**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**Факультет: Информационные технологии и управление**

**Кафедра: Компьютерная инженерия**

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

Курсовая работа

**Тема** «Разработка системы загрузки программного обеспечения онлайн»

**Группа:** 680.20

**Курc:** 3

**Специальность: 050655** Информационные технологии

**Студент**: Дуньямалыев Джалил

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

**Зав.кафедрой: доц.Рагимова Н.А.**

**Баку – 2025**

**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** **050655** Информационные технологии

**Студент** Дуньямалыев Джалил

**Зачетная книжка**

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

**Срок выдачи**

**Дата сдачи**

**Тема курсовой работы** «Разработка системы загрузки программного

обеспечения онлайн »

**Отзыв руководителя курсовой**:

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

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

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

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

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

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

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

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

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

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

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

**СОДЕРЖАНИЕ**

**ВВЕДЕНИЕ………………………………………………………………………4**

1. **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ………...………………………………………6**
   1. **Web-программирование…………………………………………….……...6**
   2. **HTML и XHTML………………………………………………...…………..7**
   3. **Dynamic HTML……………………………………………………………..13**

**1.4.1 Каскадные таблицы стилей……………………………………...13**

**1.4.2 JavaScript…………………………………………………………...16**

**1.5 Bootstrap……………………………………………………………………..17**

1. **ПРАКТИЧЕСКАЯ ЧАСТЬ…………………………………………...……20**
   1. **Описание предметной области…………………………………………...20**
   2. **Создание окружения для проекта………………………………………..21**
   3. **Верстка HTML страницы…………………………………………………22**
   4. **Использование стилей CSS………………………………………………..45**

**2.5.Функциональность с помощью JS……………………………………….49**

**3.РЕЗУЛЬТАТ…………………………………………………………………..54**

**4.ПРИЛОЖЕНИЯ……………………………………………………………...55**

**ВВЕДЕНИЕ**

Веб-системы и технологии относятся к различным инструментам, языкам и платформам, которые используются для разработки, развертывания и обслуживания веб-приложений. Эти технологии развивались на протяжении многих лет, чтобы не отставать от растущего спроса на веб-сервисы, что привело к появлению разнообразных языков программирования, фреймворков, библиотек и баз данных. Когда появились Web-технологии, пользоваться компьютерной техникой начали абсолютно новые категории граждан. Список социальных категорий населения, подсоединяющихся к сети интернет с целью поиска информации во всемирной паутине, постоянно увеличивается за счёт людей, которые не считают себя специалистами в сфере информационных технологий. Web-технология в корне изменила понятия об информационной обработке, как и работе с компьютерной техникой тоже.

Оказывается стандартные характеристики, описывающие прогресс в сфере вычислительного оборудования, а именно, производительность, пропускная способность, объём памяти, практически не включали в свой состав главное системное «узкое место», именуемое пользовательским интерфейсом. Устаревшая техника взаимоотношений пользователя с информационной системой тормозила продвижение передовых технологий и нивелировала преимущества от их внедрения. И лишь когда интерфейс пользователя, обеспечивающий взаимодействие человека и компьютера, разработчики довели до простого и естественного понимания обычными людьми, произошёл невообразимый взрывной рост интереса к достоинствам и возможностям компьютерного оборудования.

Web-технологиями является весь набор средств, позволяющих организовать WWW (World Wide Web), то есть всемирную паутину. Так как каждый сеанс является взаимодействием двух сторон, а именно, сервера и клиента, то и Web-технологии делятся на следующие группы:

1. Технологии серверной стороны (server-side).
2. Технологии клиентской стороны (client-side).

Технологии клиентской стороны включают в свой состав весь набор технологий по созданию веб-страниц (HTML, JavaScript, DHTML и другие), а технологии серверной стороны состоят из технологий доступа к информационным базам данных в сети интернет (CGI, PHP).

Серверные программы обеспечивают предоставление тех или иных ресурсов клиентским программам. Клиенты, когда им требуется какой-либо файл или просто какая-то информация от сервера, вырабатывают специальный запрос клиента и отправляют его серверу. Серверная программа выполняет обработку запроса и отправляет ответ сервера, который содержит запрошенную информацию или же извещение об ошибке, в случае недоступности требуемых данных. Данная компьютерная организация, или по другому, принципы формирования вычислительных систем или сетей, именуется архитектурой «клиент-сервер» или двухзвенной организацией. Как раз на базе двухзвенной архитектуры работают практически все интернет - сервисы, включая и WWW.

Данная курсовая работа рассматривает пример создания информационной системы для продуктовых магазинов.  
Разработка онлайн-информационной системы будет включать в себя следующие этапы:

1. Сбор требований: Первым шагом будет сбор требований от клиента, таких как продукты, которые он хочет продавать, целевая аудитория, функции, которые он хочет видеть на веб-сайте.

2. Дизайн: Следующим шагом будет разработка дизайна веб-сайта, включая создание каркасов и макетов, выбор цветовых схем и шрифтов, а также определение общего макета и пользовательского интерфейса.

3. Разработка: Это предполагает создание веб-сайта с использованием HTML, CSS, JavaScript и других веб-технологий.

4. Тестирование: После завершения разработки веб-сайт необходимо будет протестировать, чтобы убедиться, что он должным образом функционирует в различных браузерах и устройствах.

WEB-сайт Интернет-магазина – это система, ориентированная большей частью на покупателя. Он должен предоставлять сервисы просмотра товара, удобного поиска товара, легкость и удобство при заказе товара.   
 И так, целью моей курсовой работы является создание сайта Интернет-магазина, обеспечивающего удобство, безопасность и интуитивную понятность интерфейса пользователю.

1. **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**1.1** **Web-программирование**

Web-программирование (Web-разработка) - это бурно развивающийся раздел программирования, ориентированный на разработку динамических Интернет-приложений. Языки Web-программирования делятся на две группы: клиентские и серверные. Клиентские языки обрабатываются на стороне пользователя (в основном в браузере). Соответственно, обработка скрипта зависит от браузера пользователя, и пользователь имеет полномочия настроить свой браузер так, чтобы тот вообще игнорировал скрипты. При этом если браузер старый, он может не поддерживать тот или иной язык или версию языка, на которую опирался разработчик. С современными браузерами таких проблем возникать не должно, к тому же языки программирования не так уж часто кардинально обновляются (раз в несколько лет) и лучшие из них давно известны. Код клиентского скрипта может посмотреть каждый, выбрав в меню своего браузера «Исходный код страницы». Преимущество клиентского языка заключается в том, что обработка скриптов на таком языке может выполняться без отправки документа на сервер. Программа сразу проверит правильное заполнение формы перед отправкой, и, если необходимо, выведет ошибку. Отсюда же вытекает и то ограничение, что с помощью клиентского языка программирования ничто не может быть записано на сервер. Самым распространенным из клиентских языков признан JavaScript, разработчиками которого является компания Netscape совместно с компанией Sun Microsystems. Еще один популярный язык - это VBScript. Помимо этого в последнее время набрали популярность такие технологии, как AJAX, Adobe Flash, Microsoft Silverlight и др. Серверные языки программирования открывают перед программистом большие просторы для деятельности. Когда пользователь делает запрос на какую-либо страницу (переходит на нее по ссылке, или вводит адрес в адресной строке своего браузера), то вызванная страница сначала обрабатывается на сервере (то есть выполняются все программы, связанные со страницей) и только потом возвращается к посетителю по сети в виде файла. Этот файл может иметь расширения: HTML, PHP, ASP, Perl, SSI, XML, DHTML, XHTML.

**1.2****HTML и XHTML**

HyperTextMarkupLanguage (HTML) - это язык разметки документов во Всемирной паутине, принятый за стандартный. Большая доля всех Web-страниц в Интернете создана при помощи языка HTML (или XHTML), поэтому мы рассмотрим его подробно.

Язык HTML позволяет форматировать текст и другие элемента Web-страницы:

*Цвет, жирность, стиль, название шрифта* для текста.

*Позволяет выделять фрагменты и символы* например: ударение в слове, заголовок страницы или абзаца, сам абзац, пункт списка.

*Гипертекстовые ссылки*, позволяют переходить между документами и между фрагментами одного документа.

*Формы* для введения данных, как правило, данные из форм обрабатываются с помощью скриптов на языках программирования, ориентированных на Web,например PHP.

*Отображение мультимедийных файлов*, их может отображать сам браузер - изображения, аудиофайлы, или внешние приложения, взаимодействующие с браузером, например Flash-ролики, Java-апплеты.

HTML - язык разметки документов основанный на тэгах. Документ на языке HTML представляет собой набор элементов, при этом начало и конец каждого элемента обозначается служебными символами - тегами. Все тэги HTML начинаются с «<« (левой угловой скобки) и заканчиваются символом «>« (правой угловой скобки). Завершающий тег выглядит также, как начальный, и отличается от него прямым слэшем перед текстом внутри угловых скобок.

**<HTML></HTML>**

HTML регистронезависимый язык, теги могут быть написаны как строчными, так и заглавными буквами (в отличие от XHTML). Теги могут быть вложенными друг в друга.

<**HTML>**

**<HEAD>**

**<TITLE>**

Заголовок страницы

**</TITLE>**

**</HEAD>**

**</HTML>**

Тэги могут быть пустыми, то есть не содержать текста или других вложенных конструкций (например, <**br>**который переводит строку). Закрывающий тег в таком случае не указывается.

Также, элементы разметки могут иметь атрибуты, задающие их свойства (например, размер шрифта, цвет, расположение). Атрибуты задаются в начале тега.

**<ahref=«**http://www.yandex.ru**»>**Пример элемента с атрибутом **href.</a>**

Теги можно разделить на следующие группы.

*Гиперссылки*

**<AHREF=«**filename**»target=«\_self»>**текст ссылки**</A>**

где filename - имя файла (может быть и локальным) или адрес страницы вInternet, на который нужно совершить переход.

текст ссылки - текст гипертекстовой ссылки, который будет отображаться в браузере, как правило, выделяется подчеркиванием.

target - задает окно или фрейм, в котором будет открыт документ, при переходе по ссылке. Он может принимать значения:

\_top - документ откроется в текущем окне

\_blank - документ откроется в новом окне

\_self - документ откроется в текущем фрейме

\_parent - документ откроется в родительском фрейме

По умолчанию принимает значение \_self.

*Текстовые ссылки.*

**<H1></H1>**,**<H2></H2>**, … ,**<H6></H6>** - заголовки 1-6 уровней. Применяются для выделения частей выводимого текста (заголовок 1 - будет выведен очень большим, 6 - будет размером сопоставимым с обычным текстом).

**<P>** - обозначает начало нового абзаца. Закрывающий тег**</P>**, не является необязательным.

**<BR>** - переход на новую строку. Закрывающий тег**</br>**, отсутствует.

**<HR>** - горизонтальная линия.

**<BLOCKQUOTE></BLOCKQUOTE>** - цитата. Выделение заданного текста как цитаты.

**<PRE></PRE>** - режим предпросмотра. При этом текст заключается в рамку и выводиться не форматированным (то есть все теги, кроме </PRE>, игнорируются, но при этом переводы строки ставятся там, где они присутствуют в исходном документе).

**<DIV></DIV>** - блок текста (как правило, применяется для использования каскадных стилей CSS).

**<SPAN></SPAN>** - строка (как правило, применяется для использования каскадных стилей CSS).

*Теги форматирования текста*

**<EM></EM>** - выделение символа, на который падает ударение (обычно отображается курсивом).

**<STRONG></STRONG>** - выделение символа, на который падает усиленное ударение (обычно отображается жирным текстом).

**<I></I>** - выделение текста курсивом.

**<B></B>** - выделение текста жирным шрифтом.

**<U></U>** - подчёркивание текста

**<S></S>** - зачёркивание текста.

**<STRIKE></STRIKE>** - то же самое, что <S> … </S>

**<BIG></BIG>** - увеличение шрифта.

**<SMALL></SMALL>** - уменьшение шрифта.

**<BLINK></BLINK>** - мигающий текст.

**<MARQUEE></MARQUEE>** - сдвигающийся по экрану текст.

**<SUB></SUB>** - вывод текста под строкой. Например, **H<SUB>2</SUB> О**тобразиться в виде текста H2O.

**<SUP></SUP>** - вывод текста над строкой. Например, **E=mc<SUP>2</SUP>**отобразиться в виде текста E=mc2.

**<FONT атрибуты></FONT>** - задание атрибутов у используемого шрифта. Атрибуты могут быть следующими:

**COLOR=color** - указание цвета. Цвет может быть указан шестнадцатеричным числом в формате #rrggbb (первые 2 шестнадцатеричные цифры указывают интенсивность красного, следующие 2 - зелёного, последние 2 - синего) или названием самого цвета.

FACE= указываем имя шрифта.

SIZE= позволяет изменить размер шрифта. Размеры могут быть от 1 до 7, по умолчанию размер 3.

SIZE=+ размер или SIZE=-размер - размер больше или меньше стандартного. Например,SIZE=+2 указывает размер на 2 больше стандарта, то есть размер 5.

*Списки.*

Данная конструкция

**<UL>**

<LI> первый элемент списка </LI>

<LI> второй элемент списка </LI>

<LI> третий элемент списка </LI>

</UL>

создаёт список вида:

- первый элемент;

- второй элемент;

- третий элемент.

Также стоит отметить, что тегов есть параметры, позволяющие менять вид списка.

*Объекты.*

EMBED - вставка объектов различных типов

APPLET - вставка Java-апплетов

SCRIPT - вставка различных скриптов, например JavaScript

*Изображения.*

IMG - тег для вставки изображения. Это не закрывающийся тег.

SRC - имя локального файла или путь к нему в виде URL

ALT - текст картинки (отобразится, в виде текста, если не удалось отобразить картинку)

TITLE - подсказка (показывается при попадании курсора в область картинки)

WIDTH, HEIGHT - размеры изображения (выводимое изображение будет масштабировано до указанных размеров)

ALIGN - обтекание текста

*Таблицы.*

TABLE - тег создание таблицы. Тег имеет следующие параметры:

BORDER - задает толщину границу таблицы

CELLSPACING - задает расстояние от ячейки до ячейки

CAPTION - задает заголовок таблицы (необязательный тег)

TR - добавление строки в таблицы

TH - задает заголовок столбца (необязательный тег)

TD - добавление ячейки таблицы

WIDTH, HEIGHT - размеры таблицы

*Формы.*Формы ввода данных могут быть самыми разнообразными. Поэтому рассмотрим только основные теги:

FORM - тег для создания формы

INPUT - добавление элемента ввода

TEXTAREA - добавление текстового поля

SELECT - добавление списка (как правило, это выпадающее меню)

OPTION - пункт списка

*Символы.*Некоторые символы не могут быть выведены напрямую. Для их вывода требуется использовать их определения, например, символ апмерсанд& в коде HTML будет иметь вид **&amp**, символ меньше < будет иметь вид **&lt**, символ больше > будет **&gt**. Это ограничение введено, так как эти символы уже используются в языке HTML как служебные.

Любая HTML-страница должна иметь обозначение начала и конца документа обрамленные тегами**<html>** и **</html>** соответственно. Внутри них должны находиться теги заголовка **<head>** и **</head>**, итеги, обозначающие тело документа **<body>** и **</body>**. А внутри них могут быть произвольные комбинации из групп тегов описанных ранее.

Также рассмотрим ExtensibleHypertextMarkupLanguage (XHTML) это расширяемый язык разметки гипертекста. Стоит отметить, что язык XHTML это ни описание самого языка, а список отличий XHTML от HTML. Рассмотрим основные отличия HTML и XHTML. В XHTML все используемые теги должны иметь закрывающий тег. Теги, не имеющие закрывающего тега должны оканчиваться символом /. Например тег **<br>**, должен иметь закрывающий его тег **<br />**. В XHTML допускается писать теги и их атрибуты только строчными буквами. В XHTML очень строгая проверка синтаксиса не допускается использовать< и &, даже в URL, вместо них должны быть **&lt** и **&amp**. Браузеры, обнаружив ошибку синтаксиса в XHTML, должны прекратить его обработку и вывести ошибку на экран. В стандарте HTML браузер должен попытаться отобразить запрашиваемый документ. Стоит отметить, что XHTML расширяемый язык - за счет указания типа документа и возможности использовать свои теги.

Как мы видим, язык разметки HTML предоставляет широкие возможности для отображения информации, для этого в нем содержится большое количество тегов для различного форматирования выводимой информации. Язык XHTML очень похож на HTML, но более строгий, грамматические правила в XHTML менее сложные, и как следствие при созданий Web-страниц будет меньше ошибок.

**1.3 Dynamic HTML**

Dynamic HTML или DHTML - так принято называть связку языка HTML, каскадных таблиц стилей, скриптового языка и объектной модели документов. Скриптовым языком может выступать JavaScript или VisualBasic, но именно первый язык получил большую популярность и сегодня используется повсеместно. При помощи DHTML можно создавать интерактивные Web страницы, он позволяет легко и гибко обрабатывать данные запроса и формировать динамический ответ. DHTML может быть использован для реализации интерфейса Drag'n'Drop. На его основе создаются игры и другие интерактивные сервисы. Стоит отметить, что для реализации принципов DHTML достаточно лишь браузера, который будет обрабатывать содержимое страницы. То есть, нет необходимости, например, в обращениях к базе данных.

***1.3.1******Каскадные таблицы стилей***

Рассмотрим подробнее каскадные таблицы стилей - CSS (CascadingStyleSheets). Это стандарт позволяющий задавать описание внешнего вида некоторых элементов страницы на HTML. CSS используется при создании Web-страниц для определения шрифта, цвета, расположения и прочих атрибутов, используемых в документе, элементов.

Основная цель использования CSS разграничить само содержимое страницы, созданное на языке HTML и описание оформления, которое написано на CSS. Такое разделение упрощает создание и изменение документа, документом более легко управлять, а также снижается избыточность в исходном коде. Также, при помощи CSS можно легко представить один и тот же документ, но с разным форматированием или использовать разные методы вывода, такие как отображение на экране, печать документа, чтение голосом и тому подобных.

Описание стиля на CSS при открытии страницы может быть взято из разных мест: оно может быть встроенным стилем - это блоки CSS внутри страницы на языке HTML.

**<style type=«**text/css**»>**

**body {**

**color:**green;

**}**

**</style>**

Из отдельного файла .css, в котором описывается таблица стиля, на этот файл делается ссылка внутри страницы:

**<link rel=«**stylesheet**» type=«**text/css**» href=«**style.css**» />**

Это может быть Inline-стиль, при этом в HTML документе информация о стиле элемента указывается как атрибут style.

**<p style=«**font-size: 21px; color: green;**»>**Выводимый текст**</p>**

Также в любом браузере есть свой стандартный стиль, используемый по умолчанию.

Таблица стилей это набор описаний форматов. Каждое правило стиля, в свою очередь, имеет один или несколько селекторов, которые разделяются запятыми. В любом стиле также должен присутствовать блок определений, который обозначается фигурными скобками { }, и включает в себя набор свойств и их значений.

селектор, селектор {

свойство1: значение;

свойство2: значение;

свойство3: значение;

}

В стандарте CSS присутствуют приоритеты, согласно которым применяются правила стилей, например, если для элемента подходят свойства сразу нескольких правил. Это и называется каскадом. Стоит обратить внимание и на порядок расположения применяемых свойств - у свойства, указанного позже, будет более высокий приоритет.

До создания CSS оформление стиля Web-страницы производилось непосредственно внутри самого содержимого документа. Однако с использованием CSS появилась возможность принципиально разделить содержания и описание стиля документа. Благодаря этому стало возможным лёгкое применение единого стиля при создании похожих документов, а также это оформления можно изменить легко и быстро. Появилась возможность создать несколько вариантов дизайна страницы для разных устройств, на котором ее можно просмотреть. Например, на мониторе дизайн страницы будет иметь большую ширину, при отправке документа на печать буду отбрасываться такие элементы как меню, а на КПК или сотовом будет масштабирование уменьшающее искажение текста страницы. За счет того, что таблицы стилей хранятся в отдельном файле CSS, уменьшается время загрузки страницы. В этом случае браузер получает только описание структуры документа и данные, находящиеся на Web-странице, а стиль оформления этих данных загружается браузером при первом обращении и потом берется из КЭШа. Последующее изменения дизайна сайта так же значительно упрощается. Нет необходимости изменять каждую страницу, лишь нужно внести изменения в CSS-файл. Но у CSS есть также и недостатки. Различное отображение вёрстки в различных браузерах, которые по-разному отображают одни и те же данные CSS.

Для работы с CSS есть большое количество различных программных продуктов, но они не получили большого распространения. Как правило, объем CSS, кода по соотношению с HTML, не такой большой. И почти все HTML-редакторы имеют поддержку синтаксиса CSS. Из отдельных решений можно выделить TopStyle, данный программный продукт предназначен для создания стилей CSS. Очень полезным в нем является модуль Stylechecker, с помощью него можно не только проверить правильность синтаксиса таблицы стилей, но и проверить формат представления данных в зависимости от браузера. Таким образом, при его помощи можно создавать стили, которые будут одинаково отображаться в разных браузерах. Также существуют визуальные CSS-редакторы, но из них сложно выделить какой-то наиболее достойный.

***1.3.2 JavaScript***

Несмотря на то, что JavaScript не является стандартом W3C, мы рассмотрим его в данном контексте, и как часть DHTML.

JavaScript - это объектно-ориентированный скриптовый язык программирования. JavaScript встраиваемый язык, используемый для доступа к объектам в приложениях. Он нашел очень широкое применение при создании Web-страниц. JavaScript очень похож на язык Си, но все же имеет кардинальные отличия: Структуру объектов и тип объектов (например переменных, функций) можно определить в процессе выполнения кода. В JavaScript действует автоматическое приведение типов данных. Также автоматической является контроль памяти и защита от утечек. Функции в языке программирования JavaScript могут быть без имени. Все описания переменных, функции и других элементов регистрозависимы, названия переменных могут содержать буквы, символ подчёркивания, символ доллара и арабские цифры. При этом названия переменных не должны начинаться с цифры. Стоит отметить, что JavaScript не поддерживает области видимости, не умеет работать с файловой системой и потоками ввода/вывода, не имеет сетевых интерфейсов. Это сделано для обеспечения безопасности, так как скрипт на языке JavaScript легко запустить на любом компьютере, для этого будет достаточным открыть на нем Web-страницу, содержащую скрипт. Также для повышения безопасности скрипт не может получить доступ к свойствам другой страницы.

Использовать JavaScript на странице, можно при помощи тегов**<script></script>.** Например:

<**script type=«**text/javascript**»>**

**alert ('**Hello, World!**');**

**</script>**

Также скрипт может быть сохранен в отдельном файле.

<scripttype=«text/javascript»src=http://Путь\_к\_файлу\_со\_скриптом**></script>**

Язык JavaScriptдает создателю страницы очень широкие возможности для выполнения скриптов и как следствие качественно новые горизонты для создания динамических элементов на Web-страницах. JavaScript позволяет автоматически заполнять данные в формах, изменять форматирование и стиль страницы. Скрывать ненужное содержимое и отображать нужное содержимое, менять поведение клиентской части Web-приложений, изменять элементы управления на странице и так далее. При этом язык довольно простой для понимания, и большая часть функции выполняется автоматически.

Выделить какой-то редактор для JavaScript сложно, но отметим что в среде Eclipse, есть поддержка JavaScript при помощи подключаемого модуля, при этом становиться доступен весь функционал среды Eclipse. А также практически все современные браузеры имеют встроенные средства для отладки JavaScript-скриптов.

* 1. **Bootstrap**

Bootstrap — это открытый и бесплатный [HTML-](https://blog.skillfactory.ru/glossary/html/), [CSS-](https://blog.skillfactory.ru/glossary/css/) и [JS-фреймворк](https://blog.skillfactory.ru/glossary/javascript/), который используют веб-разработчики для быстрой верстки адаптивных дизайнов сайтов и веб-приложений. Включает в себя CSS- и HTML-шаблоны оформления для веб-форм, меток, типографики, кнопок, блоков навигации и других компонентов веб-интерфейса.

Bootstrap используется, когда:

* у сайта много страниц;
* страницы собраны из простых базовых элементов — кнопок или таблиц;
* не будет глобального редизайна;
* шаблонность страниц окупается скоростью внедрения.

По сути, Bootstrap — это набор файлов. После их подключения к странице для верстки станет доступно большое количество готовых компонентов и классов. Они позволяют быстро и качественно создавать адаптивный дизайн сайта.

Классы в Bootstrap делятся на 3 большие группы:

* для создания сетки — адаптивного макета страницы;
* для стилизации контента — текста, изображений, кода, таблиц и прочей информации;
* служебные — для решения популярных вспомогательных задач, таких как отображение и скрытие элементов, выравнивание текста на странице, настройка цвета фона, отступов и пр.

Кроме классов, в Bootstrap есть компоненты (готовые объекты интерфейса). Это хлебные крошки (путь от начального элемента до уровня иерархии, который сейчас просматривает пользователь), кнопки, выпадающие списки и подсказки, модальные окна, формы, навигационные меню, всплывающие панели и пр.

Чтобы начать работать с Bootstrap, нужен текстовый редактор для работы с кодом (Visual Studio Code, Atom, Sublime Text и т.п.) или IDE — интегрированная среда разработки, а также браузер, в котором можно отслеживать изменения. Он должен быть обновлен до версии, которую поддерживает последний вариант Bootstrap.

Существует несколько способов установки Bootstrap. Получить актуальную версию фреймворка можно на [официальном сайте.](https://getbootstrap.com/)

1. Подключение скомпилированных файлов через [BootstrapCDN](https://www.bootstrapcdn.com/" \t "_blank) — общедоступную сеть доставки контента. Достаточно создать шаблон HTML и разместить внутри него ссылку на фреймворк.
2. Скачивание скомпилированных файлов CSS и JS с их подключением к проекту через ссылки.
3. Скачивание исходных файлов. Для работы потребуются [компилятор CSS](https://bootstrap-5.ru/docs/5.1/getting-started/contribute/#sass) и [автопрефиксатор](https://github.com/postcss/autoprefixer" \t "_blank).
4. Установка исходных файлов с помощью менеджеров пакетов yarn, npm, RubyGems, NuGet, Composer.

**2. ПРАКТИЧЕСКАЯ ЧАСТЬ**

**2.1 Описание предметной области**

**Предметная область** — это область знаний, охватываемая в рамках определённого рассуждения или научной теории. Также под предметной областью понимается определённая часть реального мира, подлежащая исследованию с целью последующего управления и, в конечном итоге, автоматизации.  
Предметной областью данной курсовой работы является **онлайн-информационная система для продуктовых магазинов**.

**Архитектура сайта** представляет собой структуру организации информации и системы навигации, направленных на обеспечение удобного доступа пользователей к необходимым данным.  
Грамотно спроектированная архитектура сайта способствует сокращению времени на поиск нужной информации, что положительно сказывается на удобстве использования и, как следствие, повышает популярность ресурса среди пользователей.

При разработке архитектуры сайта необходимо учитывать влияние структуры представленной информации на продвижение товаров и услуг в интернет-пространстве. На этапе создания новой структуры сайта или при её оптимизации важно обеспечить такое представление данных, которое бы способствовало увеличению посещаемости ресурса и акцентировало внимание потенциальных клиентов на ключевых разделах, соответствующих основным предложениям компании.

Одним из ключевых элементов интернет-магазина является **интерфейс**, так как именно через него осуществляется взаимодействие с пользователями. Удобное и логичное расположение информационных блоков на страницах сайта позволяет клиентам быстро находить интересующие их сведения о товарах или предоставляемых услугах.

В процессе разработки использовались следующие технологии:

* **HTML** — базовый язык веб-разметки, широко применяемый при создании сайтов благодаря своей простоте и универсальности.
* **CSS** — формальный язык описания внешнего вида веб-страниц, разработанных с использованием HTML, позволяющий задавать стиль и оформление элементов интерфейса.

**2.2** **Создание окружения для проекта**

В качестве редактора кода будет использован Visual Studio Code. Создаем папку в нем, которую назовем “ kursovaya”, в нем и будут все файлы проекта.

Сайт будет состоять из трех страниц. Создадим для нее файл index.html, about.html, software-details.html. После этого создаем файл для стилей “style.css”,для функциональности файл “main.js” и папку для изображений.

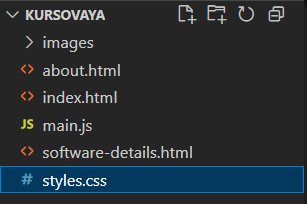
 Окружение готово и теперь мы можем приступить к самой разработке (рис. 2.1)

Рис. 2.1 Итоговый вид окружения

**2.3 Верстка HTML страницы**

Открываем ранее созданный “index.html” и пишем стандартный HTML5 код (рис. 2.2).

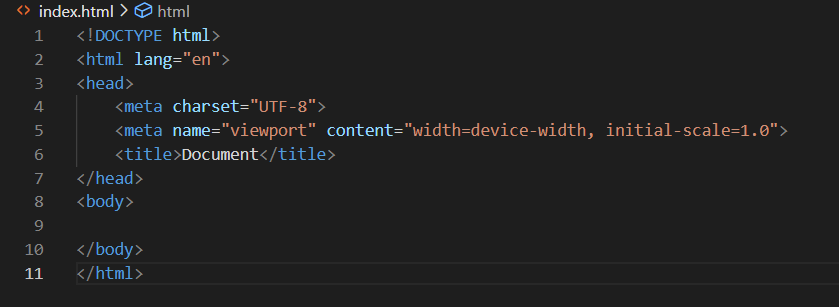


Рис. 2.2

Элемент <!DOCTYPE> используется для определения типа документа — DTD (описания структуры документа). Это необходимо для того, чтобы браузер знал, как правильно интерпретировать содержимое веб-страницы, поскольку HTML имеет несколько версий.

Тег <html> служит контейнером для всего содержимого веб-документа, включая разделы <head> и <body>. Хотя использование открывающего и закрывающего тегов <html> не является обязательным, хорошей практикой считается их всегда указывать.

Элемент <head> содержит метаинформацию о документе, предназначенную для обработки браузером, такую как заголовок страницы, ссылки на стили и скрипты.

Элемент <body> содержит основное содержимое страницы. В любом HTML-документе может быть только один такой элемент.

Подключение внешних файлов, таких как таблицы стилей, обычно осуществляется с помощью тега <link> внутри элемента <head> (см. рис. 2.3).

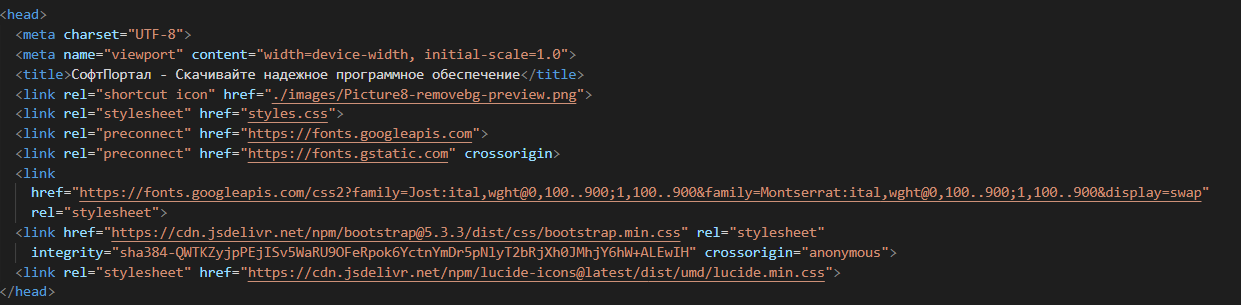


Рис. 2.3

Затем прописываем в <head> : <meta name="viewport" content="width=device-width, initial-scale=1> . Этот код задает параметры масштабирования страницы для устройств с разными размерами экранов. Атрибут content определяет метаданные, которые могут быть использованы браузером для корректного отображения содержимого в окне браузера на разных устройствах.

* Атрибут width=device-width задает ширину страницы равной ширине экрана устройства.
* Атрибут initial-scale=1 устанавливает начальный масштаб страницы, который будет использоваться при первом загрузке страницы.

Через rel="shortcut icon" ставим иконку в названии страницы :



Получаем такой результат(рис. 2.4):



Рис. 2.4

**Описание шапки сайта (Header)**

Шапка сайта (<header>) представляет собой верхнюю навигационную панель, обеспечивающую быстрый доступ к основным разделам веб-ресурса. Она выполнена с использованием современных стандартов HTML5 и CSS, что обеспечивает адаптивность и удобство взаимодействия с пользователем.

**1. Логотип и название сайта**

В левой части шапки расположен логотип с текстовым названием **«СофтПортал»**, который является кликабельной ссылкой на главную страницу (index.html).Логотип реализован с помощью элемента <a class="logo">.

<div class="header-left">

        <a href="index.html" class="logo">

          <span class="logo-text">СофтПортал</span>

        </a>

      </div>

**2. Основное навигационное меню**

Навигационное меню (<nav class="main-nav">) включает следующие пункты:

* **Главная** — возврат на стартовую страницу (index.html), текущая активная страница выделена классом active.
* **Категории** — ссылка на раздел с категориями программ (в текущей реализации заглушка #).
* **Новинки** — раздел с новыми программными продуктами (заглушка #).
* **Популярное** — раздел с популярными программами (заглушка #).

<nav class="main-nav">

        <a href="index.html" class="nav-link active">Главная</a>

        <a href="#" class="nav-link">Категории</a>

        <a href="#" class="nav-link">Новинки</a>

        <a href="#" class="nav-link">Популярное</a>

      </nav>

Меню реализовано с помощью семантического тега <nav>, что улучшает доступность для пользователей с ограниченными возможностями и поисковых систем.

**3. Поисковая строка**

* В правой части шапки (<div class="header-right">) расположен элемент поиска.
* Поле ввода (<input type="search">) с плейсхолдером *«Поиск программ...»* подсказывает пользователю его назначение.
* Иконка лупы (<img src="./images/search-icon.png">) визуально обозначает функцию поиска.

<div class="header-right">

        <div class="search-container">

          <input type="search" class="search-input" placeholder="Поиск программ...">

          <img src="./images/search-icon.png" class="search-icon">

        </div>

      </div>

После использования CSS стилей навбар примет вид(см рис. 2.5):

Рис. 2.5

**Описание секции (Hero Section)**

Секция (<section class="hero">) является ключевым элементом главной страницы, выполняющим маркетинговую и навигационную функции. Этот блок призван привлечь внимание пользователя и побудить его к взаимодействию с сайтом.

**1. Структура и содержимое**

Секция состоит из двух основных частей:

* **Текстовый блок (**<div class="hero-text">**)**:
  + Заголовок <h1> с основным предложением: "Скачивайте надежное программное обеспечение"
  + Описательный абзац <p> с уточнением преимуществ: "Безопасные и проверенные программы..."
  + Группа кнопок-призывов к действию (CTA):
    - Основная кнопка "Просмотреть категории" (класс primary)
    - Вторичная кнопка "Популярные программы" (класс outline)
* **Графический блок (**<div class="hero-image">**)**:
  + Изображение программного обеспечения (<img src="./images/logo2.png">)
  + Стилизовано с закругленными углами (border-radius: 30px)

<section class="hero">

      <div class="container">

        <div class="hero-content">

          <div class="hero-text">

            <h1>Скачивайте надежное программное обеспечение</h1>

            <p>Безопасные и проверенные программы для всех ваших потребностей. Быстрая загрузка, без вирусов.</p>

            <div class="hero-buttons">

              <a href="#" class="button primary">Просмотреть категории</a>

              <a href="#featured" class="button outline">Популярные программы</a>

            </div>

          </div>

          <div class="hero-image">

            <img src="./images/logo2.png" alt="Software Download Hero" style="border-radius: 30px;">

          </div>

      </div>

      </div>

    </section>

**2. Функциональные особенности**

* **Кнопки навигации**:
  + Первая кнопка ведет к разделу категорий (в текущей реализации заглушка #)
  + Вторая кнопка осуществляет плавный скролл к разделу популярных программ (якорь #featured)
* **Адаптивный дизайн**:
  + Предполагается (по названию классов), что блок использует flexbox или grid для корректного отображения на разных устройствах
  + Текстовая и графическая части меняют расположение при изменении размера экрана

Секция главной страницы сочетает информативность и функциональность. Центральный заголовок h1 формулирует основное предложение сервиса, сопровождаясь поясняющим текстом о преимуществах. Две кнопки-призыва к действию предлагают пользователю основные сценарии взаимодействия: изучение категорий или переход к популярным программам. Визуальная составляющая (логотип или тематическое изображение) дополняет текстовый контент, создавая целостное восприятие. Стилистическое оформление с закругленными углами соответствует современным тенденциям веб-дизайна.

В конце она будет выглядеть так(см рис.2.6):

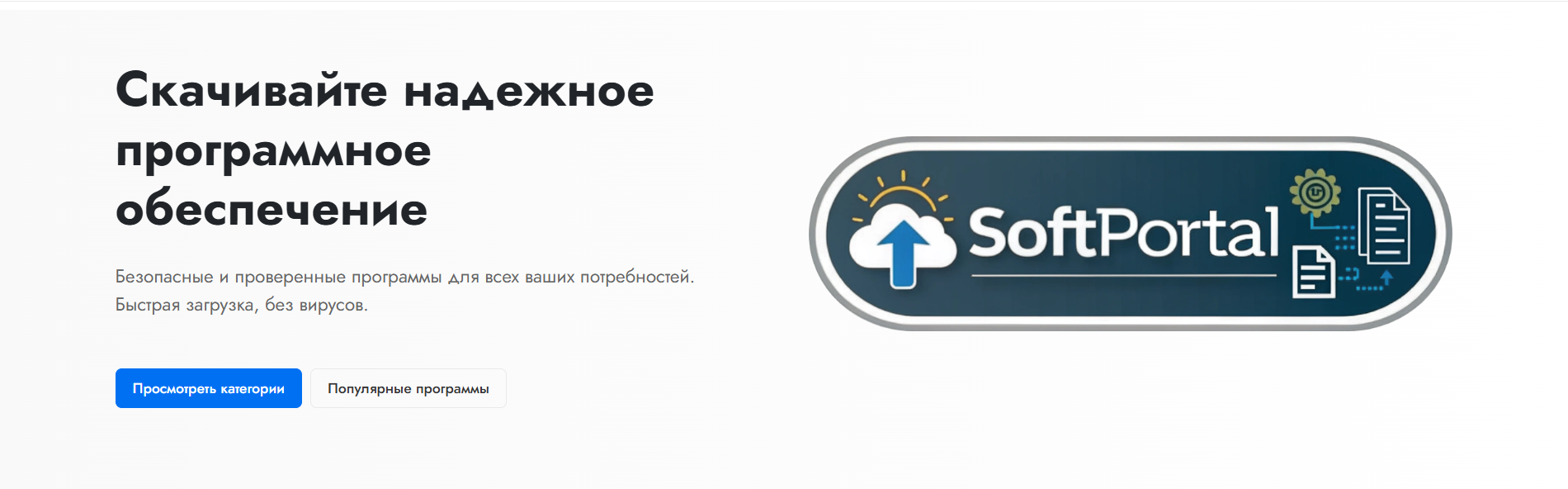


Рис. 2.6

**Пошаговое создание поискового блока на сайте**

Добавляем новый раздел сайта с помощью тега <section class="search-section">. Этот блок отвечает за привлечение внимания пользователя к поиску программного обеспечения прямо с главной страницы.

<section class="search-section">

    </section>

Внутри секции размещаем контейнер <div class="container">. Он ограничивает ширину содержимого и выравнивает элементы по центру страницы, обеспечивая аккуратный и читаемый внешний вид.

Внутри контейнера создаём блок <div class="search-content">, который включает в себя:

* Заголовок второго уровня <h2>, приглашающий пользователя начать поиск: **"Найдите нужное программное обеспечение"**.
* Краткое описание в виде абзаца <p>, информирующее о доступных более чем 10 000 программ для разных операционных систем (Windows, Mac, Linux).

Под описанием добавляем блок <div class="main-search">, внутри которого находятся:

* Поле ввода поиска <input type="search" class="main-search-input" placeholder="Поиск программ...">, где пользователь может ввести название интересующей его программы.
* Изображение иконки поиска <img src="./images/search-icon.png" class="main-search-icon">, визуально дополняющее поле поиска.

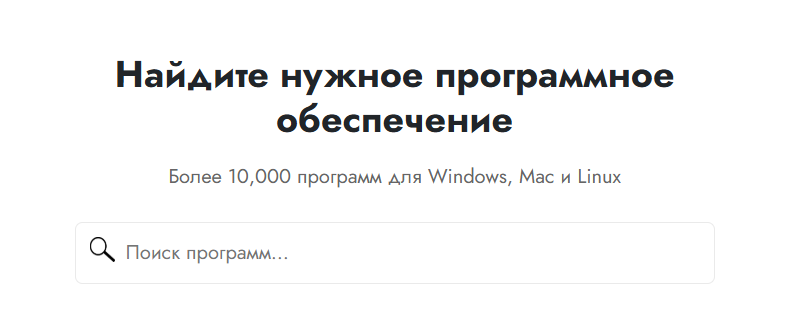


Рис. 2.7

**Создание секции популярных программ на сайте**

**Создание общей секции**

Размещаем новый блок <section id="featured" class="featured-section">, который представляет собой раздел сайта с подборкой популярных программ. Атрибут id="featured" позволяет быстро ссылаться на этот блок через якорные ссылки.

**Создание контейнера**

Внутри секции добавляем контейнер <div class="container">, который ограничивает ширину контента и выравнивает его по центру страницы, обеспечивая удобное восприятие информации пользователем.

Добавляем заголовок второго уровня <h2>, который кратко описывает содержание секции: **"Популярные программы"**.

**Создание сетки карточек программ**

Под заголовком размещаем блок <div class="software-grid" id="featured-software">, который организует карточки программ в виде сетки. Каждая программа представлена в отдельной карточке (<div class="card">).

**Структура карточки программы**

Каждая карточка включает в себя несколько элементов:

* **Изображение программы**: в блоке <div class="image-placeholder"> размещается картинка программы (<img src="..." alt="">), которая визуально представляет софт.
* **Основная информация**: внутри блока <div class="card-content">:
  + Название программы (<h3>).
  + Тег категории программы (<span class="tag">), например "Безопасность", "Графика и дизайн", "Мультимедиа".
  + Краткое описание возможностей программы (<p class="description">).
  + Блок рейтинга (<div class="rating">):
    - Звёздный рейтинг (<div class="stars">).
    - Оценка в числовом формате (<span class="score">).
    - Количество загрузок (<span class="downloads">).
  + Кнопка скачивания программы (<button class="download-btn">), иногда обёрнутая в ссылку (<a>), ведущую на страницу подробностей программы.

Результат(см рис. 2.8):

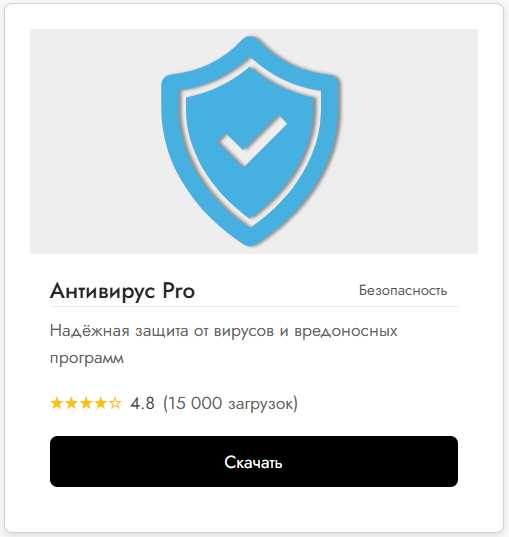


Рис. 2.8

Примеры представленных программ:

* **Антивирус Pro** — защита от вирусов.
* **Фото Редактор** — обработка изображений.
* **Видео Конвертер** — конвертация видеофайлов.
* **Офисный Пакет** — работа с документами.
* **Браузер Турбо** — быстрый и безопасный интернет-сёрфинг.
* **Архиватор Плюс** — эффективная работа с архивами.

В конце секции добавляется кнопка <a href="categories.html" class="button primary">Показать все программы</a>, которая позволяет пользователю перейти к полному списку программ на сайте.

После задания стилей получаем(см рис. 2.9):

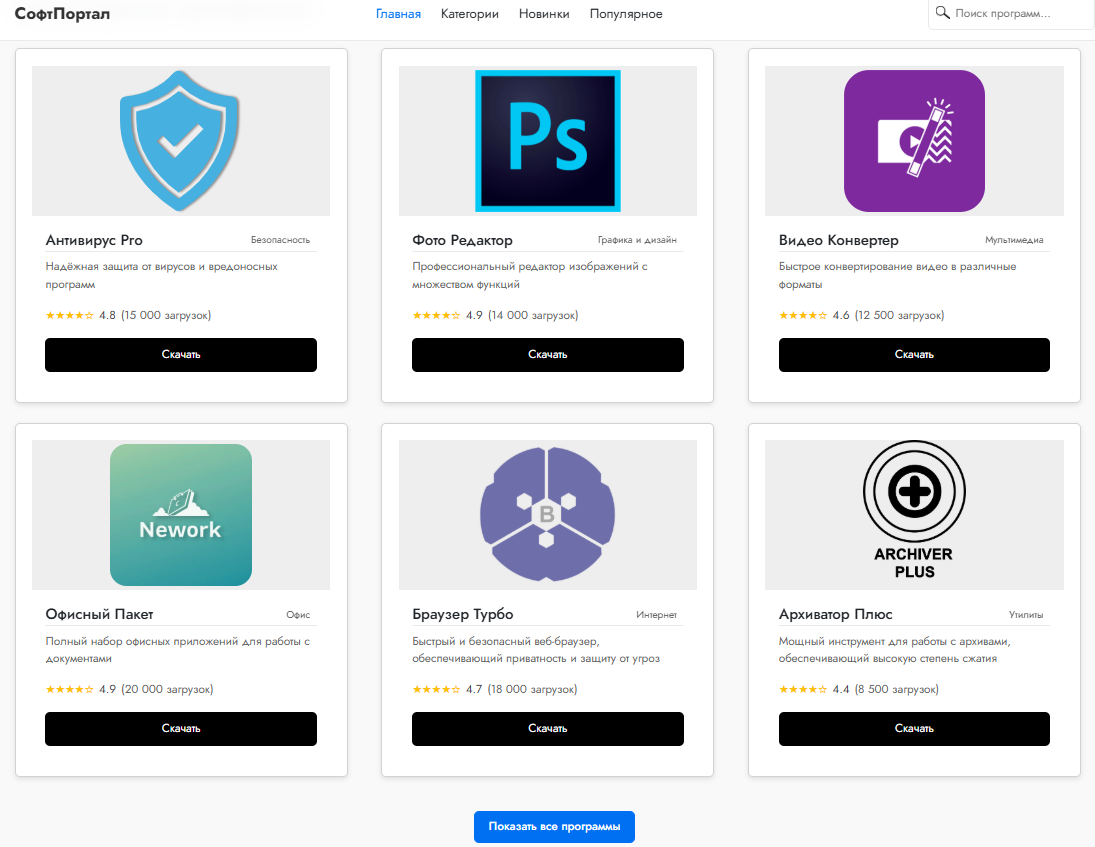


рис. 2.9

**Создание подвала сайта**

Создаём элемент <footer class="footer">, который предназначен для размещения информации о сайте в самом низу страницы. Классы помогают применить стили для подвала.

Внутри подвала создаём контейнер <div class="container footer-container">, чтобы:

* Ограничить ширину содержимого.
* Выстроить элементы подвала в две части: логотип и текст слева, ссылки справа.

В блоке <div class="footer-left"> размещаем:

* Логотип сайта в виде ссылки <a href="index.html" class="logo">, включающий:
  + Иконку загрузки <i data-lucide="download"></i>.
  + Название проекта <span class="logo-text">СофтПортал</span>.
* Текст авторских прав:

<p style="margin-bottom: 0px;">&copy; <span id="current-year"></span> СофтПортал. Все права защищены.</p>

Здесь тег <span id="current-year"></span> будет динамически подставлять актуальный год.

В блоке <div class="footer-links"> размещаем навигационные ссылки:

* "О нас" — ссылка на страницу с информацией о проекте.
* "Контакты" — форма обратной связи или контактные данные.
* "Конфиденциальность" — политика обработки персональных данных.
* "Условия использования" — пользовательское соглашение.

Все ссылки оформлены стандартным тегом <a>, чтобы сохранить простую навигацию.

Получаем(см рис. 2.10):

Рис. 2.10

Здесь наша первая страница завершена, позже мы зададим ему стили.

Начинаем верстку второй страниц, а именно файл “about.html”.Переход к нему возможен через нажатие на “О нас” в подвале прошлой страницы:





Шапку сайта создаем как на первой странице.

Далее создаем навбар:

<nav class="breadcrumbs">

                <ol class="breadcrumbs-list">

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

                    <li>/</li>

                    <li class="current">О нас</li>

                </ol>

            </nav>

И предоставляем общую информацию пользователю:

 <div class="about-header">

                <h1>О компании СофтПортал</h1>

                <p class="about-subtitle">Ваш надежный источник качественного программного обеспечения с 2010 года</p>

            </div>

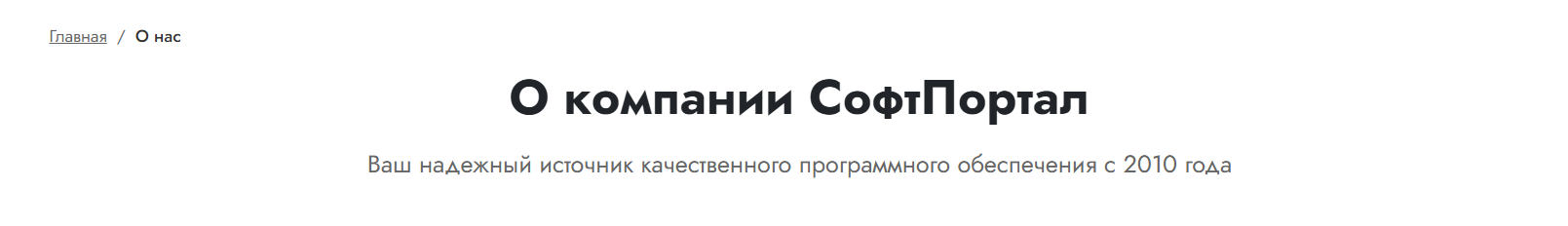


рис. 2.11

Создаем контейнер для контента с классом «about-content»

**Cоздание информационных секций "О нас"**

**Создание общей структуры**

Каждая текстовая секция оформлена в отдельном элементе <section class="about-section">, что позволяет:

* Логически разделить разные части информации о компании.
* Применять стили отдельно к каждому блоку.

**Разметка первой секции — "Наша миссия"**

В первой секции размещены:

* Блок с изображением <div class="about-image">, в котором содержится картинка офиса сайта:
* <img src="./images/ofis2.jpg" alt="Офис СофтПортал">
* Блок с текстом <div class="about-text">, содержащий:
  + Заголовок <h2>Наша миссия</h2>.
  + Два абзаца <p>, подробно описывающие цели проекта и принципы проверки программ.

Таким образом, текст и изображение располагаются рядом.

**Разметка второй секции — "История компании"**

Вторая секция имеет тот же принцип построения, но с добавлением класса reverse, который позволяет через CSS менять порядок отображения элементов (например, сначала текст, потом картинка или наоборот).

Внутри второй секции:

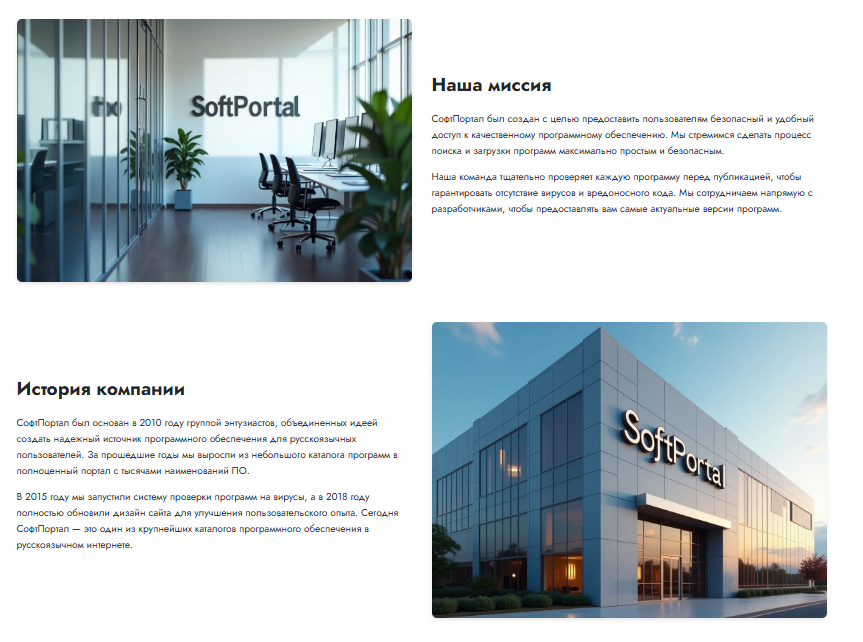
* Блок <div class="about-image"> с картинкой команды:
* <img src="./images/building.jpg" alt="Команда СофтПортал">
* Блок <div class="about-text"> с:
  + Заголовком <h2>История компании</h2>.
  + Двумя абзацами <p>, рассказывающими об основаниях и развитии проекта СофтПортал.

**Оформление через классы**

Классы about-section, about-image, about-text и reverse позволяют:

* Создать адаптивную верстку (удобную для разных экранов).
* Контролировать расположение текста и изображений.
* Выравнивать контент, добавлять отступы и стилизовать внешний вид секций.

**Итог(см рис. 2.12):**



**Рис. 2.12**

**Пошаговое создание секций "Наши ценности", "Наша команда" и "СофтПортал в цифрах"**

**Секция 1. Наши ценности**

**Основная структура**

* Вся секция обёрнута в <section class="about-values">, чтобы задать общие стили для фона, отступов и выравнивания.

**Заголовок секции**

* Добавлен заголовок второго уровня <h2>Наши ценности>, который сразу сообщает о содержании.

**Сетка карточек**

* Карточки ценностей размещены в контейнере <div class="values-grid">.
* Для каждой ценности создана отдельная карточка <div class="value-card">, внутри которой:
  + Иконка в блоке <div class="value-icon"> с изображением.
  + Подзаголовок <h3> с названием ценности.
  + Короткий текст в абзаце <p>.

Таким образом, каждая ценность красиво и компактно оформлена.

**Секция 2. Наша команда**

**Основная структура**

* Секция начинается с <section class="about-team">.

**Заголовок**

* Заголовок второго уровня <h2>Наша команда> представляет секцию.

**Сетка участников**

* Используется контейнер <div class="team-grid">, чтобы аккуратно разместить профили сотрудников в ряд.

**Карточки участников**

* Каждый член команды представлен в блоке <div class="team-member">, содержащем:
  + Фотографию в <div class="team-photo">.
  + Имя участника <h3>.
  + Должность <p class="team-role">.
  + Краткое описание задач <p class="team-description">.

Карточки визуально унифицированы, что делает блок "Наша команда" аккуратным и привлекательным.

**СофтПортал в цифрах**

**Основная структура**

* Вся секция помещена в <section class="about-stats">.

**Заголовок**

* <h2>СофтПортал в цифрах> привлекает внимание к статистике проекта.

**Сетка статистики**

* Блок <div class="stats-grid"> организует статистические карточки.

**Карточки статистики**

* Каждая карточка оформлена в <div class="stat-card">, внутри которой:
  + Крупное число или показатель <div class="stat-number">.
  + Подпись к числу <div class="stat-label">.

Статистика визуально подчеркивает масштаб и успех проекта.

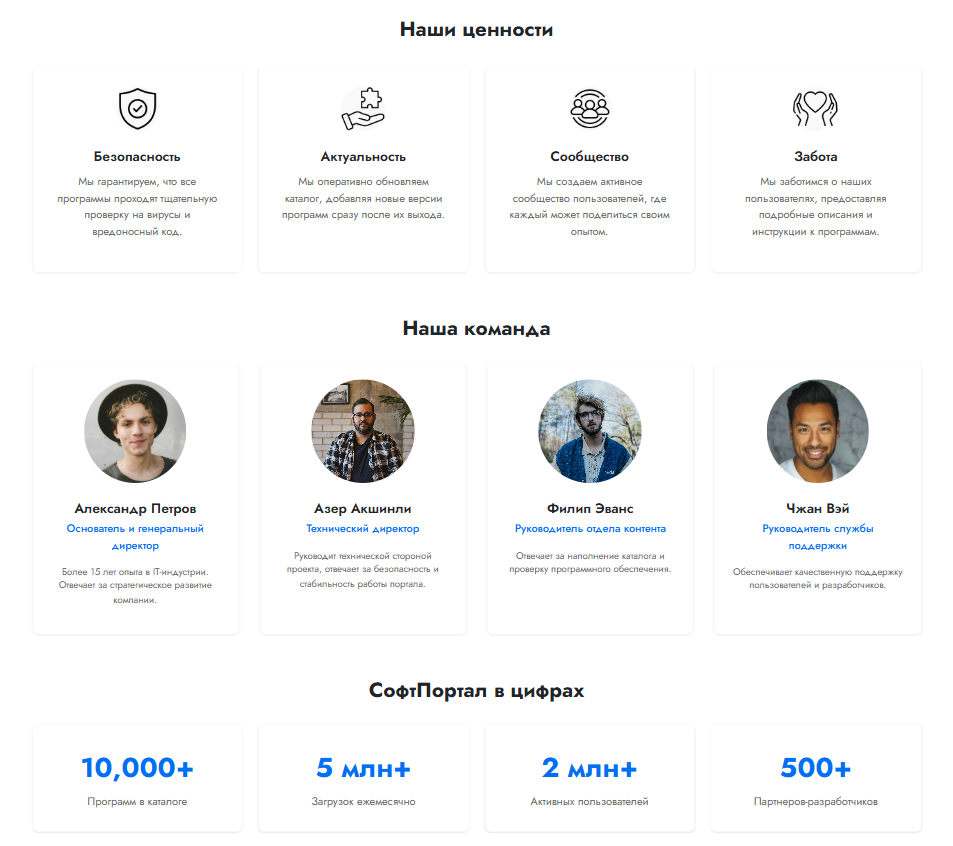


рис. 2.13

**Секция «Свяжитесь с нами»**

**Шаг 1. Основная структура**

* Секция обёрнута в тег <section class="about-contact">.
* Это позволяет отдельно оформить фон, отступы и выравнивание элементов в этой части страницы.

**Шаг 2. Заголовок секции**

* Добавлен заголовок <h2>Свяжитесь с нами>.
* Заголовок сразу дает понять пользователю, о чём эта часть сайта.

**Шаг 3. Основное содержимое: сетка контактов**

Контент разбит на две большие колонки с помощью контейнера <div class="contact-grid">:

**Левая часть — контактная информация:**

* Внутри блока <div class="contact-info"> перечислены способы связи:
  + **Email** с двумя почтовыми адресами.
  + **Телефон** с основным и бесплатным для Азербайджана номерами.
  + **Адрес** офиса с встроенной картой Google Maps (<iframe>).

Для каждого контакта используется:

* Блок <div class="contact-item">, в который входят:
  + Иконка <i data-lucide="..."> (будет заменяться на красивые SVG-иконки через библиотеку Lucide).
  + Название контакта (<h3>) и описание (<p>).

**Вставка карты:**

* Для адреса в контактах подключена карта через тег <iframe>.
* Карта на странице помогает пользователю сразу увидеть местоположение офиса.

**Шаг 4. Правая часть — форма обратной связи**

* В блоке <div class="contact-form"> размещена форма <form> для отправки сообщений.
* Форма содержит:
  + Поле для имени (input type="text").
  + Поле для email (input type="email").
  + Поле для темы сообщения (input type="text").
  + Большое поле для текста сообщения (<textarea>).

Каждое поле обёрнуто в <div class="form-group"> для удобной стилизации.

* Кнопка отправки формы оформлена через <button type="submit" class="button primary">Отправить сообщение</button>.

**Описание структуры подвала сайта**

**1. Основная обёртка**

* Подвал размещен внутри тега <footer class="footer">.
* Внутри — контейнер <div class="container footer-container">, чтобы контент был аккуратно выровнен и имел одинаковые отступы по ширине сайта.

**2. Левая часть подвала — логотип и копирайт**

В <div class="footer-left"> размещено:

* **Логотип сайта**:
* <a href="index.html" class="logo">
* <span class="logo-text">СофтПортал</span>
* </a>

Ссылка на главную страницу с текстом "СофтПортал".

* **Копирайт**:
* <p style="margin-bottom: 0px;">&copy; <span id="current-year"></span> СофтПортал. Все права защищены.</p>

Используется тег <span id="current-year">, чтобы при помощи JavaScript автоматически подставлять текущий год.

**3. Правая часть подвала — навигационные ссылки**

В <div class="footer-links"> находятся ссылки:

* **О нас** — с классом active, чтобы подсвечивать её как текущую страницу (если пользователь находится на about.html).
* **Контакты**, **Конфиденциальность**, **Условия использования** — пока ведут на заглушки (#), но при желании можно будет подключить настоящие страницы.

**Почему так сделано:**

* **Логотип** позволяет вернуться на главную страницу с любого места сайта.
* **Копирайт** демонстрирует актуальность сайта и защищенность его содержимого.
* **Навигация в футере** помогает пользователю быстро найти важную информацию, даже если он доскроллил до самого низа страницы.

Теперь приступим к третьей странице, название которого – “software-details.html”.

Шапка у него такая же как в первых двух как подвал и навбар, поэтому рассмотрим внутреннюю часть.

Создаем контейнер с классом “container software-details-container” и размещаем внутри него элементы.

**Описание раздела страницы программного обеспечения**

**1. Навигация «хлебные крошки» (breadcrumbs)**

<nav class="breadcrumbs">

<ol class="breadcrumbs-list">

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

<li>/</li>

<li><a href="#">Категории</a></li>

<li>/</li>

<li><a href="#">Безопасность</a></li>

<li>/</li>

<li class="current">Антивирус Pro</li>

</ol>

</nav>

**Что это такое?**  
Хлебные крошки показывают путь пользователя на сайте (иерархию страниц).  
**Зачем?**  
— Повышают удобство навигации.  
— Позволяют быстро вернуться на предыдущие уровни.

**2. Заголовок программы + основная информация (software-header)**

**Левая часть — изображение программы:**

<div class="software-image">

<img src="./images/Antivirus2.svg" alt="Антивирус Pro">

</div>

Показывается иконка/картинка приложения, чтобы пользователь визуально узнал продукт.

**Правая часть — текстовая информация:**

<div class="software-info">

**Внутри**:

* **Название и категория программы**:
* <div class="software-title-container">
* <h1>Антивирус Pro</h1>
* <span class="badge">Безопасность</span>
* </div>

— Заголовок H1 для SEO  
— Небольшой "бейджик" (значок категории)

* **Рейтинг и количество загрузок**:
* <div class="software-rating">
* <div class="stars">★★★★☆</div>
* <span class="rating-value">4.8</span>
* <span class="downloads">(15,000 загрузок)</span>
* </div>

— Отображается средний рейтинг в звездах  
— Числовая оценка и количество загрузок — повышают доверие пользователей.

* **Описание программы**:
* <p class="software-description">
* Надежная защита от вирусов и вредоносных программ...
* </p>

Кратко объясняется суть программы, её основная польза.

* **Кнопки действий**:
* <div class="software-actions">
* <button class="button primary download-button">Скачать</button>
* <button class="button outline">Добавить в избранное</button>
* </div>

— **Скачать** (основная кнопка, привлекает внимание).  
— **Добавить в избранное** (позволяет пользователю сохранить программу в личном списке).

Результат(см рис. 2.14):

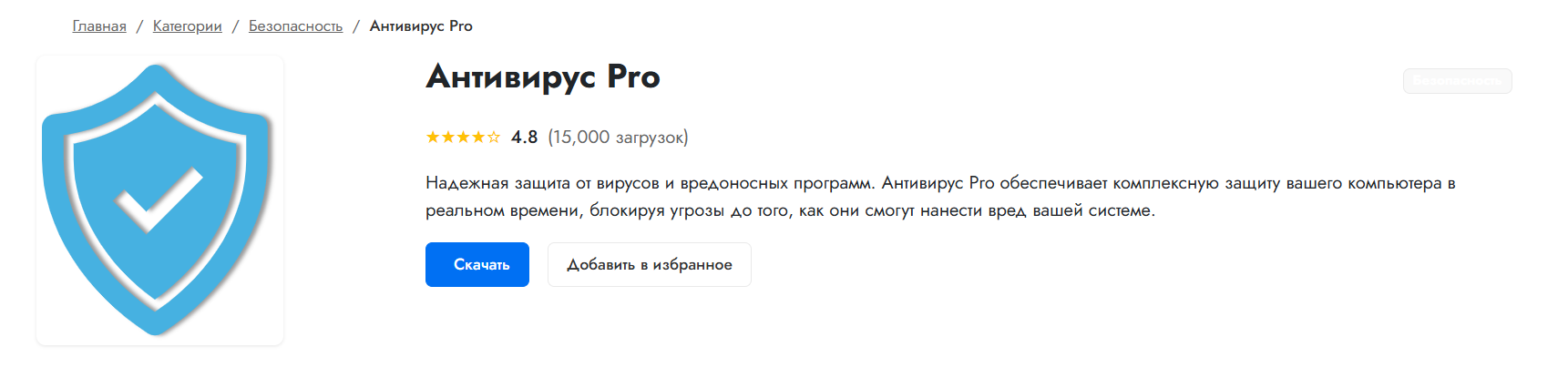


рис. 2.14

Описание компонента "Вкладки" для страницы программного обеспечения

**1. Общая структура**

Компонент представляет собой систему вкладок для отображения различной информации о программном обеспечении. Состоит из двух основных частей:

- Панель переключения вкладок (таб-бар)

- Контентные области для каждой вкладки

**2. Элементы управления (таб-бар)**

```html

<div class="tabs-list">

<button class="tab-button active" data-tab="details">Детали</button>

<button class="tab-button" data-tab="screenshots">Скриншоты</button>

<button class="tab-button" data-tab="requirements">Требования</button>

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

</div>

```

- Четыре кнопки-вкладки с соответствующими названиями

- Атрибут `data-tab` связывает кнопку с контентом

- Класс `active` отмечает текущую активную вкладку ("Детали" по умолчанию)

**3. Контентные области**

Вкладка "Детали" (активная по умолчанию)

Содержит три блока информации:

1. Карточка "Информация:

- Версия программы (12.5.1)

- Разработчик (Security Solutions Inc.)

- Дата выпуска (2023-10-15)

- Размер дистрибутива (85 MB)

- Тип лицензии (Платная с пробным периодом)

2. Карточка "Поддерживаемые ОС:

- Windows 10/11

- Windows 8

- Windows 7

(с иконками подтверждения)

3. Карточка "Возможности:

- 6 ключевых функций антивируса

- Каждая функция отмечена иконкой подтверждения

Результат(см рис. 2.15):

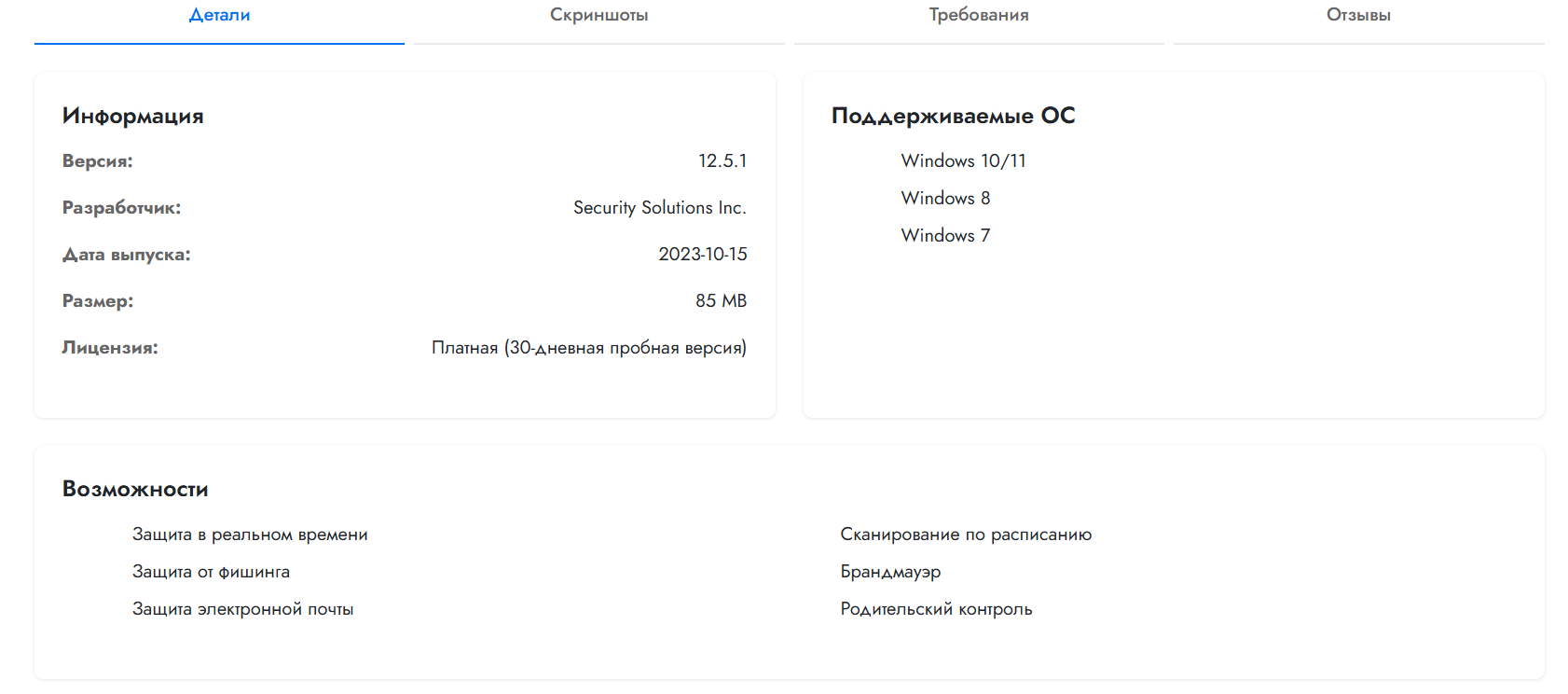


рис. 2.15

**2.4 Использование стилей CSS**

**Описание стилей в CSS файле (styles.css)**

1. Переменные CSS (Custom Properties):

- Определены переменные для цветов, фона, теней и других свойств

- Есть два набора переменных: светлая и темная темы (через класс `.dark-theme`)

- Включают основные цвета (`--primary-color`, `--text-color`), цвета состояний (`--success-color`, `--warning-color`)

2. Сброс стилей:

- Обнуление margin и padding для всех элементов

- Установка `box-sizing: border-box` для корректного расчета размеров

- Базовые настройки шрифта и цвета текста

3. Основные стили body:

- Минимальная высота 100vh

- Flex-верстка для прижатия футера к низу

- Плавные переходы для смены темы

**Компоненты**

Шапка сайта (Header)

- Фиксированное позиционирование (sticky)

- Размытый фон (backdrop-filter)

- Гибкая верстка с использованием flexbox

- Адаптивное меню (скрывается на мобильных устройствах)

- Поисковая строка с иконкой

- Переключение темы (светлая/темная)

Подвал сайта (Footer)

- Простая flex-верстка

- Адаптивное расположение элементов (столбец на мобильных, строка на десктопе)

- Ссылки на основные страницы

Кнопки

- Два основных стиля: `.primary` и `.outline`

- Эффекты при наведении

- Поддержка иконок

Карточки программ

- Тень и скругленные углы

- Адаптивная сетка (1-2-3 колонки в зависимости от ширины экрана)

- Рейтинг в виде звезд

- Кнопка скачивания

- Категории в виде бейджей

Герой-секция (Hero)

- Градиентный фон

- Адаптивное расположение текста и изображения

- Кнопки-призывы к действию

Поисковая секция

- Центрированный поиск с иконкой

- Адаптивный размер поля ввода

Страницы

Главная страница

- Сетка популярных программ (`.software-grid`)

- Стили для карточек программ (`.card`)

- Адаптивное количество колонок (1-2-3)

Страница программы (software-details.html)

- Хлебные крошки (`.breadcrumbs`)

- Табы (вкладки) для разной информации

- Детали программы в виде карточек

- Список возможностей

- Скриншоты в адаптивной сетке

Адаптивность

- Медиа-запросы для разных размеров экрана (мобильные, планшеты, десктопы)

- Изменение количества колонок в сетках

- Скрытие/показ элементов в зависимости от ширины экрана

- Адаптивные изображения (max-width: 100%)

Анимации

- Простая анимация появления (fadeIn)

- Плавные переходы для hover-эффектов

Неиспользуемые стили

1. Стили для мобильного меню (`.mobile-menu`, `.mobile-nav`) - не используются, так как нет соответствующей разметки

2. Некоторые компоненты пагинации (`.pagination-button`) - не используются в текущих страницах

3. Стили для выпадающих списков (`.dropdown`) - не используются в текущей реализации

4. Некоторые вспомогательные классы (`.sr-only`) - не применяются

Особенности реализации

1. Использование CSS Grid для сложных сеток (ценности, команда, статистика)

2. Flexbox для простых выравниваний

3. Кастомные свойства для легкой смены темы

4. Плавные переходы для интерактивных элементов

5. Современные свойства CSS (backdrop-filter для размытия фона)

**2.5.Функциональность с помощью JS**

**Описание JS-файла (main.js)**

Основные компоненты и функциональность

1. Тестовые данные

```javascript

const softwareProducts = [...];

```

- Массив объектов с данными о программных продуктах

- Каждый объект содержит:

- `id`: уникальный идентификатор

- `title`: название программы

- `description`: краткое описание

- `category`: категория

- `image`: путь к изображению

2. Инициализация поиска

```javascript

document.addEventListener('DOMContentLoaded', () => {

initRealTimeSearch();

});

```

- Запуск функции `initRealTimeSearch()` после полной загрузки DOM

- Основная точка входа для поисковой функциональности

3. Функция initRealTimeSearch()

Основные задачи:

- Находит все поля поиска на странице

- Добавляет обработчики событий

- Создает контейнеры для результатов

- Управляет отображением результатов

Ключевые элементы:

3.1 Обработчики событий:

```javascript

searchInputs.forEach(input => {

input.addEventListener('input', (e) => {...});

input.addEventListener('keypress', (e) => {...});

});

```

- `input`: Обработка ввода в реальном времени

- `keypress`: Обработка нажатия Enter

3.2 Создание контейнера результатов:

```javascript

const resultsContainer = document.createElement('div');

resultsContainer.className = 'search-results-container';

```

3.3 Обработка кликов вне поля поиска:

```javascript

document.addEventListener('click', (e) => {

// Скрывает результаты при клике вне поля поиска

});

```

4. Функция performRealTimeSearch()

Основная логика поиска:

```javascript

function performRealTimeSearch(query, inputElement) {

// Фильтрация результатов

const results = softwareProducts.filter(product =>

product.title.toLowerCase().includes(query.toLowerCase()) ||

product.description.toLowerCase().includes(query.toLowerCase()) ||

product.category.toLowerCase().includes(query.toLowerCase())

);

// Генерация HTML результатов

// Отображение/скрытие контейнера

}

```

Особенности фильтрации:

- Поиск по трем полям: название, описание и категория

- Нечувствительность к регистру (toLowerCase())

5.Генерация HTML результатов

Динамическое создание элементов:

```javascript

let html = '<ul class="search-results-list">';

results.forEach(product => {

html += `

<li class="search-result-item">

<a href="software-details.html?id=${product.id}">

<div class="search-result-image">

<img src="${product.image}" alt="${product.title}">

</div>

<div class="search-result-info">

<div class="search-result-title">${product.title}</div>

<div class="search-result-category">${product.category}</div>

</div>

</a>

</li>`;

});

html += '</ul>';

```

6.Динамическое добавление стилей

Функция `addSearchStyles()`:

```javascript

function addSearchStyles() {

const style = document.createElement('style');

style.textContent = `...`;

document.head.appendChild(style);

}

```

- Создает элемент `<style>` с CSS для результатов поиска

- Добавляет стили непосредственно в head документа

7. Особенности реализации

1. Универсальность:

- Работает с любым количеством поисковых полей

- Поддерживает классы `.search-input` и `.main-search-input`

2. Производительность:

- Использует делегирование событий

- Оптимизированная фильтрация данных

3. Безопасность:

- Экранирование данных через шаблонные строки

- Проверка существования изображения (использует fallback через ||)

4. Доступность:

- Правильная семантика HTML (списки, ссылки)

- Поддержка клавиатурной навигации (Enter)

8. Неиспользуемые возможности

- Обработчик Enter (строка 43) содержит закомментированный код для перенаправления

- Возможность расширения для работы с сервером (AJAX-запросы)

9. Архитектурные особенности

- Чистая функция фильтрации

- Разделение логики и представления

- Инкапсуляция стилей поиска

- Независимость от конкретной реализации HTML

Результат JavaScript-кода :

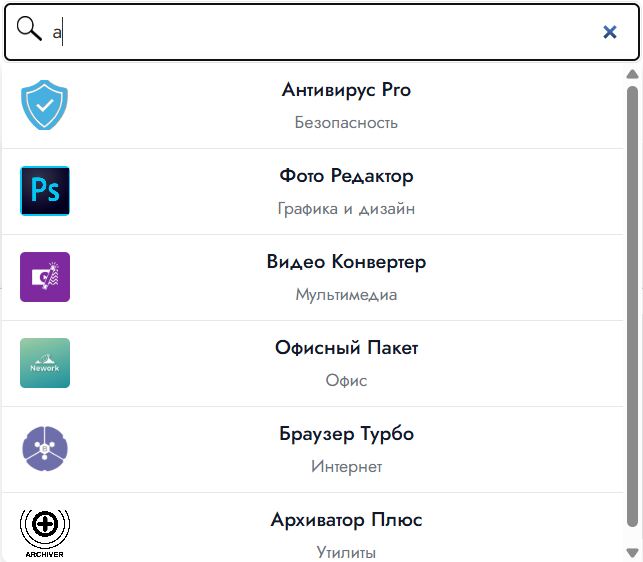


Рис. 2.16

**3. РЕЗУЛЬТАТ**

За время выполнения курсовой работы были изучены принципы разработки web-сайта:

1. Мы создали веб-страницу используя HTML.
2. Стили выполнены с использованием CSS
3. Создана адаптивная веб-страница с использованием медиа-запросов.
4. Добавлена функциональность с помощью Javascript

В результате была создана современная, функциональная и эстетичная веб-страница, которая удовлетворяет требованиям заказчиков, удобна для пользователей и соответствует современным трендам в веб-дизайне.

Ссылка на веб-страницу: <https://gleeful-zabaione-0a0a71.netlify.app/software-details>

**4. ПРИЛОЖЕНИЯ**

Репозиторий:

* <https://github.com/celil-dunya2005/Kursovaya>

Страница сайта на хостинге GitHub Pages:

* <https://gleeful-zabaione-0a0a71.netlify.app/software-details>