

1. **Внешние (external)** — подключаются отдельным файлом с расширением .css через тег <link>. Это наиболее предпочтительный способ в профессиональной разработке, так как обеспечивает повторное использование стилей и отделение логики от оформления.  
   Пример:

**<link rel="stylesheet" type="text/css" href="styles.css">**

CSS тесно связан с объектной моделью документа (DOM), так как позволяет применять стили к элементам, находящимся в определённых состояниях или обладающим определёнными классами, идентификаторами или вложенностью. Например:

**#menu ul li a:hover {**

**background-color: #ccc;**

**color: #000;**

**}**

Этот селектор изменит стиль ссылки внутри меню при наведении курсора.

Современные возможности CSS (начиная с CSS3) включают:

* **Flexbox и Grid** для продвинутой компоновки элементов;
* **Псевдоклассы и псевдоэлементы** (:first-child, :before, :after) для точного позиционирования и стилизации;
* **Анимации и переходы** (@keyframes, transition) для создания плавных эффектов;
* **Переменные CSS** (--main-color: #3498db) для повторного использования значений;
* **Фильтры** и **тени** (box-shadow, filter: blur()), создающие визуальные эффекты без использования изображений;
* **Поддержка адаптивного дизайна** через медиазапросы, позволяющие изменять стили в зависимости от ширины экрана:

**@media (max-width: 768px) {**

**.container {**

**flex-direction: column;**

**}**

**}**

CSS играет важнейшую роль в реализации **динамического HTML**. Совместно с JavaScript, стили могут изменяться в процессе взаимодействия пользователя со страницей. Например, с помощью JavaScript можно изменить стиль любого элемента на лету, обратиться к нему по идентификатору или классу и задать новый цвет, размер или эффект:

**document.getElementById("myDiv").style.backgroundColor = "yellow";**

Это позволяет реализовывать всплывающие окна, раскрывающиеся меню, анимации, модальные окна и другие динамичные элементы интерфейса.

CSS не только делает страницу красивой, но и обеспечивает её **удобство, доступность и адаптивность**. Без стилей HTML-документ представлял бы собой просто череду текста и изображений, а благодаря CSS сайты превращаются в полноценные, профессиональные и интуитивно понятные веб-приложения.

* + 1. **JavaScript**

**JavaScript** — это объектно-ориентированный, интерпретируемый скриптовый язык программирования, разработанный для встраивания в веб-страницы. Он получил широкое распространение благодаря своей простоте, гибкости и возможности тесной интеграции с HTML и CSS. Синтаксис JavaScript во многом напоминает язык C, однако имеет ряд принципиальных отличий, таких как динамическая типизация, возможность создания анонимных функций, а также автоматическое управление памятью.

Одной из ключевых особенностей JavaScript является то, что структура и типы объектов (включая переменные и функции) могут быть определены во время выполнения программы. Это делает язык чрезвычайно гибким, позволяя адаптировать поведение скриптов в зависимости от текущего состояния страницы. Кроме того, JavaScript автоматически приводит типы данных при необходимости, что упрощает работу с переменными.

Функции в JavaScript могут быть именованными или анонимными, что особенно удобно при работе с обратными вызовами (callback-функциями). Все идентификаторы в языке чувствительны к регистру: переменные MyVar и myvar считаются разными. Допустимыми символами для имён переменных являются латинские буквы, символ подчёркивания (\_), долларовый знак ($) и цифры, при этом имя не может начинаться с цифры.

С точки зрения безопасности, JavaScript имеет определённые ограничения: он не поддерживает прямую работу с файловой системой, потоками ввода-вывода и сетевыми соединениями. Эти ограничения введены намеренно, чтобы предотвратить возможность вредоносных действий на стороне клиента. Например, скрипт, встроенный в страницу, не может получить доступ к содержимому других открытых вкладок или окон браузера, а также к локальным файлам пользователя. Всё это делает JavaScript безопасным средством для реализации интерактивности в веб-пространстве.

Для вставки скриптов JavaScript непосредственно в HTML-документ используются теги <script>. Пример простейшего кода, выводящего сообщение:

**<script type="text/javascript">**

**alert('Hello, World!');**

**</script>**

Также возможна внешняя загрузка скрипта из отдельного файла с помощью атрибута src:

**<script type="text/javascript" src="script.js"></script>**

Такой подход считается предпочтительным при разработке сложных веб-приложений, поскольку облегчает повторное использование кода и делает структуру страницы более чистой.

С помощью JavaScript можно реализовать огромное количество функций: автоматическое заполнение форм, изменение оформления страницы, скрытие или отображение блоков, обработка нажатий клавиш и событий мыши, валидация данных, создание анимаций, манипулирование DOM-элементами, изменение URL без перезагрузки страницы и многое другое. Всё это делает JavaScript незаменимым инструментом для создания современных динамичных пользовательских интерфейсов.

Инструментарий для работы с JavaScript также очень развит. Хотя язык не требует специализированной среды разработки, многие IDE и текстовые редакторы поддерживают автодополнение, подсветку синтаксиса и отладку JavaScript. К примеру, популярная среда **Eclipse** позволяет подключить модуль поддержки JavaScript, расширяя функционал среды. Кроме того, практически каждый современный браузер (Google Chrome, Mozilla Firefox, Microsoft Edge и др.) содержит встроенные средства разработчика, включающие консоль, инспектор DOM, инструменты профилирования и пошаговой отладки скриптов.

На сегодняшний день JavaScript продолжает активно развиваться. Современные стандарты ECMAScript (особенно ES6 и выше) ввели множество новых возможностей: классы, модули, стрелочные функции, промисы, async/await и другие синтаксические улучшения. Всё это делает язык ещё более удобным и мощным для создания не только клиентских, но и серверных приложений (например, с использованием платформы Node.js).

Таким образом, JavaScript является неотъемлемой частью современной веб-разработки, а в составе DHTML — основным инструментом, отвечающим за интерактивность и поведение элементов на странице.

1. **ПРАКТИЧЕСКАЯ ЧАСТЬ**
2. **Описание предметной области**

**Предметная область данной курсовой работы — онлайн информационно-справочная система, предназначенная для аптек.** Данная система создаётся с целью предоставления пользователям удобного доступа к информации о лекарственных препаратах, услугах аптек, наличии товаров и других связанных с фармацевтической деятельностью данных.

Основой любой веб-системы является её архитектура. Архитектура сайта — это тщательно организованная система представления информации и способов её навигации, разработанная для того, чтобы помочь пользователям эффективно и быстро находить необходимую им информацию. Чем более продуманной является структура сайта, тем проще пользователю ориентироваться в нём, что напрямую влияет на общее впечатление от работы с системой.

Грамотно реализованная архитектура онлайн-ресурса обеспечивает сокращение времени, требуемого на поиск интересующих сведений, и способствует росту популярности системы среди конечных пользователей за счёт удобства и понятного интерфейса. При создании структуры аптечной справочной системы важно учитывать, каким образом представление информации может повлиять на продвижение фармацевтических товаров и услуг в интернете.

На этапе проектирования новой структуры сайта, а также при её последующей оптимизации, необходимо добиваться того, чтобы представление данных способствовало повышению посещаемости ресурса и акцентировало внимание потенциальных клиентов на ключевых разделах. Это особенно актуально для аптечного сегмента, где важно выделять популярные и востребованные препараты, услуги доставки, акции и консультации специалистов.

Неотъемлемым элементом онлайн-системы является **интерфейс** — он служит основным каналом взаимодействия между системой и конечным пользователем. От того, насколько логично и понятно расположены элементы на странице, зависит успешность всего проекта. Интерфейс должен обеспечивать доступ к информации о лекарствах, их показаниях, дозировках, наличии и стоимости. Клиент должен иметь возможность легко найти интересующий его препарат, сравнить аналоги, узнать о возможных противопоказаниях и, при необходимости, оформить заказ или запрос консультации.

Для реализации веб-интерфейса и взаимодействия с пользователем в данной работе применяются современные веб-технологии. В частности:

* **HTML (HyperText Markup Language)** — базовый язык разметки, который используется для создания структуры и содержимого веб-страниц. Он обеспечивает размещение на странице всех текстовых и визуальных элементов: заголовков, таблиц, списков, изображений и других компонентов. HTML обладает простой и интуитивно понятной синтаксической основой, что делает его незаменимым инструментом при разработке информационных систем в интернете.
* **CSS (Cascading Style Sheets)** — каскадные таблицы стилей, применяемые для визуального оформления страниц. С помощью CSS можно задать цветовую схему, шрифты, размеры и отступы элементов, а также создать адаптивный дизайн, подходящий для различных устройств (смартфоны, планшеты, ПК). Использование CSS позволяет отделить визуальное оформление от логической структуры документа, обеспечивая удобство при разработке и сопровождении системы.
* **JavaScript** — язык программирования, предназначенный для добавления интерактивности на веб-страницы. В контексте аптечной справочной системы JavaScript используется для реализации динамических функций: поиск препаратов в режиме реального времени, фильтрация и сортировка товаров, отображение всплывающих подсказок, переключение вкладок и другие интерактивные элементы. Также он позволяет обрабатывать действия пользователя без необходимости перезагрузки страницы, что положительно влияет на пользовательский опыт и скорость взаимодействия с системой.

В совокупности, применение данных технологий позволяет создать полнофункциональную, современную и удобную в использовании справочную систему для аптек. Такой подход обеспечивает быстрый доступ к информации, улучшает восприятие ресурса пользователями и способствует более эффективной работе аптечной сети в условиях цифровизации фармацевтической отрасли.

1. **Создание окружения для проекта**

Для разработки сайта будет использоваться редактор кода **Visual Studio Code**.

На первом этапе создаём папку проекта с именем **"kursovaya"**, в которой будут размещаться все файлы. Сайт будет состоять из нескольких страниц, каждая из которых будет иметь свои файлы разметки, стилей и скриптов.

Создаём основные HTML-страницы:

* **index.html** — главная страница сайта;
* **catalog.html** — страница с каталогом товаров;
* **product-detail.html** — страница с подробной информацией о товаре.

Для оформления каждой страницы создаются отдельные CSS-файлы:

* **index.css** — стили для главной страницы;
* **catalog.css** — стили для каталога;
* **product-detail.css** — стили для страницы описания товара.

Чтобы добавить динамическую функциональность, создаются JavaScript-файлы:

* **index.js** — скрипты для главной страницы;
* **catalog.js** — скрипты для страницы каталога.

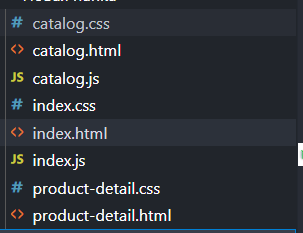
Таким образом, окружение для проекта полностью подготовлено, и можно приступать к разработке интерфейса и функциональности сайта. (Рис. 1)

Рис. 2. Окружение сайта

1. **Верстка HTML страницы**
   * 1. **Главная страница**

Создание сайта всегда начинается с формирования **структуры HTML-документа**. Это основа, которая задаёт порядок элементов и их связи между собой.

Для сайта фармакологии «ФармИнфо» мы начинаем с создания основного файла index.html. Внутри него мы сначала пишем базовый каркас документа.

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>ФармИнфо - Главная</title>

    <link rel="stylesheet" href="index.css">

</head>

<body>

<!-- Контент страницы будет здесь -->

<script src="index.js"></script>

</body>

</html>

Строка <!DOCTYPE html> сообщает браузеру, что перед ним документ стандарта HTML5, а не какой-либо старый формат. Это важно для правильного отображения элементов.  
Тег <html> открывает корень всего HTML-документа, и с помощью атрибута lang="ru" мы уточняем, что сайт русскоязычный.  
Внутри блока <head> содержится информация, которую браузер использует для работы страницы: кодировка текста (UTF-8), установка адаптивности на мобильных устройствах (viewport), заголовок вкладки в браузере (<title>), а также подключение внешнего файла стилей (index.css), который будет отвечать за внешний вид.  
Основное содержимое сайта находится в <body>. В самом конце перед закрытием </body> подключается файл index.js, в котором будут находиться скрипты для интерактивности сайта.

Следующим шагом мы создаём **шапку сайта**, которая содержит логотип и меню навигации. Шапка размещается внутри тега <header>.  
Внутри <header> используется контейнер для выравнивания содержимого по центру. Далее размещается блок <div class="logo">, где пишется название сайта — "ФармИнфо".  
После логотипа создаётся навигационное меню через тег <nav>, внутри которого список <ul>. Каждый пункт меню оформлен тегами <li> и <a>, ведущими на разные страницы: Главная, Каталог препаратов, Аптеки, Справочник и Контакты.

<header>

        <div class="container header-content">

            <div class="logo">ФармИнфо</div>

            <nav>

                <ul>

                    <li><a href="index.html">Главная</a></li>

                    <li><a href="catalog.html">Каталог</a></li>

                    <li><a href="#">Аптеки</a></li>

                </ul>

            </nav>

        </div>

    </header>

Это меню позволит пользователю удобно перемещаться по основным разделам сайта.

После шапки мы создаём **поисковую строку**. Поиск очень важен для фармакологического сайта, ведь пользователи часто хотят быстро найти конкретное лекарство.

Поисковая строка оформляется через форму <form>, в которой есть поле ввода (<input type="text">) и кнопка отправки (<button type="submit">).

div class="search-bar">

        <div class="container">

            <form class="search-form">

                <input type="text" placeholder="Поиск лекарств, действующих веществ, аптек...">

                <button type="submit">Найти</button>

            </form>

        </div>

    </div>

Когда пользователь вводит запрос и нажимает "Найти", форма передаёт введённую информацию на сервер или в JavaScript для обработки на месте.

Далее идёт **главный баннер** — это большая яркая секция, которая сразу привлекает внимание. Здесь мы размещаем главный заголовок, короткое описание сайта и кнопку, ведущую в каталог препаратов.

<section class="hero">

        <div class="container">

            <h1>Информационно-справочная система аптек</h1>

            <p>Найдите необходимые лекарства, сравните цены и получите актуальную информацию о наличии в аптеках вашего города</p>

            <a href="catalog.html" class="btn">Перейти в каталог</a>

        </div>

    </section>

Эта секция обычно занимает значительную часть экрана на первом экране сайта и задаёт пользователю направление его действий.

Следующий логический блок — **«Наши преимущества»**. Он объясняет посетителю, чем полезен сайт. Например, мы показываем, что у нас есть огромный каталог препаратов, удобный поиск по аптекам и свежая информация.

Каждое преимущество оформляется через карточку:

<section class="features">

        <div class="container">

            <div class="features-grid">

                <div class="feature-card">

                    <div class="feature-icon">📋</div>

                    <div class="feature-content">

                        <h3>Полный каталог препаратов</h3>

                        <p>Более 10 000 наименований лекарственных средств с подробным описанием, показаниями к применению и противопоказаниями.</p>

                    </div>

                </div>

            </div>

        </div>

    </section>

Каждая карточка включает иконку, заголовок и короткое описание. Иконка здесь сделана через эмодзи, но в реальном проекте можно заменить её на SVG-иконку или картинку.

Следом мы создаём секцию **«Популярные препараты»**. Здесь мы выводим самые востребованные лекарства в виде карточек с изображением, названием, ценой и кнопкой для просмотра подробной информации.

<section class="popular-section">

        <div class="container">

            <div class="section-header">

                <h2>Популярные препараты</h2>

                <p>Наиболее востребованные лекарственные средства с информацией о наличии в аптеках</p>

            </div>

            <div class="medicine-grid">

                <div class="medicine-card">

                    <div class="medicine-image">

                        <img src="https://borimed.com/sites/default/files/product/%D0%9F%D0%B0%D1%80%D0%B0%D1%86%D0%B5%D1%82%D0%B0%D0%BC%D0%BE%D0%BB-200-v2.jpg" alt="Парацетамол" height=250px width=300px>

                    </div>

                    <div class="medicine-info">

                        <div class="medicine-name">Парацетамол</div>

                        <div class="medicine-price">5 azn</div>

                        <div class="medicine-availability">В наличии в 15 аптеках</div>

                        <a href="product.html" style="display: inline-block; margin-top: 0.5rem; color: var(--accent-color); text-decoration: none;">Подробнее</a>

                    </div>

                </div>

            <div class="view-all">

                <a href="catalog.html" class="btn-secondary">Посмотреть все препараты</a>

            </div>

        </div>

    </section>

Эта секция помогает пользователю быстро найти популярные товары и перейти к полному каталогу.

Далее мы создаём **блок поиска ближайших аптек**. Эта секция предлагает пользователю ввести свой адрес и найти аптеку с нужным препаратом рядом с собой.

 <section class="pharmacy-finder">

        <div class="container">

            <div class="finder-container">

                <div class="finder-image">

                    <iframe

                        src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d15486.783373727614!2d50.049444349961675!3d40.46035108650008!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x4030606eefea1ed1%3A0x41cc40173c5f5edc!2z0JzQtdC20LTRg9C90LDRgNC-0LTQvdGL0Lkg0LDRjdGA0L7Qv9C-0YDRgiDQuNC80LXQvdC4INCT0LXQudC00LDRgNCwINCQ0LvQuNC10LLQsA!5e1!3m2!1sru!2saz!4v1745689723128!5m2!1sru!2saz"

                        width="100%"

                        height="350"

                        style="border:0; border-radius: 8px;"

                        allowfullscreen=""

                        loading="lazy"

                        referrerpolicy="no-referrer-when-downgrade">

                    </iframe>

                </div>

                <div class="finder-content">

                    <h2>Найдите ближайшие аптеки</h2>

                    <p>Введите свой адрес или разрешите доступ к геолокации, чтобы найти ближайшие аптеки с нужными вам препаратами.</p>

                    <p>Сравнивайте цены, смотрите режим работы и наличие лекарств в режиме реального времени.</p>

                    <form class="location-form">

                        <input type="text" placeholder="Введите ваш адрес">

                        <button type="submit" class="btn-secondary">Найти</button>

                    </form>

                </div>

            </div>

        </div>

    </section>

Таким образом, пользователь может прямо с сайта определить, куда ему удобнее всего обратиться.

Последний элемент на странице — это **подвал сайта** (footer). Здесь размещаются полезные ссылки на документы, информацию о компании и контакты.

<footer>

        <div class="container footer-content">

            <div class="footer-column">

                <h3>Поддержка</h3>

                <ul class="footer-links">

                    <li><a href="#">Часто задаваемые вопросы</a></li>

                    <li><a href="#">Контакты службы поддержки</a></li>

                    <li><a href="#">Сообщить о проблеме</a></li>

                    <li><a href="#">Предложить улучшение</a></li>

                </ul>

            </div>

        </div>

        <div class="container copyright">

            &copy; 2025 ФармИнфо. Все права защищены.

        </div>

    </footer>

Подвал нужен для завершения страницы и размещения важной информации, которую пользователь может искать на любом сайте.

* + 1. **Страница каталога**

Когда мы создаём страницу каталога, наша основная цель — **показать пользователю большой список препаратов**, удобно отсортированных и отфильтрованных по разным параметрам: цена, категория, наличие, производитель, и т.д.

Чтобы это работало, HTML-страница должна быть правильно организована:

1. Шапка сайта — навигация.
2. Строка поиска — чтобы быстро находить нужные лекарства.
3. Хлебные крошки — для удобной навигации между разделами.
4. Основная часть — два блока:
   * Левая колонка: фильтры и категории.
   * Правая колонка: список препаратов.
5. Подвал сайта — информация о проекте, контакты.

## Шапка сайта (Header)

В самом верху страницы — **шапка**. Она делает две главные вещи:

* Показывает логотип сайта.
* Даёт пользователю ссылки для перемещения по сайту: главная страница, каталог, аптеки, контакты.

Код выглядит так:

<header>

        <div class="container">

            <div class="header-content">

                <div class="logo">ФармИнфо</div>

                <nav>

                    <ul>

                        <li><a href="index.html">Главная</a></li>

                        <li><a href="catalog.html">Каталог препаратов</a></li>

                    </ul>

                </nav>

            </div>

        </div>

    </header>

Шапка оборачивается в <header>, а внутри обязательно есть <nav>, где хранится меню ссылок (<ul> и <li>).

## Строка поиска

Чтобы пользователи могли быстро найти нужный препарат, мы добавляем **строку поиска** сразу под шапкой:

<div class="search-bar">

        <div class="container">

            <form class="search-form">

                <input type="text" placeholder="Поиск препаратов...">

                <button type="submit">Поиск</button>

            </form>

        </div>

    </div>

Форма <form> позволяет отправлять поисковый запрос. Ввод осуществляется через <input type="text">, а отправка через кнопку <button type="submit">.

## Хлебные крошки (Breadcrumbs)

Дальше идёт **заголовок страницы** и **хлебные крошки** — это маленький элемент навигации, который показывает путь, где пользователь сейчас находится:

<div class="page-title">

        <div class="container">

            <div class="breadcrumbs">

                <a href="index.html">Главная</a>

                <span>></span>

                <a href="catalog.html">Каталог препаратов</a>

            </div>

            <h1>Каталог лекарственных препаратов</h1>

        </div>

    </div>

Хлебные крошки делают сайт удобным: человек всегда видит, где он находится и может быстро вернуться назад.

## Основная часть каталога: Категории и Товары

Главный контент страницы делится на две части:

* Слева — **фильтры и категории**.
* Справа — **список препаратов**.

Структура выглядит так:

<div class="catalog-container">

            <div class="sidebar">

<!-- Категории и фильтры -->

</div>

            <div class="catalog-content">

<!-- Сортировка и карточки препаратов -->

</div>

</div>

### ***Левая колонка: Категории и Фильтры***

**Категории** позволяют пользователю быстро выбрать группу препаратов (например, "Антибиотики", "Витамины", "Анальгетики").

<h2>Категории</h2>

                <ul class="category-list">

                    <li><a href="#">Антибиотики</a></li>

                    <li><a href="#">Анальгетики</a></li>

                    <li><a href="#">Противовирусные</a></li>

                </ul>

**Фильтры** позволяют уточнить результаты поиска:

* По цене:

<div class="filter-section">

                    <h3>Цена</h3>

                    <div class="price-filter">

                        <input type="number" placeholder="от">

                        <input type="number" placeholder="до">

                    </div>

* По наличию в аптеках:

h3>Наличие</h3>

                    <div class="checkbox-filter">

                        <label>

                            <input type="checkbox" checked> В наличии

                        </label>

                        <label>

                            <input type="checkbox"> Под заказ

                        </label>

                    </div>

* По производителю:

<h3>Производитель</h3>

                    <div class="checkbox-filter">

                        <label>

                            <input type="checkbox"> Отечественные

                        </label>

                        <label>

                            <input type="checkbox"> Импортные

                        </label>

                    </div>

* По рецептурности:

<h3>Рецептурные</h3>

                    <div class="checkbox-filter">

                        <label>

                            <input type="checkbox"> По рецепту

                        </label>

                        <label>

                            <input type="checkbox"> Без рецепта

                        </label>

                    </div>

В конце — кнопка "Применить фильтры".

<button class="filter-button">Применить фильтры</button>

                </div>

            </div>

### ***Правая колонка: Сортировка и Карточки товаров***

Вначале — панель сортировки:

<div class="sorting">

                    <div class="sort-options">

                        <label>Сортировать по:</label>

                        <select>

                            <option>Популярности</option>

                            <option>Цене (низкая - высокая)</option>

                            <option>Цене (высокая - низкая)</option>

                            <option>Названию (А-Я)</option>

                        </select>

Пользователь может отсортировать препараты, например, по цене или по алфавиту.

Далее — **список препаратов**:

Каждый препарат — это карточка:

 <div class="product-card"

                        data-price="12"

                        data-availability="В наличии"

                        data-producer="Отечественный"

                        data-prescription="По рецепту"

                        data-category="Антибиотики">

                        <div class="product-image">

                            <img src="https://ft.by/wp-content/uploads/drugs/amoksicillin-ft\_image2.png" alt="Амоксициллин" height=200px width=300px>

                        </div>

                        <div class="product-info">

                            <div class="product-name">Амоксициллин</div>

                            <div class="product-description">Антибиотик широкого спектра действия, 500 мг, 20 таблеток</div>

                            <div class="product-price">12 azn</div>

                            <div class="product-availability in-stock">В наличии</div>

                            <div class="product-action">

                                <a href="product-detail.html?id=1" class="details-link">Подробнее</a>

                                <a href="#" class="pharmacy-link">Аптеки</a>

                            </div>

                        </div>

                    </div>

**Важно:** У каждой карточки есть специальные атрибуты data-\*, чтобы потом можно было с помощью JavaScript делать фильтрацию и сортировку прямо на странице!

## Подвал сайта (Footer)

Подвал делает сайт завершённым. Обычно в подвале размещают информацию о проекте, контакты, ссылки на политику конфиденциальности:

 <footer>

        <div class="container">

            <div class="footer-content">

                <div class="footer-column">

                    <h3>О нас</h3>

                    <ul>

                        <li><a href="#">О компании</a></li>

...

</ul>

                </div>

...

 </div>

            <div class="copyright">

                © 2025 ФармИнфо. Все права защищены.

            </div>

        </div>

    </footer>

Создание страницы подробного описания препарата — это важный этап разработки фармакологического сайта. Такая страница должна быть удобной, хорошо структурированной и давать пользователю максимум нужной информации.  
Теперь подробно разберём, как правильно построить её структуру на основе HTML и подготовить для дальнейшей работы с CSS и JavaScript.

* + 1. **Страница деталей препаратов**

## Шапка сайта и строка поиска

Шапка (header) и строка поиска (search-bar) являются частью общего стиля сайта и должны быть одинаковыми на всех страницах.

Шапка содержит:

* логотип "ФармИнфо",
* меню навигации по основным разделам сайта: Главная, Каталог, Аптеки, О нас, Контакты.

Под шапкой располагается строка поиска, позволяющая пользователю сразу начать поиск другого препарата без возврата на главную страницу.

Это делает сайт удобным и единообразным на всех уровнях.

## Название страницы и хлебные крошки

Следующий важный элемент — **хлебные крошки** (breadcrumbs) и **заголовок** страницы.

<div class="page-title">

        <div class="container">

            <div class="breadcrumbs">

                <a href="index.html">Главная</a>

                <span>></span>

                <a href="catalog.html">Каталог препаратов</a>

                <span>></span>

                <a href="#">Антибиотики</a>

                <span>></span>

                Амоксициллин

            </div>

            <h1>Детальная информация о препарате</h1>

        </div>

    </div>

Хлебные крошки позволяют пользователю понимать, где он находится, и при необходимости быстро вернуться на уровень выше.  
Заголовок <h1> нужен как для пользователя (понять о чём страница), так и для поисковых систем (SEO-оптимизация).

## Основная структура страницы

Контент страницы разбит на **две основные колонки**:

* слева — фотогалерея препарата,
* справа — текстовое описание, цена, наличие, кнопки действий и вкладки с дополнительной информацией.
* <div class="product-detail-container">
* <div class="product-gallery">

...</div> <!-- Фото -->

            <div class="product-details">

... </div> <!-- Текстовые данные -->

</div>

Такая организация помогает визуально разгрузить страницу и сделать её более понятной.

## Галерея фотографий препарата

Фотографии — важная часть, потому что пользователи хотят видеть внешний вид упаковки.

<div class="product-gallery">

                <div class="main-image">

                    <img src="https://ft.by/wp-content/uploads/drugs/amoksicillin-ft\_image2.png" alt="Амоксициллин">

                </div>

                <div class="thumbnail-container">

                    <div class="thumbnail active">

                        <img src="https://ft.by/wp-content/uploads/drugs/amoksicillin-ft\_image2.png" alt="Амоксициллин фото 1" height=50px width= 70px>

                    </div>

**Основная картинка** (main-image) показывает крупное изображение препарата.  
**Миниатюры** (thumbnail-container) позволяют выбрать другое фото (например, разные стороны упаковки). Позже через JavaScript мы сделаем их кликабельными.

## Основная информация о препарате

В правой части (product-details) находится текстовая информация о лекарстве.

 <div class="product-details">

                <h2 class="product-title">Амоксициллин</h2>

                <div class="product-subtitle">Антибиотик широкого спектра действия, 500 мг, 20 таблеток</div>

                <div class="price-availability">

                    <div class="product-price">12 azn</div>

                    <div class="product-availability in-stock">В наличии</div>

                </div>

Здесь мы видим:

* Название препарата,
* Краткое описание формы выпуска,
* Цену,
* Наличие в аптеках.

Это важная информация, которую пользователь хочет увидеть сразу.

### ***Кнопки действий***

Ниже основных данных размещаются кнопки:

<div class="action-buttons">

                    <a href="#" class="action-button find-pharmacy">Найти в аптеках</a>

                    <a href="#" class="action-button add-favorite">

<svg <!-- ... --></svg>

                        Добавить в избранное

                    </a>

                </div>

Первая кнопка поможет найти препарат в ближайших аптеках.  
Вторая кнопка — добавить препарат в список "Избранное" пользователя, чтобы легко вернуться к нему позже.

## Вкладки с дополнительной информацией

Ниже кнопок находится блок с вкладками (tabs).

Навигация по вкладкам:

<div class="tabs-nav">

                        <button class="tab-button active" data-tab="description">Описание</button>

                        <button class="tab-button" data-tab="properties">Характеристики</button>

                        <button class="tab-button" data-tab="usage">Применение</button>

                        <button class="tab-button" data-tab="side-effects">Побочные эффекты</button>

                        <button class="tab-button" data-tab="reviews">Отзывы</button>

                    </div>

Каждая вкладка связана с отдельным блоком контента:

<div class="tab-content active" id="description">

... </div>

<div class="tab-content" id="properties">

... </div>

<div class="tab-content" id="usage">

... </div>

<div class="tab-content" id="side-effects">

... </div>

<div class="tab-content" id="reviews">

... </div>

При клике на кнопку происходит переключение вкладок (это реализуется через скрипт).

## Что находится в каждой вкладке?

**Описание** — общее описание действия препарата, механизм работы, группа лекарств.  
**Характеристики** — действующее вещество, форма выпуска, дозировка, производитель, условия хранения.  
**Применение** — как правильно принимать препарат, рекомендованные дозировки для взрослых и детей.  
**Побочные эффекты** — возможные негативные реакции, противопоказания, меры предосторожности.  
**Отзывы** — возможность для реальных пользователей оставить своё мнение о препарате и поставить ему оценку.

Таким образом, каждая вкладка содержит строго свою категорию информации, а не перегружает пользователя сразу всем текстом.

1. **Использование стилей CSS**
   * 1. **Главный сайт**

## 1. ****Переменные через****

:root {

    --primary-color: #4CAF50;

    --secondary-color: #E8F5E9;

    --accent-color: #2E7D32;

    --text-color: #333;

    --light-gray: #f5f5f5;

    --border-color: #ddd;

}

* Здесь задаются **CSS-переменные**. Они объявляются в :root, чтобы быть доступны **во всём документе**.
* Теперь вместо повторяющихся цветов можно использовать var(--имя-переменной).
* Это делает стили более удобными для поддержки: хочешь поменять основной цвет — меняешь его только тут!

## 2. ****Базовые стили для всего****

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

* Убирается стандартный внешний отступ (margin) и внутренний (padding) для всех элементов.
* box-sizing: border-box — упрощает расчёт размеров блоков (ширина включает отступы и границы).
* Устанавливается современный красивый шрифт для всего сайта.

## 3. ****Стилизация тела страницы****

body {

color: var(--text-color);

background-color: #f9f9f9;

line-height: 1.6;

}

* Цвет текста — тёмно-серый.
* Фон — светло-серый.
* line-height: 1.6 — увеличенная высота строк для лучшей читаемости.

## 4. ****Структурные контейнеры****

.container {

max-width: 1200px;

margin: 0 auto;

padding: 0 15px;

}

* Центрирует контент и ограничивает его ширину до 1200px.
* Немного внутреннего отступа по бокам.

## 5. ****Хедер и навигация****

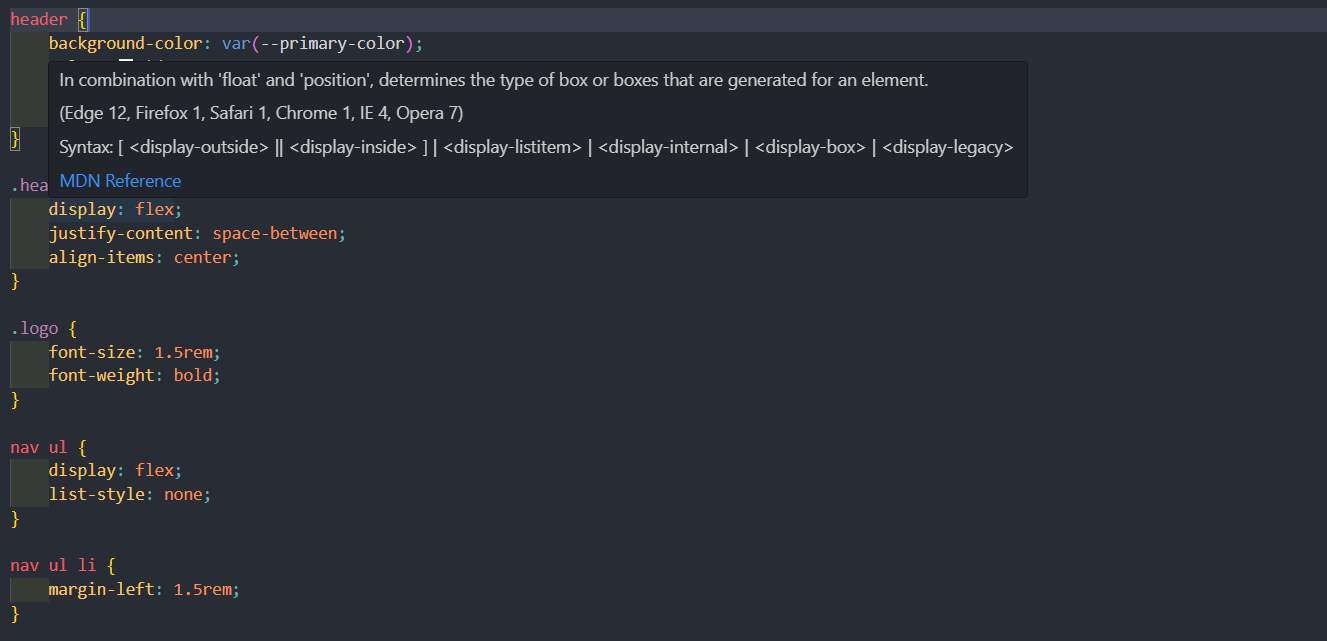


Рис. 2.2 Верхняя панель с навигатором

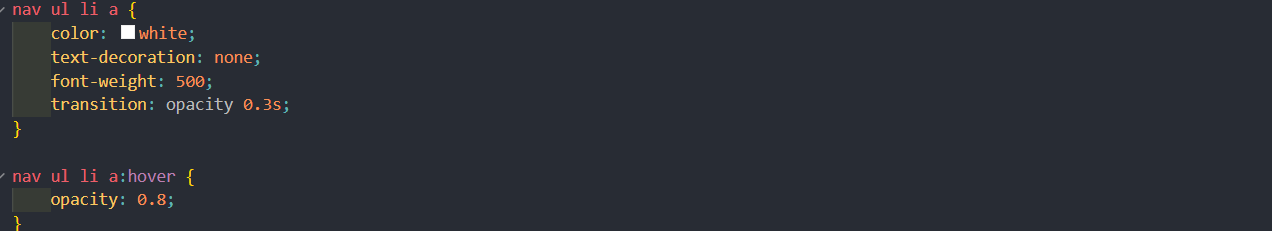


Рис. 2. hover

* Верхняя панель (header) зелёная, с белым текстом.
* Навигация (nav ul) строится в горизонтальный ряд.
* У ссылок плавное затемнение при наведении (hover), через opacity.

## 6. ****Поисковая строка*****Рис. 2.4 поиск*

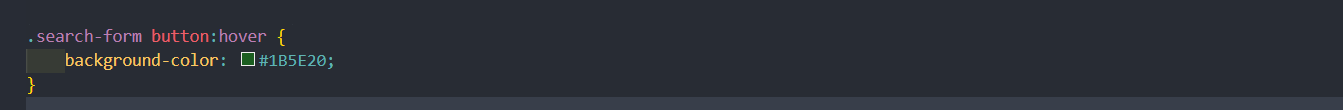


Рис. 2. hover

* Блок поиска с мягким светлым фоном.
* Кнопка поиска зелёная и красиво реагирует на наведение (меняет цвет фона).

## 7. ****Hero-секция****

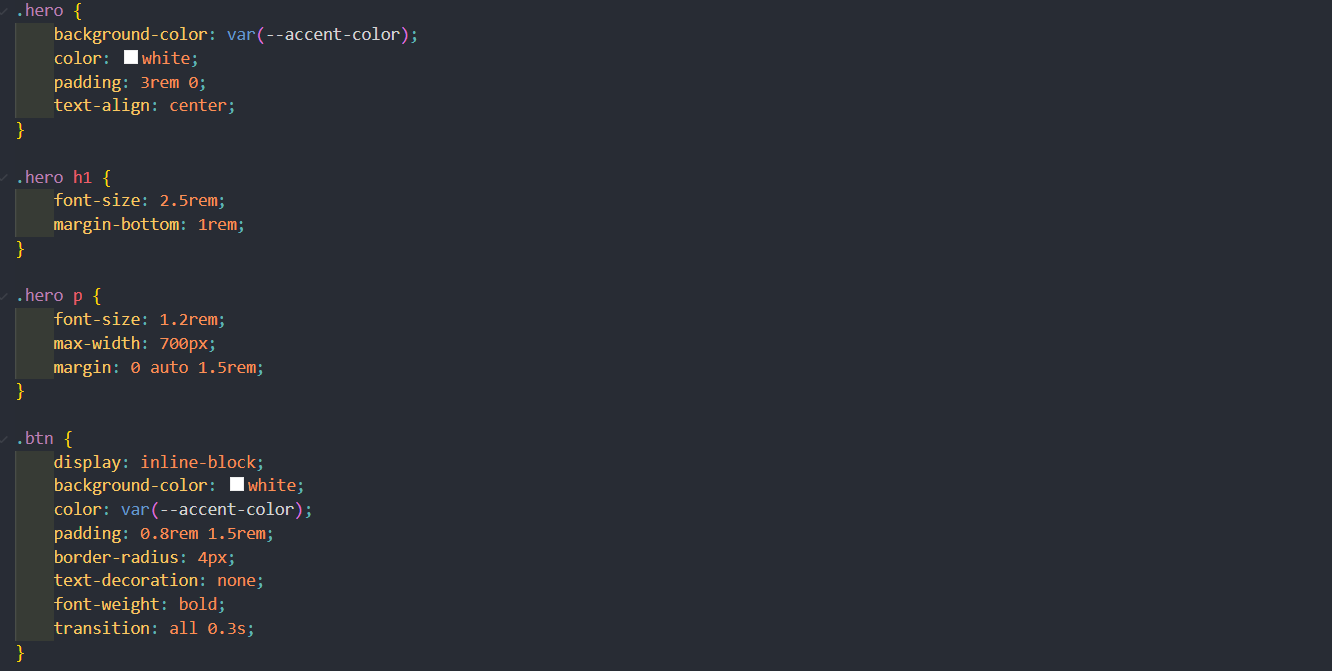


Рис. 2. hero

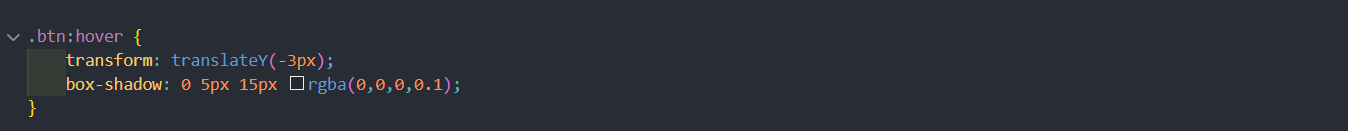


Рис. 2. btn hover

* Крупный заголовок и текст по центру страницы на насыщенном зелёном фоне.
* Кнопка ("btn") белая, эффектная, чуть "приподнимается" при наведении (transform: translateY(-3px)).

## 8. ****Блок "особенности"****

## 

Рис. 2. особенности

* Используется **CSS Grid**: элементы строятся в сетку.
* Каждая карточка немного поднимается при наведении (hover) и отбрасывает лёгкую тень.

## 9. ****Популярные товары****

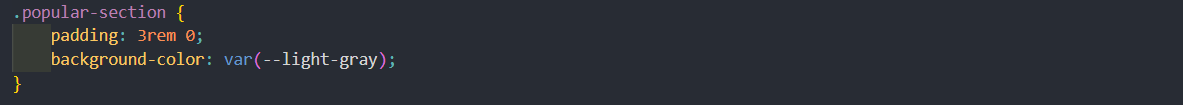
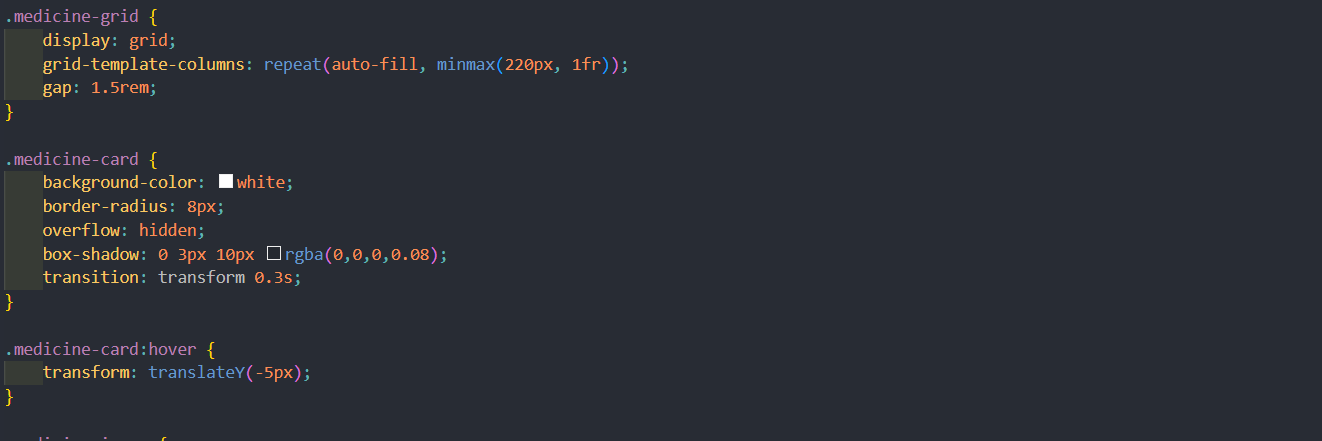


Рис. 2. сетка популярных

Рис. 2. блоки для лекарств

* Сетка для карточек лекарств.
* Карточки тоже "оживают" при наведении — лёгкое движение вверх.

## 10. ****Поиск аптеки****

## 

Рис. 2. поиск

* Блок с изображением и текстом о поиске ближайшей аптеки.
* Изображение и текст стоят рядом (флекс-контейнер).

## 11. ****Футер*****Рис. 2.12 нижняя часть сайта*

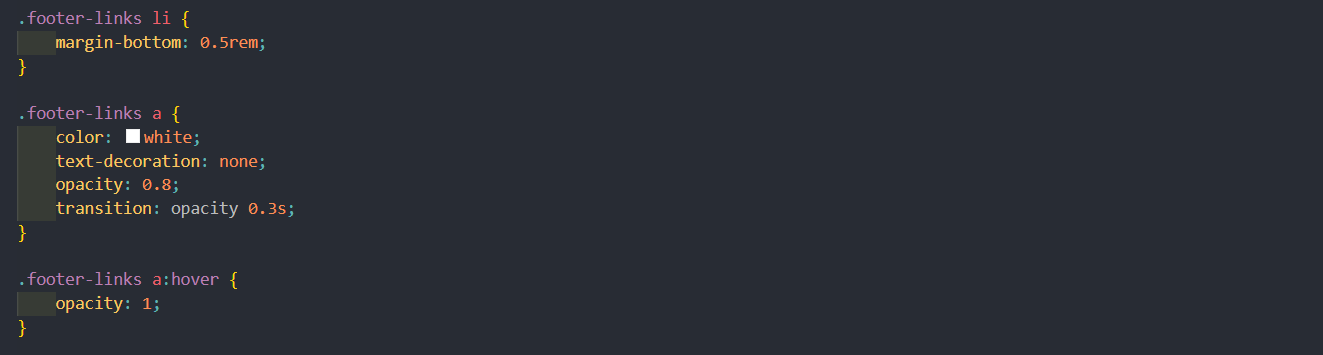


Рис. 2. hover

* Тёмный футер.
* Колонки футера стоят в ряд.
* Ссылки в футере становятся ярче при наведении.

## 12. ****Адаптивность для мобильных (****@media****)****

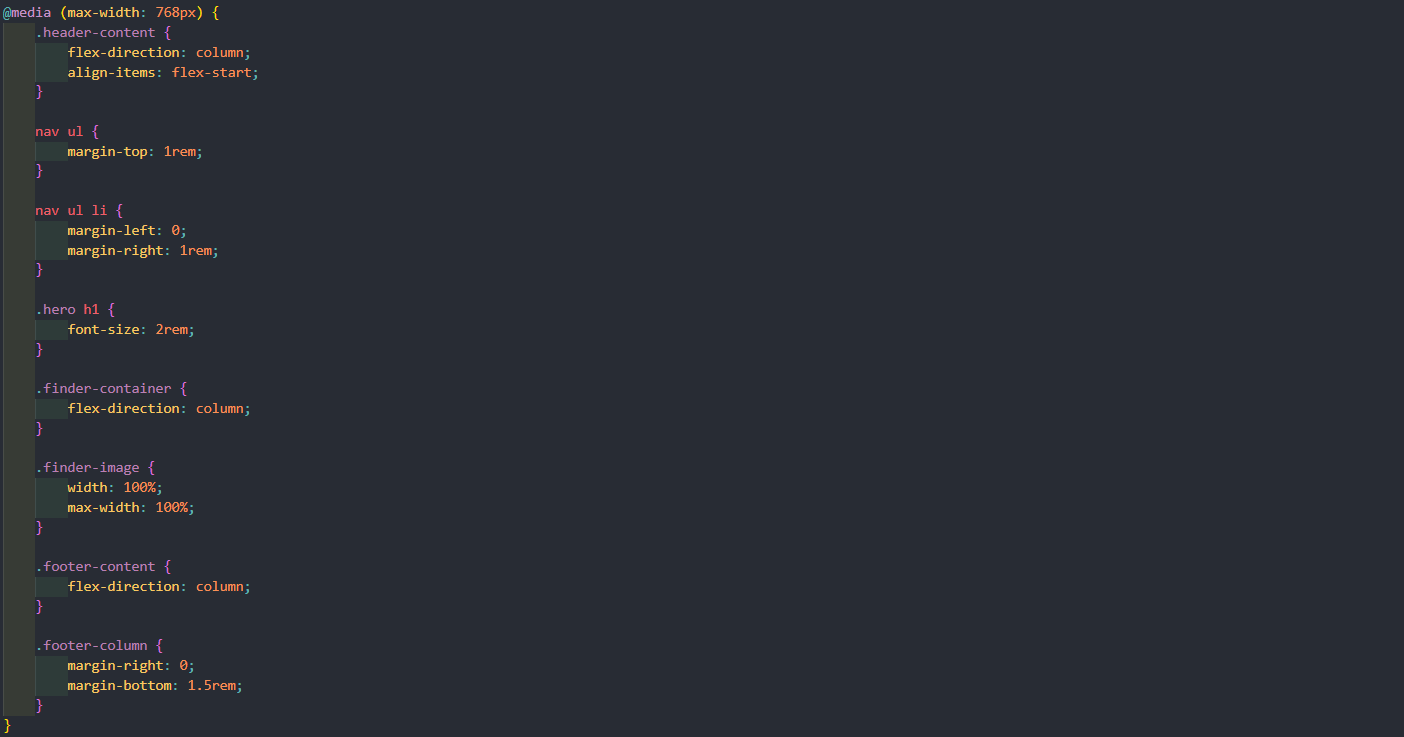


Рис. 2. структура для телефонов

* Когда ширина экрана меньше 768px:
  + Навигация (header-content) становится вертикальной.
  + Сетки меняются на столбцы.
  + Футер тоже переходит на вертикальное расположение.

Это делает сайт удобным для просмотра на телефонах и планшетах.

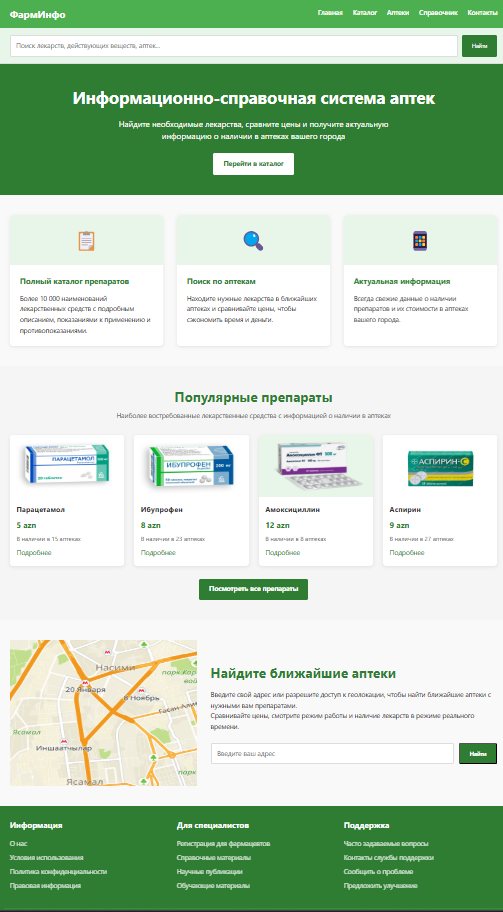


Рис. 2. Главная страница

* + 1. **Страница каталога**

## 1. Основные переменные

:root {

--primary-color: #4CAF50;

--secondary-color: #E8F5E9;

--accent-color: #2E7D32;

--text-color: #333;

--light-gray: #f5f5f5;

--border-color: #ddd;

}

* Здесь определены **CSS-переменные**.
* Они позволяют удобно управлять цветами по всему проекту: меняешь значение в одном месте — и оно меняется везде.

## 2. Базовая стилизация для всех элементов

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

* Сброс отступов (margin и padding в 0).
* box-sizing: border-box — чтобы ширина элементов учитывала padding и border.
* Единый шрифт на весь сайт.

## 3. Оформление тела страницы

body {

color: var(--text-color);

background-color: #f9f9f9;

line-height: 1.6;

}

* Основной цвет текста.
* Светлый фон страницы.
* Увеличенный межстрочный интервал для лёгкого чтения.

## 4. Контейнеры и структура

.container {

max-width: 1200px;

margin: 0 auto;

padding: 0 15px;

}

* Центрирование контента и ограничение ширины.
* Защита контента с небольшими отступами слева и справа.

## Шапка и навигация

## 

Рис. 2. Шапка

* **header** — зелёный фон, белый текст, тень снизу для объёмности.
* **.header-content** — флекс-контейнер, чтобы логотип и меню располагались по краям.
* **nav ul** — горизонтальное меню без маркеров.
* **Ссылки в меню** плавно меняют прозрачность при наведении.

## Поисковая строка

## 

Рис. 2. поиск

* Фон чуть светлее основного зелёного.
* Форма поиска — горизонтальный флекс-контейнер с зазором.
* Красивое закругление у полей и кнопок.
* Кнопка поиска меняет цвет при наведении.

## Заголовок страницы и хлебные крошки

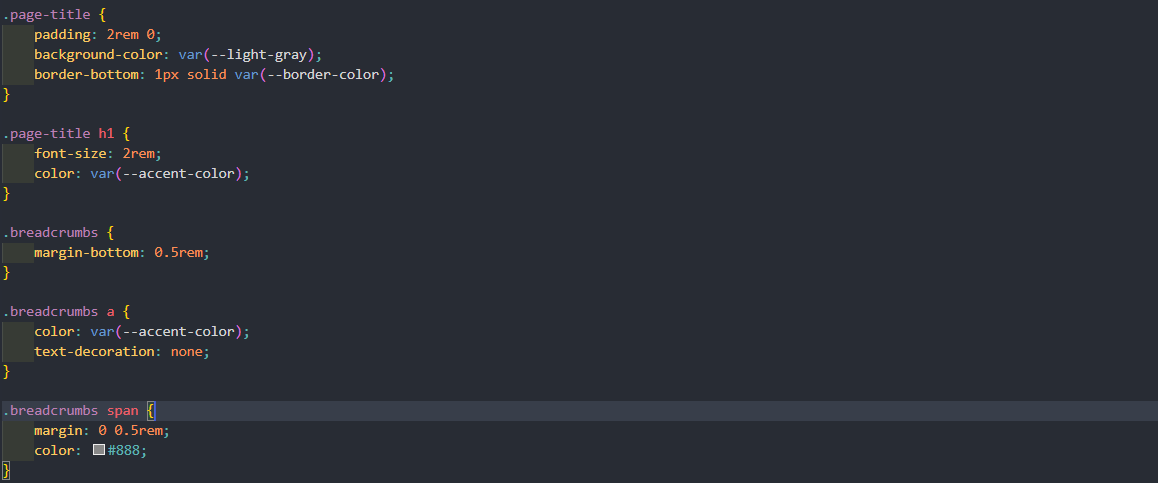


Рис. 2. элементы навигации

* Светлый фон с серой полоской снизу.
* Хлебные крошки: маленькие стрелочки (span) между ссылками (a), чтобы показать путь пользователя.

## Каталог товаров

### ***Боковая панель (sidebar)***

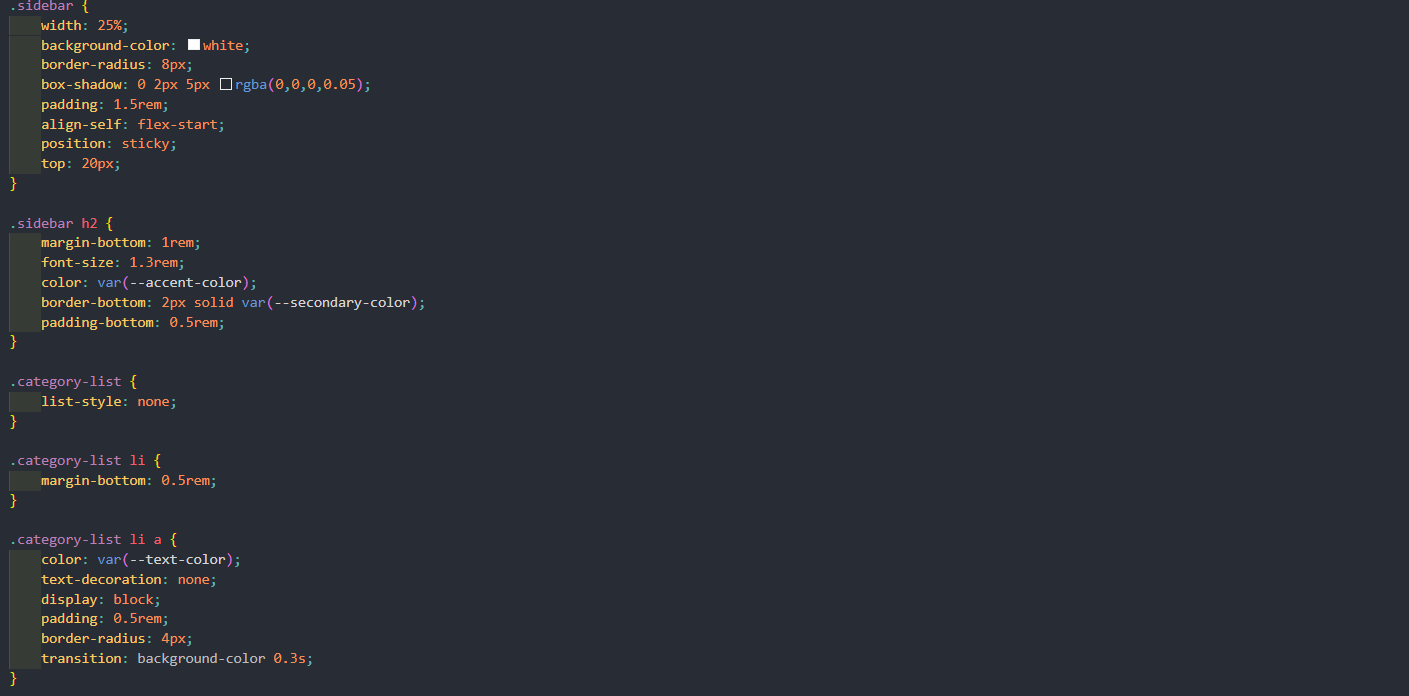


Рис. 2. боковая панель

* Ширина 25% от контейнера.
* Белый фон, тень, закруглённые углы.
* Фиксация при прокрутке (position: sticky).

### ***Фильтры***

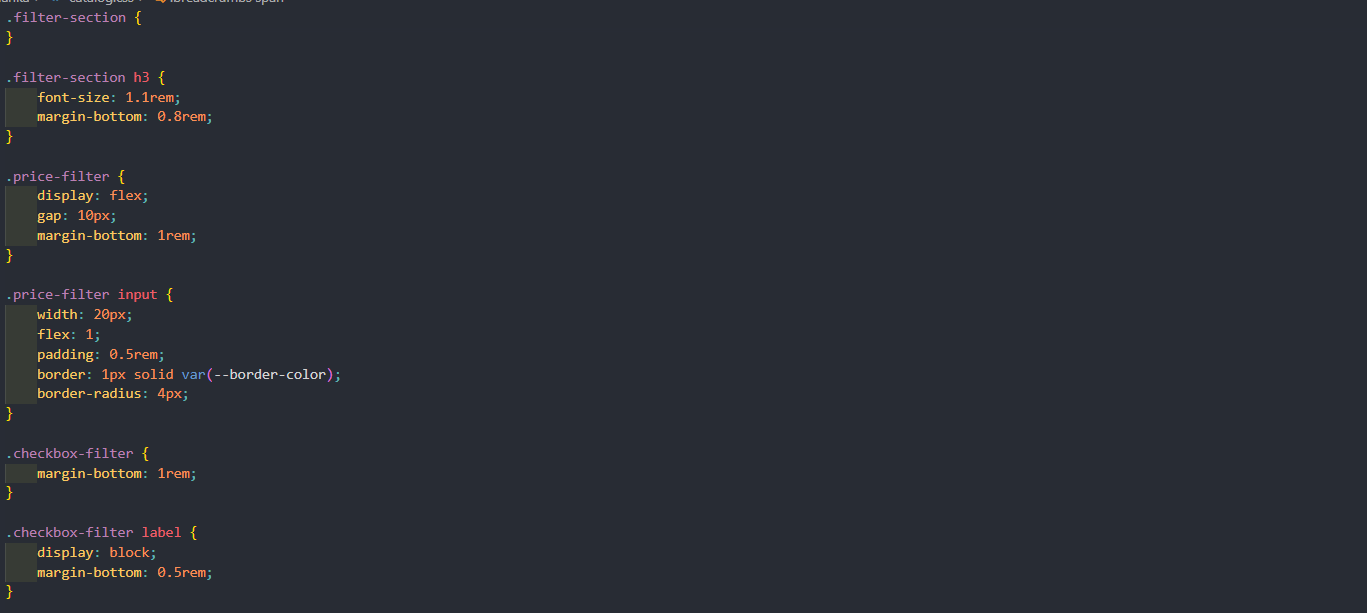


Рис. 2.20 кнопки для фильтров

* Секции для фильтрации по цене и по признакам (чекбоксы).
* Кнопка фильтрации красиво меняет цвет при наведении.

### ***Основная часть каталога (catalog-content)***

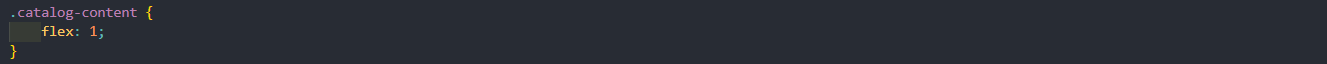


Рис. 2.21

* Товары отображаются в сетке (display: grid), адаптивно под ширину экрана.

## Карточки товаров

## 

Рис. 2.23 лекарства

* Белая карточка с тенью и закруглениями.
* На карточке — изображение (фон-заглушка) и текстовая информация.
* При наведении карточка приподнимается (transform: translateY(-5px)).

### ***Доступность товара***

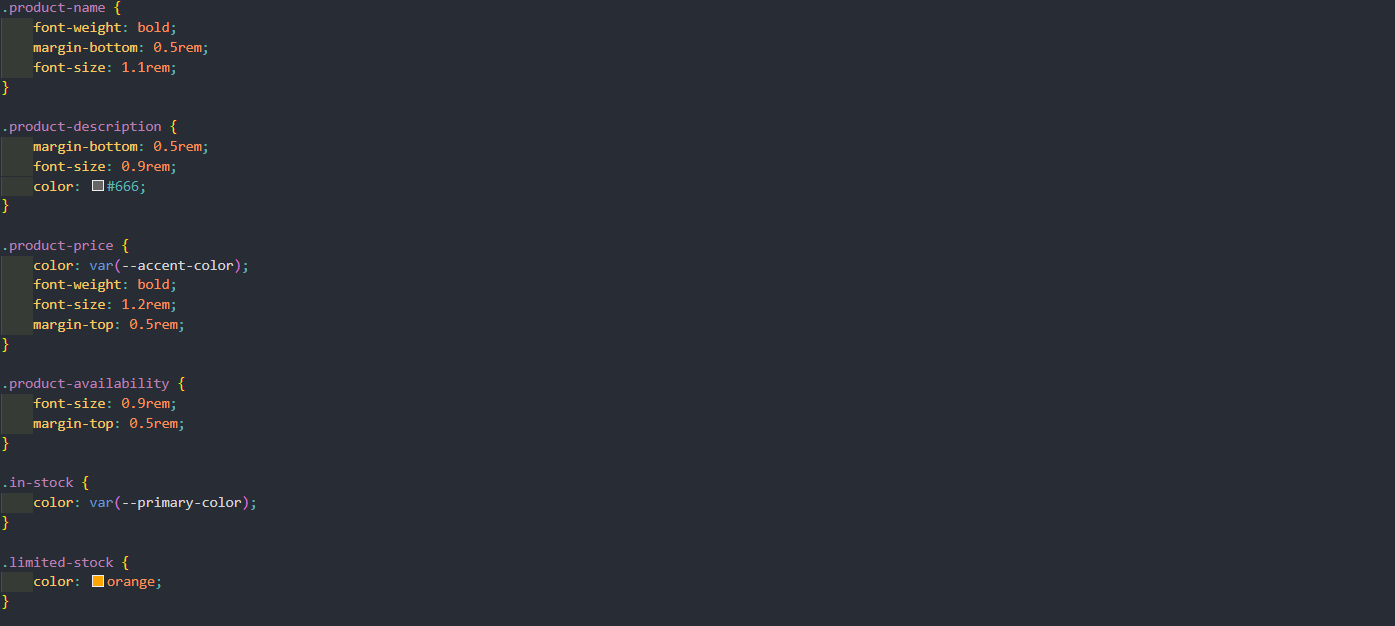


Рис. 2.24 доступность товара цветовым обозначением

* **In stock** — зелёный текст.
* **Limited stock** — оранжевый текст.

## Действия в карточке

## 

Рис. 2.25 кнопка для детального рассмотрения лекарства

* Кнопки "Подробнее" и "Где купить".
* Разный цвет кнопок.
* Анимация изменения цвета при наведении.

## Пагинация

## 

Рис. 2.26 пагинация

* Красивая пагинация: цифры с рамкой, активная страница выделяется зелёным цветом.

## Футер

## 

Рис. 2.27

* Светлый фон с верхней границей.
* Контент футера размещён в несколько колонок.
* Ссылки в футере меняют цвет при наведении.

## Адаптивность

@media (max-width: 768px) { ... }

* На мобильных устройствах:
  + Меню и футер переходят в вертикальный режим.
  + Карточки и формы адаптируются под узкие экраны.
  + Уменьшается размер заголовков.

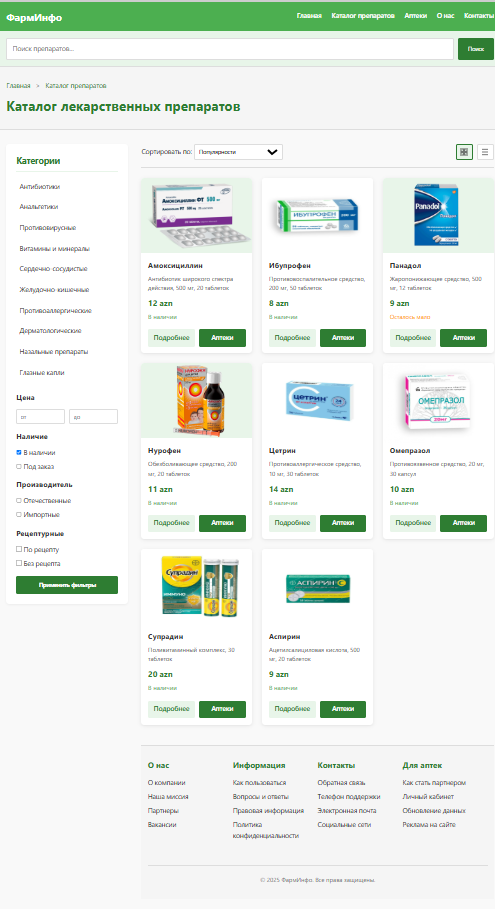


Рис. 2.28 страница каталога

* + 1. **Страница деталей продукта**

### 1. **Переменные (Root-переменные)**

:root {

--primary-color: #4CAF50;

--secondary-color: #E8F5E9;

--accent-color: #2E7D32;

--text-color: #333;

--light-gray: #f5f5f5;

--border-color: #ddd;

}

* Здесь задаются **CSS-переменные** (начинаются с --), которые можно использовать по всему стилю через var(--название-переменной).
* Это позволяет **легко управлять цветами и другими общими значениями** — чтобы, например, поменять цветовую схему, достаточно изменить только здесь!

### 2. **Базовые стили**

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: ...;

}

* Сброс отступов (margin, padding) для всех элементов.
* box-sizing: border-box; — размеры элементов теперь учитывают паддинги и границы (очень удобно для верстки сеток).
* Определяется общий шрифт для всего проекта.

### 3. **Стилизация** body

body {

color: var(--text-color);

background-color: #f9f9f9;

line-height: 1.6;

}

* Цвет текста и фона.
* line-height — высота строки для лучшей читаемости текста.

### 4. **Контейнер**

.container {

max-width: 1200px;

margin: 0 auto;

padding: 0 15px;

}

* Центрирует контент по центру страницы (margin: 0 auto) и ограничивает ширину до 1200px.
* Делает отступы по краям внутри контейнера.

### **Шапка (Header)**

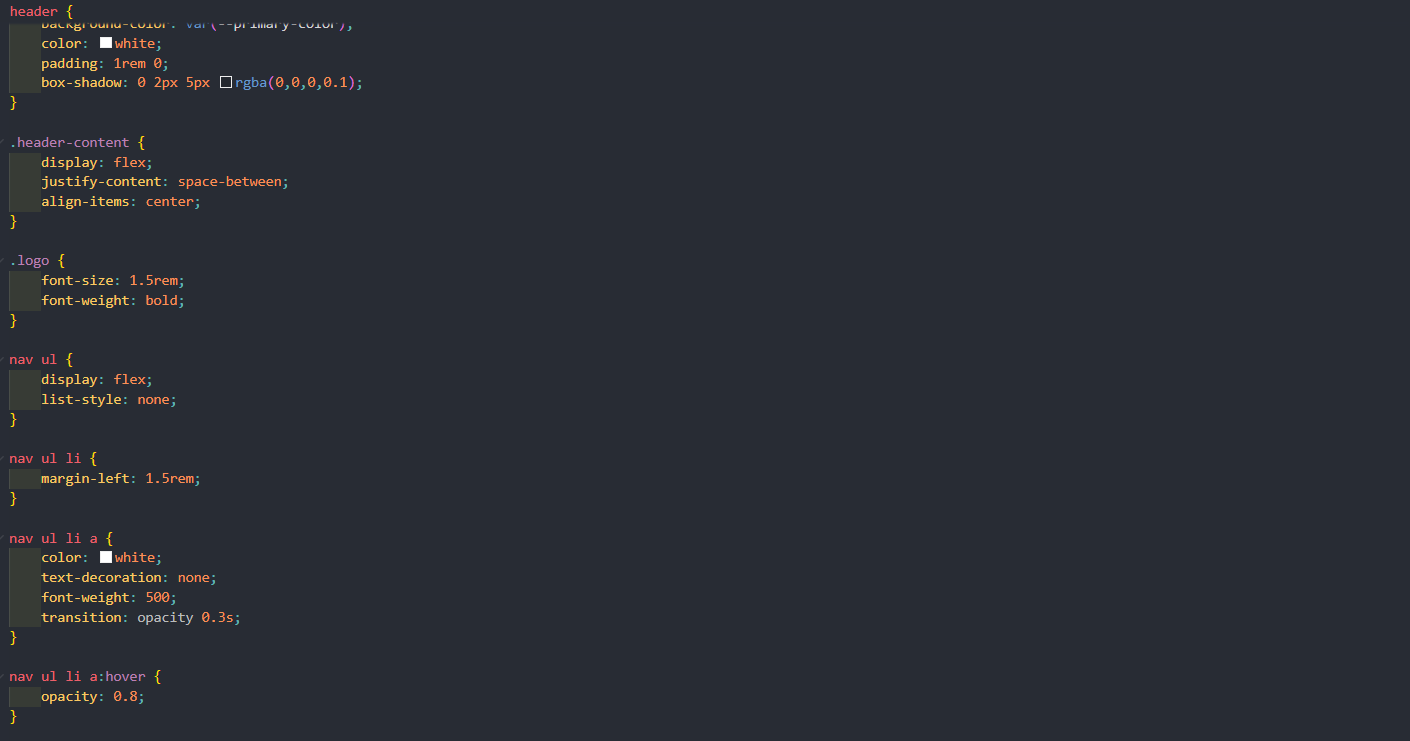


Рис. 2.29 шапка

* Шапка сайта с цветным фоном и белым текстом.
* Используется Flexbox (display: flex;) для **горизонтального расположения** логотипа и навигации.
* Навигация (nav) через ненумерованный список (ul).
* При наведении (:hover) ссылки становятся полупрозрачными (эффект через opacity).

### **Поисковая строка**

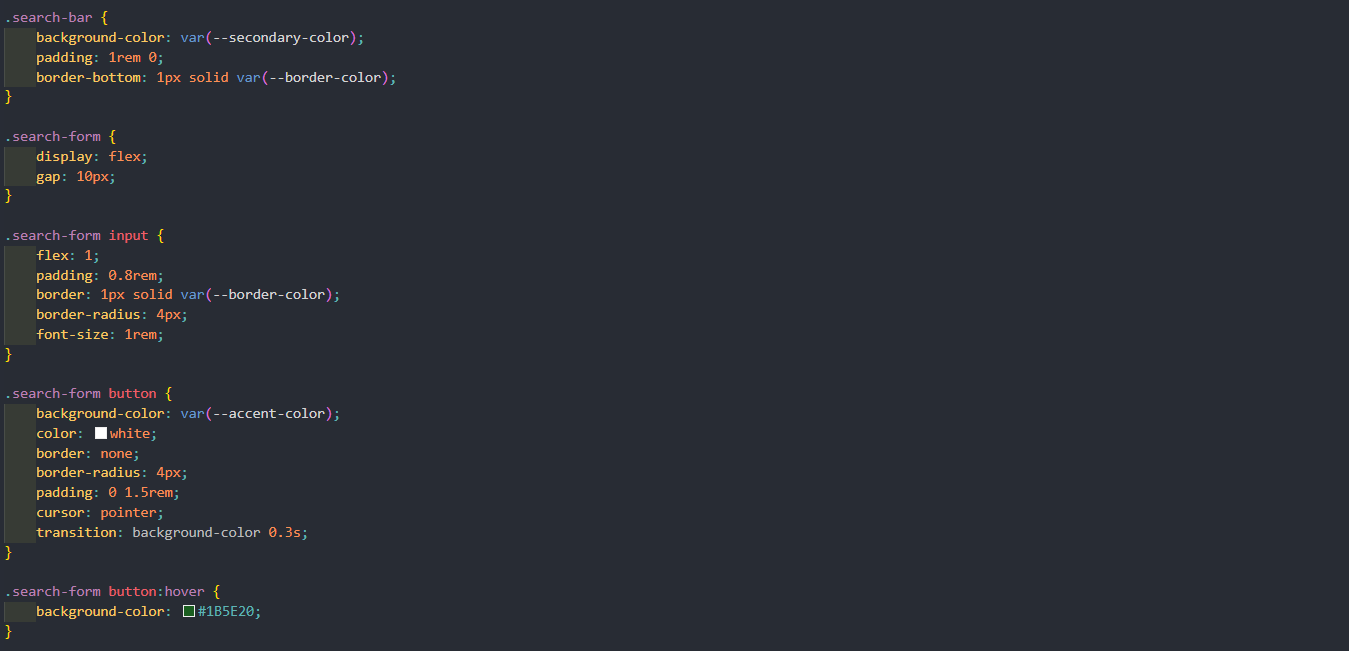


Рис. 2.30 поиск

* Блок поиска с мягким фоном.
* Поле ввода растягивается на всю ширину (flex: 1).
* Кнопка поиска с анимацией изменения цвета при наведении.

### **Название страницы и хлебные крошки (Breadcrumbs)**

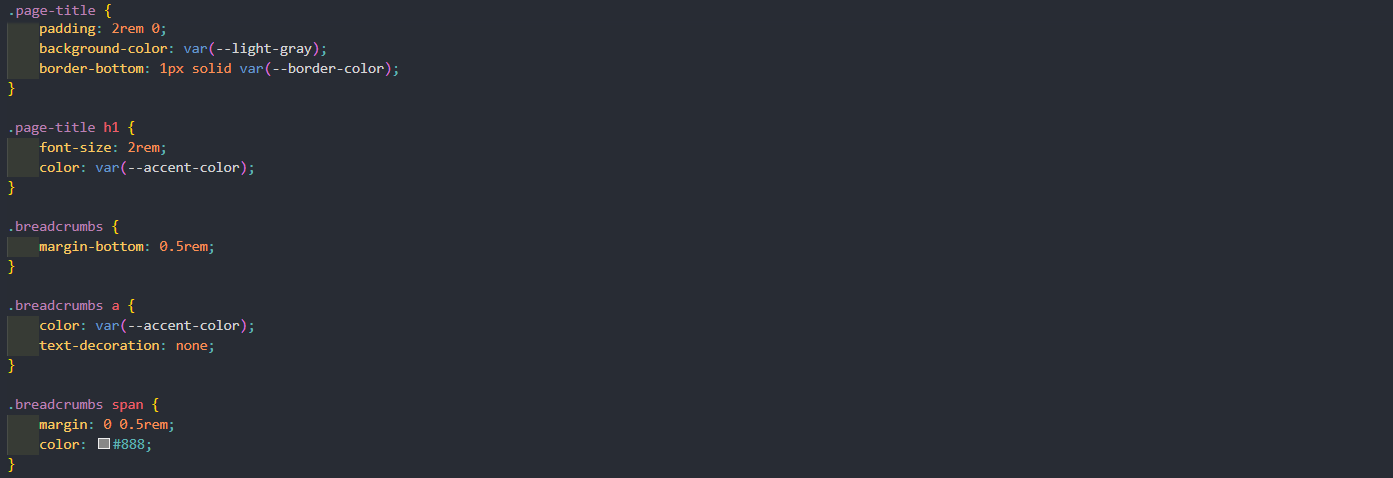


Рис. 2.31 хлебные крошки

* Фон для заголовка и навигации.
* Разделение ссылок "крошек" через маленький отступ и цвет.

### **Футер (Подвал сайта)**

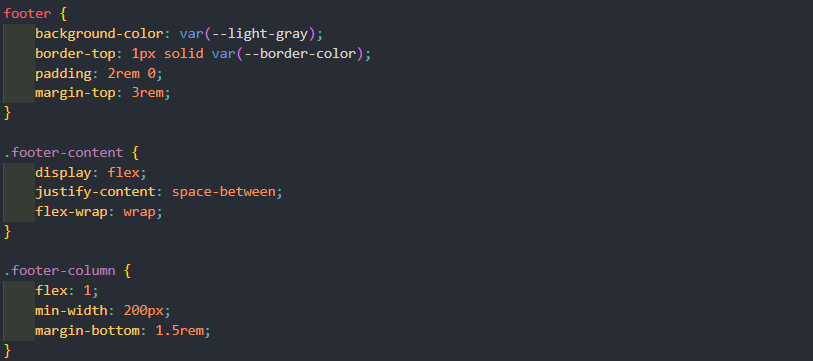


Рис. 2.32 футер

* Используется Flexbox для размещения колонок в футере.
* Красивые заголовки, списки ссылок с эффектом наведения.

### **Детали товара (Product Detail Page)**

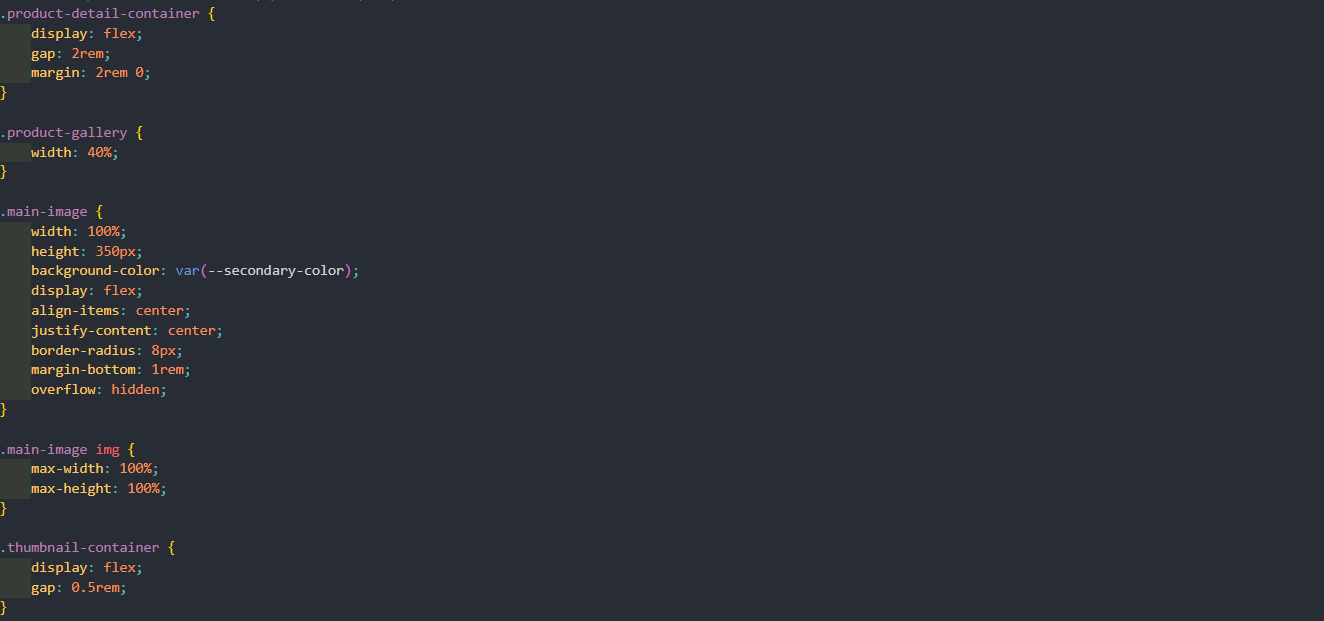


Рис. 2.33 детали

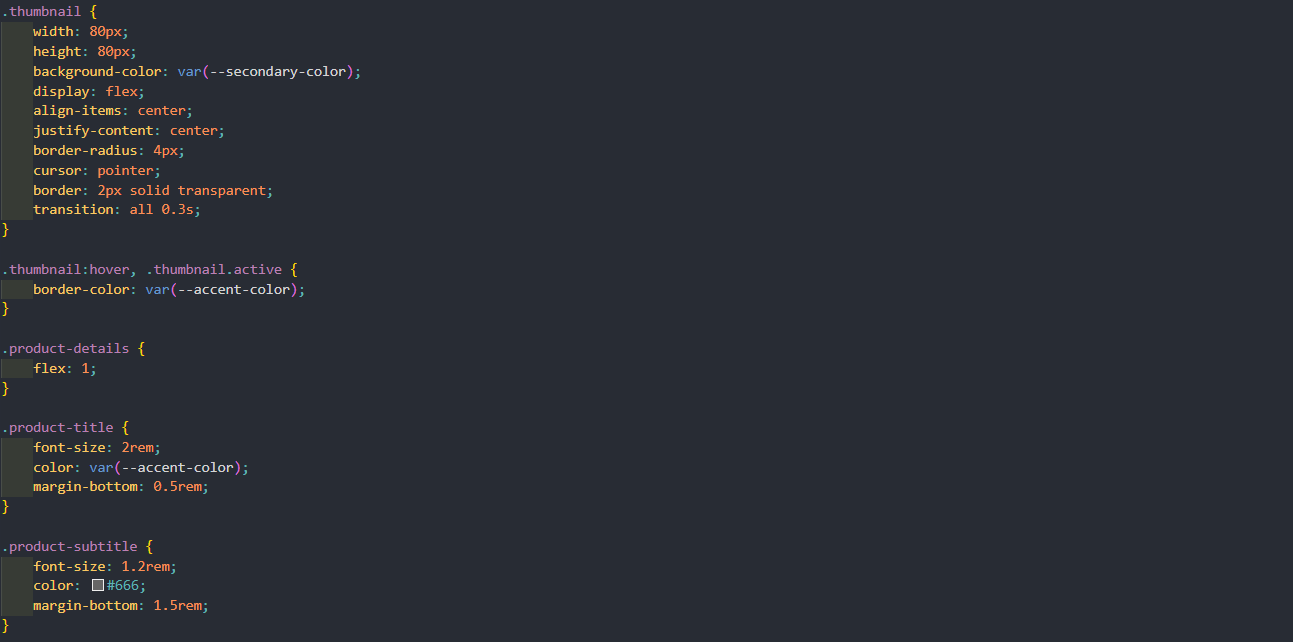


Рис. 2.34 thumbnails и описание

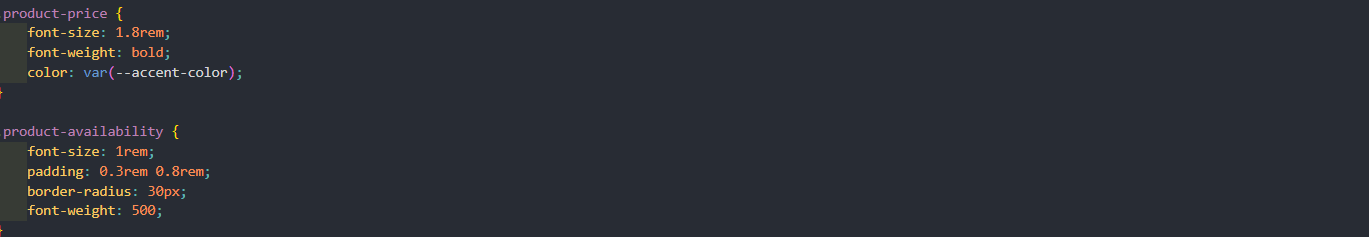


Рис. 2.35 цена и наличие

* Визуальная галерея товара:
  + Большая картинка товара.
  + Миниатюры (thumbnails) с возможностью выбрать другую картинку.
  + Активация миниатюры через добавление класса active.
* Секция с описанием товара (заголовок, подзаголовок, цена, наличие на складе).

### **Кнопки действий**

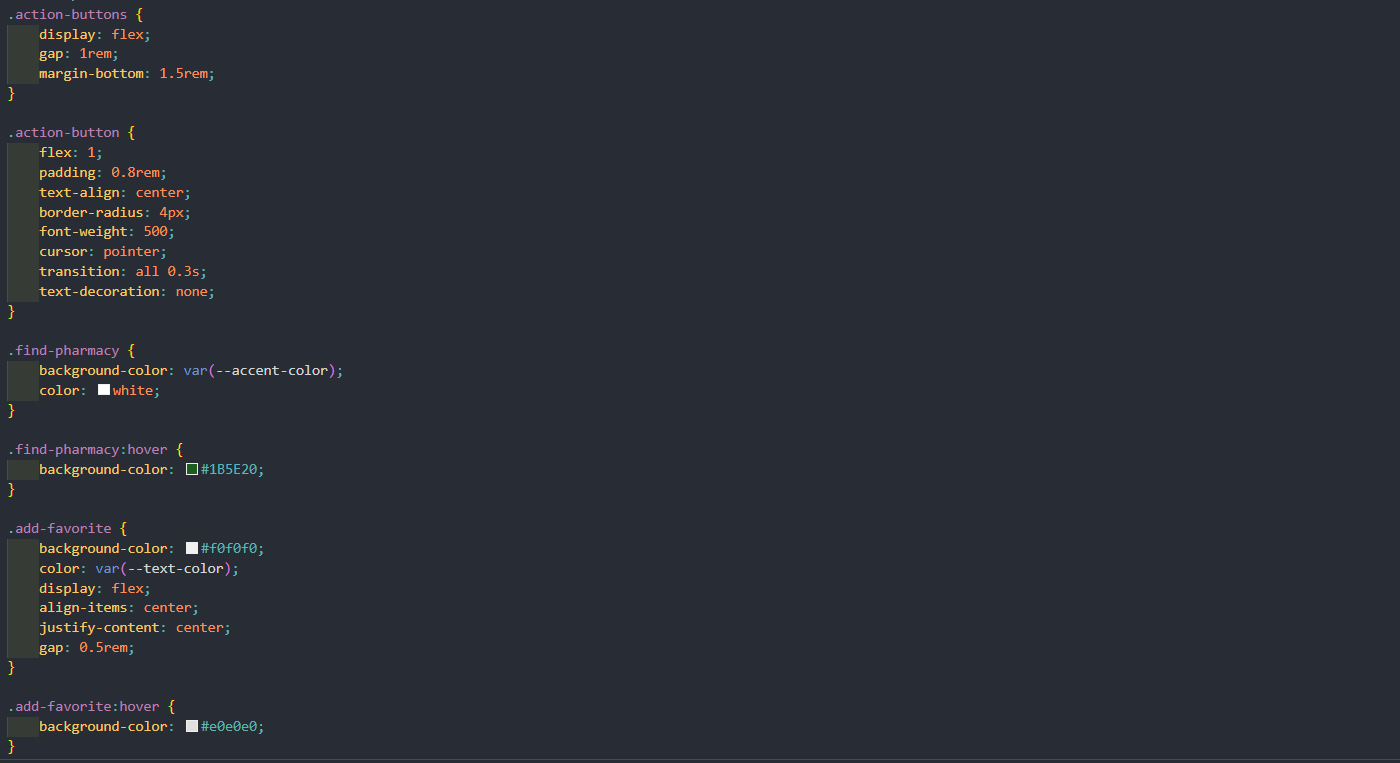


Рис. 2.36 кнопки действий

* Кнопки "Найти в аптеке" и "Добавить в избранное" с красивыми эффектами наведения.

### **Табы (Вкладки с информацией)**

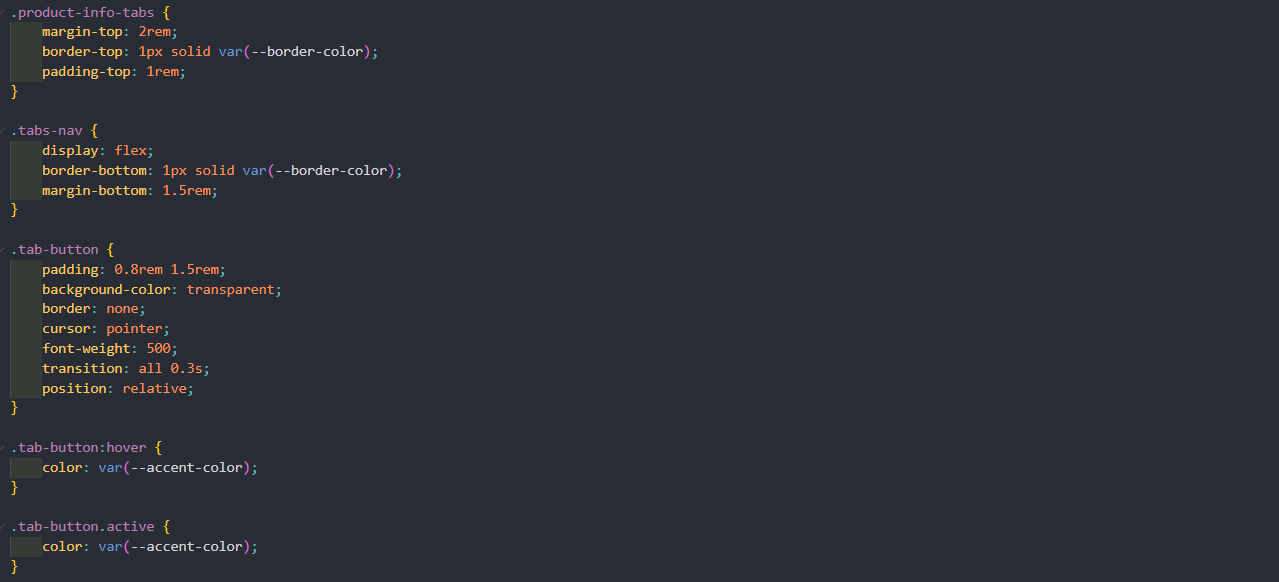


Рис. 2.37 вкладки с информацией

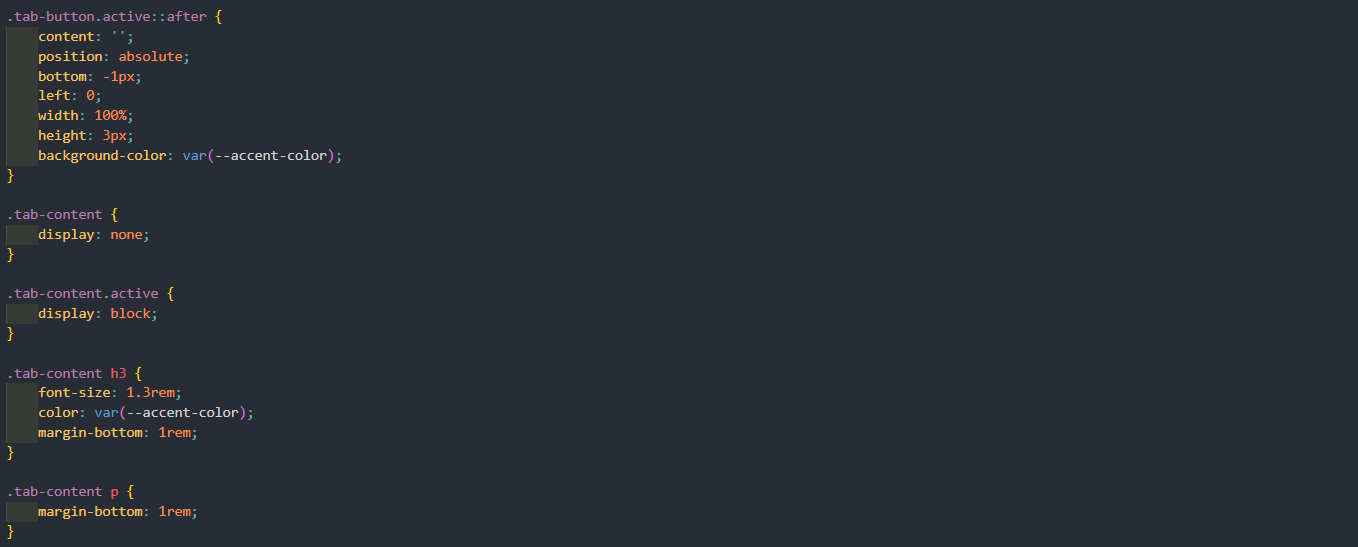


Рис. 2.38 вкладки с информацией

* Секция вкладок (Tabs):
  + При нажатии одна вкладка активируется (.active класс), и появляется связанный контент.
  + Нижняя линия под активной вкладкой (::after псевдоэлемент).

### **Списки характеристик, дозировка и побочные эффекты**

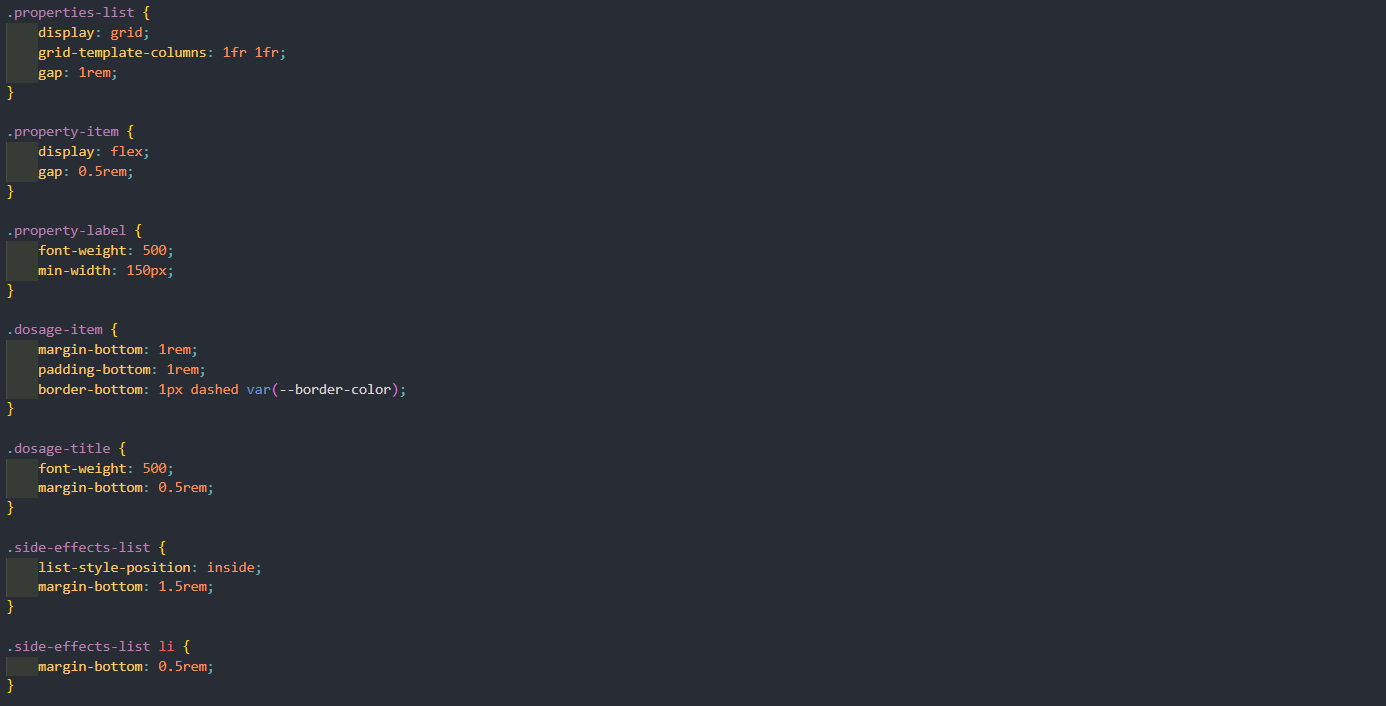


Рис. 2.39 характеристики товаров

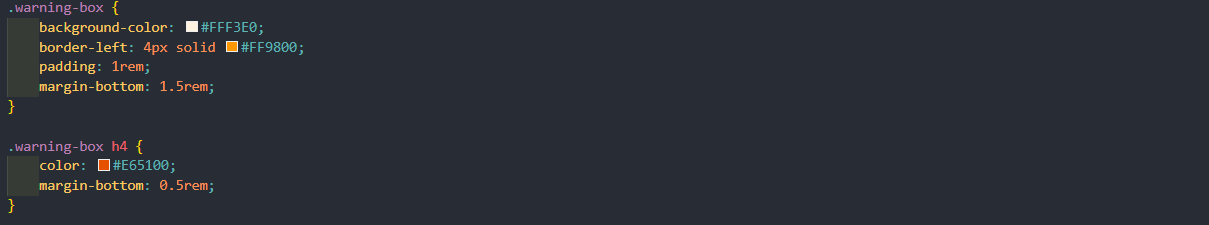


Рис. 2.40 предупреждающее сообщение

* Используется сетка (grid) для аккуратного отображения характеристик товара.
* Блоки с информацией отделены линиями (border-bottom).

### **Аналоги товара**

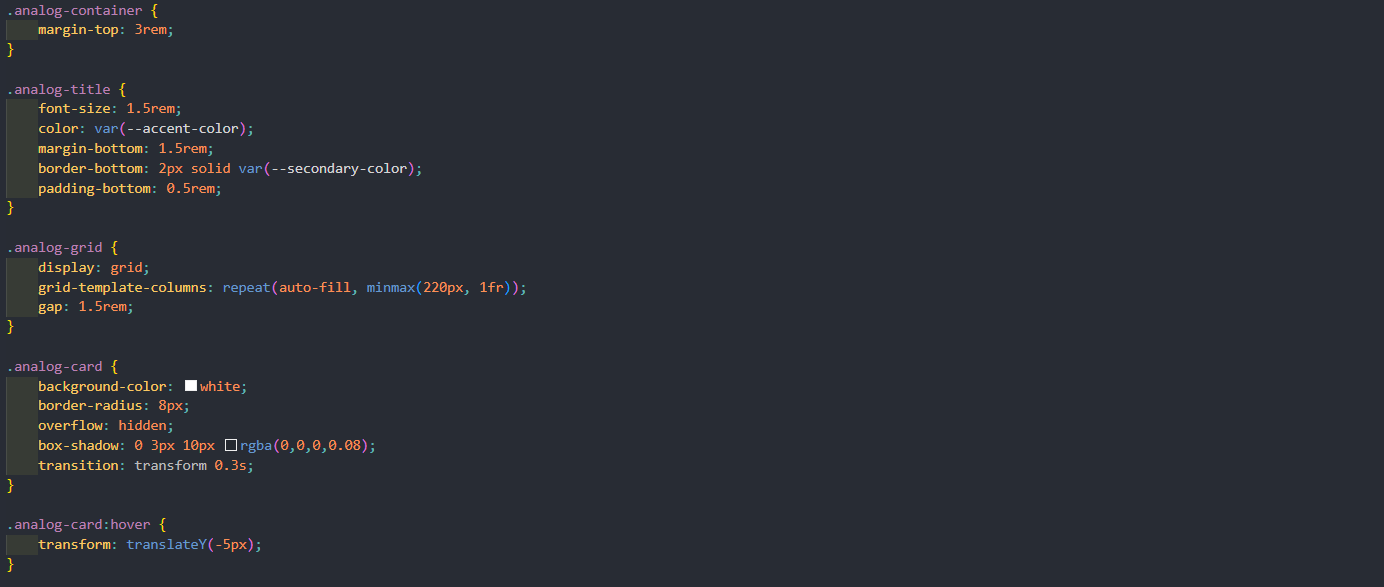


Рис. 2.41 аналоги лекарств

* Сетка карточек аналогичных товаров (grid-template-columns).
* При наведении карточка немного "приподнимается" (transform: translateY(-5px)).

### **Отзывы (Reviews)**

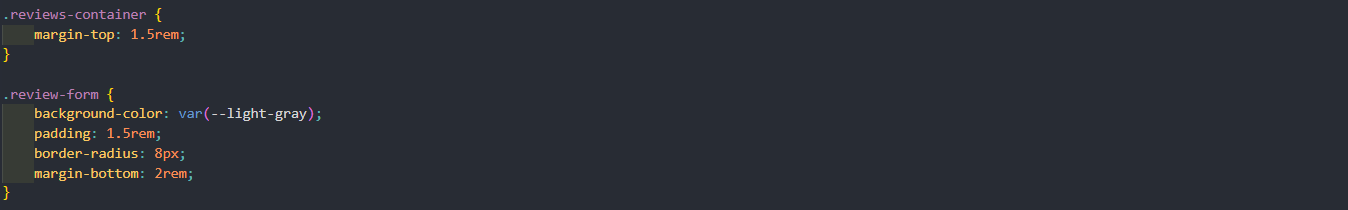


Рис. 2.42 отзывы

* Блок формы отзыва.
* Список оставленных отзывов со звездочками (rating-stars), цветными звездами при оценке (.filled).

### **Кнопка "Назад в каталог"**

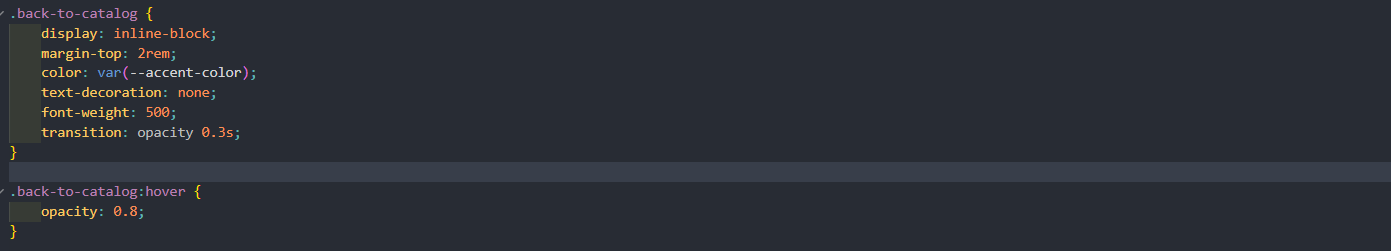


Рис. 2.44 кнопка назад

* Лёгкая, красивая ссылка с эффектом наведения.

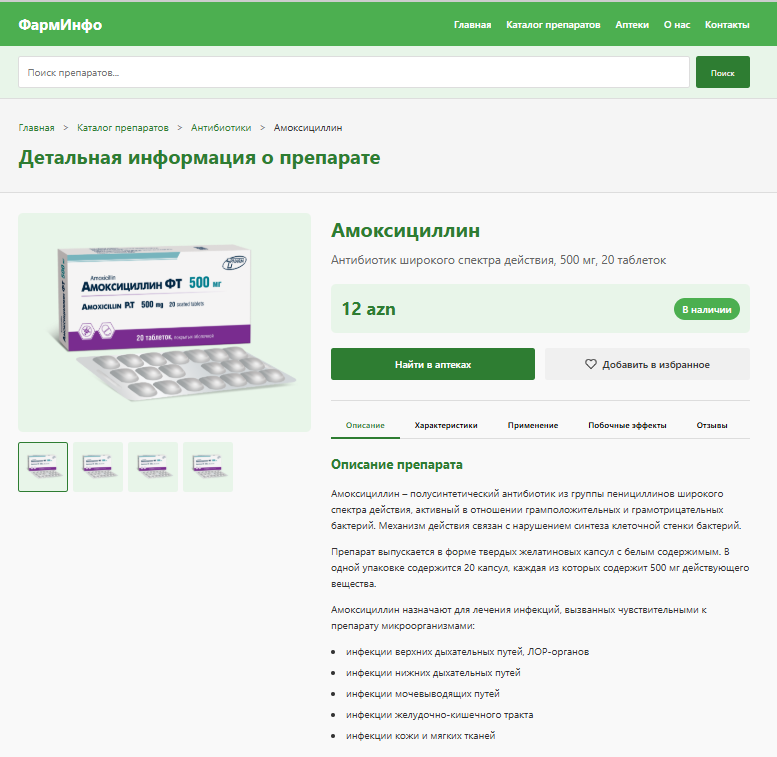


Рис. 2.45 Страница деталей препарата

1. **Использование JavaScript**
2. **JavaScript функции на главной странице**

### 1. **Ждём полной загрузки страницы**

document.addEventListener("DOMContentLoaded", () => { ... });

* Сначала весь HTML загружается.
* Только после этого можно работать с элементами страницы в JS.

### 2. **Обработка формы поиска**

const searchForm = document.querySelector(".search-form");

searchForm.addEventListener("submit", (e) => { ... });

* Ищется форма поиска по классу .search-form.
* При отправке формы:
  + Останавливается стандартное поведение (e.preventDefault()), чтобы страница не перезагружалась.
  + Берётся значение из поля ввода (input).
  + Если текст введён — показывается alert с запросом и пишется сообщение в консоль.
  + Если нет текста — выводится просьба ввести запрос.

### 3. **Обработка формы местоположения**

const locationForm = document.querySelector(".location-form");

locationForm.addEventListener("submit", (e) => { ... });

* Аналогично предыдущей форме.
* Пользователь вводит адрес.
* Если адрес есть — показывается alert о поиске аптек рядом.
* Иначе — сообщение о необходимости ввести адрес.

### 4. **Подсказка для ссылок "Подробнее"**

const detailLinks = document.querySelectorAll('.medicine-info a');

detailLinks.forEach(link => {

link.title = "Нажмите, чтобы узнать больше о препарате";

});

* Для всех ссылок внутри .medicine-info устанавливается атрибут title.
* При наведении на ссылку браузер покажет всплывающую подсказку.

### 5. **Плавный скролл до блока "Аптеки"**

const navLinks = document.querySelectorAll('nav a');

navLinks.forEach(link => {

if (link.textContent.includes("Аптеки")) {

link.addEventListener("click", (e) => {

e.preventDefault();

document.querySelector('.pharmacy-finder').scrollIntoView({ behavior: 'smooth' });

});

}

});

* Все ссылки в меню проверяются.
* Если текст ссылки содержит слово "Аптеки":
  + При клике отменяется переход по ссылке.
  + И запускается **плавная прокрутка** к блоку с классом .pharmacy-finder.

1. **JavaScript функции на странице каталогов**

Этот JavaScript-код отвечает за **фильтрацию и сортировку карточек товаров** на странице после загрузки HTML-документа.

### 1. DOMContentLoaded **— запуск кода после загрузки страницы**

document.addEventListener('DOMContentLoaded', function () {

// ... остальной код

});

Этот блок гарантирует, что скрипт выполнится **только после полной загрузки DOM** (структуры HTML-документа). То есть, все элементы уже будут на странице и доступны для взаимодействия.

### 2. **Переменные с элементами интерфейса**

const applyFiltersButton = document.querySelector('.filter-button');

const sortSelect = document.getElementById('sort-select');

* applyFiltersButton — кнопка "Применить фильтры".
* sortSelect — выпадающий список для сортировки товаров.

### 3. **Фильтрация товаров по кнопке**

applyFiltersButton.addEventListener('click', function (e) {

e.preventDefault();

...

});

Когда пользователь нажимает кнопку "Фильтры", выполняется функция, которая:

* **отменяет стандартное поведение** (например, если кнопка в <form> — предотвращает перезагрузку страницы),
* **собирает данные из фильтров** (цена, наличие, производитель, рецепт),
* **перебирает карточки товаров**, определяет, подходит ли каждая товарная карточка под заданные фильтры,
* **разделяет подходящие и неподходящие карточки**,
* **перестраивает содержимое контейнера** .product-grid, сначала показывая подходящие, затем остальные.

### 4. **Извлечение данных из фильтров**

const priceMin = parseFloat(...) || 0;

const priceMax = parseFloat(...) || Infinity;

* Получаются значения минимальной и максимальной цены из двух input'ов (по placeholder'у "от" и "до").
* Если пользователь ничего не ввёл, берётся значение по умолчанию: 0 и бесконечность.

Затем аналогично собираются значения галочек:

const inStockChecked = ...;

const byOrderChecked = ...;

// и т.д.

Каждая пара чекбоксов (например, «в наличии» и «под заказ») проверяется на выбранность.

### 5. **Проверка соответствия каждой карточки**

products.forEach(product => {

const price = parseFloat(product.dataset.price);

const availability = product.dataset.availability;

const producer = product.dataset.producer;

const prescription = product.dataset.prescription;

let match = true;

...

});

* Каждая карточка товара (.product-card) хранит данные в data-\* атрибутах (data-price, data-availability и т.д.).
* Проверяется: если товар не укладывается в фильтры, переменная match становится false.

Если товар удовлетворяет всем условиям (match === true), он добавляется в массив matched, иначе — в unmatched.

### 6. **Перестроение DOM: matched + unmatched**

container.innerHTML = '';

[...matched, ...unmatched].forEach(product => {

container.appendChild(product);

product.style.display = "block";

});

* Очищаем контейнер товаров (.product-grid),
* Сначала вставляем подходящие товары (matched), затем — неподходящие (unmatched),
* Все карточки видимы (даже неподходящие), но можно при желании скрывать unmatched (например, через product.style.display = "none").

### 7. **Обработка сортировки**

sortSelect.addEventListener('change', function () {

const sortType = sortSelect.value;

sortProducts(sortType);

});

Когда пользователь выбирает тип сортировки (например, «по возрастанию цены»), вызывается функция sortProducts.

### 8. **Функция сортировки** sortProducts

function sortProducts(sortType) {

const products = Array.from(container.querySelectorAll('.product-card'));

...

sortedProducts.forEach(product => {

container.appendChild(product);

product.style.display = "block";

});

}

* Собирает все карточки продуктов.
* В зависимости от значения sortType сортирует массив:
  + price-asc — по возрастанию цены,
  + price-desc — по убыванию.
* Перестраивает порядок карточек в контейнере.

1. **РЕЗУЛЬТАТ**

В ходе выполнения курсовой работы были изучены основные принципы создания веб-сайта:

1. Создана структура страницы с использованием HTML.
2. Выполнено оформление внешнего вида через CSS.
3. Реализована адаптивность сайта с помощью медиа-запросов для корректного отображения на разных устройствах.
4. Добавлен JavaScript для интерактивности: переключение вкладок, работа фотогалереи, фильтрация и обработка поиска.

В результате была разработана современная, удобная и функциональная веб-страница, соответствующая требованиям пользователей и современным стандартам веб-дизайна.

Ссылка на веб-страницу: <https://emil222222.github.io/UniProject/>

1. **ЛИТЕРАТУРА**
2. **Джон Дакетт — HTML и CSS. Разработка и дизайн веб-сайтов**
3. **Джон Дакетт — JavaScript и jQuery. Интерактивная веб-разработка**
4. **Эрик Фримен, Элизабет Робсон — Изучаем HTML, XHTML и CSS**
5. **Дэвид Флэнаган — JavaScript. Подробное руководство**
6. **Николас Закас — JavaScript для профессиональных веб-разработчиков**
7. **Марка Майерс — Изучаем JavaScript. Руководство для начинающих**
8. **Этан Брасс — Адаптивный веб-дизайн**
9. **Джеффри Зельдман — Дизайн веб-стандартов**
10. **Эрик Мейер — CSS. Дизайн веб-сайтов**
11. **Брюс Лоусон, Реми Шарп — HTML5. Разработка веб-приложений для современных браузеров**
12. **ПРИЛОЖЕНИЕ**

Репозиторий:

* <https://github.com/Emil222222/UniProject>

Страница сайта на хостинге GitHub Pages:

* [<https://emil222222.github.io/UniProject/>](https://medina003.github.io/Coursework/)