КОМИТЕТ ОБРАЗОВАНИЯ, НАУКИ И МОЛОДЕЖНОЙ ПОЛИТИКИ

ВОЛГОГРАДСКОЙ ОБЛАСТИ

ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ПРОФЕССИОНАЛЬНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

«КАМЫШИНСКИЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

**КУРСОВАЯ РАБОТА**

|  |  |  |  |
| --- | --- | --- | --- |
| на тему: | | «*Городское бюро по найму*» |  |
| *по дисциплине «МДК 02.01 Информационные технологии и платформы разработки информационных систем»* | | | |
| Студента | *Рыбалка Вячеслава Сергеевичa* | | |
|  | *(фамилия, имя, отчество)* | | *(подпись)* |
| Группа | *ИС-407* | | |
| Специальность | *09.02.04 «Информационные системы (по отраслям)»* | | |
|  | *(код по ОКСО, наименование специальности)* | | |
| Проверил | *Николаев А.Ю.* | | |
|  | *(фамилия, инициалы)* | | *(подпись)* |

Камышин 2022г.

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 1](#_Toc121138621)

[ВВЕДЕНИЕ 2](#_Toc121138622)

[Анализ предметной области 3](#_Toc121138623)

[СТРУКТУРА ПРОЕКТА 4](#_Toc121138624)

[Интерфейс пользователя 8](#_Toc121138625)

[СПИСОК ЛИТЕРАТУРЫ 15](#_Toc121138626)

# ВВЕДЕНИЕ

Развитие рынка труда привело к тому, что на сегодняшний день намного проще найти работу виртуально, так как жизнь почти каждого человека связана с сетью Интернет. Если еще несколько лет назад люди предпочитали устраиваться на работу, приходя лично к работодателю, и мало кто сильно освоил сеть Интернет, то современные люди обучены гораздо большему кругу возможностей.

Современный Интернет позволяет своим пользователям делать буквально почти всё, что угодно виртуально, например, слушать музыку, заходить на различные сайты в поисках какой-либо информации, просматривать карты и, конечно, устраиваться на работу.

Целью создания веб-сайта «Городское бюро по найму» послужило решение автоматизировать и сделать более комфортными в выполнении задачи современного человека.

Массовое распространение веб-сайтов привело к массовому желанию людей заниматься их разработкой. Этому послужили следующие причины: популярность веб-сайтов, высокие заработные платы, интересная, сложная и творческая работа.

Для создания веб-сайта разработчику необходимо пройти несколько этапов: разработка, тестирование и внедрение в эксплуатацию веб-сайта.

Задачи курсового проекта:

* Определить базовые таблицы
* Спроектировать отчеты
* Выполнить нормализацию таблиц
* Разработать PHP-запросы
* Создать пользовательский интерфейс
  + Авторизация
  + Регистрация
  + Основная страница
  + Страницы обработчики

Разработанный веб-сайт планируется ввестись в эксплуатацию для людей, которые находятся в поисках работы или для тех, кто предоставляет одно или более рабочее место.

# Анализ предметной области

* 1. **Потенциальные пользователи**

Данный проект рассчитан на следующую аудиторию:

* Люди, находящиеся в поиске работы;
* Люди, находящиеся в поиске работника;
* Люди, интересующиеся востребованностью той или иной профессии.
  1. **Анализ предметной области**

База данных должна содержать сведения о следующих объектах:

* Информация о клиентах – регистрационный номер, фамилия, имя, отчество, адрес, телефон, пол, образование, номер квитанции об уплате регистрационного взноса, заявки (желаемая должность, минимальная зарплата, район города);
* Информация о работодателях – название фирмы, организационно-правовая форма, форма собственности, адрес, телефон, инспектор по кадрам, номер договора об оказании услуг по найму, предложение (профессия, количество рабочих мест, зарплата, район города, ограничение на пол, возраст, образование).

Выходные документы

* Квитанция (приходной ордер) об уплате регистрационного взноса;
* Для клиента и работодателя (выборочно или для всех) – список согласованных предложений-заявок;

Бизнес-правила

* Каждый клиент может зарегистрировать несколько заявок на поиск работы;
* Каждый работодатель может предложить несколько вакансий;
* Сведения о клиентах и работодателях сохраняются в течение года.
  1. **Описание предметной области**

В современном мире для обеспечения комфортной и беззаботной жизни нужны

деньги, так как почти любая развитая страна имеет капиталистические направленности и взгляды. Естественно, деньги никому просто так не достаются и нужно работать, но у многих возникают проблемы с поиском работы, кому-то просто не хочется куда-то идти и с кем-то разговаривать, кто-то боится и так далее. В целях удобства и эффективности были созданы интернет-ресурсы для поиска работы, например, городское бюро по найму.

Деятельность бюро организована следующим образом: бюро готово искать работников для различных работодателей и вакансии для ищущих работу специалистов различного профиля. При обращении к вам клиента-работодателя его стандартные данные (название, вид деятельности, адрес, телефон) фиксируются в базе данных. При обращении к вам клиента-соискателя его стандартные данные (фамилия, имя, отчество, квалификация, профессия, иные данные) также фиксируются в базе данных. По каждому факту удовлетворения интересов обеих сторон составляется документ.

# СТРУКТУРА ПРОЕКТА

1. **Логическая схема БД**

Логическая модель БД описывает понятия предметной области, их взаимосвязь, а также ограничения на данные, налагаемые предметной областью без учета её реализации в конкретной СУБД [4]. Основным средством разработки логической модели данных в настоящий момент являются различные варианты ER-диаграмм (Entity-Relationship, диаграммы сущность-связь).

На рисунке 1 представлена логическая схема базы данных предметной области «kursovaya» и на рисунке 1.1 представлен общий вид базы. На логической схеме изображены следующие сущности: `users`, `offers`, `clients`, `receipts`, `bosses`. Описание сущностей содержится в таблице 1.

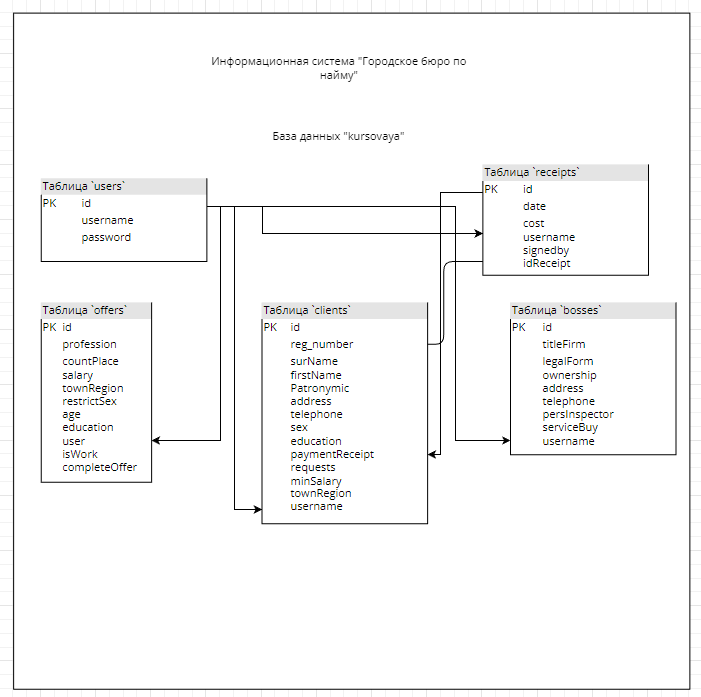


Рисунок 1 - Логическая схема БД

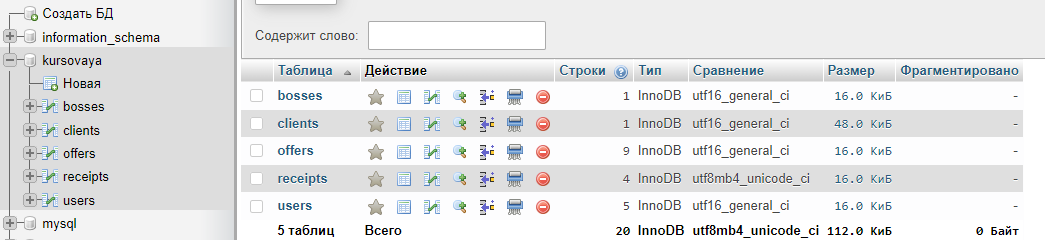


Рисунок 1.1 – Общий вид базы данных

Таблица 1 - Описание сущностей логической модели данных

|  |  |
| --- | --- |
| Название | Назначение |
| users | Описывает пользователей, зарегистрировавшихся на сайте. Содержит атрибуты:   * уникальный идентификатор; * имя пользователя; * пароль пользователя. |
| offers | Описывает заявки, добавленные пользователями. Содержит атрибуты:   * уникальный идентификатор; * профессия; * количество рабочих мест; * зарплата, район города; * ограничения по полу; * возраст; * образование; * имя пользователя; * isWork (заявка от работника или предложение от работодателя); * completeOffer (согласована ли уже сделка). |
| clients | Описывает клиентов, заполнивших профиль на сайте. Содержит атрибуты:   * уникальный идентификатор; * регистрационный номер; * фамилия; * имя; * отчество; * адрес; * телефон; * пол; * образование; * номер квитанции об уплате регистрационного взноса; * заявки (желаемая профессия); * минимальная зарплата; * район города; * имя пользователя. |
| receipts | Описывает завершенные чеки об оплате регистрационного взноса. Содержит атрибуты:   * уникальный идентификатор; * дата оплаты регистрационного взноса; * стоимость регистрационного взноса; * имя пользователя; * кем подписана квитанция об оплате; * номер квитанции об уплате регистрационного взноса. |
| bosses | Описывает работодателей, заполнивших профиль на сайте. Содержит атрибуты:   * уникальный идентификатор; * название фирмы; * организационно-правовая форма; * форма собственности; * адрес; * телефон; * инспектор по кадрам; * номер договора об оказании слуг по найму; * имя пользователя. |

1. **Технологии**

При разработке информационной системы использовались следующие технологии:

* HTML;
* CSS;
* PHP;
* Open Server Panel (и его модуль PhpMyAdmin);
* Framework Bootstrap v5.0;
* Framework jQuery v3.6.1;
* JavaScript.

С помощью технологии **HTML** я сделал разметку страницы, разделил контент по смысловой нагрузке, а с помощью **CSS** – добавил небольшую стилизацию, которую не смог выполнить на фреймворке Bootstrap v5.0, при помощи которого я делал основную стилизацию сайта и его адаптивность под разные устройства и экраны. Всё это позволяет создать интерактивные страницы с приятным дизайном.

«**PHP**: препроцессор гипертекста»; первоначально PHP/FI (Personal Home Page / Form Interpreter), а позже названный Personal Home Page Tools — «Инструменты для создания персональных веб-страниц») — C-подобный скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.

Технология PHP позволяет подключаться к базе данных, установленной на локальном хосте Open Server Panel, а конкретнее, на его модуле PhpMyAdmin. Также одна из функций PHP является манипуляция данными из базы, например, выгрузить из базы `offers` все данные при помощи LINQ (SQL-запросов): «” SELECT \* FROM `offers`”». Но зачастую «голый» PHP без помощи каких-либо фреймворков или других вспомогательных средств не может выполнить какую-либо задачу, поэтому в проект при помощи CDN (Сеть CDN — это географически распределенная сеть, состоящая из прокси-серверов и файловых серверов в центрах обработки данных, подключенных высокоскоростными магистральными сетями) фреймворк jQuery, который позволяет использовать AJAX-запросы.

**AJAX** — это аббревиатура, которая означает Asynchronous Javascript and XML. На самом деле, AJAX не является новой технологией, так как и Javascript, и XML существуют уже довольно продолжительное время, а AJAX — это синтез обозначенных технологий. AJAX чаще всего ассоциируется с термином Web 2.0 и преподносится как новейшее Web-приложение.

При использовании AJAX нет необходимости обновлять каждый раз всю страницу, так как обновляется только ее конкретная часть. Это намного удобнее, так как не приходится долго ждать, и экономичнее, так как не все обладают безлимитным интернетом. Правда в этом случае, разработчику необходимо следить, чтобы пользователь был в курсе того, что происходит на странице. Это можно реализовать с использованием индикаторов загрузки, текстовых сообщений о том, что идёт обмен данными с сервером.

1. **Обоснования использования**

**Open Server Panel**. Является очень удобным и обширным в использовании программным обеспечением, которое включает в себя сразу весь необходимый, так называемый, «набор джентльмена», который, в свою очередь, включает в себя HTTP-сервис, то есть сам локальный сервер, модуль для хранения базы данных (phpMyAdmin), язык программирования PHP, консоль, SQL-менеджер, а также в нем есть множество других модулей, таких как MySQL / MariaDB разных версий, PostgreSQL разных версий, MongoDB разных версий, технологии Memcached, Redis, DNS. Приятным бонусом являются предлагаемые при установке программы, который также помогаю в верстке – JustColorPicker, Bright Master и WinSCP.

**PHP**. В процессе изучения вопроса «кто же на свете всех милее…», часто возникали мысли, а стоит ли вообще переходить куда-либо, такие ли уж большие преимущества у других языков, и стоят ли они трудозатрат на обучение, поиска новых специалистов и прочее. Идеальных языков нет и быть не может, поэтому, скорее всего у нового языка и соответственно новых фреймворков будут свои «подводные камни». Поэтому в конце я пришел к совершенно обычному выводу, который и давали в форумах, наиболее умудренные опытом люди: пока сами не попробуем ответа на эти вопросы не получим.

Язык PHP обладает рядом неоспоримых преимуществ:

* Высокая скорость работы и, соответственно, общая производительность ресурсов.
* Бюджетность, экономичность. Найти специалиста не представляется проблемой, стоимость написания программ на php не высока.
* Простота освоения, простой синтаксис.
* Отличная совместимость и переносимость — php-коды работают одинаково хорошо с разными платформами.
* Набор текста кода и его редактирование можно осуществлять в любом текстовом или html-редакторе.
* Высокая гибкость, емкость и функциональность. PHP-программу можно составлять отдельно от разработки веб-страницы, без привязки, после чего совместить. Это существенно упрощает жизнь дизайнеров и программистов.
* Многозадачность и широкие возможности — создание любых веб-приложений, блогов, гостевых книг, интернет-магазинов, сайтов, работа с редиректами, заголовками, pdf-документами, базами данных, электронной почтой и пр.

**jQuery**. Это библиотека JavaScript (то есть, она написана на JavaScript), предназначенная для абстрагирования, выравнивания, исправления и упрощения скриптинга при работе с узлами HTML-элементов в браузере или для работы в браузере без графического интерфейса. Итак:

* Абстрагируется интерфейс объектной модели документа (он же DOM API).
* Выравниваются все различия реализаций DOM, существующие между браузерами.
* Исправляются известные браузерные баги, связанные с CSS и DOM.

Все это оборачивается в более простой и менее корявый API, нежели нативный API DOM – вот вам и библиотека jQuery. Также из плюсов **кроссплатформенность и кроссбраузерность**. Благодаря jQuery отпадает необходимость заботиться о синтаксисе и особенностях взаимодействия различных браузеров и операционных систем с кодом ввиду идентичной реализации работы фреймворка во всех средах. Кстати, крупнейшие поисковые системы уже перешли на jQuery., и еще один важный плюс - скорость работы с селекторами и простота эксплуатации AJAX. Независимые тесты демонстрируют, насколько быстр jQuery на фоне аналогичных библиотек. И это без использования кэшей, когда нужно многократно обращаться к одному элементу. Молниеносность достигается благодаря использованию селекторов – механизму быстрого обращения к любому объекту HTML-документа.

**Bootstrap.** Есть три причины, по которым я выбрал именно фреймворк Bootstrap**:**

1. Сокращения времени, необходимого для создания макета сайта. Фреймворк предлагает достаточное количество шаблонов и готовых решений;
2. Кросс-браузерность и адаптивность. Сайты, созданные с помощью Bootstrap, идентично отображаются на разных устройствах и в современных браузерах;
3. Простота использования. Для работы с фреймворком потребуются элементарные навыки верстки;
4. Легкость в освоении. Достаточное количество обучающих видеоматериалов поможет вам быстро и без особых трудностей освоить Bootstrap.

# Интерфейс пользователя

1. **Разработка и описание интерфейса**

При разработке интерфейса необходимо следовать следующим принципам: интерфейс должен помогать выполнять задачу, а не становиться ею. Он должен быть дружелюбным, удобным, интуитивно-понятным, чтобы минимизировать время, которое пользователь тратит на осознание того, как работать с системой.

На начальной странице пользователь должен зарегистрироваться, если нет уже существующего аккаунта, а после пройти авторизацию, введя в соответствующие поля свои логин и пароль (см. Рисунок 2).

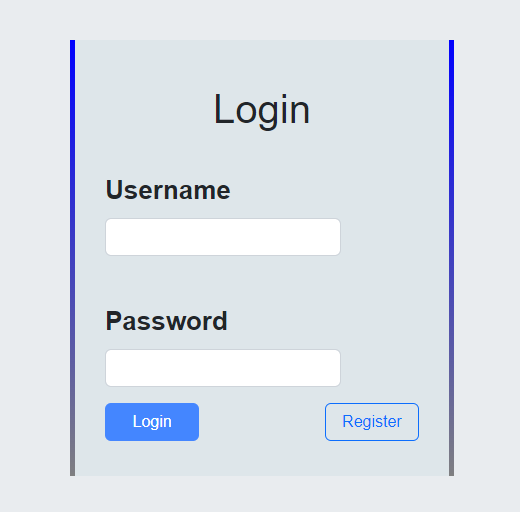
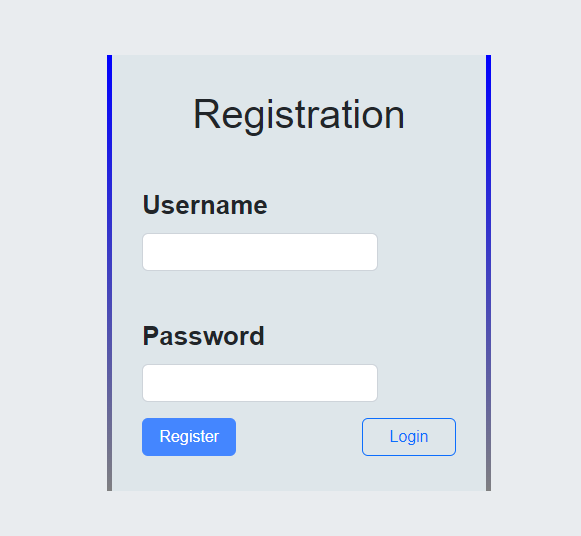


Рисунок 2 - Формы регистрации и авторизации пользователя

При нажатии на кнопку «Login» на форме авторизации произойдет вход в систему и появится главный экран, содержащий основной контент страницы (см. Рисунок 3).

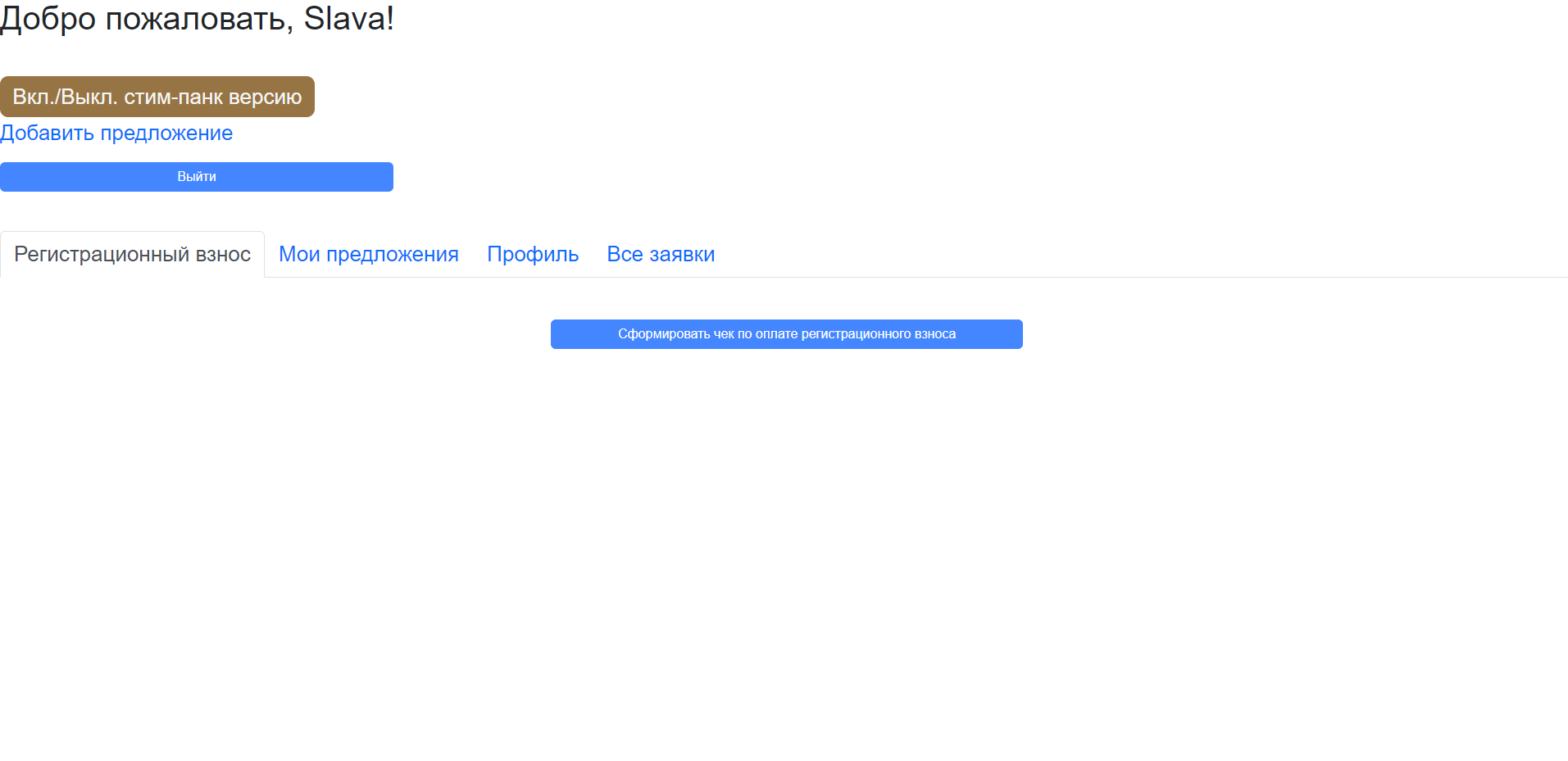


Рисунок 3 - Главная страница сайта

При нажатии на кнопку «Сформировать чек по оплате регистрационного взноса» появится модальное окно с квитанцией об уплате регистрационного взноса, в котором автоматически сгенерируется номер квитанции, кем подписана квитанция, кто ее оплатил, дата оплаты и стоимость. После нажатия на кнопку «Сохранить чек» данные с чека сохранятся в базе в таблице «receipts» (см. Рисунок 4).

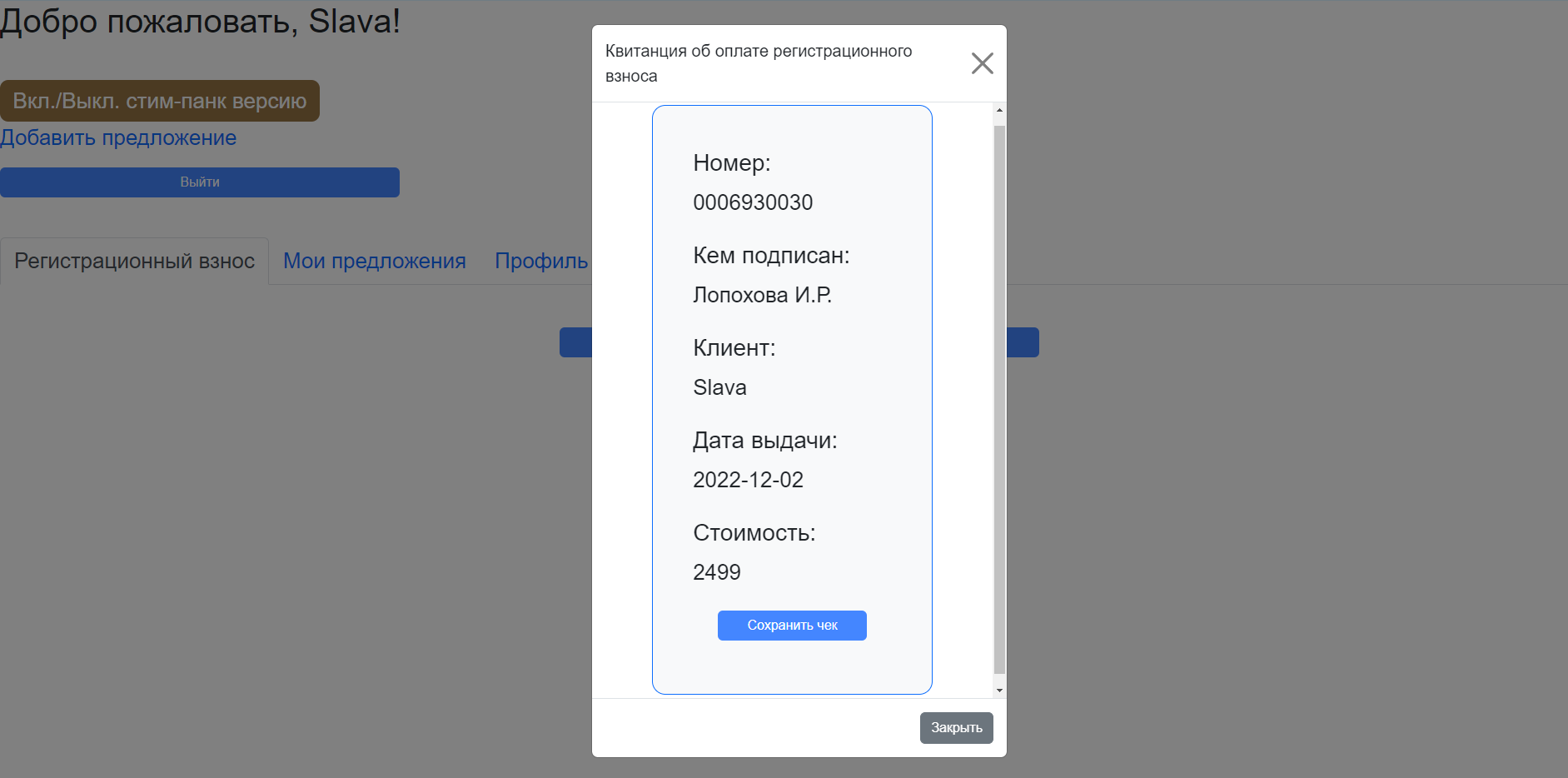


Рисунок 4 - Модальное окно квитанции об уплате регистрационного взноса

При помощи Bootstrap мне легко удалось реализовать переход по вкладкам без перезагрузки страницы. Переходя во вкладку «Мои предложения», пользователь увидит форму для заполнения заявки на поиск работы, либо на поиск работника (все зависит от поставленной в чекбоксе галочки), у любого пользователя нет ограничений на количество добавленных заявок. Также в левом верхнем углу присутствует кнопка «Добавить предложение», которое добавляет еще одну форму идентичную первой. На форме есть две кнопки, первая – «Предложить заявку», которая отвечает за сохранения данных, полученных из формы, в таблицу «offers» базы данных, и вторая – «Выгрузить», отвечающая за выгрузку (показ) всех сохраненных заявок конкретного пользователя на текущий момент (см. Рисунки 5-7)

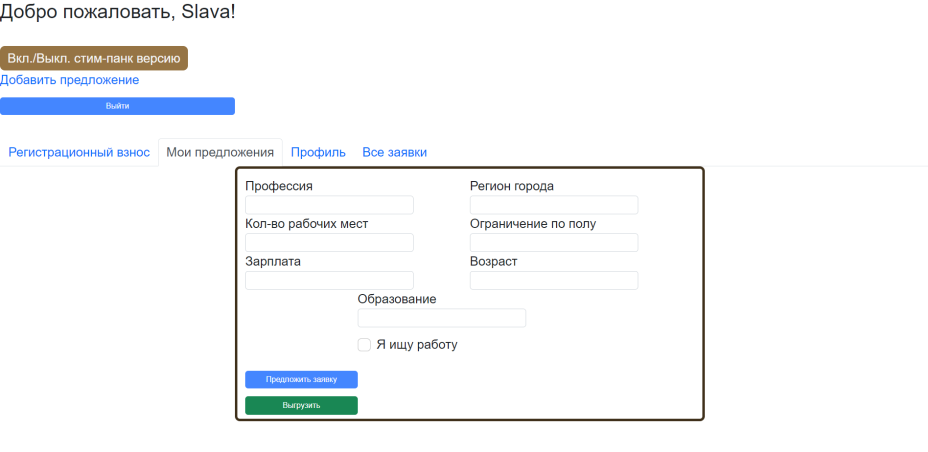


Рисунок 5 - Вкладка «Мои предложения»

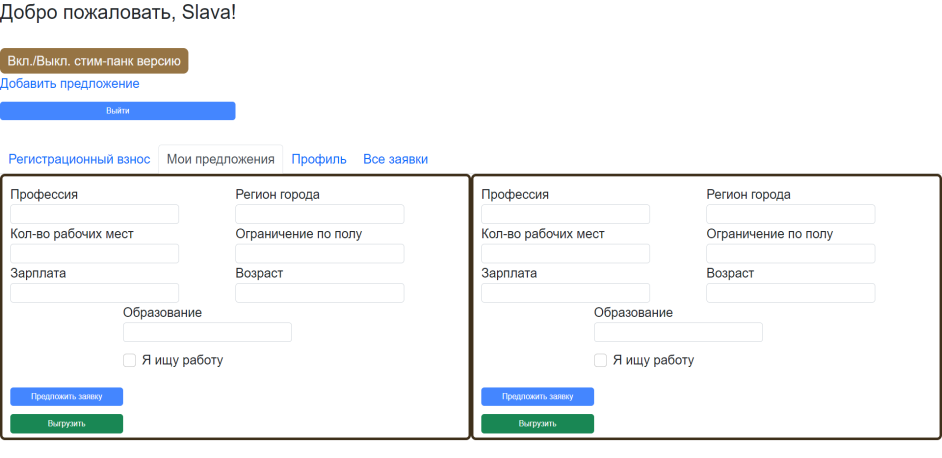


Рисунок 6 - Результат нажатия на кнопку «Добавить предложение»



Рисунок 7 - Результат нажатия на кнопку «Выгрузить»

На сайте предполагается просмотр профиля пользователя, записей, опубликованных им и, по моему мнению, пользователь может опубликовывать предложения и заявки, как работодатель, так и работник, поэтому на вкладке «Профиль» я реализовал заполнение сразу двух профилей (работника и работодателя). Достаточно простой и минимальный интерфейс с плейсхолдерами для того, чтобы было более понятно какую информацию и в каком формате нужно вводить. В теории, если развить эту работу дальше, то есть возможность усовершенствовать систему заявок. К примеру, предложить заявку может только пользователь, заполнивший формы на вкладке «Профиль», а также если пользователь уже вводил данные в эту форму, то подгружать их при следующем заходе на сайт (или при обновлении страницы, или через AJAX-запрос без перезагрузки страницы) (см. Рисунок 8).

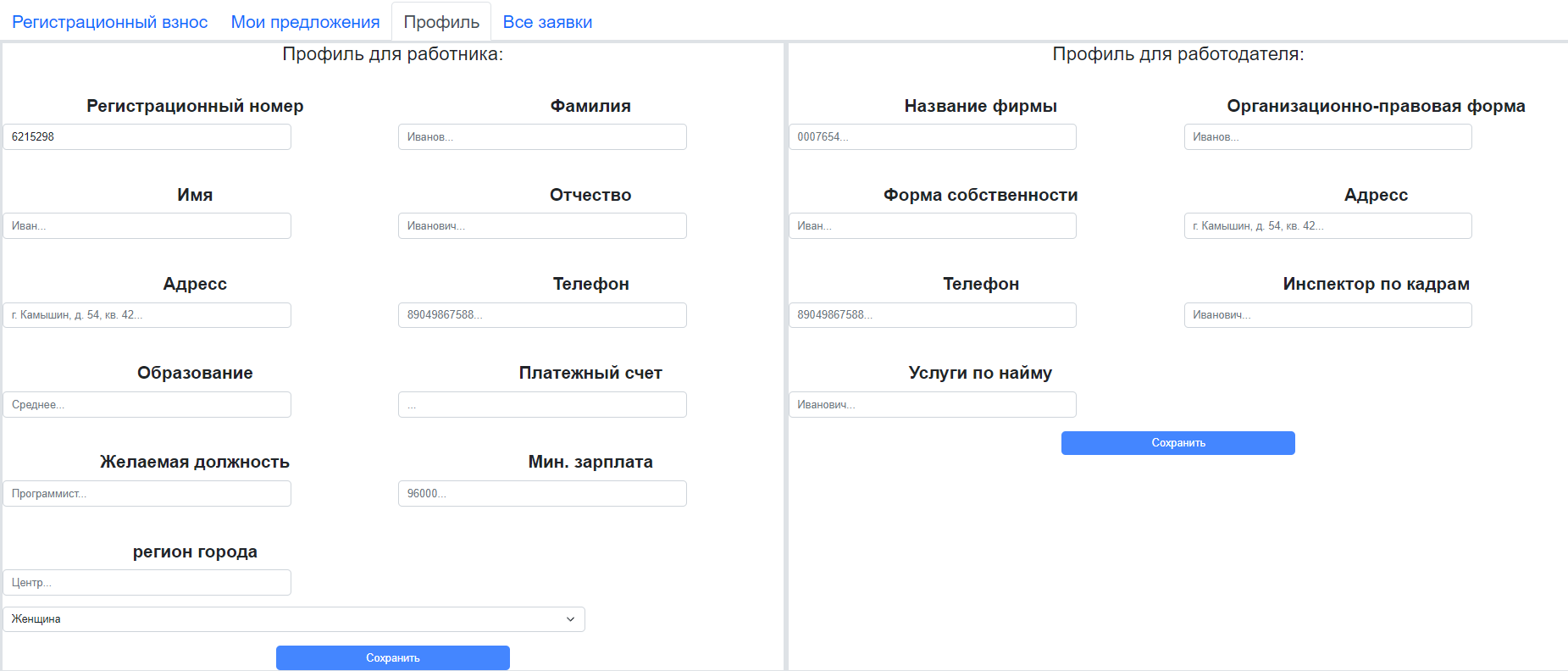


Рисунок 8 - Вкладка «Профиль»

Перейдя на вкладку «Все заявки», где реализован фильтр данных, в котором по умолчанию стоит значение «Все», пользователь увидит все заявки в табличном виде, выгруженные из базы в табличном виде (см. Рисунок 9). Таблица генерируется при заходе на сайт в файле «index.php» в достаточно интересной форме. Сначала генерируются заголовки таблицы, и выводится открывающийся тег «<form method =”POST”>», после, запускается цикл «for» по длину массива «$result», в котором хранятся все записи таблицы «offers», а потом я записываю в переменные данные, соответствующие полям таблицы (в id записываю id из таблицы). Причем поля таблицы не просто надписи, а input’ы (чтобы читать данные формы нужны input’ы) с атрибутом «readonly», чтобы пользователь не смог изменить данные полей. Проделав необходимую «подготовку», я генерирую саму таблицу, всё в том же цикле «for», по строкам с помощью индекса «$i». (Генерацию таблицы см. на рисунке 10).

Чтобы уберечь данные от хакерских атак, таких как XSS-атак или SQL-инъекций, реализована защита переменных при помощи функции «htmlspecialchars()» (также можно было реализовать то же самое при помощи аналога – «htmlentities()»). Вкратце, в HTML есть сущности или мнемоники. Это когда я пишу прямо в HTML определённую последовательность символов, например &copy; а браузер отображает соответствующий этой мнемонике символ, в данном случае © - значок копирайта. Когда мы запускаем функцию htmlspecialchars(), она берёт нашу строку и заменяет некоторые символы в ней (кавычки, угловые скобки и т.д.) на мнемоники, чтобы браузер гарантированно вывел нашу строку на экран как строку, не пытаясь выполнять её как код. То есть когда мы введём в нашу форму текст <script>alert('hello')</script>, функция htmlspecialchars() превратит его в &lt;script&gt;alert('hello')&lt;/script&gt;. Разумеется, браузер уже не воспримет такой код как javascript и просто выведет на экран как есть. Делая вывода, злоумышленник не сможет своровать данные при помощи XSS инъекций javascript’а непосредственно в HTML.

В сгенерированной таблице нужно реализовать функцию согласования заявок и для этого в таблице в последнем столбике генерируется кнопка «Согласовать». Здесь как раз и пригодился AJAX-запрос. Причина проста – без AJAX запроса очень тяжело (или даже невозможно) согласовать ту заявку, на которую пользователь нажал. Так как в цикле «for» можно вытаскивать элементы только явно, например «$result[3]» вытянет с массива только 4 элемент, поэтому в SQL запросе мне бы не удалось автоматизировать согласование заявок. То есть на каждую кнопку пришлось бы писать отдельный обработчик. В любом случае, AJAX-запрос в этом случае очень сильно упрощает жизнь, здесь нужно всего лишь задать событие по клику на кнопку и в скрипте написать функцию (см. рисунок 11), принимающую ID конкретной строки, с AJAX. Осталось записать в глобальную переменную POST значение ID строки и передать ее в обработчик «do\_Trade.php» (см. рисунок 12).

Находясь в поисках работы, пользователь может отсортировать список и в меню выбора поставить значение «Ищу работу» (см. Рисунок 13). После перезагрузки страницы и перехода на вкладку «Все заявки», пользователь увидит только предложения от работодателей (см. Рисунок 14), и наоборот, если поставить значение «Ищу работника», то после вышеперечисленных действий, пользователь увидит только заявки от работников (см. Рисунок 15).

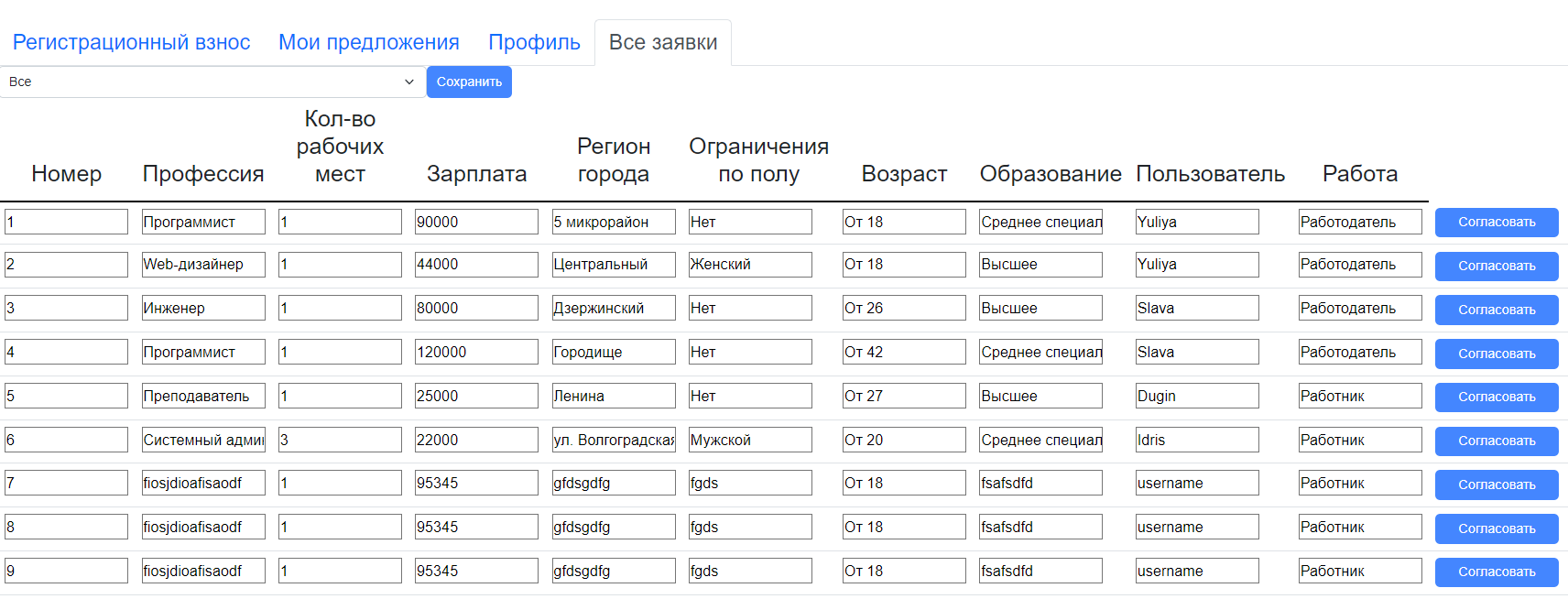


Рисунок 9 - Вкладка «Все заявки»

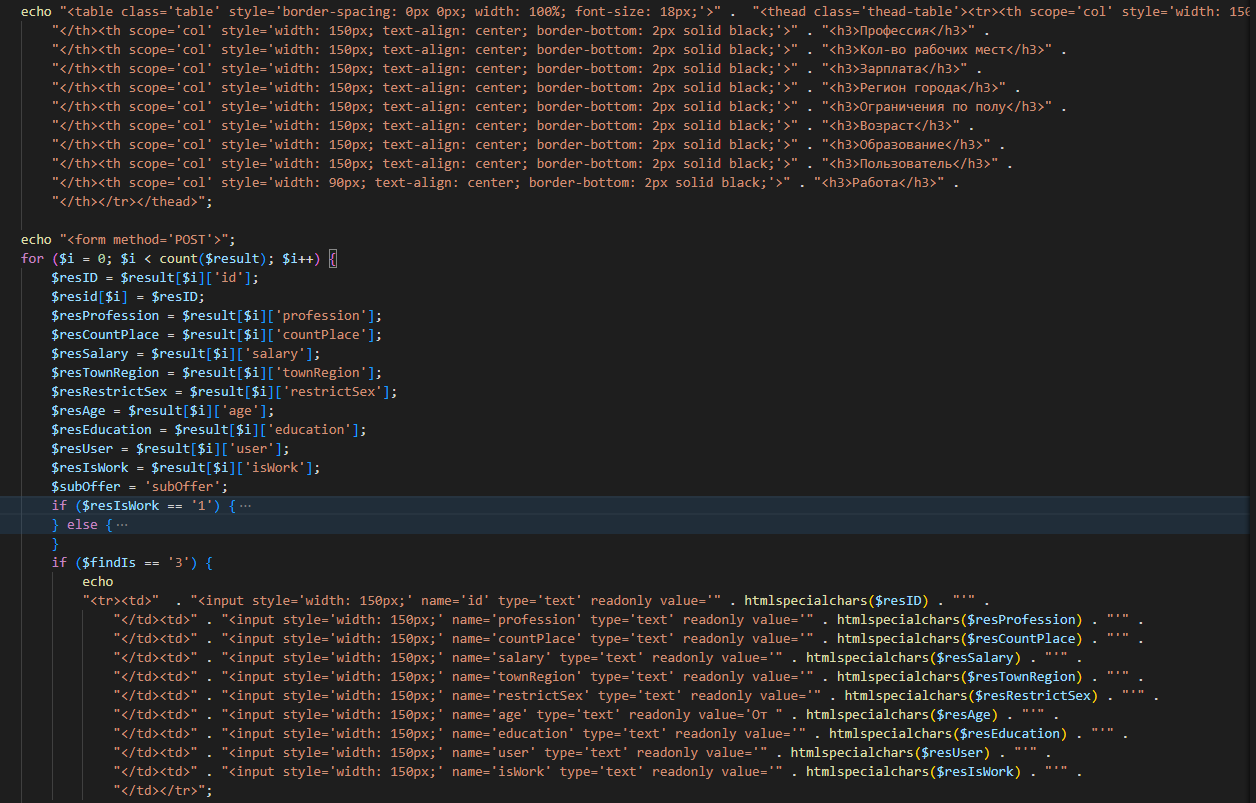


Рисунок 10 - Генерация таблицы



Рисунок 11 - Создание обработчика по событию клика на кнопке



Рисунок 12 - AJAX-запрос

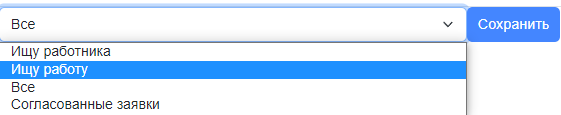


Рисунок 13 - Фильтрация таблицы

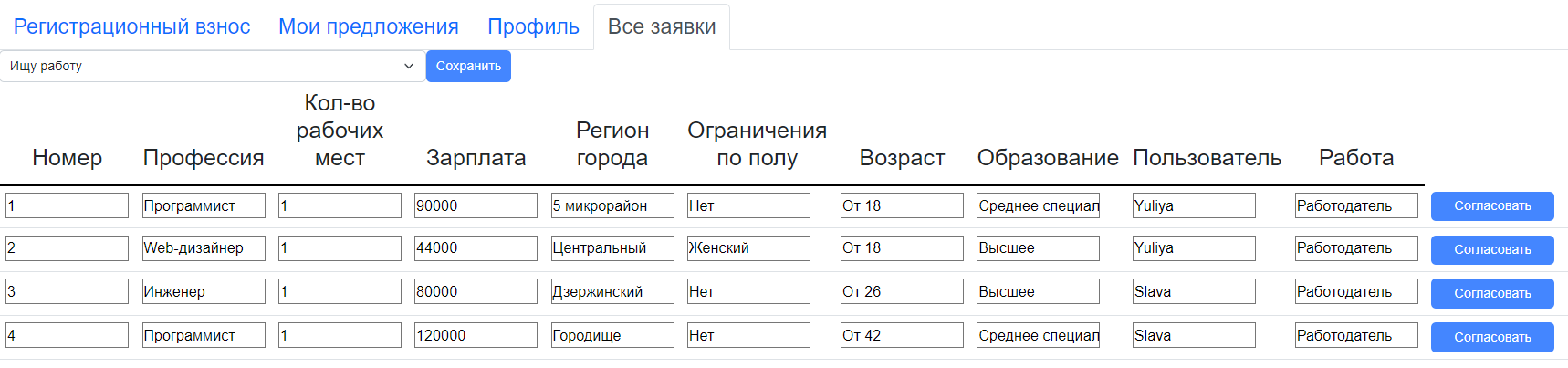


Рисунок 14 - Результат с примененной фильтрацией записей по значению «Ищу работу»

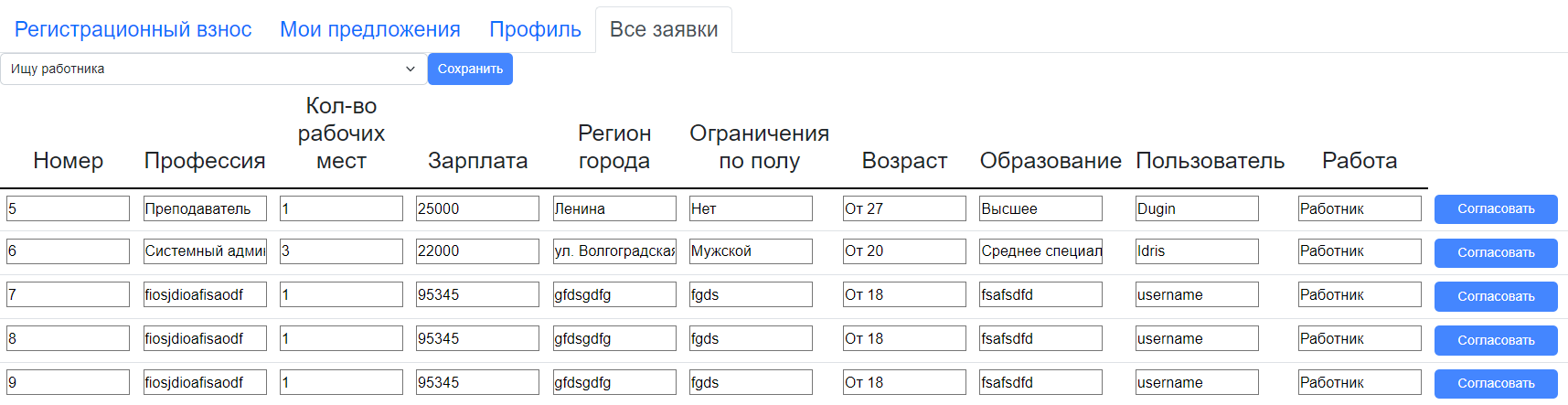


Рисунок 15 - Результат с примененной фильтрацией записей по значению «Ищу работника»

# СПИСОК ЛИТЕРАТУРЫ

1. <https://habr.com/ru>;
2. [PHP: Hypertext Preprocessor](https://www.php.net/);
3. <https://ru.wikipedia.org>;
4. [Форум программистов и сисадминов Киберфорум (cyberforum.ru)](https://www.cyberforum.ru/);
5. [Stack Overflow - Where Developers Learn, Share, & Build Careers](https://stackoverflow.com/);
6. <https://www.bibliofond.ru/>;
7. <https://getbootstrap.com/docs/5.2/>;
8. [Русская документация по API jQuery (jquery-docs.ru)](https://jquery-docs.ru/);
9. [Сайт code.mu - лучший учебник по языкам программирования | Трепачёв Дмитрий](https://www.code.mu/ru/);
10. [Современный учебник JavaScript](https://learn.javascript.ru/).