Министерство науки и высшего образования РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«**СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ**»

Институт управления бизнес-процессами

Кафедра «Бизнес информатика и моделирование бизнес-процессов»

**КУРСОВАЯ РАБОТА**

«Создание web-приложения с применением современных web-технологий по тематике «Рецепты по ингредиентам»

Руководитель \_\_\_\_\_\_\_\_\_\_ Никитин А.И.

подпись, дата

Студент УБ22-08Б, 432214925 \_\_\_\_\_\_\_\_\_\_ Плясун Е.П.

подпись, дата

Красноярск 2023

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc156048908)

[1 Теоретическая часть 4](#_Toc156048909)

[1.1 HTML 4](#_Toc156048910)

[1.2 CSS 8](#_Toc156048916)

[1.3 JavaScript 9](#_Toc156048917)

[1.4 PHP 13](#_Toc156048920)

[1.5 MySQL 14](#_Toc156048922)

[2 Практическая часть 16](#_Toc156048923)

[2.1 Требования и описание реализации 16](#_Toc156048924)

[2.2 Реализация проекта 17](#_Toc156048925)

[ЗАКЛЮЧЕНИЕ 27](#_Toc156048932)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 28](#_Toc156048933)

# ВВЕДЕНИЕ

В современном мире web-разработка играет ключевую роль в сфере информационных технологий. Это обосновывается тем, что web-приложения и сайты стали неотъемлемой частью жизни каждого человека. Web-программирование позволяет создавать интерактивные и динамичные web-страницы, обеспечивая пользователей удобным доступом к различным сервисам и контенту через Интернет.

Каждый человек хотя бы раз в жизни сталкивался с такой проблемой: необходимо приготовить обед или ужин, но он не знает, что именно можно сделать из имеющихся дома продуктов. Или же человек знает, что хочет приготовить, но не знает точный рецепт блюда. В этих случаях большинство людей привыкло использовать Интернет, ведь там можно найти ответ на любой вопрос. Как правило, в таких ситуациях люди делают выбор в пользу сайта, посвященному кулинарии.

В данной работе необходимо создать web-сайт, позволяющий по определенным продуктам получать рецепты. Пользователи могут размещать рецепты с указанием состава продуктов, которые имеются в системе. Также они могут выбрать определенные продукты и сервис должен показывать, что из этого можно приготовить. Под каждым рецептом можно оставлять отзывы. Продукты в систему заносятся администратором.

Целью данной работы является разработка сайта с использованием технологий HTML, CSS, Javascript, PHP, MySQL согласно теме.

Задачи данной работы:

1. ознакомление с технологиями HTML, CSS, Javascript, PHP, MySQL;

2. создание web-сайта с применением изученных технологий.

Объектом исследования является web-приложение с применением различных технологий.

Предметом исследования является процесс создания web-приложения, базы данных.

# 1 Теоретическая часть

## HTML

HTML (Hypertext Markup Language) - это код, который используется для структурирования и отображения веб-страницы и её контента. Например, контент может быть структурирован внутри множества параграфов, маркированных списков или с использованием изображений и таблиц данных.

HTML не является языком программирования, это язык разметки, и используется, чтобы сообщать браузеру, как отображать веб-страницы. HTML состоит из ряда [элементов](https://developer.mozilla.org/ru/docs/Glossary/Element), которые используются, чтобы вкладывать или оборачивать различные части контента, чтобы заставить контент отображаться или действовать определённым образом. Ограждающие [теги](https://developer.mozilla.org/ru/docs/Glossary/Tag) могут сделать слово или изображение ссылкой на что-то ещё, могут сделать слова курсивом, сделать шрифт больше или меньше и так далее.

Главными частями элемента являются:

1. **открывающий тег (Opening tag)**: состоит из имени элемента, заключённого в открывающие и закрывающие **угловые скобки**. Открывающий тег указывает, где элемент начинается или начинает действовать;
2. **закрывающий тег (Closing tag):** это то же самое, что и открывающий тег, за исключением того, что он включает в себя «/» перед именем элемента. Закрывающий элемент указывает, где элемент заканчивается. Отсутствие закрывающего тега является одной из наиболее распространённых ошибок начинающих и может приводить к странным результатам;
3. **контент (Content)**: это содержание элемента;
4. **элемент(Element)**: открывающий тег, закрывающий тег и контент вместе составляют элемент.

Элементы также могут иметь атрибуты, которые содержат дополнительную информацию об элементе, которую нет необходимости показывать в фактическом контенте. Атрибут всегда должен иметь:

1. пробел между ним и именем элемента (или предыдущим атрибутом, если элемент уже имеет один или несколько атрибутов);
2. имя атрибута, за которым следует знак равенства;
3. значение атрибута, заключённое с двух сторон в кавычки.

### [Вложенные элементы](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D0%B2%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5_%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B)

Также можно располагать элементы внутри других элементов — это называется **вложением**. Если необходимо заявить, что «кошка **очень** раздражена», то можно заключить слово "очень" в элемент [<strong>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/strong), который указывает, что слово должно быть акцентированно.

### [Пустые элементы](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D0%BF%D1%83%D1%81%D1%82%D1%8B%D0%B5_%D1%8D%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82%D1%8B)

Некоторые элементы не имеют контента, и называются **пустыми элементами**. Например, элемент [<img>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/img) (Рисунок 1):



Рисунок 1 – Элемент «img»

Он содержит три атрибута, но не имеет закрывающего тега </img>, и никакого внутреннего контента. Это потому, что элемент изображения не оборачивает контент для влияния на него. Его целью является вставка изображения в HTML страницу в нужном месте.

### [Анатомия HTML документа](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D0%B0%D0%BD%D0%B0%D1%82%D0%BE%D0%BC%D0%B8%D1%8F_html_%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0)

Теперь рассмотрим, как отдельные элементы объединяются в целую HTML страницу (Рисунок 2).

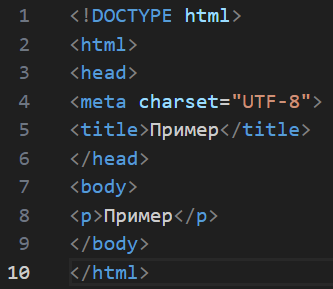


Рисунок 2 – Базовые теги HTML

Здесь имеются:

* <!DOCTYPE html> — доктайп. Раньше доктайпы должны были выступать в качестве ссылки на набор правил, которым HTML страница должна была следовать, чтобы считаться хорошим HTML, что могло означать автоматическую проверку ошибок и другие полезные вещи. Однако в наши дни это просто исторический артефакт, который должен быть включён для того, чтобы все работало правильно;
* <html></html> — элемент [<html>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/html). Этот элемент оборачивает весь контент на всей странице, известен как корневой элемент;
* <head></head> — элемент [<head>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/head). Этот элемент выступает в качестве контейнера для всего, что необходимо включить на HTML страницу, но не является контентом, который показывается пользователям страницы. К ним относятся такие вещи, как ключевые слова и описание страницы, которые будут появляться в результатах поиска, CSS стили контента, кодировка и многое другое;
* <body></body> — элемент [<body>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/body). В нем содержится весь контент, который будет показан пользователям, когда они посещают страницу, будь то текст, изображения, видео, игры, проигрываемые аудиодорожки или что-то ещё;
* <meta charset="utf-8"> — этот элемент устанавливает UTF-8 кодировку документа, которая включает в себя большинство символов из всех известных человечеству языков. По сути, теперь документ может обрабатывать любой текстовый контент, который в него вложен. Нет причин не устанавливать её, так как это может помочь избежать некоторых проблем в дальнейшем;
* <title></title> — элемент [<title>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/title). Этот элемент устанавливает заголовок для страницы, который является названием, появляющимся на вкладке браузера загружаемой страницы, и используется для описания страницы, когда вы добавляете её в закладки/избранное.

### [Разметка текста](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%82%D0%BA%D0%B0_%D1%82%D0%B5%D0%BA%D1%81%D1%82%D0%B0)

* [Заголовки](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D0%B7%D0%B0%D0%B3%D0%BE%D0%BB%D0%BE%D0%B2%D0%BA%D0%B8). Элементы заголовка позволяют указывать определённые части вашего контента в качестве заголовков или подзаголовков. Точно так же, как книга имеет название, названия глав и подзаголовков, HTML документ может содержать то же самое. HTML включает шесть уровней заголовков <h1> – <h6>, хотя обычно используется не более 4-х.
* [Абзацы](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D0%B0%D0%B1%D0%B7%D0%B0%D1%86%D1%8B). Элемент [<p>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/p) предназначен для абзацев текста.
* [Списки](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D1%81%D0%BF%D0%B8%D1%81%D0%BA%D0%B8). Большая часть веб-контента является списками и HTML имеет специальные элементы для них. Разметка списка всегда состоит по меньшей мере из двух элементов. Наиболее распространёнными типами списков являются нумерованные и ненумерованные списки:

1. **Ненумерованные списки** - это списки, где порядок пунктов не имеет значения. Они оборачиваются в элемент [<ul>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/ul).
2. **Нумерованные списки -** это списки, где порядок пунктов имеет значение. Они оборачиваются в элемент [<ol>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/ol).

Каждый пункт внутри списков располагается внутри элемента [<li>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/li) (list item, элемент списка).

## [Ссылки](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics#%D1%81%D1%81%D1%8B%D0%BB%D0%BA%D0%B8). Чтобы добавить ссылку, нужно использовать простой элемент — [<a>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/a) — это сокращение от "anchor" ("якорь"). Чтобы текст в абзаце стал ссылкой, необходимо выполнить следующие действия:

1. выбрать некоторый текст и обернуть его в элемент [<a>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/a);
2. задать элементу [<a>](https://developer.mozilla.org/ru/docs/Web/HTML/Element/a) атрибут href;
3. заполнить значение этого атрибута веб-адресом, на который будет указана ссылку.

Таким образом, HTML — это простой язык разметки, который легко изучить и использовать. Код на нем можно писать в любом текстовом редакторе. На HTML можно создавать простые сайты, например, портфолио, каталоги, справочники, инструкции, небольшие блоги [2].

## CSS

CSS (Cascading Style Sheets) — это код, который используется создателями веб-страниц для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц.

Как и HTML, CSS не является языком программирования. Это не язык разметки - это язык таблицы стилей. Это означает, что он позволяет применять стили выборочно к элементам в документах HTML. Но всё равно нужно применить CSS к HTML документу. В противном случае, CSS стиль не повлияет на то, как браузер отобразит HTML документ. Для подключения CSS необходимо открыть файл HTML и вставить строку, указанную на рисунке 3, в шапку, между <head> и </head> тегами.



Рисунок 3 – Подключение CSS

Рассмотрим структуру CSS (Рисунок 4) немного более подробно:

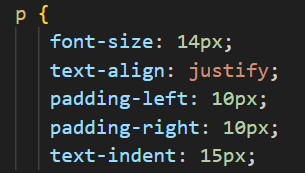


Рисунок 4 – Подключение CSS

* [Селектор (Selector)](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics#%D1%81%D0%B5%D0%BB%D0%B5%D0%BA%D1%82%D0%BE%D1%80_selector). Имя HTML-элемента в начале набора правил. Он выбирает элемент(ы) для применения стиля (в данном случае, элементы p). Для стилизации другого элемента необходимо изменить селектор.
* [Объявление (Declaration)](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics#%D0%BE%D0%B1%D1%8A%D1%8F%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_declaration). Правило, которое указывает, какие из **свойств** элемента необходимо стилизовать.
* [Свойства (Properties)](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics#%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0_properties). Способы, которыми можно стилизовать определённый HTML-элемент. В CSS можно выбрать, какие свойства нужно затронуть в правиле.
* [Значение свойства (Property value)](https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/CSS_basics#%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%B0_property_value). Справа от свойства, после двоеточия, есть **значение свойства**, которое выбирает одно из множества возможных признаков для данного свойства.

Важно отметить, что:

* Каждый набор правил (кроме селектора) должен быть обёрнут в фигурные скобки ({}).
* В каждом объявлении необходимо использовать двоеточие (:), чтобы отделить свойство от его значений.
* В каждом наборе правил необходимо использовать точку с запятой (;), чтобы отделить каждое объявление от следующего.
* Таким образом, CSS – это инструмент для оформления сайтов, без которого невозможно сделать красивую страницу. Глубокое изучение каскадных таблиц поможет сделать сайты более стильными [1].

## JavaScript

JavaScript — это интерпретируемый язык программирования, который используют для написания frontend- и backend-частей сайтов, а также мобильных приложений. Часто в текстах и обучающих материалах название языка сокращают до JS. Это язык программирования высокого уровня, то есть код на нем понятный и хорошо читается.

JavaScript называют языком скриптов или сценариев. Скрипты — это набор инструкций, которые выполняются при загрузке страницы. Браузер самостоятельно интерпретирует код на JavaScript, для этого даже не требуется компиляция (перевод языка программирования в машинный код).

Скрипты можно прописать внутри кода страницы или подключить к HTML отдельным файлом.

JS — мультипарадигменный язык, так как он поддерживает разные парадигмы (стили) программирования и особенности:

* + - Объектно-ориентированный — представляет программу как систему отношений между объектами — наборами данных и функций.
    - Императивный — в этом стиле программирования используют инструкции, которые выполняются последовательно, а для доступа к данным используются переменные, значение которых может быть переписано в ходе вычисления.
    - Функциональный — в нем вычисление основано на математических функциях. Так как функциональное программирование не опирается на переменные, которые могут менять значение, результаты запросов будут всегда одинаковыми.
    - Динамическая типизация: Переменные в JavaScript не требуют объявления типа данных. Они могут изменять свой тип в процессе выполнения, что делает код более гибким, но также требует внимательного управления типами.
    - Обработка событий: JavaScript является мощным инструментом для обработки событий на веб-страницах. Он может реагировать на действия пользователей, такие как клики мышью, нажатия клавиш и другие.
    - Асинхронные операции: JavaScript позволяет выполнять асинхронные операции, такие как загрузка данных с сервера, без блокирования выполнения других задач.
    - Серверный JavaScript: С помощью платформы Node.js JavaScript может быть использован на серверной стороне для создания высокопроизводительных и масштабируемых веб-приложений.
    - Множество фреймворков и библиотек: JavaScript имеет обширное сообщество разработчиков и большое количество библиотек и фреймворков, таких как React, Angular, Vue.js, и другие, что упрощает создание сложных веб-приложений.
    - Кроссплатформенность: JavaScript может быть использован для разработки веб-приложений на различных платформах, включая веб, мобильные устройства и даже десктоп.

### Где применяется JavaScript

* + Клиентская часть веб—приложений (frontend). Это интерфейс страницы, то есть всё, что видит пользователь: контент, кнопки, формы обратной связи, меню. С помощью JS интерфейс реагирует на действия пользователя (клики мыши, нажатия клавиш), также язык отвечает за запоминание данных и автозаполнение форм.
  + Серверная часть веб—приложений (backend). Серверный код пишут на платформе Node.js. На JS работают, например, запросы AJAX (asynchronous javascript and XML), которые отправляются на сервер в фоновом режиме, без перезагрузки веб-страницы, и push-уведомления — всплывающие сообщения в браузере, которые реализуются с помощью технологии Comet. Такие уведомления приходят со специального Comet-сервера, который постоянно поддерживает соединение с браузером. Как раз с помощью JavaScript устанавливается это соединение.
  + Мобильные приложения на Android, iOS, Windows Mobile — когда нужно кросс-платформенное приложение или адаптация веб-приложения, а языков Kotlin (для Android) и Swift (для iOS) недостаточно, то используется JavaScript.

### Для чего нужен JS

В веб-приложениях HTML отвечает за разметку страницы. Расположение контента, внешний вид интерфейса описывается с помощью CSS, а JavaScript отвечает за интерактивность и функционал этого интерфейса. Вот примеры задач, которые решает JS:

* + Взаимодействие с пользователем. Всплывающие окна с полем для ввода почты или других данных, уведомления об использовании cookie.
  + Обработка данных HTML. JS помогает заполнить необходимые поля в формах, проверять буквенные и числовые значения, а также управлять HTML-элементами на странице. Например, в конструкторе лендингов Tilda можно перетаскивать блоки и по клику менять шрифты, цвета текста, фонов и другие значения.
  + Анимация. Это не только движущиеся объекты на странице и кнопки, но и карусель в галерее или плавная прокрутка страницы. Например, Pinterest с помощью JavaScript создал эффект бесконечной ленты.
  + Математические вычисления. С помощью JS-скриптов на сайтах реализованы калькуляторы, например, подсчет суммы заказа в корзине интернет-магазина.
  + Мобильные приложения: С использованием фреймворков, таких как React Native и NativeScript, JavaScript может быть использован для создания мобильных приложений для разных платформ.
  + Серверный JavaScript: используется на серверной стороне с использованием платформы Node.js для создания высокопроизводительных и масштабируемых веб-приложений.
  + Обработка событий: JavaScript используется для обработки событий, таких как клики мышью, нажатия клавиш, отправка форм и другие действия пользователя. Это позволяет создавать реактивные и отзывчивые интерфейсы [5].

## PHP

PHP (рекурсивный акроним словосочетания PHP: Hypertext Preprocessor) - это распространённый язык программирования общего назначения с открытым исходным кодом. PHP специально сконструирован для веб-разработок и его код может внедряться непосредственно в HTML.

Основная задача PHP — это «оживление» HTML страниц. Обычные HTML-страницы статичны. Статичность (или неизменность) означает, что после того как страницу создали и загрузили на сайт, при каждом обращении к этой странице браузер покажет её любому пользователю в неизменном виде. Но этого не всегда достаточно.

Почти всегда пользователи приходят на сайт за информацией, которая всё время меняется, и нужно отображать её актуальное состояние, например:

* показать курс валют;
* подсказать погоду на завтра;
* вывести счётчик посещений страницы.

Если использовать только HTML, то решить такие задачи не получится. PHP принимает входящий запрос от веб-сервера, выполняет сценарий и возвращает веб-серверу результат в виде готового HTML-код. Сервер отправляет этот результат в браузер, который, в свою очередь, отображает её пользователю. После этого видно свежий курс валют, погоду и другое.

РНР позволяет изменять веб-страницу на сервере прямо перед тем, как она будет отправлена браузеру. PHP умеет исполнять код — так называемые сценарии. При исполнении PHP может изменить или динамически создать любой HTML-код, который и является результатом исполнения сценария. Затем сервер отправляет этот код браузеру. При этом браузеру не известно, как была сформирована эта страница — статично свёрстана верстальщиком, или динамически создана при участии PHP. Это не важно, так как браузер всегда работает только с тем, что получил от сервера. Выполнение сценария также называют его интерпретацией, а сам PHP — интерпретатором.

### Где используется PHP

Основная сфера применения языка PHP — веб (сайты), которые пользователи каждый день посещают через браузер компьютера или смартфона. Важно понимать, что веб — это не весь интернет, а только самая большая и популярная его часть. Кроме веба, через интернет работает электронная почта, мессенджеры, файлообменные сервисы, сетевые игры и многое другое.

Практически каждый сайт, который есть в интернете, работает на PHP. Этот язык отлично подходит для любых динамических веб-сайтов, среди которых:

* социальные сети;
* блоги и форумы;
* интернет-магазины;
* браузерные игры.

Типичный PHP-сценарий — это набор выражений. Каждое выражение начинается с новой строчки и заканчивается точкой с запятой.

Выражение — это инструкция, которая приказывает PHP-интерпретатору выполнить одно действие, например, сложить два числа или вывести на экран информацию. Любой PHP-сценарий обязательно начинается с такой строки: <?php — так веб-серверу сообщается, что дальше в файле находится код на языке PHP.

Таким образом, PHP используют в веб-разработке, его можно встроить в статичные HTML-файлы и модифицировать их с помощью инструкций [4].

## MySQL

MySQL — одна из наиболее используемых систем управления базами данных. MySQL управляет реляционными базами данных, то есть такими, в которых таблицы связаны между собой. MySQL работает по принципу клиент-сервер. Компьютер пользователя (клиент) отправляет запрос. Сервер баз данных его обрабатывает и предоставляет ответ. Именно поэтому часто можно услышать понятие MySQL-сервер. Это сервер, на котором хранится база данных.

Система MySQL написана на языках программирования C и C++. Для работы MySQL используется язык структурированных запросов SQL. SQL (Structured Query Language) — это язык программирования, при помощи которого можно управлять информацией: добавлять, модифицировать, удалять и получать данные. Запросы к базе данных формируются на языке SQL.

SQL используется не только в MySQL. Многие РСУБД (реляционные системы управления базами данных) используют этот язык для работы с данными. Например:

* Microsoft SQL Server;
* PostgreSQL в облаке;
* Oracle Database;
* MariaDB;
* SQLite.

Для взаимодействия с базой данных используется специальное ПО, например, [phpMyAdmin](https://help.reg.ru/support/hosting/bazy-dannykh/rabota-v-phpmyadmin) — популярная программа среди разработчиков сайтов. Эта утилита позволяет работать с БД без ввода SQL-запросов. Но умение формировать запросы всё равно не помешает [3].

# 2 Практическая часть

## 2.1 Требования и описание реализации

Курсовая работа предполагает создание web-приложение, используя технологии HTML, CSS, Javascript, PHP, MySQL. Требуется реализовать добавление, редактирование, просмотр и удаление пользователей. Также предусмотреть функции авторизации и аутентификации.

Описание реализации:

- неавторизированные пользователи должны иметь возможность регистрироваться;

- авторизированные пользователи разделить на 2 группы: администраторы и авторизированные пользователи;

- администраторы имеют право просматривать, добавлять, редактировать и удалять пользователей;

- авторизированные пользователи имеют право просматривать и редактировать только свои данные;

- при регистрации пользователю присваивается статус авторизированного пользователя.

Требования к проекту:

1. Должна быть предусмотрена авторизация и регистрация пользователей.
2. Зарегистрированный пользователь имеет право редактировать свои данные и просматривать свои услуги.
3. Администраторы имеют право редактировать пользователей и добавлять услуги.
4. Неавторизированный пользователи имеют право просматривать список услуг.
5. Сайт должен быть выдержан в определенном стиле и корректно открываться на любом разрешении.
6. Общий функционал может быть продуман самостоятельно.
7. Важно продемонстрировать не большой объем функционала, а минимальные требования к проекту. Проект может максимально простым, но при этом содержать все необходимые функции.

## 2.2 Реализация проекта

Для начала работы необходимо установить на компьютер дистрибутив для сборки локального веб-сервера. Например, Open Server Panel (OSP). Это программное обеспечение, которое предоставляет пользователю графический интерфейс для управления веб-сервером, базами данных и другими компонентами веб-инфраструктуры. OSP обычно используется для локальной разработки и тестирования веб-приложений, а также для управления хостинг-серверами. С помощью Open Server Panel можно устанавливать и настраивать веб-серверы (например, Apache), базы данных (MySQL), интерпретаторы языков программирования (PHP) и другие компоненты веб-стека. Это делает процесс создания и тестирования веб-приложений более удобным и эффективным.

### База данных

После установки OSP необходимо создать базу данных. Для этого нужно перейти во вкладку «Дополнительно», затем в «PhpMyAdmin». Созданная база данных представлена на рисунке 5.

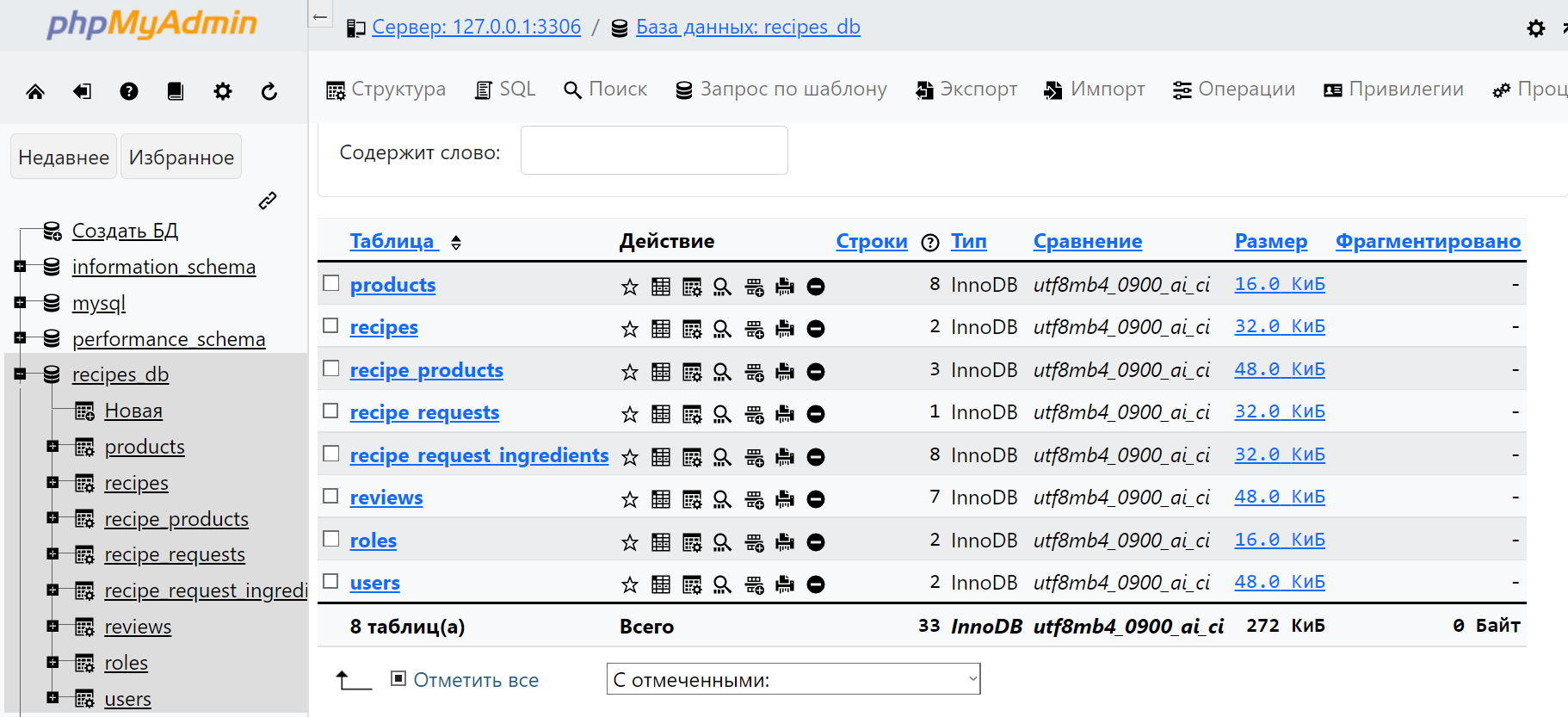


Рисунок 5 — База данных

Она содержит следующие таблицы:

* Products. Содержит информацию о всех продуктах. Их задает администратор.
* Recipes. Содержит информацию о названии и описании рецептов, уже принятых администратором.
* Recipe\_products. Содержит информацию о том, какой продукт в каком количестве нужен в определенном рецепте.
* Recipe\_requests. Содержит информацию о рецептах, отправленных на модерацию.
* Recipe\_request\_ingredients. Содержит информацию об ингредиентах для рецептов, отправленных на модерацию.
* Reviews. Содержит информацию об отзывах на рецепт: кто писал, для какого рецепта, текст отзыва.
* Roles. Содержит информацию о ролях. По умолчанию пользователям присваивается роль «user», поменять её может администратор в OSP.
* Users. Содержит информацию о пользователях и их ролях.

### Регистрация

Страница представлена в файле register.php, код реализует страницу регистрации пользователей с использованием PHP и HTML, используются следующие методы:

* include 'connection.php'; - подключение файла connection.php, который содержит код для установления соединения с базой данных. Подобная строка содержится и на других страницах, где необходимо подключение к базе данных.
* session\_start(); - запуск сессии для работы с сеансами пользователя. Этот пункт также повторяется в других файлах.
* if (isset($\_SESSION['user\_id'])) {...} - проверка, авторизован ли пользователь.
* Получение данных из формы с помощью $\_POST.
* Проверка уникальности email в базе данных с использованием подготовленного запроса и методов PDO.
* Хеширование пароля с помощью password\_hash.
* Вставка данных в базу данных с использованием подготовленного запроса и методов PDO.
* HTML-разметка формы регистрации: использование тегов <form>, <input>, <label> для создания формы.
* Обработка ошибок и вывод сообщений об ошибках при заполнении формы.

Страница регистрации представлена на рисунке 6.

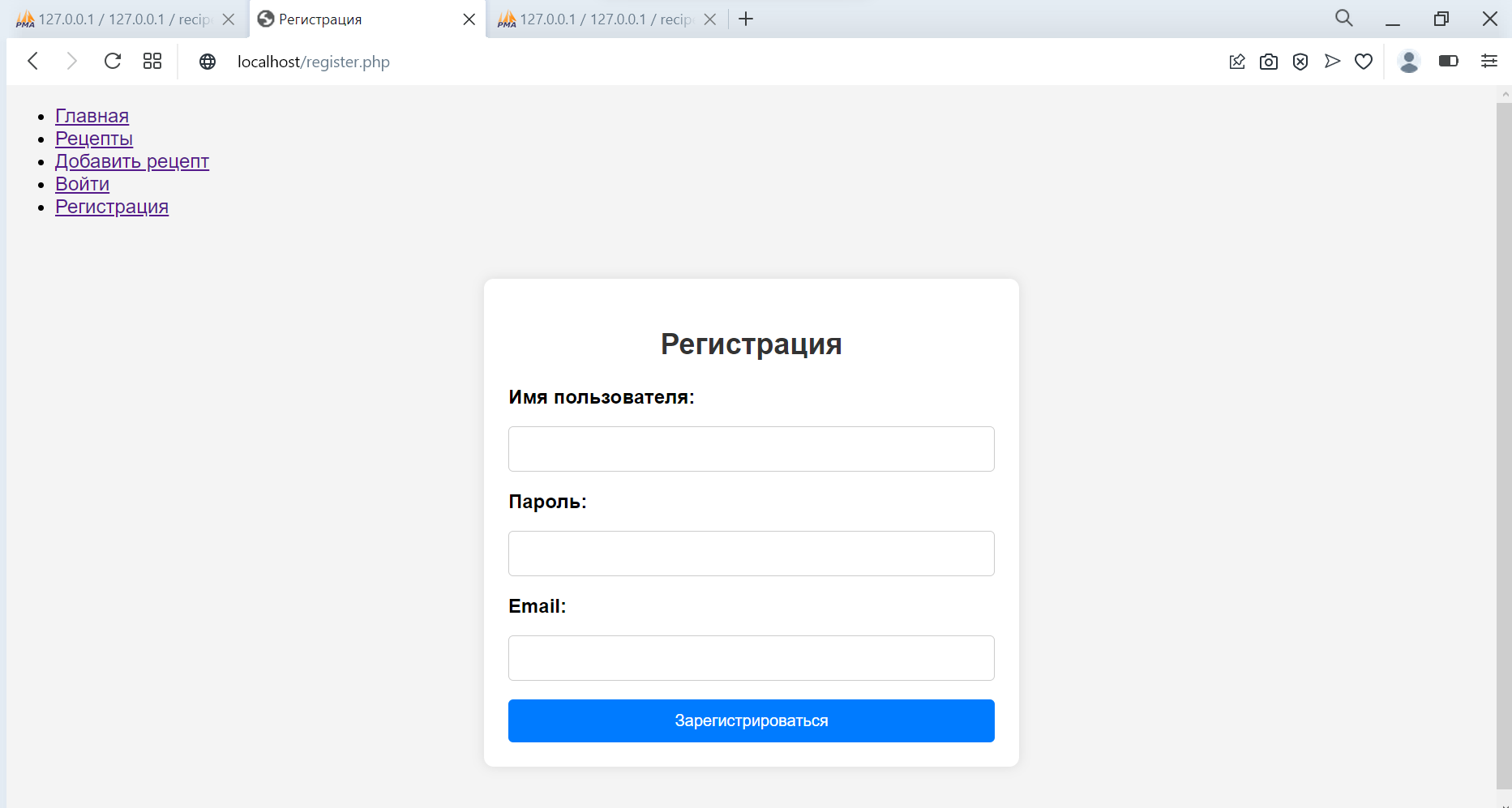


Рисунок 6 — Регистрация

### Вход

Страница представлена в файле login.php, код реализует страницу входа пользователей с использованием PHP и HTML, используются следующие методы:

* if (isset($\_SESSION['user\_id'])) {...} - проверка, авторизован ли пользователь. Если пользователь уже авторизован, происходит перенаправление на страницу index.php.
* Получение данных из формы с помощью $\_POST.
* Подготовленный запрос для выборки пользователя из базы данных по email.
* Использование методов PDO для выполнения запроса и получения данных.
* Проверка соответствия введенного пароля хешированному паролю в базе данных с помощью password\_verify.
* Установка значений сессии для авторизации пользователя.
* Перенаправление на страницу index.php в случае успешной авторизации.
* HTML-разметка формы входа: использование тегов <form>, <input>, <label> для создания формы, обработка ошибок и вывод сообщений об ошибках при входе.

Страница входа представлена на рисунке 7.

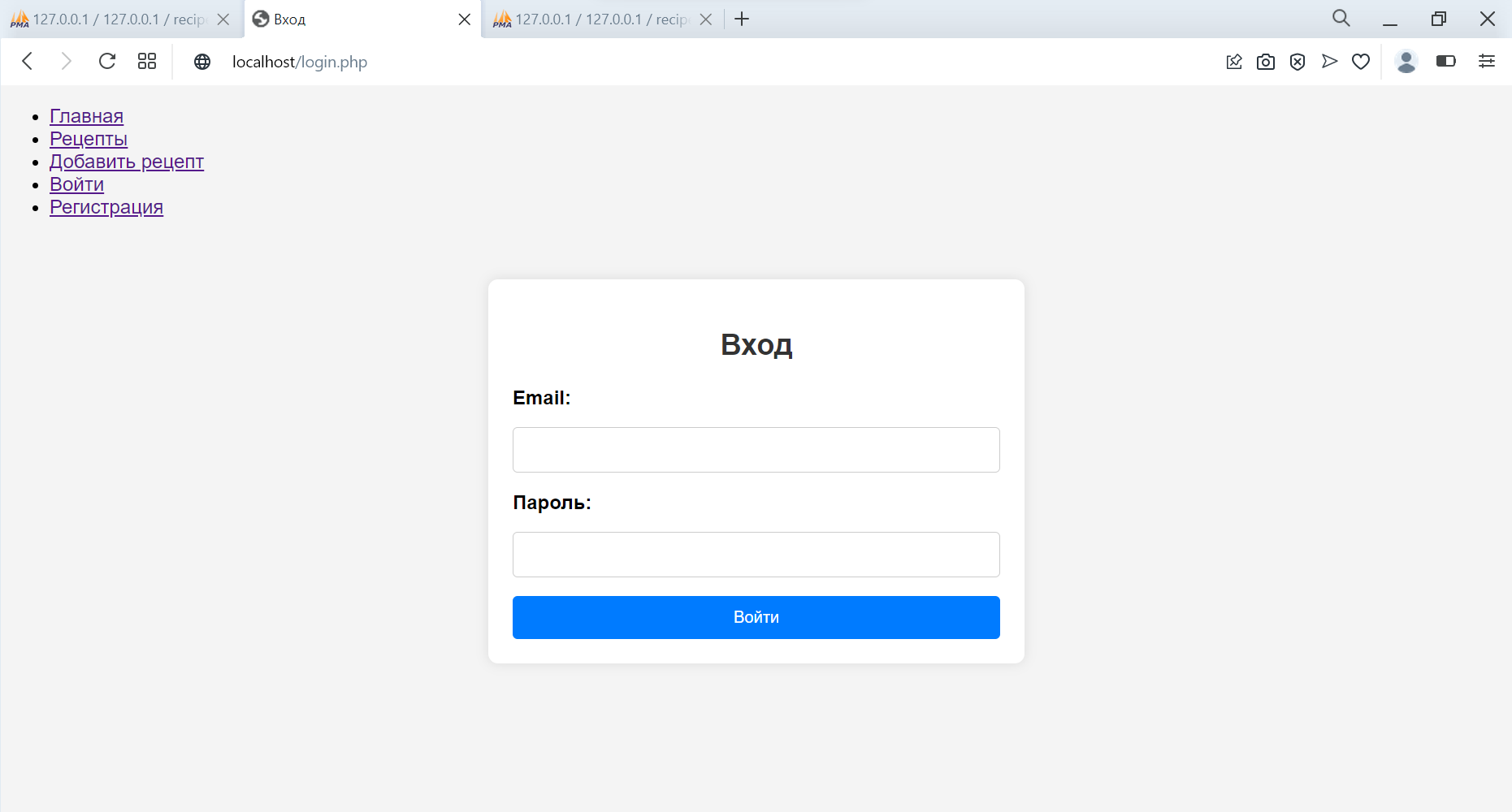


Рисунок 7 — Авторизация в web-приложении

### Главное меню

Пройдя процесс регистрации или входа в аккаунт, пользователь попадает в главное меню (index.php) (Рисунок 8):

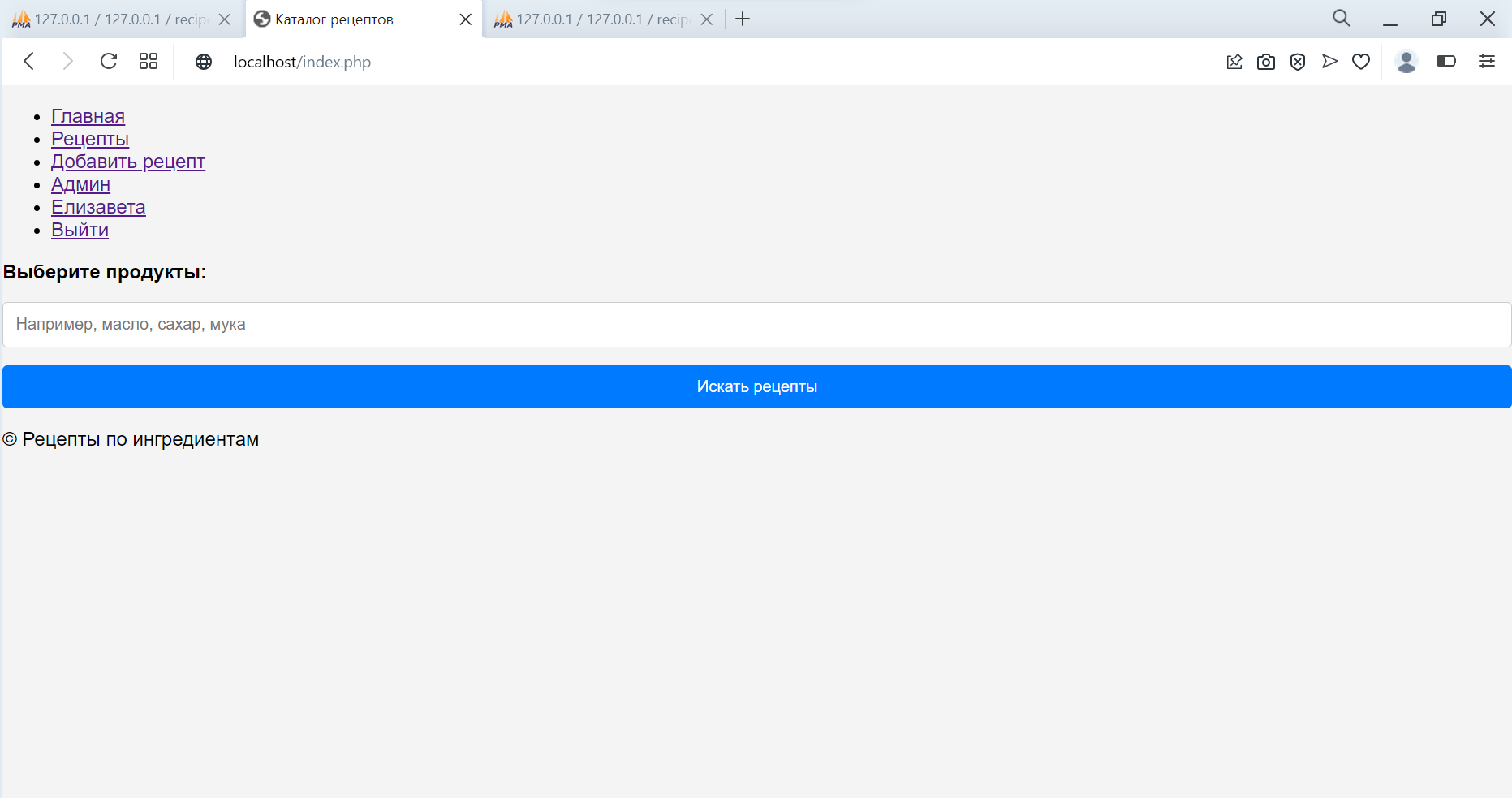


Рисунок 8 — Главное меню

Это страница для поиска рецептов по ингредиентам с использованием HTML и PHP. Методы и функции, используемые для реализации страницы:

* require\_once 'layout.php' - подключение файла layout.php, который содержит общий макет страницы.
* HTML-разметка формы поиска: использование тегов <form>, <input>, <label> для создания формы, использование атрибутов action="search.php" и method="post" для указания адреса обработчика формы и метода передачи данных.
* Использование стилей: подключение стилей с помощью тега <link> и атрибута href="auth.css".

На этой странице есть окно поиска рецептов по ингредиентам. Например, найдем рецепты, которые имеют в составе помидор (Рисунок 9).

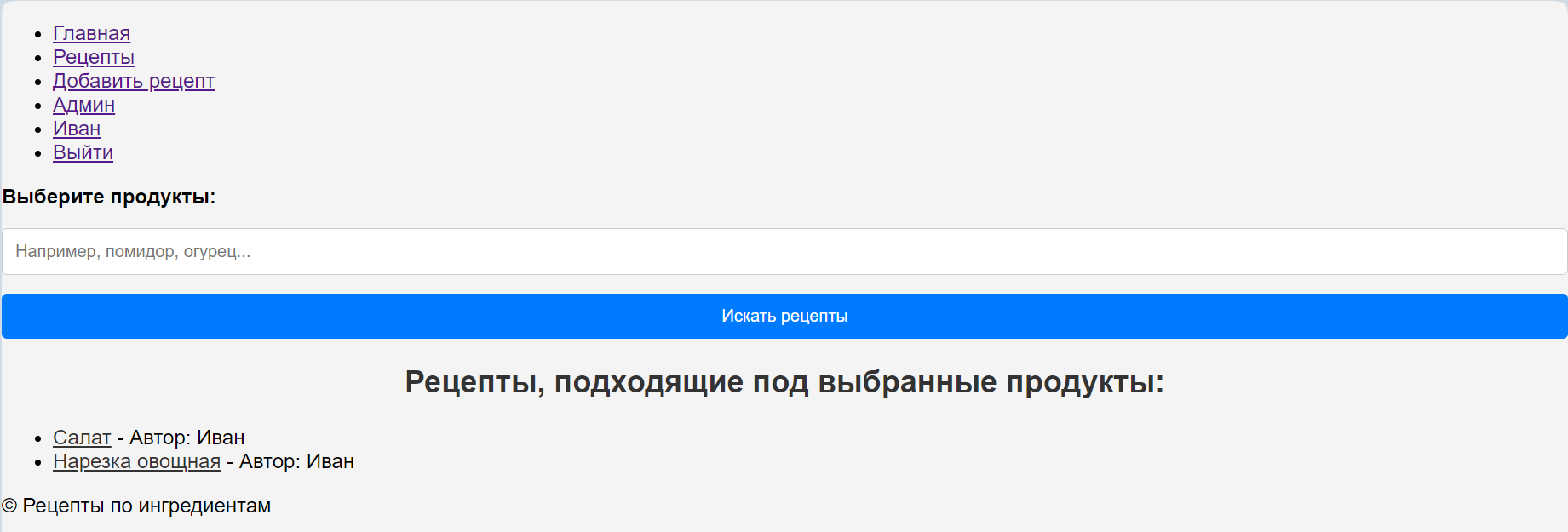


Рисунок 9 — Поиск по ингредиенту

Нашлись 2 рецепта, удовлетворяющие запросу. Перейдем на страницу рецепта «Салат», чтобы ознакомиться с интерфейсом (Рисунок 10)

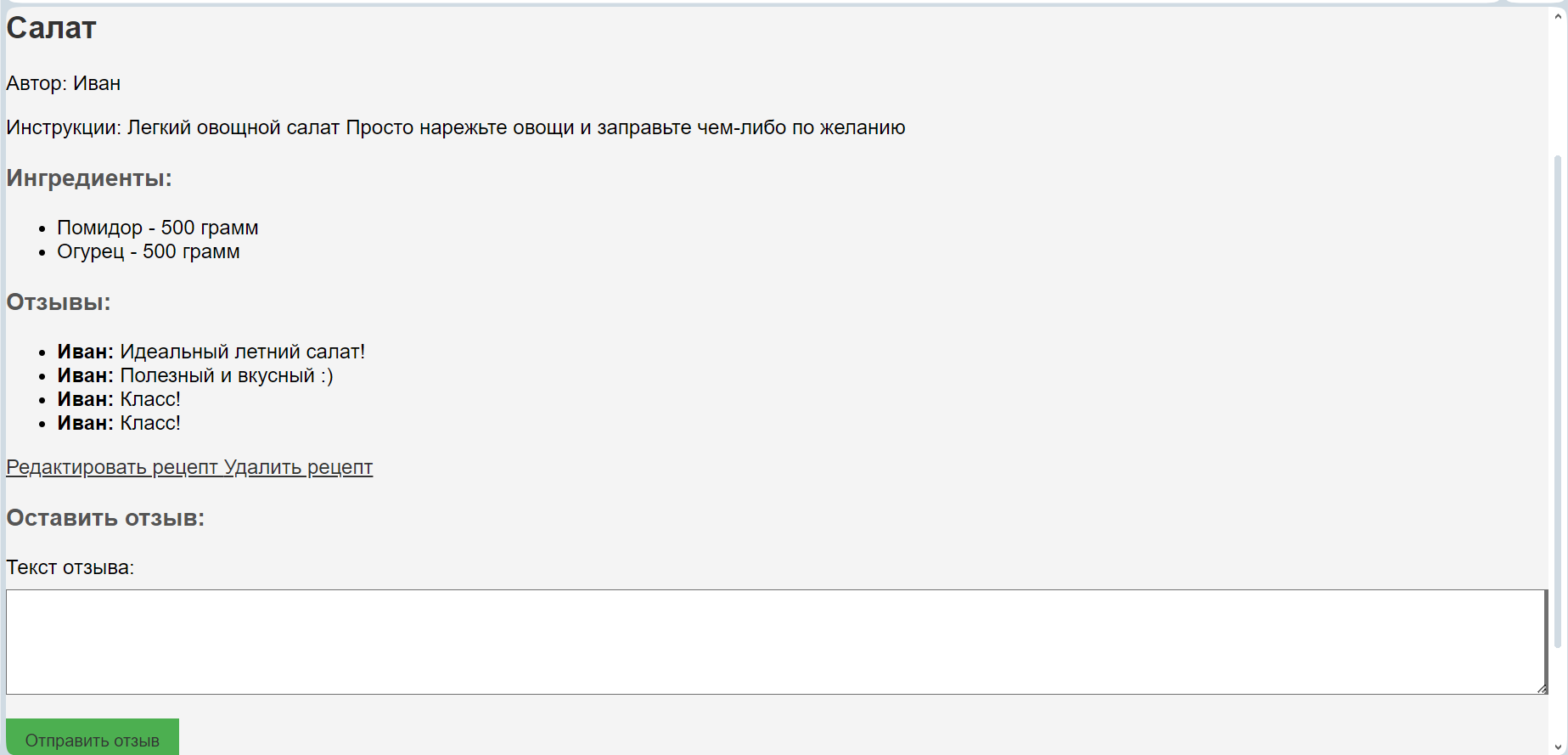


Рисунок 10 — Поиск по ингредиенту

На данной странице представлены: название блюда, автор, инструкция (описание) приготовления блюда, ингредиенты с указанием веса в граммах, отзывы. Также автор рецепта может редактировать и удалять рецепт. Любой пользователь может оставить отзыв.

Из главного меню пользователь может перейти в раздел «Рецепты», где представлены все имеющиеся на портале рецепты.

Также имеется раздел «Добавить рецепт» (Рисунок 11)

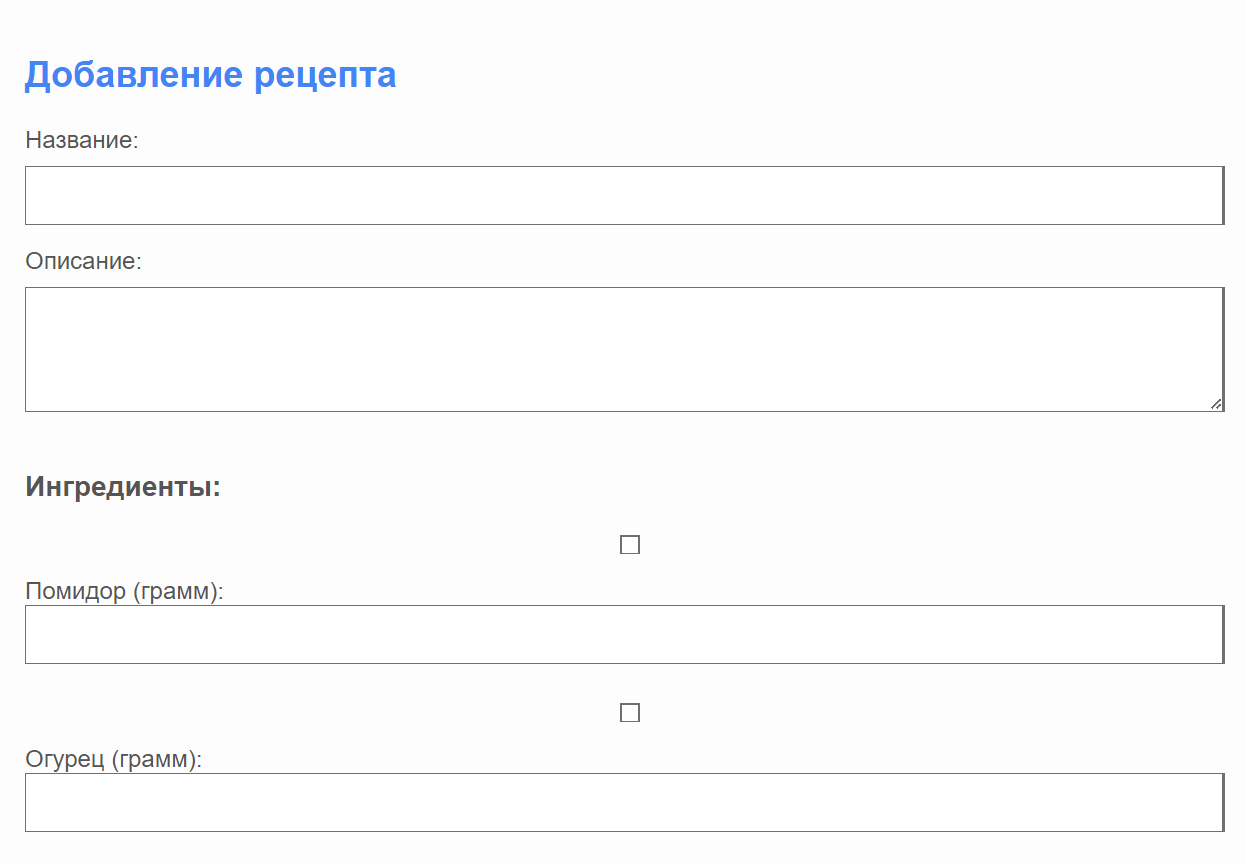


Рисунок 11 — Добавление рецепта

Пользователь указывает название блюда, описание его приготовления, а также выбирает из списка необходимые ингредиенты с указанием грамм. К сожалению, интерфейс выбора ингредиентов не совсем удобен пользователю, так как приходится из всего списка ингредиентов искать необходимый. Поэтому его необходимо доработать.

### Администратор

Этот раздел доступен только тем пользователям, у которых role = «admin». В этом разделе можно добавлять новые продукты, рецепты, а также принимать заявки на добавление рецептов от пользователей. Интерфейс страницы представлен на рисунке 12.

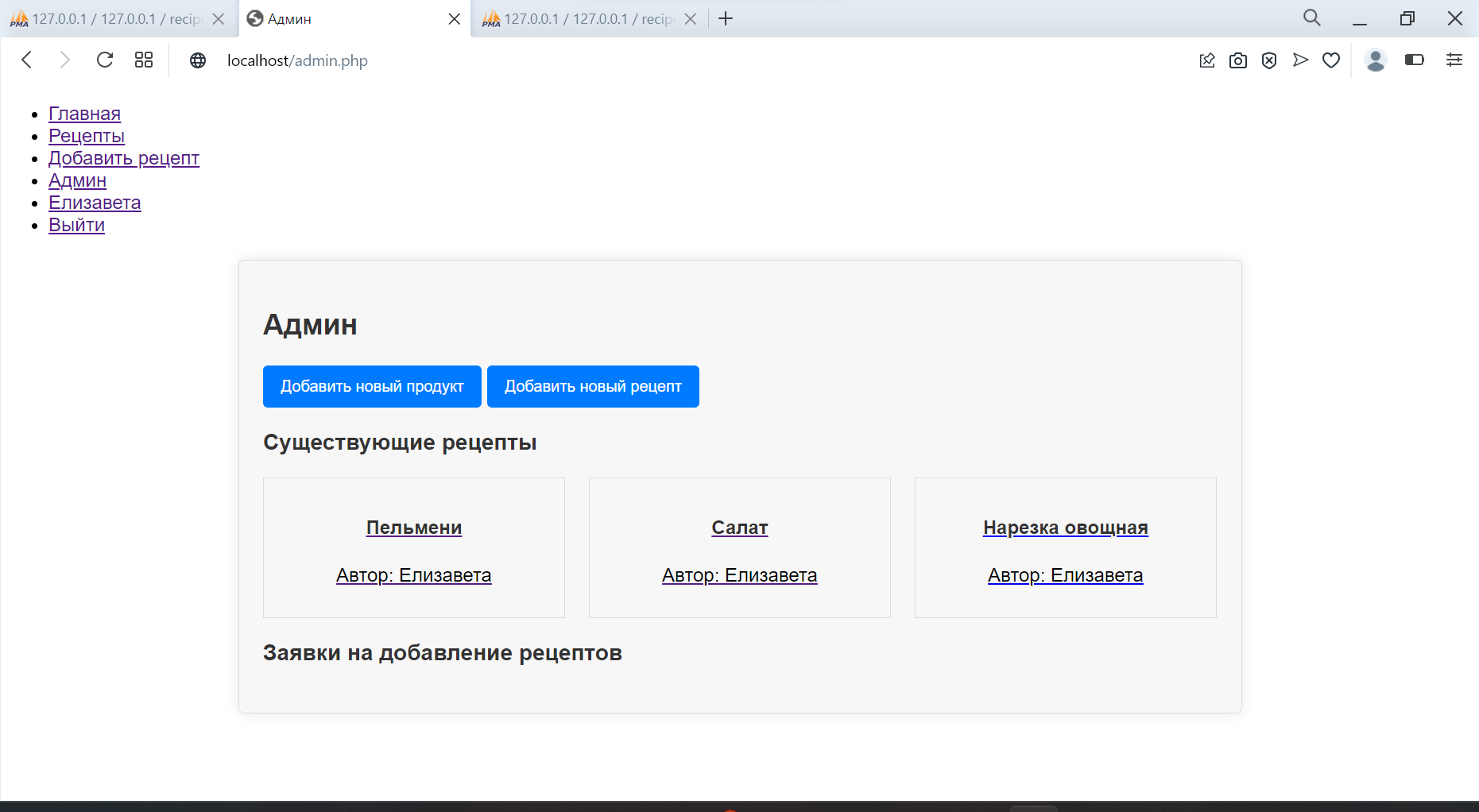


Рисунок 12 — Раздел администрирования

### Личный кабинет

Эта страница содержит в себе информацию о пользователе: имя, почта, роль, а также рецепты, которые он добавлял. Интерфейс страницы представлен на рисунке 13.

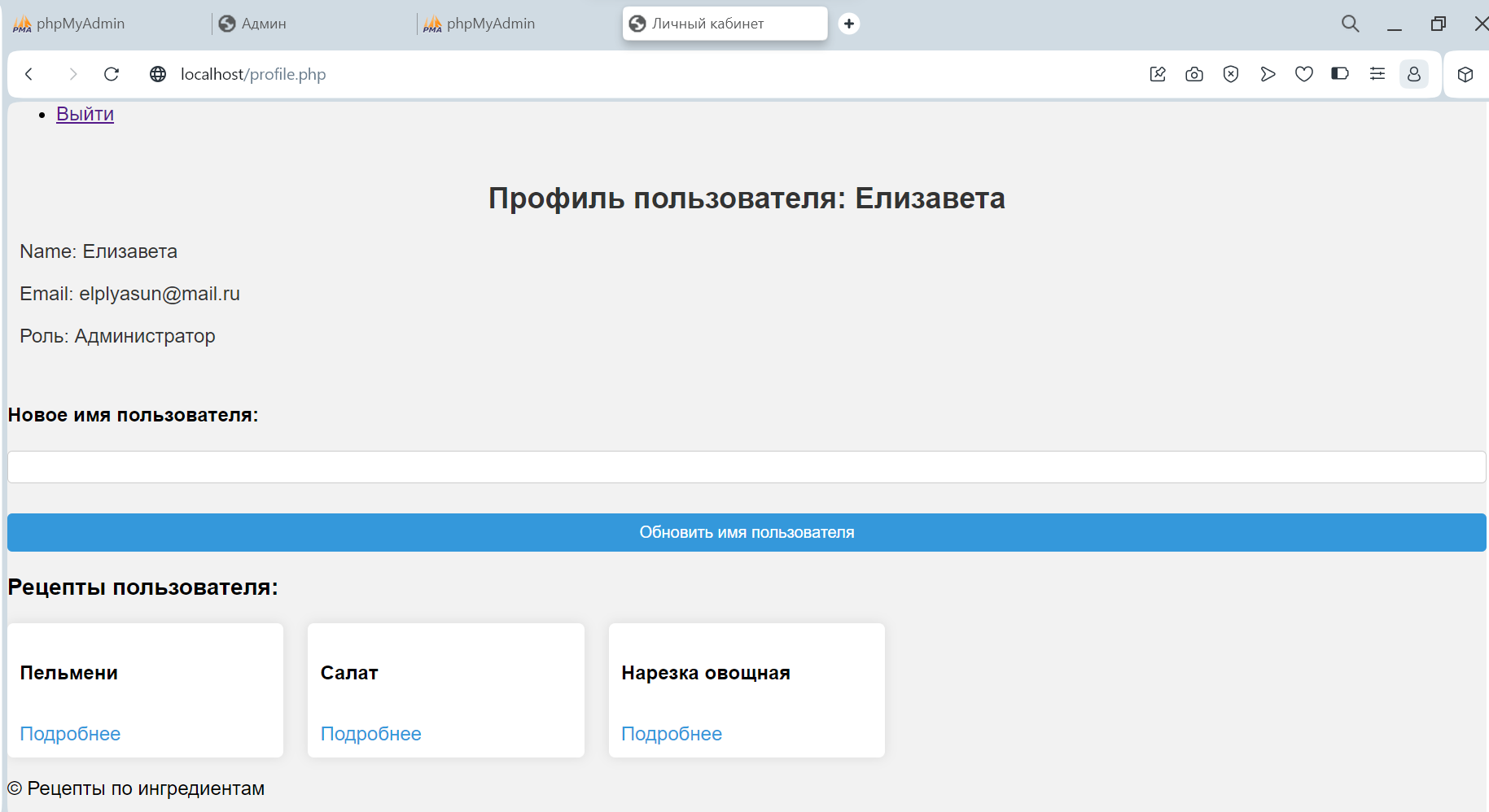


Рисунок 13 — Личный кабинет

Кроме того, пользователь может изменить свое имя, и оно также поменяется в базе данных. Например, поменяем пользователя Елизавету на Ивана (Рисунки 14-15)

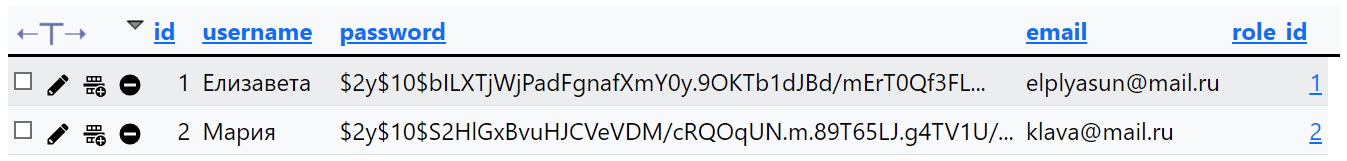


Рисунок 14 — До смены имени

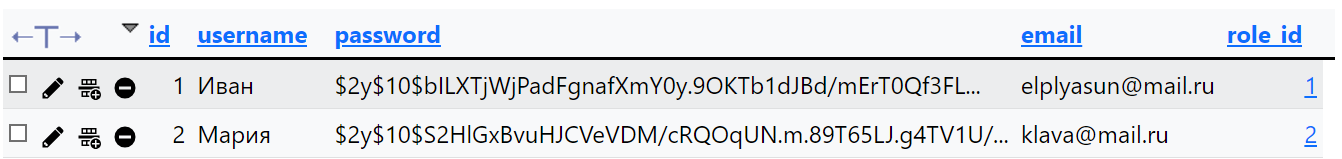


Рисунок 15 — После смены имени

Заметно, что все данные, кроме имени, остались неизменными.

Таким образом, данный веб-сайт позволяет искать, добавлять, редактировать рецепты, а также оставлять отзывы.

# ЗАКЛЮЧЕНИЕ

В ходе работы были изучены технологии HTML, CSS, JavaScript, PHP и MySQL, а также разработан веб-сайт по поиску рецептов. В процессе создания такого сайта необходимо учитывать удобство использования, безопасность передачи данных и возможность масштабирования функционала.

В заключение, web-программирование играет важную роль в создании современных сайтов и приложений, предоставляющих пользователю удобный и понятный интерфейс. Кроме того, веб-программирование позволяет реализовать различные функциональности на сайте, такие как поиск рецептов по различным критериям, создание персональных профилей пользователей, возможность комментировать рецепты, а также обеспечивает удобное взаимодействие с базой данных для хранения и обработки информации.

Таким образом, web-программирование является ключевым инструментом для разработки современных сайтов, обеспечивающих пользователю удобный доступ к информации о рецептах и возможность делиться собственными кулинарными находками.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. Основы CSS [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/CSS\_basics (дата обращения: 05.12.2023).
  2. Основы HTML [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Learn/Getting\_started\_with\_the\_web/HTML\_basics (дата обращения: 05.12.2023).
  3. Что такое MySQL [Электронный ресурс]. URL: https://help.reg.ru/support/hosting/bazy-dannykh/chto-takoye-mysql (дата обращения: 30.12.2023).
  4. Что такое PHP [Электронный ресурс]. URL: https://htmlacademy.ru/blog/php/introduction-to-php (дата обращения: 28.12.2023).
  5. JavaScript: что это такое [Электронный ресурс]. URL: https://blog.skillfactory.ru/glossary/javascript/ (дата обращения: 21.12.2023).