Частное учреждение образования

«Колледж бизнеса и права»

РЕАЛИЗАЦИЯ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ   
ПОИСКА ИСПОЛНИТЕЛЕЙ ДЛЯ РЕАЛИЗАЦИИ ПРОЕКТОВ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту по учебному предмету  
«Базы данных и системы управления базы данными»

КП Т.093026.401

Руководитель проекта (С.В.Банцевич)

Обучающийся (В.А.Черняков)

2024

**СОДЕРЖАНИЕ**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

КП Т.093026.401 ПЗ

Разраб.

Черняков В.А.

Провер.

Банцевич С.В.

Реценз.

Н. Контр.

Утверд.

*Реализация веб-приложения для поиска исполнителей для реализации проектов*

Лит.

Листов

КБиП

50

У

[Введение 3](#_Toc162303860)

[1 Описание задачи 5](#_Toc162303861)

[1.1 Анализ предметной области 5](#_Toc162303862)

[1.2 Постановка задачи 5](#_Toc162303863)

[2 Проектирование веб-приложение 7](#_Toc162303864)

[2.1 Проектирование модели 7](#_Toc162303865)

[2.2 Требования к веб-приложению 9](#_Toc162303866)

[2.3 Структура веб-приложения 9](#_Toc162303867)

[2.4 Проектирование макета веб-приложения 10](#_Toc162303868)

[2.5 Программно-технические средства, необходимые для разработки приложения 11](#_Toc162303869)

[2.6 Защита и сохранность данных 11](#_Toc162303870)

[2.7 Организация и ведение информационной базы (модели) 12](#_Toc162303871)

[3 Реализация веб-приложения 15](#_Toc162303872)

[3.1 Описание разделов веб-приложения 15](#_Toc162303873)

[3.2 Разработка административной части приложения 15](#_Toc162303874)

[3.3 Описание используемых функций и процедур 16](#_Toc162303875)

[3.4 Функциональное тестирование 17](#_Toc162303876)

[4 Применение 19](#_Toc162303877)

[4.1 Назначение веб-приложения 19](#_Toc162303878)

[4.2 Программно-аппаратное обеспечение сервера и клиента 19](#_Toc162303879)

[Заключение 21](#_Toc162303880)

[Список используемых источников 22](#_Toc162303881)

[Приложение А (обязательное) Структура веб-приложения 23](#_Toc162303882)

[Приложение Б (обязательное) Текст модулей веб-приложения 24](#_Toc162303883)

[Приложение В (обязательное) Результаты работы приложения 50](#_Toc162303884)

# Введение

На фоне быстрого развития информационных технологий и индустрии проектного управления, эффективное формирование команды для разработки проектов становится ключевым аспектом успешной реализации идей. Научные исследования и практика показывают, что поиск подходящих участников, сопоставление их навыков с требованиями проекта, а также управление командой, это неотъемлемые компоненты успешной реализации проектных задач.

Целью курсового проекта являться разработка проекта для поиска участников проекта и организации их работы со следующими возможностями:

- поиск участника для проекта;

- организация менеджмента проекта;

- и другое.

Решение поставленных задач содержится в пояснительной записке из шести разделов для веб-приложения.

В первом разделе «Описание задачи» содержится описание поставленной задачи и её анализа, список основных задач, которых предстоит реализовать в рамках предметной области. Раздел так же содержит анализ предметной области.

Во втором разделе «Проектирование веб-приложения» содержится подробное описание наработок будущих прототипов и моделей, а так же другие программные решения и средства для последующей реализации веб-приложения.

В третьем разделе «Реализация веб-приложения» содержится описание разработанного веб-приложения, применяемые инструменты разработки и технологии, порядок авторизации пользователей, организация данных в рамках выбранной системе управления базами данных, логическая и физическая организация функций, входные и выходные данные. В разделе так же приводится тестирование на основании подготовленных модульных тестов, описывается справочная система.

В четвертом разделе «Применение» содержится описание назначения программного средства и условий его при применения.

В пятом разделе «Охрана труда и окружающей среды» содержится информация о правовых, нормативных, социально-экономических и организационных вопросах охраны труда, о разработке мер по нормализации и защите работающих от статического электричества при эксплуатации персональной электронно-вычислительной машины (ПЭВМ), о требованиях пожарной безопасности и охране окружающей среды.

В шестом экономическом разделе приводится обоснования необходимости выведения программного средства на рынок, составляется план на разработку программного средства и рассчитываются затраты на его разработку, а также рассчитывается экономический эффект у разработчика и пользователя программного средства.

В заключении содержится общий итог о проделанной работе, анализируется степень реализации функций, извлекаются соответствующие выводы о качестве разработанного программного средства.

В приложении А представлена структура веб-приложения.

В приложении Б представлен текст программы.

В приложении В представлены результаты работы программы.

Графическая часть представлена диаграммами вариантов использования, деятельности, «Концептуальная модель данных» и «Логическая модель данных»

# Описание задачи

## Анализ предметной области

Предметной областью решаемой задачи является сфера деятельности по поиску и координации исполнителей для проектов различной направленности. В данной предметной области осуществляется контроль и управление процессом поиска, назначения и выполнения проектных задач.

Веб-приложение "Cukierka" направлено на обеспечение эффективного поиска исполнителей для различных проектов. Процесс работы с приложением выглядит следующим образом:

- регистрация и авторизация пользователей, предоставляющих проекты и желающих найти исполнителей;

- подача заявки на поиск исполнителя, где пользователи описывают требования к проекту и необходимые критерии исполнителя;

- подбор исполнителя, где система автоматически или с помощью администратора выбирает наиболее подходящих кандидатов для выполнения проекта;

- выполнение задачи, где выбранные исполнители приступают к реализации проекта в соответствии с требованиями заказчика;

- завершение и отчётность, где после завершения проекта происходит закрытие заявки, а в системе делается запись о выполненной работе и затраченных ресурсах.

На сегодняшний день для ведения учёта проектов и поиска исполнителей может использоваться ряд различных инструментов, но важно обеспечить оперативность реагирования на запросы и эффективное ведение учёта проектов для обеспечения успешной реализации проектов заказчиками.

## Постановка задачи

Исходя из анализа предметной области и требований проекта "Cukierka" выделяются следующие задачи, подлежащие автоматизации:

- запись и систематизация информации о проектах и требованиях к исполнителям;

- отслеживание статуса проектов и доступность исполнителей.

Программа "Cukierka" предоставляет следующие преимущества:

- удобство использования, приложение обладает интуитивно понятным интерфейсом, который позволяет пользователям легко регистрировать и отслеживать проекты и соответствующих им исполнителей;

- модульность, приложение может быть настроено под конкретные потребности пользователей, включая различные типы проектов и требований к исполнителям.

На сегодняшний день решение указанных задач осуществляется разрозненно и вручную. Пользователи вынуждены использовать различные инструменты для регистрации и отслеживания проектов, что часто приводит к потере данных и неэффективности в работе.

Хотя на рынке существуют аналогичные решения, они часто требуют значительных финансовых затрат и сложны в внедрении. Программа "Cukierka" разрабатывается с учетом потребностей пользователей и направлена на предоставление более гибкого и простого в использовании решения для поиска исполнителей и управления проектами.

# Проектирование веб-приложение

## Проектирование модели

Модель предметной области (инфологическая модель) – это описание предметной области, выполненное без ориентации на используемые в дальнейшем программные и технические средства. Цель инфологического проектирования заключается в представлении семантики предметной области. Инфологическая модель используется на этапе проектирования базы данных. Она помогает разработчикам базы данных понять структуру предметной области и определить, какие данные необходимо хранить в базе данных.

Диаграмма вариантов использования (UML) – это графическое представление функциональности системы с точки зрения пользователей. Диаграмма вариантов использования показывает, как пользователи взаимодействуют с системой для достижения своих целей. Диаграммы вариантов использования являются важным инструментом для проектирования и разработки программного обеспечения. Они помогают понять функциональность системы и общаться с заинтересованными сторонами. Суть диаграммы вариантов использования заключается в том, что проектируемая система представляется в виде множества сущностей или актеров, взаимодействующих с системой с помощью, так называемых вариантов использования.

Диаграмма вариантов использования представлена на листе 1 графической части.

Концептуальная модель веб-приложения представляет собой абстрактное представление структуры, основных компонентов и взаимосвязей веб-приложения. Она помогает разработчикам и дизайнерам лучше понять функциональные и нефункциональные требования к приложению, определить его основные возможности и поведение, а также логику взаимодействия между пользователем и системой.

В концептуальной модели веб-приложения мы определяем абстрактное представление структуры и компонентов приложения. На основе анализа были выделены сущности "Резюме", "Пользователи", "Вакансии" и "Отклики" с их соответствующими атрибутами.

Сущность «Резюме» имеет следующие атрибуты:

- заголовок;

- содержимое;

- навыки;

- заработная плата;

- дата создания.

Сущность «Пользователи» имеет следующие атрибуты:

- почта;

- имя;

- фамилия;

- пароль;

- дата создания;

- вид пользователя.

Сущность «Вакансии» имеет следующие атрибуты:

- заголовок;

- содержимое;

- требуемые навыки;

- заработная плата;

- дата создания.

Сущность «Отклики» имеет следующие атрибуты:

- содержимое.

Концептуальная модель реляционной базы данных представлена на листе 2 графической части.

В логической модели веб-приложения структура данных описывается с использованием сущностей и их атрибутов. Сущности представляют объекты, которые будут храниться в базе данных и взаимодействовать между собой.

На основе логической модели можно выделить сущности "Резюме", "Пользователи", "Вакансии" и "Отклики" с их соответствующими атрибутами.

Сущность «Резюме» имеет следующие атрибуты:

- уникальный идентификатор;

- заголовок;

- содержимое;

- навыки;

- заработная плата;

- дата создания.

Сущность «Пользователи» имеет следующие атрибуты:

- уникальный идентификатор;

- резюме (внешний ключ к таблице «Резюме»);

- почта;

- имя;

- фамилия;

- пароль;

- дата создания;

- вид пользователя.

Сущность «Вакансии» имеет следующие атрибуты:

- уникальный идентификатор;

- пользователь (внешний ключ к таблице «Пользователи»)

- заголовок;

- содержимое;

- требуемые навыки;

- заработная плата;

- дата создания.

Сущность «Отклики» имеет следующие атрибуты:

- уникальный идентификатор;

- пользователь (внешний ключ к таблице «Пользователи»)

- вакансия (внешний ключ к таблице «Вакансии»)

- содержимое.

Логическая модель реляционной базы данных представлена на листе 3 графической части.

## Требования к веб-приложению

Дизайн веб-приложения "Cukierka" играет важную роль в обеспечении удобства и удовлетворительного пользовательского опыта. Основные требования к графическому интерфейсу веб-приложения включают:

- простая цветовая схема, использование 2-3 основных цветов, обеспечивающих читаемость и приятный визуальный опыт;

- деловой стиль, соответствие деловому стилю, исключая элементы, которые могут создать впечатление несерьезности или некомпетентности;

- нейтральная цветовая гамма, использование нейтральных цветов, таких как белый, зеленый и чёрный, для создания профессионального впечатления;

- минималистичный дизайн, предотвращение перегруженности интерфейса, обеспечивая четкость и ясность информации;

- логотип, включение логотипа, который отражает идентичность и название веб-приложения;

- ограниченное количество шрифтов, использование не более 2-3 шрифтов на сайте для единообразия и читаемости текста;

- кросс-браузерная и адаптивная совместимость, корректное отображение интерфейса в различных браузерах (Chrome, Firefox, Safari, Edge) и на различных устройствах (ПК, планшеты, смартфоны), обеспечивая доступность для всех пользователей.

## Структура веб-приложения

Ментальная карта веб-приложения – это эффективный инструмент для понимания структуры и взаимосвязей между компонентами веб-приложения. Она может быть использована как разработчиками, так и пользователями для улучшения понимания приложения.

Ментальная карта приложения представлена на рисунке 2.1

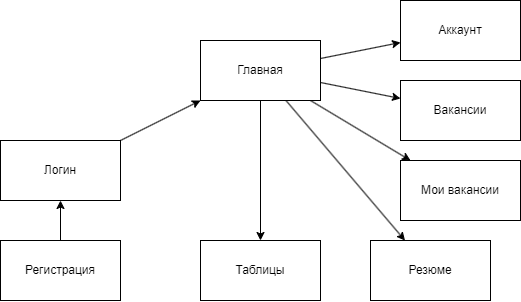
.

Рисунок 2.1

## Проектирование макета веб-приложения

Дизайн веб-приложения "Cukierka" сосредотачивается на обеспечении простоты, удобства использования и минимализма для создания профессионального визуального опыта.

Основные аспекты дизайна:

- разделение информации, информация на странице разделена на столбцы для легкого восприятия данных;

- понятные заголовки, каждый столбец снабжен понятным заголовком, который четко выделяет суть представленной информации;

- сортировка данных, пользователи имеют возможность сортировать данные по различным критериям, обеспечивая легкий доступ к необходимой информации;

- максимальное использование экранного пространства, весь экран используется для эффективного отображения информации без перегрузки интерфейса;

- баланс информативности и удобства чтения, информация представлена в удобочитаемом формате, обеспечивая максимальную информативность при минимальном напряжении для восприятия;

- поиск, реализован эффективный механизм поиска, который обеспечивает управление большим объемом данных;

- простые цвета и формы, используются простые цвета и формы, соответствующие общему стилю приложения и обеспечивающие единый и профессиональный вид интерфейса.

Макет главной страницы представляет собой форму, содержащую всю необходимую информацию для электромонтёров. Форма также включает кнопку, позволяющую пользователям загрузить отчет в формате CVS.

## Программно-технические средства, необходимые для разработки приложения

Комплекс программно-технических средств был тщательно подобран с учетом требований к веб-приложению "Cukierka" и ориентированности на эффективность разработки и поддержки. Ниже приведены основные инструменты и технологии, которые были выбраны для разработки приложения:

- операционная система Windows 10;

- среда разработки Visual Studio Code (VSCode);

- язык программирования Rust;

- язык запросов SQL;

- система управления базами данных PostgreSQL;

- система контроля версий Git.

Операционная система Windows 10 – Проприетарная операционная система для персональных компьютеров, обеспечивающая удобную среду разработки и обширную поддержку инструментов и приложений.

Среда разработки Visual Studio Code (VSCode) – Редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как легкий редактор кода для кроссплатформенной разработки веб и облачных приложений.

Язык программирования Rust – Высокоуровневый системный язык программирования, который обеспечивает безопасность памяти без использования сборщика мусора.

Язык запросов SQL – Для взаимодействия с базой данных PostgreSQL. Так же позволяет делать запросы и к другим СУБД.

Система управления базами данных PostgreSQL – Мощная и расширяемая реляционная база данных, обеспечивающая надежное хранение данных приложения.

Система контроля версий Git – Распределенная система управления версиями файлов, которая обеспечивает коллаборацию и управление исходным кодом проект

## Защита и сохранность данных

В современном мире безопасность данных является одной из главных проблем при разработке веб-приложений. Далее подробнее представленкаждый из этих аспектов.

Ограничение доступа к данным в веб-приложении осуществляется через механизмы аутентификации и авторизации. Сайт предоставляет разделение доступа для пользователей на основе от того, вошли они в аккаунт или нет. Администраторы имеют полный доступ ко всем данным и функциональности приложения, а незарегистрированные пользователи могут иметь ограниченный доступ ко всему функционалу. Все остальные пользователи получают стандартный доступ.

В программном средстве механизмы управления доступом к данным реализуются с помощью учетных записей и ролей базы данных.

## Организация и ведение информационной базы (модели)

В любой информационной системе данные хранятся в определенной структуре. Эта структура называется моделью данных. Модель данных определяет, как данные будут организованы и как к ним будет осуществляться доступ. Разрабатываемое программное средство содержит 4 таблиц для работы веб-приложения. Краткое описание таблиц проходит в таблицах 2.1-2.4.

Таблица «Резюме» хранит информацию о бригадах. Структура таблицы представлена в таблице 2.1.

Таблица 2.1

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип данных | Размер | Ограничения |
| id | UUID | 16 bytes | PRIMARY KEY |
| title | VARCHAR | varies | NOT NULL |
| body | VARCHAR | varies | NOT NULL |
| payment | INTEGER | 4 bytes | NOT NULL |
| skill | VARCHAR[] | varies | NOT NULL |
| created | TIMESTAMPTZ | 8 bytes | NOT NULL |

Таблица «Пользователи» хранит информацию о электромонтёрах. Структура таблицы представлена в таблице 2.2.

Таблица 2.2

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип данных | Размер | Ограничения |
| id | UUID | 16 bytes | PRIMARY KEY |
| resume\_id | UUID | 16 bytes | FOREIGN KEY (Resumes.id) ON DELETE CASCADE |
| email | VARCHAR | varies | NOT NULL |
| firstname | VARCHAR | varies | NOT NULL |
| secondname | VARCHAR | varies | NOT NULL |
| password | VARCHAR | varies | NOT NULL |
| created | TIMESTAMPTZ | 8 bytes | NOT NULL |
| type | CHAR | 1 byte | NOT NULL |

Таблица «Вакансии» хранит информацию о причинах поломки. Структура таблицы представлена в таблице 2.3.

Таблица 2.3

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип данных | Размер | Ограничения |
| id | UUID | 16 bytes | PRIMARY KEY |
| user\_id | UUID | 16 bytes | FOREIGN KEY (Users.id) ON DELETE CASCADE |
| title | VARCHAR | varies | NOT NULL |
| body | VARCHAR | varies | NOT NULL |
| payment | INTEGER | 4 bytes | NOT NULL |
| skill | VARCHAR[] | varies | NOT NULL |
| created | TIMESTAMPTZ | 8 bytes | NOT NULL |

Таблица «Отклики» хранит информацию различных моделях ТСО. Структура таблицы представлена в таблице 2.4.

Таблица 2.4

|  |  |  |  |
| --- | --- | --- | --- |
| Столбец | Тип данных | Размер | Ограничения |
| id | UUID | 16 bytes | PRIMARY KEY |
| user\_id | UUID | 16 bytes | FOREIGN KEY (Users.id) ON DELETE CASCADE |
| vacancy\_id | UUID | 16 bytes | FOREIGN KEY (Vacancies.id) ON DELETE CASCADE |
| body | VARCHAR | varies | NOT NULL |
| created | TIMESTAMPTZ | 8 bytes | NOT NULL |

Модели данных делятся на логические и физические. Логическая модель определяет структуру данных независимо от их физического хранения, включая сущности, атрибуты и отношения. Физическая модель определяет конкретные способы хранения данных на диске, такие как типы данных, индексы и организация файлов. Обе модели играют ключевую роль в разработке информационных систем, обеспечивая эффективное управление и использование данных.

Физическая модель данных разрабатываемого веб-приложения представлена на рисунке 2.2.

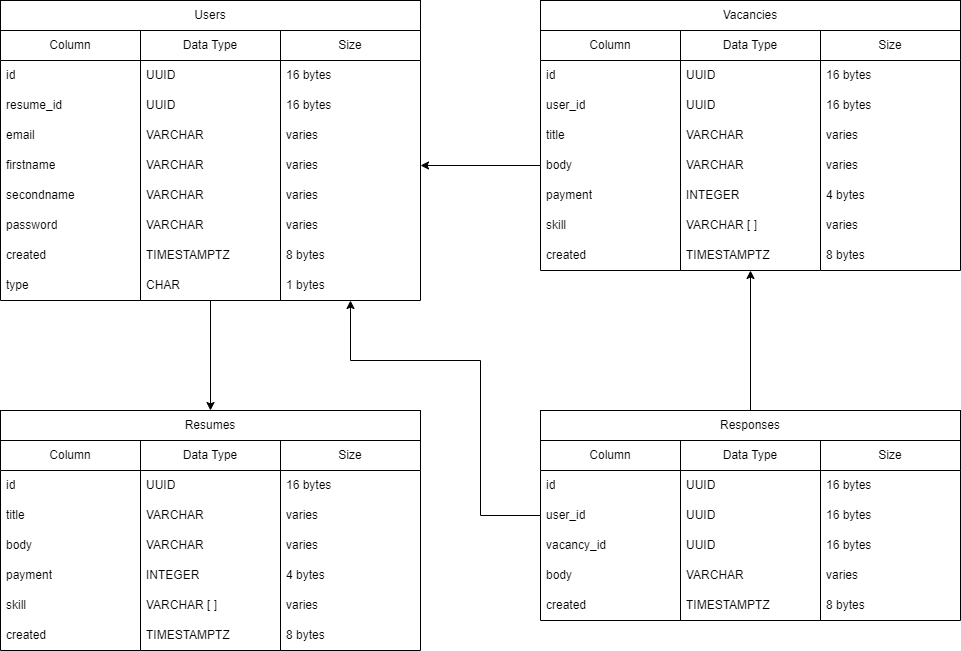


Рисунок 2.2

# Реализация веб-приложения

## Описание разделов веб-приложения

На главной странице представлено навигационное меню веб-приложения и форма для предоставления всей информации, которая необходима электромонтёрам. Форма также может иметь кнопку, которая позволяет пользователю получить отчёт в CVS-форме. Так же обладает поиском.

Навигационное меню – это область интерфейса, на которой представлены навигационные опции, которые позволяют пользователям находить контент веб-приложения.

Навигационное меню предоставляет удобный доступ к основным разделам приложения, обеспечивая ему возможность просмотра, добавления и редактирования информации.

Структура навигационного меню на главная странице представлена ниже:

- главная страница;

- страница резюме;

- страница с вакансиями;

- страница с «моими» вакансиями;

- аккаунт;

- таблицы;

- выйти.

## Разработка административной части приложения

Логическая карта – концептуальная модель системы, которая описывает ее структуру и основные компоненты на абстрактном уровне. Это высокоуровневое представление системы, независимое от конкретных технологий и деталей реализации. Логическая карта обычно включает в себя описание ключевых компонентов системы, их взаимосвязей и основных функциональных возможностей.

Физическая структура – реальные компоненты и артефакты системы, такие как файлы, папки, базы данных. Это низкоуровневое представление системы, которое учитывает конкретные технологии, инфраструктуру и детали реализации. Физическая структура определяет, как компоненты системы фактически реализуются и взаимодействуют между собой в реальной среде.

Физическая структура для администратора веб-приложения представлена на рисунке 3.1.

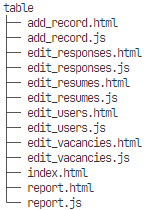


Рисунок 3.1 – Структура веб-приложения

Навигационное меню должно быть интуитивно понятным и легкодоступным, чтобы пользователь мог быстро ориентироваться и переходить между различными страницами приложения. Оно предоставляет быстрый доступ к ко всем функциям приложения, позволяя пользователям эффективно управлять данными и осуществлять необходимые действия.

## Описание используемых функций и процедур

Для обеспечения функциональности веб-приложения "Cukierka" был разработан набор JavaScript-функций и процедур. Ниже приведено описание каждой из этих функций:

Функция getCookie используется для извлечения значения cookie по его имени. Она принимает имя cookie в качестве аргумента и возвращает его значение. Если cookie с указанным именем не найден, функция возвращает null. Код функции представлен ниже.

function getCookie(name) {

var cookies = document.cookie.split(";"); // Разделяем строку куки на отдельные куки

for (var i = 0; i < cookies.length; i++) {

var cookie = cookies[i].trim(); // Удаляем лишние пробелы

if (cookie.startsWith(name + "=")) { // Проверяем, начинается ли текущая куки с искомого имени

return cookie.substring(name.length + 1); // Возвращаем значение куки (без имени)

}

}

return null; // Если куки с указанным именем не найдено

}

Функция deleteCookie удаляет cookie с указанным именем. Она устанавливает истекший срок действия (expires) для cookie на прошедшую дату, чтобы браузер мог удалить его. Код функции представлен ниже.

function deleteCookie(name) {

document.cookie = name + "=; expires=Thu, 01 Jan 1970 00:00:00 UTC; path=/;";

}

Функция setCookie устанавливает новый cookie с заданным именем, значением и сроком действия в днях. Если срок действия не указан, cookie будет действовать только в текущей сессии. Код функции представлен ниже.

function setCookie(name, value, days) {

var expires = "";

if (days) {

var date = new Date();

date.setTime(date.getTime() + (days \* 24 \* 60 \* 60 \* 1000));

expires = "; expires=" + date.toUTCString();

}

document.cookie = name + "=" + value + expires + "; path=/";

}

Функция sendSql отправляет SQL-запрос на сервер для выполнения. Она принимает строку SQL в качестве аргумента и возвращает данные, полученные в ответ на запрос. Код функции представлен ниже.

async function sendSql(sql) {

const response = await fetch("/api/sql", {

method: "POST",

headers: {

"Content-Type": "application/json"

},

body: JSON.stringify({ sql: sql })

});

const data = await response.json();

return data;

}

Функция deleteCookie удаляет cookie с указанным именем. Она устанавливает истекший срок действия (expires) для cookie на прошедшую дату, чтобы браузер мог удалить его. Код функции представлен ниже.

function deleteCookie(name) {

document.cookie = name + "=; expires=Thu, 01 Jan 1970 00:00:00 UTC; path=/;";

}

Функция getQueryParam извлекает значение параметра из строки запроса URL текущей страницы по его имени. Она принимает имя параметра в качестве аргумента и возвращает его значение. Код функции представлен ниже.

async function digestMessage(message) {

const msgUint8 = new TextEncoder().encode(message); // encode as (utf-8) Uint8Array

const hashBuffer = await crypto.subtle.digest("SHA-256", msgUint8); // hash the message

const hashArray = Array.from(new Uint8Array(hashBuffer)); // convert buffer to byte array

const hashHex = hashArray

.map((b) => b.toString(16).padStart(2, "0"))

.join(""); // convert bytes to hex string

return hashHex;

}

## Функциональное тестирование

Функциональное тестирование – это тестирование функций приложения на соответствие всем требованиям и проводится для выявления различных неполадок и недочетов программной реализации веб-приложения на этапе её сдачи в эксплуатацию. Представляет собой комплекс ключевых мероприятий по проверке программного обеспечения, по результатам которых устанавливается соответствие этого ПО исходным требованиям заказчика.

В таблице 3.1 представлены тест-кейсы для проведения функционального тестирования. Предварительное условие: запустить программу и находиться главной странице веб-приложения.

Таблица 3.1

|  |  |  |  |
| --- | --- | --- | --- |
| № | Модуль/Функция | Шаги воспроизведения | Результат |
| 1 | Регистрация пользователя | Перейти на страницу регистрации. Ввести данные: имя, электронная почта, пароль. Нажать кнопку "Зарегистрироваться". | Успешное отображение страницы регистрации. Пользователь успешно зарегистрирован. Перенаправление на страницу авторизации. |
| 2 | Авторизация пользователя | Перейти на страницу авторизации. Ввести данные: электронная почта, пароль. Нажать кнопку "Войти". | Успешное отображение страницы авторизации. Пользователь успешно авторизован. Перенаправление на домашнюю страницу. |
| 3 | Добавление резюме | Авторизоваться. Перейти на страницу добавления резюме. Заполнить обязательные поля. Нажать кнопку "Сохранить". | Успешное отображение страницы добавления резюме. Резюме успешно добавлено. Перенаправление на список резюме. |
| 4 | Управление веб-приложением | Авторизоваться как администратор. Перейти в раздел управления пользователями. Выбрать пользователя для блокировки. Сохранить изменения. | Успешное отображение страницы управления пользователями. Пользователь успешно заблокирован. Изменения сохранены. |
| 5 | Формирование отчетов | Авторизоваться как администратор. Перейти в раздел формирования отчетов. Выбрать тип и период отчета. Выбрать формат экспорта. | Успешное отображение страницы формирования отчетов. Отчет сформирован. Отчет экспортирован в CVS. |

# Применение

## Назначение веб-приложения

Веб-приложение "Cukierka" разработано для облегчения процесса поиска исполнителей для реализации проектов и является важным инструментом для эффективного управления заданиями и услугами. Оно обеспечивает оперативное взаимодействие между заказчиками и исполнителями, способствуя быстрому подбору подходящих кандидатов и решению возникающих задач.

Целью разработки данного веб-приложения является автоматизация и упрощение процесса поиска исполнителей и управления проектами. Это позволяет сократить время на подбор исполнителей, улучшить качество выполнения работ и повысить удовлетворенность заказчиков.

Установка и запуск веб-приложения осуществляются на удаленном сервере без необходимости использования дополнительных веб-серверов или WSGI-серверов. Это обеспечивает простоту и гибкость в настройке и запуске приложения.

Программное обеспечение развернуто на сервере и включает в себя все необходимые компоненты для его функционирования, обеспечивая готовность к использованию без дополнительных конфигураций.

Веб-приложение предоставляет пользователям возможность регистрации, добавления и управления своими резюме или вакансиями, а также отклика на соответствующие запросы. Оно ориентировано на пользователей, ищущих возможности для сотрудничества в рамках различных проектов.

Однако следует отметить, что веб-приложение имеет ограниченный функционал и предназначено исключительно для управления поиском исполнителей и проектами. Доступ к приложению имеют только авторизованные пользователи, обеспечивая безопасность данных и контроль над использованием приложения.

## Программно-аппаратное обеспечение сервера и клиента

Для корректной работы веб-приложения на сервере требуется определенное программное обеспечение, которое включает в себя:

- процессор Intel Pentium Gold G6400 OEM;

- памяти 32 ГБ ОЗУ;

- памяти 32 ГБ свободного места;

- операционная система Arch Linux;

- язык программирования Rust;

- система управления базами данных PostgreSQL;

- фреймворк HTTP-сервер Warp.

После установки и настройки необходимого программного обеспечения на сервере веб-приложение может корректно функционировать и обрабатывать запросы от клиентов.

На стороне клиента требования более общие, которые представлены ниже:

- веб-браузер (Google Chrome, Mozilla Firefox, Safari, Microsoft Edge или Opera);

- стабильное интернет-соединение.

С учетом этих требований веб-приложение будет доступно для использования как на стороне сервера, так и на стороне клиента, обеспечивая стабильную и надежную работу для пользователей.

# Заключение

В ходе выполнения поставленной задачи было разработано веб-приложение "Cukierka" для поиска исполнителей для реализации проектов. Процесс разработки включал создание функционала регистрации и аутентификации пользователей, управления резюме и вакансиями, а также добавление и управление услугами и заданиями.

Анализируя выполнение поставленной задачи, можно отметить, что все её компоненты были реализованы в соответствии с требованиями и ожиданиями. Веб-приложение успешно обеспечивает эффективный поиск исполнителей для проектов и автоматизирует учет информации о них.

Проектные решения полностью соответствуют поставленному заданию. Веб-приложение было успешно развернуто на удаленном сервере. Оно ориентировано на конкретную область применения и эффективно решает поставленные задачи.

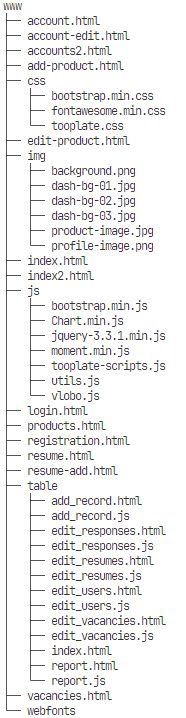
При анализе степени соответствия проектных решений заданию не было обнаружено несоответствий. Проект был выполнен в полном объеме и в срок, соответствуя всем требованиям.

Таким образом, можно сделать вывод о успешном выполнении поставленной задачи и высоком уровне соответствия проектных решений её требованиям. Веб-приложение функционирует стабильно и эффективно, обеспечивая удобство использования как для пользователей, так и для администраторов.

# Список используемых источников

1. Багласова, Т.Г. Методические рекомендации по организации и проведению производственной практики специальности 2-40 01 01 «Программное обеспечение информационных технологий» / К.О. Яхимович, Т.Г. Багласова Минск; КБП, 2022 33 с
2. Багласова, Т.Г. Методические указания по оформлению курсовых проектов, дипломных проектов и отчетов для учащихся специальности 2-40 01 01 «Программное обеспечение информационных технологий» / К.О. Яхимович, С.В. Банцевич, Т.Г. Багласова Минск; КБП, 2022 49 с
3. Возможности Dr.Explain - быстрое создание пользовательской и справочной документации [Электронный ресурс]. – Режим доступа : https://www.drexplain.ru/features. – Дата доступа : 14.03.2024
4. Документация по основам .NET [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/dotnet/fundamentals. – Дата доступа : 13.03.2024
5. Полная документация WPF — элементы управления — пользовательский интерфейс Telerik для WPF [Электронный ресурс]. –  Режим доступа : https://docs.telerik.com/devtools/wpf/introduction. – Дата доступа : 14.03.2024
6. Прайс Марк, Дж. C# 7 и .NET Core. Кросс-платформенная разработка для профессионалов / Дж. Прайс Марк – 3-е изд. – СПб.: Питер, 2018. – 640 с.: ил. – Серия : «Библиотека программиста»
7. Техническая документация по Windows для разработчиков и ИТ-специалистов [Электронный ресурс]. – Режим доступа : https://learn.microsoft.com/ru-ru/windows. – Дата доступа : 14.03.2024
8. Microsoft SQL Server [Электронный ресурс]. – Режим доступа : http://www.tadviser.ru/index.php/Продукт:Microsoft\_SQL\_Server. – Дата доступа : 14.03.2024.
9. SQL Server Management Studio components and configuration [Электронный ресурс]. – Microsoft, 2023. – Режим доступа : https://www.diagrams.net/. – Дата доступа : 14.03.2024.
10. Visual Studio [Электронный ресурс]. – Microsoft, 2022. – Режим доступа : https://visualstudio.microsoft.com/vs/. – Дата доступа : 14.03.2024.

# Приложение А (обязательное) Структура веб-приложения



# Приложение Б (обязательное) Текст модулей веб-приложения

function getCookie(name) {

var cookies = document.cookie.split(";"); // Разделяем строку куки на отдельные куки

for (var i = 0; i < cookies.length; i++) {

var cookie = cookies[i].trim(); // Удаляем лишние пробелы

if (cookie.startsWith(name + "=")) { // Проверяем, начинается ли текущая куки с искомого имени

return cookie.substring(name.length + 1); // Возвращаем значение куки (без имени)

}

}

return null; // Если куки с указанным именем не найдено

}

function deleteCookie(name) {

document.cookie = name + "=; expires=Thu, 01 Jan 1970 00:00:00 UTC; path=/;";

}

function setCookie(name, value, days) {

var expires = "";

if (days) {

var date = new Date();

date.setTime(date.getTime() + (days \* 24 \* 60 \* 60 \* 1000));

expires = "; expires=" + date.toUTCString();

}

document.cookie = name + "=" + value + expires + "; path=/";

}

// "SELECT table\_name FROM information\_schema.tables WHERE table\_schema = 'public';";

async function sendSql(sql) {

const response = await fetch("/api/sql", {

method: "POST",

headers: {

"Content-Type": "application/json"

},

body: JSON.stringify({ sql: sql })

});

const data = await response.json();

return data;

}

async function digestMessage(message) {

const msgUint8 = new TextEncoder().encode(message); // encode as (utf-8) Uint8Array

const hashBuffer = await crypto.subtle.digest("SHA-256", msgUint8); // hash the message

const hashArray = Array.from(new Uint8Array(hashBuffer)); // convert buffer to byte array

const hashHex = hashArray

.map((b) => b.toString(16).padStart(2, "0"))

.join(""); // convert bytes to hex string

return hashHex;

}

async function getQueryParam(parameterName) {

// Получаем параметры строки запроса URL текущей страницы

const queryString = window.location.search;

// Создаем объект URLSearchParams из строки запроса

const urlParams = new URLSearchParams(queryString);

// Получаем значение параметра по его имени

return urlParams.get(parameterName);

}

let user

async function checkAuth() {

var login = getCookie("\_e");

var password = getCookie("\_p");

if (!login || !password) {

window.location.href = "login.html";

}

result = await sendSql(`SELECT \* FROM Users WHERE email = '${login}' and password = '${password}'`);

if (result.error !== null && result.rows !== null && result.rows.length == 0) {

window.location.href = "login.html";

} else {

user = result.rows[0];

document.getElementById("nav-my-account").href = "account.html?id=" + user.id;

if (user.resume\_id == null) {

document.getElementById("nav-resume").href = "resume-add.html";

}else{

document.getElementById("nav-resume").href = "resume.html?id=" + user.resume\_id;

}

if (user.type == 'a') {

document.getElementById("nav-table-block").style.display = "block";

}

}

}

async function login() {

let email = document.getElementById("email").value;

let password = await digestMessage(document.getElementById("password").value);

let sqlQuery = `SELECT \* FROM Users WHERE email = '${email}' AND password = '${password}';`;

let result = await sendSql(sqlQuery);

console.log(result)

if (result.error !== null && result.rows !== null) {

if (result.rows.length >= 1) {

setCookie("\_e", email, 30);

setCookie("\_p", password, 30);

window.location.href = "index.html";

}

alert("Неверный логин или пароль. Перепроверьте данные.")

} else {

alert("Не удалось выполнить операцию. Перепроверьте данные или попробуйте позже.")

}

}

async function userload() {

let id = await getQueryParam('id')

result = await sendSql(`SELECT \* FROM Users WHERE id = '${id}'`);

if (result.error !== null && result.rows !== null) {

if (result.rows.length >= 1) {

account = result.rows[0];

console.log(account)

document.getElementById("firstname").value = account.firstname;

document.getElementById("secondname").value = account.secondname;

document.getElementById("email").value = account.email;

document.getElementById("usertype").value = account.type;

document.getElementById("datecreated").value = account.created;

} else {

window.location.href = "404";

}

} else {

alert("Не удалось выполнить операцию. Перепроверьте данные или попробуйте позже.")

}

let email = document.getElementById("email").value;

let firstname = document.getElementById("firstname").value;

let secondname = document.getElementById("secondname").value;

let usertype = document.getElementById("usertype").value;

let userId = await crypto.randomUUID();

let currentDate = new Date().toISOString();

}

async function resumeload() {

let id = await getQueryParam('id')

result = await sendSql(`SELECT \* FROM Resumes WHERE id = '${id}'`);

if (result.error !== null && result.rows !== null) {

if (result.rows.length >= 1) {

resume = result.rows[0];

console.log(resume)

document.getElementById("title").value = resume.title;

document.getElementById("body").value = resume.body;

document.getElementById("payment").value = resume.payment;

document.getElementById("skill").value = resume.skill;

document.getElementById("datecreated").value = resume.created;

} else {

window.location.href = "404";

}

} else {

alert("Не удалось выполнить операцию. Перепроверьте данные или попробуйте позже.")

}

let email = document.getElementById("email").value;

let firstname = document.getElementById("firstname").value;

let secondname = document.getElementById("secondname").value;

let usertype = document.getElementById("usertype").value;

let userId = await crypto.randomUUID();

let currentDate = new Date().toISOString();

}

async function exit() {

deleteCookie("\_e");

deleteCookie("\_p");

window.location.href = "login.html";

}

async function registration() {

let email = document.getElementById("email").value;

let firstname = document.getElementById("firstname").value;

let secondname = document.getElementById("secondname").value;

let usertype = document.getElementById("usertype").value;

let password = await digestMessage(document.getElementById("password").value);

let userId = await crypto.randomUUID();

let currentDate = new Date().toISOString();

let sqlQuery = `INSERT INTO Users (id, email, firstname, secondname, password, created, type)

VALUES ('${userId}', '${email}', '${firstname}', '${secondname}', '${password}', '${currentDate}', '${usertype}');`;

let result = await sendSql(sqlQuery);

console.log(result)

if (result.error !== null) {

window.location.href = "login.html";

} else {

alert("Не удалось выполнить операцию. Перепроверьте данные или попробуйте позже.")

}

}

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Редактировать пользователя</title>

<!--

Template 2108 Dashboard

http://www.tooplate.com/view/2108-dashboard

-->

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,600">

<!-- https://fonts.google.com/specimen/Open+Sans -->

<link rel="stylesheet" href="css/fontawesome.min.css">

<!-- https://fontawesome.com/ -->

<link rel="stylesheet" href="jquery-ui-datepicker/jquery-ui.min.css" type="text/css" />

<!-- http://api.jqueryui.com/datepicker/ -->

<link rel="stylesheet" href="css/bootstrap.min.css">

<!-- https://getbootstrap.com/ -->

<link rel="stylesheet" href="css/tooplate.css">

</head>

<body class="bg02">

<div class="container">

<div class="row">

<div class="col-12">

<nav class="navbar navbar-expand-xl navbar-light bg-light">

<a class="navbar-brand" href="#">

<i class="fas fa-3x fa-circle-notch tm-site-icon"></i>

<h1 class="tm-site-title mb-0">Cukierka</h1>

</a>

<button class="navbar-toggler ml-auto mr-0" type="button" data-toggle="collapse"

data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"

aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<ul class="navbar-nav mx-auto">

<li class="nav-item">

<a class="nav-link" href="/">Главная</a>

</li>

<li class="nav-item">

<a class="nav-link" href="resume.html">Резюме</a>

</li>

<li class="nav-item">

<a class="nav-link" href="vacancies.html">Вакансии</a>

</li>

<li class="nav-item">

<a class="nav-link" href="myvacancies.html">Мои вакансии</a>

</li>

<li class="nav-item">

<a class="nav-link active" id="nav-my-account" href="account.html">Аккаунт

<span class="sr-only">(current)</span>

</a>

</li>

</ul>

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link d-flex" onclick="exit()">

<i class="far fa-user mr-2 tm-logout-icon"></i>

<span>Выйти</span>

</a>

</li>

</ul>

</div>

</nav>

</div>

</div>

<!-- row -->

<div class="row tm-mt-big">

<div class="col-xl-8 col-lg-10 col-md-12 col-sm-12">

<div class="bg-white tm-block">

<div class="row">

<div class="col-12">

<h2 class="tm-block-title d-inline-block">Пользователь</h2>

</div>

</div>

<div class="row mt-4 tm-edit-product-row">

<div class="col-xl-7 col-lg-7 col-md-12">

<form action="" class="tm-edit-product-form">

<div class="input-group mb-3">

<label for="firstname"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Имя</label>

<input placeholder="Имя пользователя" id="firstname" name="firstname" type="text"

class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7" disabled>

</div>

<div class="input-group mb-3">

<label for="secondname"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Фамилия

</label>

<input placeholder="Фамилия пользователя" id="secondname" name="secondname"

type="text" class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7"

disabled>

</div>

<div class="input-group mb-3">

<label for="email" class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Почта

</label>

<input placeholder="Почта пользователя" id="email" name="email" type="email"

class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7" disabled>

</div>

<div class="input-group mb-3">

<label for="usertype" class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Вид

пользователя</label>

<select class="custom-select col-xl-9 col-lg-8 col-md-8 col-sm-7" id="usertype"

disabled>

<option value="l">Юридическое лицо</option>

<option value="n">Физическое лицо</option>

<option value="a" disabled>Администратор</option>

</select>

</div>

<div class="input-group mb-3">

<label for="datecreated"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Дата создания

</label>

<input placeholder="Дата создания пользователя" id="datecreated" name="datecreated"

type="text" class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7"

disabled>

</div>

<div class="input-group mb-3">

<div class="ml-auto col-xl-8 col-lg-8 col-md-8 col-sm-7 pl-0">

<button class="btn btn-primary">Резюме

</button>

</div>

</div>

<div class="input-group mb-3">

<div class="ml-auto col-xl-8 col-lg-8 col-md-8 col-sm-7 pl-0">

<button class="btn btn-primary" disabled>Редактировать

</button>

</div>

</div>

<div class="input-group mb-3">

<div class="ml-auto col-xl-8 col-lg-8 col-md-8 col-sm-7 pl-0">

<button class="btn btn-primary" disabled>Удалить

</button>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<footer class="row tm-mt-big">

<div class="col-12 font-weight-light">

<p class="d-inline-block tm-bg-black text-white py-2 px-4">

Авторские права &copy; 2024 Владислав Черняков

</p>

</div>

</footer>

</div>

<script src="js/jquery-3.3.1.min.js"></script>

<!-- https://jquery.com/download/ -->

<script src="js/bootstrap.min.js"></script>

<!-- https://getbootstrap.com/ -->

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

<script>

$(function () {

checkAuth();

userload();

});

</script>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Пользователь</title>

<!--

Template 2108 Dashboard

http://www.tooplate.com/view/2108-dashboard

-->

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,600">

<!-- https://fonts.google.com/specimen/Open+Sans -->

<link rel="stylesheet" href="css/fontawesome.min.css">

<!-- https://fontawesome.com/ -->

<link rel="stylesheet" href="jquery-ui-datepicker/jquery-ui.min.css" type="text/css" />

<!-- http://api.jqueryui.com/datepicker/ -->

<link rel="stylesheet" href="css/bootstrap.min.css">

<!-- https://getbootstrap.com/ -->

<link rel="stylesheet" href="css/tooplate.css">

</head>

<body class="bg02">

<div class="container">

<div class="row">

<div class="col-12">

<nav class="navbar navbar-expand-xl navbar-light bg-light">

<a class="navbar-brand" href="#">

<i class="fas fa-3x fa-circle-notch tm-site-icon"></i>

<h1 class="tm-site-title mb-0">Cukierka</h1>

</a>

<button class="navbar-toggler ml-auto mr-0" type="button" data-toggle="collapse"

data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"

aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<ul class="navbar-nav mx-auto">

<li class="nav-item">

<a class="nav-link" href="/">Главная</a>

</li>

<li class="nav-item">

<a class="nav-link" id="nav-resume" href="resume-add.html">Резюме</a>

</li>

<li class="nav-item">

<a class="nav-link" href="vacancies.html">Вакансии</a>

</li>

<li class="nav-item">

<a class="nav-link" href="my-vacancies.html">Мои вакансии</a>

</li>

<li class="nav-item">

<a class="nav-link active" id="nav-my-account" href="account.html">Аккаунт

<span class="sr-only">(current)</span>

</a>

</li>

<li class="nav-item" id="nav-table-block" style="display: none;">

<a class="nav-link" href="/table/index.html">Таблица</a>

</li>

</ul>

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link d-flex" onclick="exit()">

<i class="far fa-user mr-2 tm-logout-icon"></i>

<span>Выйти</span>

</a>

</li>

</ul>

</div>

</nav>

</div>

</div>

<!-- row -->

<div class="row tm-mt-big">

<div class="col-xl-8 col-lg-10 col-md-12 col-sm-12">

<div class="bg-white tm-block">

<div class="row">

<div class="col-12">

<h2 class="tm-block-title d-inline-block">Пользователь</h2>

</div>

</div>

<div class="row mt-4 tm-edit-product-row">

<div class="col-xl-7 col-lg-7 col-md-12">

<form action="" class="tm-edit-product-form">

<div class="input-group mb-3">

<label for="firstname"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Имя</label>

<input placeholder="Имя пользователя" id="firstname" name="firstname" type="text"

class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7" disabled>

</div>

<div class="input-group mb-3">

<label for="secondname"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Фамилия

</label>

<input placeholder="Фамилия пользователя" id="secondname" name="secondname"

type="text" class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7"

disabled>

</div>

<div class="input-group mb-3">

<label for="email" class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Почта

</label>

<input placeholder="Почта пользователя" id="email" name="email" type="email"

class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7" disabled>

</div>

<div class="input-group mb-3">

<label for="usertype" class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Вид

пользователя</label>

<select class="custom-select col-xl-9 col-lg-8 col-md-8 col-sm-7" id="usertype"

disabled>

<option value="l">Юридическое лицо</option>

<option value="n">Физическое лицо</option>

<option value="a" disabled>Администратор</option>

</select>

</div>

<div class="input-group mb-3">

<label for="datecreated"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Дата создания

</label>

<input placeholder="Дата создания пользователя" id="datecreated" name="datecreated"

type="text" class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7"

disabled>

</div>

<div class="input-group mb-3">

<div class="ml-auto col-xl-8 col-lg-8 col-md-8 col-sm-7 pl-0">

<a class="btn btn-primary">Резюме

</a>

</div>

</div>

<div class="input-group mb-3">

<div class="ml-auto col-xl-8 col-lg-8 col-md-8 col-sm-7 pl-0">

<a class="btn btn-primary" disabled>Редактировать

</a>

</div>

</div>

<div class="input-group mb-3">

<div class="ml-auto col-xl-8 col-lg-8 col-md-8 col-sm-7 pl-0">

<a class="btn btn-primary" disabled>Удалить

</a>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<footer class="row tm-mt-big">

<div class="col-12 font-weight-light">

<p class="d-inline-block tm-bg-black text-white py-2 px-4">

Авторские права &copy; 2024 Владислав Черняков

</p>

</div>

</footer>

</div>

<script src="js/jquery-3.3.1.min.js"></script>

<!-- https://jquery.com/download/ -->

<script src="js/bootstrap.min.js"></script>

<!-- https://getbootstrap.com/ -->

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

<script>

$(function () {

checkAuth();

userload();

});

</script>

</body>

</html>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Авторизация</title>

<!--

Template 2108 Dashboard

http://www.tooplate.com/view/2108-dashboard

-->

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,600">

<!-- https://fonts.google.com/specimen/Open+Sans -->

<link rel="stylesheet" href="css/fontawesome.min.css">

<!-- https://fontawesome.com/ -->

<link rel="stylesheet" href="css/bootstrap.min.css">

<!-- https://getbootstrap.com/ -->

<link rel="stylesheet" href="css/tooplate.css">

</head>

<body class="bg03">

<div class="container">

<div class="row tm-mt-big">

<div class="col-12 mx-auto tm-login-col">

<div class="bg-white tm-block">

<div class="row">

<div class="col-12 text-center">

<i class="fas fa-3x fa-circle-notch tm-site-icon text-center"></i>

<h2 class="tm-block-title mt-3">Авторизация</h2>

</div>

</div>

<div class="row mt-2">

<div class="col-12">

<form id="login" class="tm-login-form">

<div class="input-group">

<label for="email"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Почта</label>

<input name="email" type="text"

class="form-control validate col-xl-9 col-lg-8 col-md-8 col-sm-7" id="email"

required>

</div>

<div class="input-group mt-3">

<label for="password"

class="col-xl-4 col-lg-4 col-md-4 col-sm-5 col-form-label">Пароль</label>

<input name="password" type="password" class="form-control validate" id="password"

required>

</div>

<div class="input-group mt-3">

<button type="submit" class="btn btn-primary d-inline-block mx-auto">Войти</button>

</div>

<div class="input-group mt-3">

<a class="page-link mx-auto" href="registration.html">Регистрация</a>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

<footer class="row tm-mt-big">

<div class="col-12 font-weight-light text-center">

<p class="d-inline-block tm-bg-black text-white py-2 px-4">

Авторские права &copy; 2024 Владислав Черняков

</p>

</div>

</footer>

</div>

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

<script>

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

document.getElementById("login").addEventListener("submit", function (event) {

event.preventDefault();

login();

});

});

</script>

</body>

</html>

const width\_threshold = 480;

async function sendSql(sql) {

const response = await fetch("/api/sql", {

method: "POST",

headers: {

"Content-Type": "application/json"

},

body: JSON.stringify({ sql: sql })

});

const data = await response.json();

return data;

}

async function drawLineChart() {

if ($("#lineChart").length) {

ctxLine = document.getElementById("lineChart").getContext("2d");

// Fetch data for line chart

const lineChartData = await sendSql("SELECT \* FROM CountVacanciesResponsesResumesByDay");

const vacancyHits = [];

const responseHits = [];

const resumeHits = [];

const labels = [];

lineChartData.rows.forEach(row => {

vacancyHits.push(row.vacancy\_count);

responseHits.push(row.response\_count);

resumeHits.push(row.resume\_count);

labels.push(row.day);

});

optionsLine = {

scales: {

yAxes: [

{

scaleLabel: {

display: true,

labelString: "Количество"

}

}

]

}

};

// Set aspect ratio based on window width

optionsLine.maintainAspectRatio = $(window).width() < width\_threshold ? false : true;

configLine = {

type: "line",

data: {

labels: labels,

datasets: [

{

label: "Вакансии",

data: vacancyHits,

fill: false,

borderColor: "rgb(75, 192, 192)",

lineTension: 0.1

},

{

label: "Резюме",

data: resumeHits,

fill: false,

borderColor: "rgba(255,99,132,1)",

lineTension: 0.1

},

{

label: "Откликов",

data: responseHits,

fill: false,

borderColor: "rgba(99,255,132,1)",

lineTension: 0.1

}

]

},

options: optionsLine

};

lineChart = new Chart(ctxLine, configLine);

}

}

async function drawBarChart() {

if ($("#barChart").length) {

ctxBar = document.getElementById("barChart").getContext("2d");

// Fetch data for bar chart

const barChartData = await sendSql("SELECT \* FROM Top10PopularSkills");

const skillLabels = [];

const skillCounts = [];

barChartData.rows.forEach(row => {

skillLabels.push(row.skill);

skillCounts.push(row.skill\_count);

});

optionsBar = {

responsive: true,

scales: {

yAxes: [

{

ticks: {

beginAtZero: true

},

scaleLabel: {

display: true,

labelString: "Количество"

}

}

]

}

};

optionsBar.maintainAspectRatio = $(window).width() < width\_threshold ? false : true;

configBar = {

type: "bar",

data: {

labels: skillLabels,

datasets: [

{

label: "Требуемые навыки",

data: skillCounts,

backgroundColor: [

"rgba(255, 99, 132, 0.2)",

"rgba(54, 162, 235, 0.2)",

"rgba(255, 206, 86, 0.2)",

"rgba(75, 192, 192, 0.2)",

"rgba(153, 102, 255, 0.2)",

"rgba(255, 159, 64, 0.2)"

],

borderColor: [

"rgba(255,99,132,1)",

"rgba(54, 162, 235, 1)",

"rgba(255, 206, 86, 1)",

"rgba(75, 192, 192, 1)",

"rgba(153, 102, 255, 1)",

"rgba(255, 159, 64, 1)"

],

borderWidth: 1

}

]

},

options: optionsBar

};

barChart = new Chart(ctxBar, configBar);

}

}

async function drawPieChart() {

if ($("#pieChart").length) {

ctxPie = document.getElementById("pieChart").getContext("2d");

// Fetch data for pie chart

const pieChartData = await sendSql("SELECT \* FROM CountVacanciesResumes");

const data\_count = [pieChartData.rows[0].vacancy\_count, pieChartData.rows[0].resume\_count]

const labels = ["Количество вакансий", "Количество резюме"];

optionsPie = {

responsive: true,

maintainAspectRatio: false

};

configPie = {

type: "pie",

data: {

datasets: [

{

data: data\_count,

backgroundColor: ["rgba(54, 162, 235, 0.4)", "rgba(255, 206, 86, 0.4)"],

label: "All"

}

],

labels: labels

},

options: optionsPie

};

pieChart = new Chart(ctxPie, configPie);

}

}

function updateChartOptions() {

if ($(window).width() < width\_threshold) {

if (optionsLine) {

optionsLine.maintainAspectRatio = false;

}

if (optionsBar) {

optionsBar.maintainAspectRatio = false;

}

} else {

if (optionsLine) {

optionsLine.maintainAspectRatio = true;

}

if (optionsBar) {

optionsBar.maintainAspectRatio = true;

}

}

}

function updateLineChart() {

if (lineChart) {

lineChart.options = optionsLine;

lineChart.update();

}

}

function updateBarChart() {

if (barChart) {

barChart.options = optionsBar;

barChart.update();

}

}

function reloadPage() {

setTimeout(function () {

window.location.reload();

}); // Reload the page so that charts will display correctly

}

/\*

Template 2108 Dashboard

http://www.tooplate.com/view/2108-dashboard

COLOR CODES

Dark Gray: #656565

Light Gray: #E6E6E6

Blue: #0062BB

\*/

html {

font-size: 17px;

}

body {

font-family: "Open Sans", Helvetica, Arial, sans-serif;

overflow-x: hidden;

}

body {

background-image: url(../img/background.png);

background-size: cover;

background-repeat: no-repeat;

background-position: center;

background-attachment: fixed;

padding-left: 50px;

padding-right: 50px;

}

.bg02 {

background-image: url(../img/background.png);

}

.bg03 {

background-image: url(../img/background.png);

}

.fs14{

font-size: 14px;

}

a {

transition: all 0.3s ease;

}

a:hover,

a:focus {

text-decoration: none;

}

.navbar-brand {

display: flex;

align-items: center;

}

.tm-site-icon {

color: #656565;

}

.tm-site-title {

display: inline-block;

text-transform: uppercase;

font-size: 2rem;

margin-left: 1rem;

color: #656565;

letter-spacing: 1px;

}

.navbar {

height: 100px;

padding-left: 20px;

padding-right: 20px;

margin-top: 50px;

}

.tm-logout-icon {

font-size: 1.5em;

}

.tm-bg-black {

background-color: rgba(0, 0, 0, 0.5);

}

.tm-mt-big {

margin-top: 70px;

}

.tm-mt-small {

margin-top: 20px;

}

.tm-block {

padding: 50px 45px;

}

.tm-block-title {

font-size: 1.4rem;

color: #333;

margin-bottom: 30px;

}

.navbar-light .navbar-nav .nav-link {

color: #656565;

}

.navbar-light .navbar-nav .nav-link,

.dropdown-item {

padding: 15px 20px;

}

.navbar-light .navbar-nav .active > .nav-link,

.navbar-light .navbar-nav .nav-link.active,

.navbar-light .navbar-nav .nav-link.show,

.navbar-light .navbar-nav .show > .nav-link,

.navbar-light .navbar-nav a:hover {

background-color: #e6e6e6;

}

.nav-item {

margin-right: 30px;

}

.nav-item:last-child {

margin-right: 0;

}

.tm-content-row {

justify-content: space-between;

margin-left: -20px;

margin-right: -20px; /\* To offset col padding \*/

}

.tm-col {

padding-left: 20px;

padding-right: 20px;

margin-bottom: 50px;

}

.tm-col-big {

width: 39%;

}

.tm-col-small {

width: 21.95%;

}

.tm-link-black {

color: black;

}

.tm-link-black:hover,

.tm-link-black:focus {

color: #0266c4;

}

ol {

margin-bottom: 0;

}

.tm-list-group {

counter-reset: myOrderedListItemsCounter;

padding-left: 0;

}

.tm-list-group > li {

list-style-type: none;

position: relative;

cursor: pointer;

transition: all 0.3s ease;

padding: 8px;

}

.tm-list-group-pad-big > li {

padding: 20px;

}

.tm-list-group > li:hover {

color: #0266c4;

}

.tm-list-group-alternate-color > li:nth-child(odd) {

background-color: #e6e6e6;

}

.tm-list-group > li:before {

counter-increment: myOrderedListItemsCounter;

content: counter(myOrderedListItemsCounter) ".";

margin-right: 0.5em;

}

.tm-list {

padding-left: 30px;

}

.tm-list > li {

margin-bottom: 20px;

}

.form-control {

margin-bottom: 23px;

padding: 19px 18px;

border-radius: 0;

height: 50px;

}

label {

margin-bottom: 18px;

}

.btn {

border-radius: 0;

padding: 13px 28px;

transition: all 0.2s ease;

max-width: 100%;

}

.btn-small {

padding: 10px 24px;

}

.btn-primary {

background-color: transparent;

color: black;

border-color: #999999;

}

.btn-primary:hover,

.btn-primary:active,

.btn-primary:not(:disabled):not(.disabled).active,

.btn-primary:not(:disabled):not(.disabled):active,

.show > .btn-primary.dropdown-toggle {

color: black;

background-color: #cdd4da;

border-color: #999999;

}

.btn-danger {

color: #9f1321;

background-color: transparent;

border-color: #9f1321;

}

.btn-danger:hover {

color: #9f1321;

background-color: rgba(159, 19, 32, 0.27);

border-color: #9f1321;

}

.custom-file-input {

cursor: pointer;

}

.custom-file-label {

border-radius: 0;

}

.tm-btn-right {

text-align: right;

}

.table td,

.table th {

padding: 20px 24px;

}

.table-hover tbody tr:hover {

background-color: white;

color: #3aabd0;

}

.tm-bg-gray {

background-color: rgba(0, 0, 0, 0.05);

}

.tm-table-striped-even.table-striped tbody tr:nth-of-type(even) {

background-color: rgba(0, 0, 0, 0.05);

}

.tm-table-striped-even.table-striped tbody tr:nth-of-type(odd) {

background-color: white;

}

.tm-table-mt {

margin-top: 66px;

}

.page-item:first-child .page-link {

border-top-left-radius: 0;

border-bottom-left-radius: 0;

}

.page-item:last-child .page-link {

border-top-right-radius: 0;

border-bottom-right-radius: 0;

}

.page-link,

.tm-dots {

padding: 12px 18px;

}

.page-link,

.page-link:hover {

color: black;

}

.page-item {

margin-right: 18px;

}

.page-item:last-child {

margin-right: 0;

}

.page-item.active .page-link {

background-color: #e9ecef;

border-color: #dee2e6;

color: black;

}

.tm-pagination {

justify-content: flex-end;

}

.tm-pagination-label {

padding-bottom: 15px;

padding-right: 15px;

display: inline-block;

}

.tm-table-actions-row {

display: flex;

justify-content: space-between;

}

.tm-table-actions-col-right {

text-align: right;

}

input[type="checkbox"] {

cursor: pointer;

-webkit-appearance: none;

appearance: none;

border: 1px solid black;

box-sizing: border-box;

position: relative;

box-sizing: content-box;

width: 24px;

height: 24px;

transition: all 0.1s linear;

}

input[type="checkbox"]:checked {

background-color: #9f1321;

}

input[type="checkbox"]:focus {

outline: 0 none;

box-shadow: none;

}

.tm-trash-icon {

color: #6e6c6c;

cursor: pointer;

}

.tm-trash-icon:hover {

color: #9f1321;

}

.tm-trash-icon-cell {

width: 15px;

}

footer {

margin-bottom: 35px;

}

.custom-select {

height: 50px;

border-radius: 0;

}

.tm-product-img-dummy {

max-width: 100%;

width: 240px;

height: 240px;

border: 1px solid #cccccc;

display: flex;

align-items: center;

justify-content: center;

color: #c8c8c8;

}

.tm-login-col {

max-width: 600px;

}

@media (min-width: 992px) {

.navbar-expand-lg .navbar-nav .nav-link {

padding: 15px 20px;

}

}

@media (min-width: 1200px) {

.container {

max-width: 1630px;

}

}

@media (max-width: 1275px) and (min-width: 1200px) {

.nav-item {

margin-right: 15px;

}

}

@media (max-width: 1350px) and (min-width: 1200px) {

.tm-table-actions-row {

display: block;

}

.tm-table-actions-col-right {

margin-top: 30px;

}

}

@media (max-width: 1350px) {

.nav-item {

margin-right: 1px;

}

}

@media (max-width: 1199px) {

.tm-col-big,

.tm-col-small {

width: 49.65%;

}

.navbar-collapse {

background: white;

padding: 15px;

box-shadow: rgba(108, 117, 125, 0.27) 0px 1px 1px 0px;

position: absolute;

top: 77px;

right: 20px;

width: 245px;

z-index: 1000;

}

.navbar-nav .nav-link {

padding-right: 15px;

padding-left: 15px;

}

.nav-item {

margin-right: 0;

}

}

@media (max-width: 991px) {

.tm-col-big,

.tm-col-small {

width: 100%;

min-height: 383px;

height: auto;

}

.tm-block {

padding: 30px;

}

.tm-table-actions-row {

display: block;

}

.tm-table-actions-col-right {

text-align: left;

margin-top: 30px;

}

.tm-edit-product-row {

flex-direction: column-reverse;

}

}

@media (max-width: 633px) {

.pagination {

flex-wrap: wrap;

}

.tm-pagination {

justify-content: flex-start;

}

.tm-pagination-label {

display: inline-block;

margin-right: 0;

}

.page-item {

margin-top: 10px;

}

}

@media (max-width: 574px) {

.navbar-collapse {

top: 70px;

}

.tm-btn-right {

text-align: left;

margin-top: 20px;

}

.navbar {

margin-top: 30px;

height: auto;

}

body {

padding-left: 15px;

padding-right: 15px;

}

.tm-site-title {

font-size: 1.4rem;

margin-left: 7px;

}

.tm-site-icon {

font-size: 2em;

}

}

@media (max-width: 480px) {

.tm-mt-big {

margin-top: 45px;

}

}

@media (max-width: 424px) {

.navbar-collapse {

top: 107px;

}

}

async function sendSql(sql) {

const response = await fetch("/api/sql", {

method: "POST",

headers: {

"Content-Type": "application/json"

},

body: JSON.stringify({ sql: sql })

});

const data = await response.json();

return data;

}

async function generateCSV() {

const responseResumes = await sendSql("SELECT \* FROM Resumes");

const responseUsers = await sendSql("SELECT \* FROM Users");

const responseVacancies = await sendSql("SELECT \* FROM Vacancies");

const responseResponses = await sendSql("SELECT \* FROM Responses");

const resumes = responseResumes.rows;

const users = responseUsers.rows;

const vacancies = responseVacancies.rows;

const responses = responseResponses.rows;

const csvContent = "data:text/csv;charset=utf-8," +

"Resumes:\n" +

"ID,Title,Body,Payment,Skill,Created\n" +

resumes.map(resume => `"${resume.id}","${resume.title}","${resume.body}","${resume.payment}","${resume.skill.join(', ')}","${resume.created}"`).join('\n') +

"\n\nUsers:\n" +

"ID,Resume ID,Email,Firstname,Secondname,Password,Created,Type\n" +

users.map(user => `"${user.id}","${user.resume\_id}","${user.email}","${user.firstname}","${user.secondname}","${user.password}","${user.created}","${user.type}"`).join('\n') +

"\n\nVacancies:\n" +

"ID,User ID,Title,Body,Payment,Skill,Created\n" +

vacancies.map(vacancy => `"${vacancy.id}","${vacancy.user\_id}","${vacancy.title}","${vacancy.body}","${vacancy.payment}","${vacancy.skill.join(', ')}","${vacancy.created}"`).join('\n') +

"\n\nResponses:\n" +

"ID,User ID,Vacancy ID,Body,Created\n" +

responses.map(response => `"${response.id}","${response.user\_id}","${response.vacancy\_id}","${response.body}","${response.created}"`).join('\n');

const encodedUri = encodeURI(csvContent);

const link = document.createElement("a");

link.setAttribute("href", encodedUri);

link.setAttribute("download", "database\_report.csv");

document.body.appendChild(link);

link.click();

}

async function sendSql(sql) {

const response = await fetch("/api/sql", {

method: "POST",

headers: {

"Content-Type": "application/json"

},

body: JSON.stringify({ sql: sql })

});

const data = await response.json();

return data;

}

window.onload = async function () {

// Load resumes for dropdowns

const resumesResponse = await sendSql("SELECT id, title FROM Resumes");

const resumes = resumesResponse.rows;

const userSelects = document.querySelectorAll('#addUsersForm select[name="resume\_id"]');

userSelects.forEach(select => {

resumes.forEach(resume => {

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

option.value = resume.id;

option.textContent = resume.title;

select.appendChild(option);

});

});

// Load users for dropdowns

const usersResponse = await sendSql("SELECT id, email FROM Users");

const users = usersResponse.rows;

const vacancyUserSelects = document.querySelectorAll('#addVacanciesForm select[name="user\_id"]');

vacancyUserSelects.forEach(select => {

users.forEach(user => {

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

option.value = user.id;

option.textContent = user.email;

select.appendChild(option);

});

});

const responseUserSelects = document.querySelectorAll('#addResponsesForm select[name="user\_id"]');

responseUserSelects.forEach(select => {

users.forEach(user => {

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

option.value = user.id;

option.textContent = user.email;

select.appendChild(option);

});

});

// Load vacancies for dropdowns

const vacanciesResponse = await sendSql("SELECT id, title FROM Vacancies");

const vacancies = vacanciesResponse.rows;

const responseVacancySelects = document.querySelectorAll('#addResponsesForm select[name="vacancy\_id"]');

responseVacancySelects.forEach(select => {

vacancies.forEach(vacancy => {

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

option.value = vacancy.id;

option.textContent = vacancy.title;

select.appendChild(option);

});

});

};

// Add record to Resumes

document.getElementById('addResumesForm').addEventListener('submit', async function (event) {

event.preventDefault();

const formData = new FormData(this);

const data = {};

formData.forEach((value, key) => {

data[key] = value;

});

// Generate UUID for id

data.id = await crypto.randomUUID();

const sql = `INSERT INTO Resumes (id, title, body, payment, skill, created) VALUES ('${data.id}', '${data.title}', '${data.body}', '${data.payment}', ARRAY['${data.skill}']::VARCHAR[], NOW())`;

await sendSql(sql);

alert("Запрос на добавление отправлен!");

});

// Add record to Users

document.getElementById('addUsersForm').addEventListener('submit', async function (event) {

event.preventDefault();

const formData = new FormData(this);

const data = {};

formData.forEach((value, key) => {

data[key] = value;

});

// Generate UUID for id

const userId = await crypto.randomUUID();

const sql = `INSERT INTO Users (id, email, firstname, secondname, password, resume\_id, type, created) VALUES ('${userId}', '${data.email}', '${data.firstname}', '${data.secondname}', '${data.password}', '${data.resume\_id}', '${data.type}', NOW())`;

await sendSql(sql);

alert("Запрос на добавление отправлен!");

});

// Add record to Vacancies

document.getElementById('addVacanciesForm').addEventListener('submit', async function (event) {

event.preventDefault();

const formData = new FormData(this);

const data = {};

formData.forEach((value, key) => {

data[key] = value;

});

// Generate UUID for id

const vacancyId = await crypto.randomUUID();

const sql = `INSERT INTO Vacancies (id, user\_id, title, body, payment, skill, created) VALUES ('${vacancyId}', '${data.user\_id}', '${data.title}', '${data.body}', '${data.payment}', ARRAY['${data.skill}']::VARCHAR[], NOW())`;

await sendSql(sql);

alert("Запрос на добавление отправлен!");

});

// Add record to Responses

document.getElementById('addResponsesForm').addEventListener('submit', async function (event) {

event.preventDefault();

const formData = new FormData(this);

const data = {};

formData.forEach((value, key) => {

data[key] = value;

});

// Generate UUID for id

const responseId = await crypto.randomUUID();

const sql = `INSERT INTO Responses (id, user\_id, vacancy\_id, body, created) VALUES ('${responseId}', '${data.user\_id}', '${data.vacancy\_id}', '${data.body}', NOW())`;

await sendSql(sql);

alert("Запрос на добавление отправлен!");

});

# Приложение В (обязательное) Результаты работы приложения

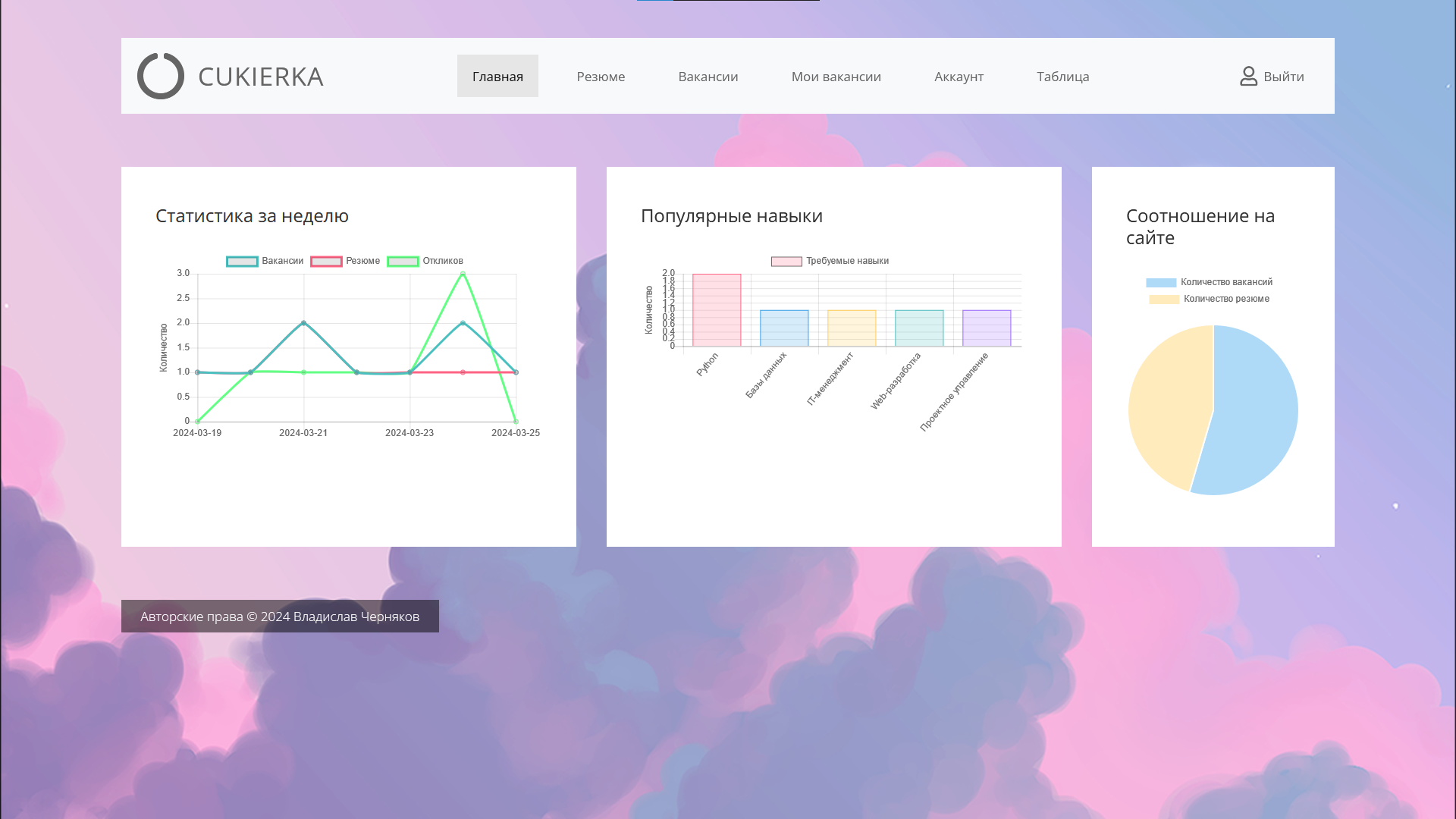


Рисунок В.1

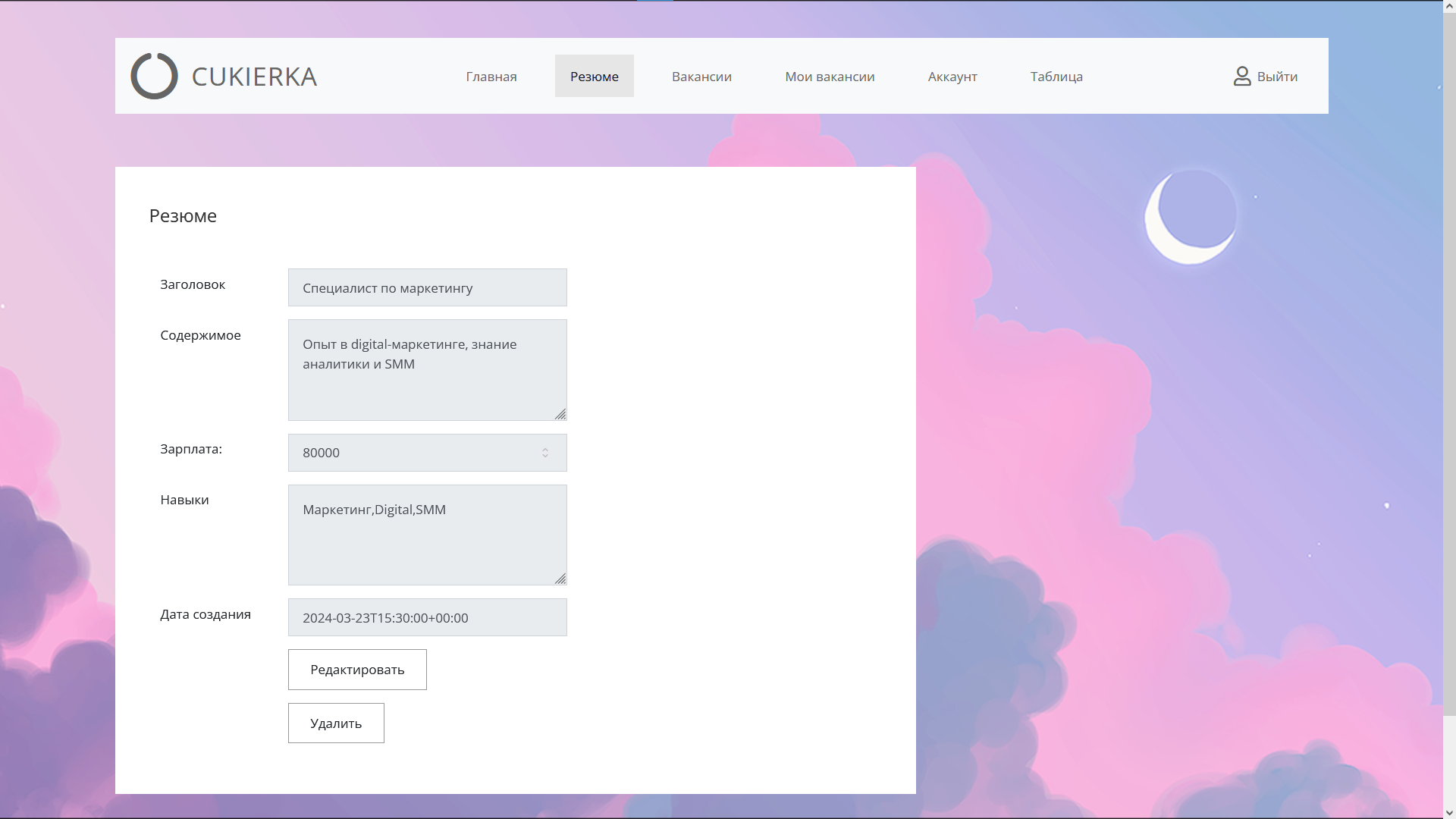


Рисунок В.2