

МИНИСТЕРСТВО ОБРАЗОВАНИЯ КИРОВСКОЙ ОБЛАСТИ

Кировское областное государственное профессиональное

образовательное бюджетное учреждение

«Слободской колледж педагогики

и социальных отношений»

**КУРСОВАЯ РАБОТА**

по профессиональному модулю ПМ.01. «Разработка программных модулей»

на тему:

**«Разработка программного модуля для учёта и подбора вакансий центра занятости населения»**

Хохрин Никита

Владимирович

Специальность 09.02.07 -

Информационные системы

и программирование

Курс 21П-1

Форма обучения: очная

Руководитель:

Махнев Александр Анатольевич

Дата защиты курсовой работы:

Оценка за защиту курсовой работы:

Председатель ПЦК:

Слободской

2024

## ОГЛАВЛЕНИЕ

Стр.

Введение.....	3
1. Анализ предметной области .....	5
2. Техническое задание на разработку программного модуля.....	12
3. Алгоритм и схема функционирования программного модуля.....	15
4. Тестирование программного модуля .....	20
Заключение .....	26
Список литературы .....	28
Приложения .....	32

## **ВВЕДЕНИЕ**

В последние годы наблюдается стремительное развитие информационных технологий, что приводит к необходимости их активного внедрения в различные сферы деятельности, включая рынок труда. Традиционные методы поиска работы, такие как объявления в газетах и обращение в кадровые агентства, становятся менее эффективными в условиях быстрого изменения требований к специалистам и роста объема информации. В связи с этим, автоматизация процессов подбора и учета вакансий становится важным аспектом деятельности центров занятости населения.

Современные центры занятости сталкиваются с рядом проблем, связанных с ручной обработкой данных, низкой скоростью обновления информации о вакансиях и соискателях, а также сложностью поиска соответствий между требованиями работодателей и квалификацией кандидатов. Разработка программного модуля, который позволит автоматизировать эти процессы, повысит эффективность работы центров занятости, сократит время на поиск работы для соискателей и облегчит процесс подбора персонала для работодателей.

Одной из первостепенных задач центра занятости является организация эффективного взаимодействия между работодателями и гражданами, ищущими работу. В условиях экономической нестабильности наличие актуальных и качественно подобранных вакансий является залогом успешного трудоустройства. Оценивая динамику рынка труда, можно выделить несколько значимых факторов, влияющих на изменение ситуации. К ним можно отнести технологические инновации, изменения в законодательстве, демографические сдвиги и глобализацию экономики. Эти факторы требуют от центров занятости гибкости в подходах к учету и подбору вакансий.

**Объект исследования:** процесс разработки программного модуля для учёта и подбора вакансий центра занятости населения.

**Предмет исследования:** разработка программного модуля для учёта и подбора вакансий центра занятости населения.

**Цель базы данных:** Учёт и подбор вакансий – автоматизация процесса учёта и подбора вакансий центра занятости населения.

**Задачи:**

1. Разработать техническое задание на создание программного продукта.
2. Описать алгоритмы и функционирование программного продукта.
3. Разработать руководство пользователя.

**Методы исследования:** анализ предметной области, разработка и тестирование программного модуля.

**Практическая значимость:** программное обеспечение может быть использовано для учёта и подбора вакансий.

В процессе выполнения курсового проекта будут использоваться следующие программные продукты: Система управления базами данных (СУБД) Microsoft SQL Server 2014, Microsoft Visual Studio 2022.

## 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

**Центр занятости населения** — это государственное учреждение, которое занимается вопросами трудоустройства граждан, содействия в поиске работы и обеспечением занятости населения. Основная цель центра заключается в улучшении ситуации на рынке труда через предоставление различных услуг как для соискателей, так и для работодателей.

Одним из ключевых аспектов работы центра является **учет вакансий**. Это процесс сбора и систематизации информации о доступных рабочих местах, которые предлагают работодатели. Центры занятости активно ведут базы данных вакансий, что позволяет им отслеживать изменения на рынке труда и предоставлять актуальную информацию о потребностях работодателей.

На основе данных о вакансиях центры занятости осуществляют **подбор вакансий** для соискателей. Это включает в себя анализ квалификаций и навыков граждан, а также их предпочтений, чтобы предложить им наиболее подходящие варианты трудоустройства. Центры помогают наладить связь между работодателями и потенциальными работниками, что способствует более эффективному процессу трудоустройства и снижению уровня безработицы.

В ходе анализа предметной области были рассмотрены аналоги создаваемого программного модуля и проведено их сравнение:

**HeadHunter** - крупнейшая российская компания интернет-рекрутмента, развивающая бизнес в России, Белоруссии, Казахстане. Клиентами HeadHunter являются свыше 515 тыс. компаний. Обширная база соискателей на hh.ru содержит более чем 55 млн резюме, а среднее дневное количество вакансий превышает 933 тысячи.

#### Достоинства:

1. Широкий выбор вакансий: огромный ассортимент вакансий в различных сферах и регионах
2. Расширенные возможности поиска: разнообразные фильтры для поиска вакансий по ключевым словам, уровню зарплаты, месту работы и другим критериям
3. Мобильное приложение: наличие мобильного приложения для более удобного доступа к вакансиям и откликам

#### Недостатки:

1. Спам: иногда пользователь может получать рекламные сообщения от работодателей
2. Платные услуги: доступ к некоторым расширенным функциям может требовать оплаты
3. Обновление данных: иногда пользователю могут встретиться устаревшие вакансии, что может вызвать неудобства

**SuperJob** - один из лидеров на рынке онлайн-рекрутмента в России. Ежедневно в базу данных сайта добавляется более 65 000 новых резюме специалистов и публикуется более 10 000 объявлений о вакансиях. Ежегодно на портале SuperJob.ru более 3 500 000 человек находят работу и более 200 000 компаний-работодателей подбирают нужный персонал.

Достоинства:

1. Широкий спектр вакансий: большое количество вакансий в разных сферах и регионах
2. Разнообразие фильтров: предоставляет многофункциональные фильтры для поиска вакансий
3. Мобильное приложение: наличие мобильного приложения для более удобного доступа к вакансиям и откликам

Недостатки:

1. Высокая конкуренция: из-за большого числа пользователей, конкуренция за популярные вакансии может быть высокой, что усложняет процесс трудоустройства
2. Устаревшие вакансии: время от времени появляются устаревшие или неактуальные вакансии
3. Спам-рассылки: некоторые пользователи могут получать рекламные сообщения от работодателей

**Работа.ру** - сервис, предоставляющий услуги по подбору персонала и поиску работы, являющийся третьим по величине в России. По данным портала, ежедневно на нём присутствует порядка 250 тыс. вакансий, 16 млн резюме.

Достоинства:

1. Разнообразные фильтры поиска: предоставляет различные фильтры для поиска вакансий
2. Мобильное приложение: наличие мобильного приложения для более удобного доступа к вакансиям и откликам
3. Большой выбор вакансий: предлагает широкий спектр вакансий в различных отраслях и регионах

Недостатки:

1. Актуальность вакансий: некоторые вакансии являются устаревшими или неактуальными
2. Спам: некоторые пользователи могут получать спам или нежелательные предложения работы
3. Высокая конкуренция: из-за большого числа пользователей, конкуренция за популярные вакансии может быть высокой, что усложняет процесс трудоустройства



Также в процессе анализа было проведено исследование предметной области, в результате которого была создана диаграмма вариантов использования, которая отражает актёров (пользователей) и выполняемые ими функции (рисунок 1).

На ней видно, что в системе учета и подбора вакансий выделяется два центральных пользователя:

- **Соискатель**, который может просматривать доступные вакансии, добавлять вакансии в избранное, а также составлять, изменять или удалять резюме для трудоустройства
- **Работодатель**, который может просматривать доступные резюме, добавлять резюме в избранное, а также добавлять, изменять или закрывать вакансии

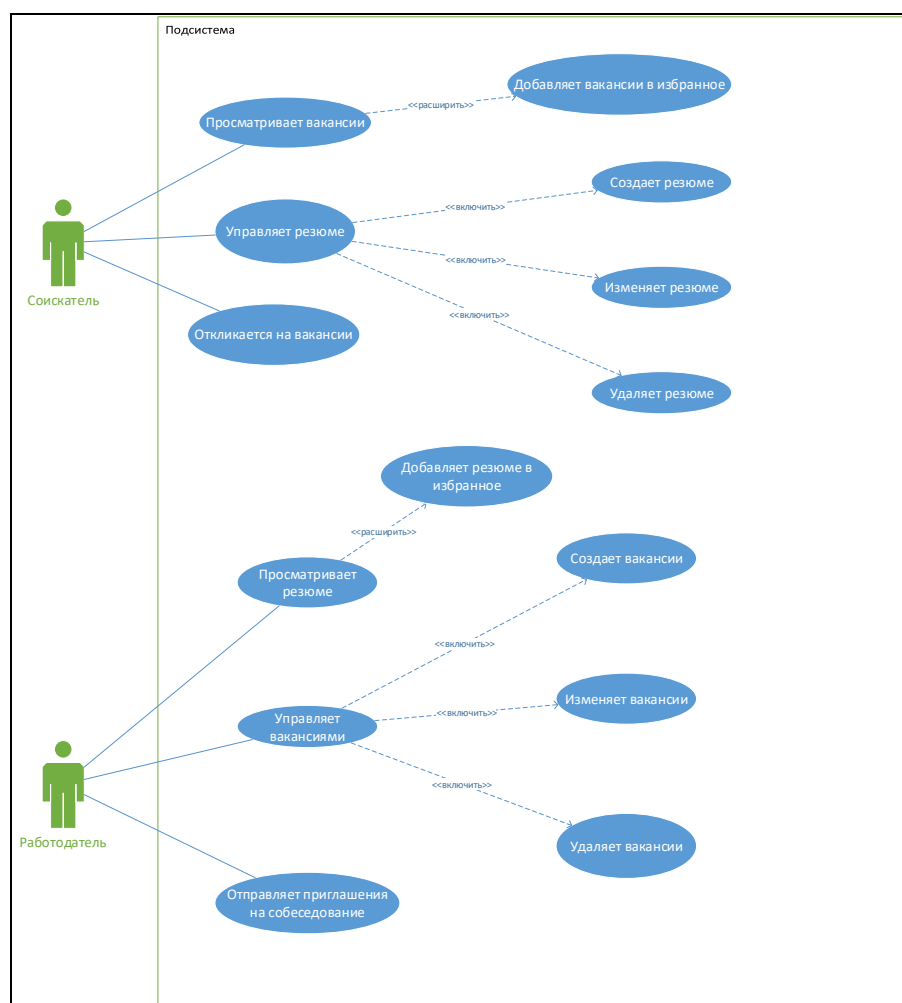


Рисунок 1 - Диаграмма вариантов использования

После изучения вышеперечисленных аналогов разрабатываемого программного модуля были выделены следующие сущности и атрибуты:

### **1) Пользователи**

- КодПользователя (int)
- Логин (nvarchar(MAX))
- Пароль (nvarchar(MAX))
- Роль (nvarchar(MAX))
- Фамилия (nvarchar(MAX))
- Имя (nvarchar(MAX))
- Отчество (nvarchar(MAX))
- ДатаРождения (nvarchar(MAX))
- НомерТелефона (nvarchar(MAX))

### **2) Вакансии**

- КодВакансии (int)
- КодПользователя (int)
- Наименование (nvarchar(MAX))
- Город (nvarchar(MAX))
- Студентам (nvarchar(MAX))
- ОпытРаботы (nvarchar(MAX))
- Специализация (nvarchar(MAX))
- ГрафикРаботы (nvarchar(MAX))
- Отрасль (nvarchar(MAX))
- Зарплата (bigint)
- Образование (nvarchar(MAX))
- Статус (nvarchar(MAX))

### **3) Резюме**

- КодРезюме (int)
- КодПользователя (int)
- Наименование (nvarchar(MAX))
- ЖелаемаяДолжность (nvarchar(MAX))
- Зарплата (nvarchar(MAX))
- Образование (nvarchar(MAX))
- ЭлПочта (nvarchar(MAX))
- ГрафикРаботы (nvarchar(MAX))
- ПрофессиональныеНавыкиИЗнания(nvarchar(MAX))

### **4) Отклики**

- КодОтклика (int)
- КодПользователя (int)
- КодВакансии (int)

### **5) Приглашения**

- КодПриглашения (int)
- КодПользователя (int)
- КодРезюме (int)

### **6) ИзбранныеРезюме**

- КодИзбранногоРезюме
- КодПользователя
- КодРезюме

## 7) ИзбранныеВакансии

- КодИзбраннойВакансии
- КодПользователя
- КодВакансии

На основании этих данных была создана модель базы данных (Рисунок 2).

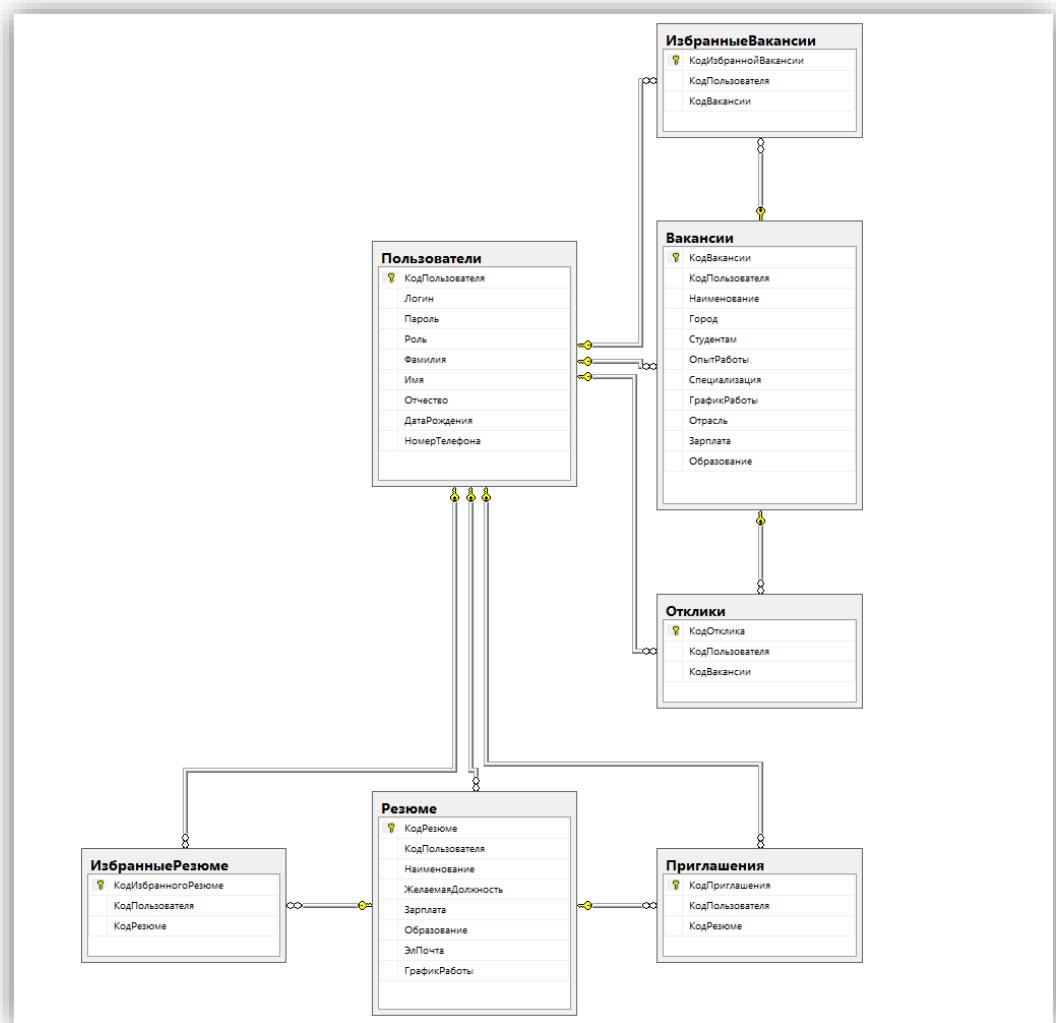


Рисунок 2 - Модель базы данных в СУБД MS SQL Server

**Вывод:** была проведена работа по изучению аналогов создаваемого программного модуля, их преимуществ и недостатков, на основании которой была создана UML-диаграмма вариантов использования. Также была спроектирована база данных и её диаграмма в СУБД MS SQL Server.

## **2. ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА РАЗРАБОТКУ ПРОГРАММНОГО МОДУЛЯ**

Техническое задание разрабатывалось в соответствии с ГОСТ 19.201-78 [2] и ГОСТ 19.101-77 [1]

Наименование программы – программный модуль для учета и подбора вакансий центра занятости населения. Шифр программы - «Найти Работу». Создаваемый программный модуль будет применяться пользователем для управления вакансиями, резюме, а также откликами на вакансии. Основанием для разработки является учебный план специальности 09.02.07 «Информационные системы и программирование» Слободского колледжа педагогики и социальных отношений. Функциональным назначением разрабатываемого программного модуля является возможность автоматизации составления резюме, откликов на вакансии, управления вакансиями и резюме. Программный продукт будет эксплуатироваться на персональном компьютере конечным пользователем. Разрабатываемый модуль выполняет следующие функции:

- Просмотр доступных вакансий - возможность имеется сразу после запуска программы при условии подключения к базе данных;
- Просмотр доступных резюме - возможность имеется сразу после запуска программы при условии подключения к базе данных;
- Управление вакансиями – возможность добавления, изменения и закрытия вакансии. Добавление вакансии в избранное и возможность отклика на вакансию;
- Управление резюме – возможность добавления, изменения и удаления резюме. Добавление резюме в избранное и возможность отправки приглашения по резюме;

Все внесённые данные хранятся в СУБД. Ввод данных в систему выполняется с помощью SQL-запросов. Валидация данных выполняется на стороне пользователя:

- Специализация, Город, ОпытРаботы, ГрафикРаботы, Отрасль, Зарплата, Образование;
- НаименованиеРезюме, ЖелаемаяДолжность, ЖелаемаяЗарплата, Образование, ЭлПочта, ГрафикРаботы, ПрофессиональныеНавыкиИЗнания;

После добавления пользователем данных в базу данных, новая информация должна отобразиться в модуле не позднее, чем через 10 секунд. Вероятность безотказной работы системы должна составлять не менее чем 99,99% (при условии успешного соединения программного модуля и базы данных). Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением заказчиком следующего перечня организационно-технических мероприятий:

- организацией бесперебойного питания технических средств;
- использованием лицензионного программного обеспечения;
- регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. «Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;
- регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов.

Время восстановления после отказа из-за сбоя электропитания технических средств и не фатальным крахом операционной системы не должно превышать 10 минут. Время восстановления после отказа работы технических средств и (или) фатальным сбоем операционной системы не должно превышать времени, затраченного на устранение ошибок в работе технических средств или восстановления после сбоя.

Эксплуатируемая программа будет запускаться на компьютере пользователя, сервер базы данных – на другом компьютере. Между ними должна существовать устойчивая связь. Специальных климатических условий и условий для обслуживания не предъявляется. В процессе эксплуатации программой будет пользоваться конечный пользователь.

Для корректной работы программы должны быть соблюдены следующие минимальные технические требования:

- Процессор с тактовой частотой не менее 1 ГГц;
- Оперативная память не менее 1 Гб;
- Видеокарта, монитор, мышь, клавиатура

Программный модуль передаётся посредством скачивания из сети Internet. Специальным требованием является то, что программа должна работать на графическом интерфейсе, разработанном производителем операционной системы.

Приемосдаточные испытания программы должны проводиться согласно разработанной исполнителем и согласованной заказчиком «Программы и методики испытаний».

Ход проведения приемо-сдаточных испытаний заказчик и исполнитель документируют в протоколе испытаний.

На основании протокола испытаний исполнитель совместно с заказчиком подписывают акт приемки-сдачи программы в эксплуатацию.

**Вывод:** было создано техническое задание, описывающее основные функции создаваемого программного модуля.

### **3. АЛГОРИТМ И СХЕМА ФУНКЦИОНИРОВАНИЯ ПРОГРАММНОГО МОДУЛЯ**

#### **Функционирование программы**

При запуске программы происходит отображение формы авторизации и регистрации, на которой пользователь может ввести свои учетные данные и войти в систему.

После успешной авторизации пользователю открывается главная форма, где он, в зависимости от своей роли, может просматривать, добавлять, изменять и удалять резюме/вакансии. Добавление происходит посредством ввода данных в специальную форму. Все добавленные с помощью программного продукта данные хранятся в базе данных SQL.

Рассмотрим алгоритмы, используемые в создаваемом программном модуле. Ниже приведено их краткое описание:

- **Запуск программы**

При запуске программы происходит подключение к базе данных посредством метода SqlConnection. После запуска главного окна программы в dataGridView загружаются данные из таблицы с помощью SQL-запроса (Запрос 1). Все данные сохраняются в базу данных SQL Server.



## Запрос 1 – Загрузка данных в dataGridView из таблицы базы данных

```
public static class DataBaseConfig
{
    public static string ConnectionString { get; private set; }

    static DataBaseConfig()
    {
        string serverName = Environment.MachineName + @"\SQLEXPRESS";
        string databaseName = "НайтиРаботу";
        ConnectionString = $"Data Source={serverName};Initial Catalog={databaseName};Integrated
Security=True";
    }
}
```

```
try
{
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
    string query = $"select КодРезюме, Фамилия, Имя, Отчество, Наименование,
ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Пользователи inner join
Резюме on Пользователи.КодПользователя = Резюме.КодПользователя ";
    SqlCommand command = new SqlCommand(query, con);
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    DataTable dt = new DataTable();
    adapter.Fill(dt);
    ОтображениеДанных.DataSource = dt;
}
catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
}
```

- **Добавление и удаление записи**

В модулях Form «AddResumeForm» и «AddVacancyForm», после ввода данных в текстовые поля, добавление данных в таблицу производится с помощью SQL-запроса, а в модулях Form «PerCabForm» и «PerCabRabForm» при нажатии специальной кнопки происходит удаление записи из таблицы dataGridView и таблицы базы данных SQL.

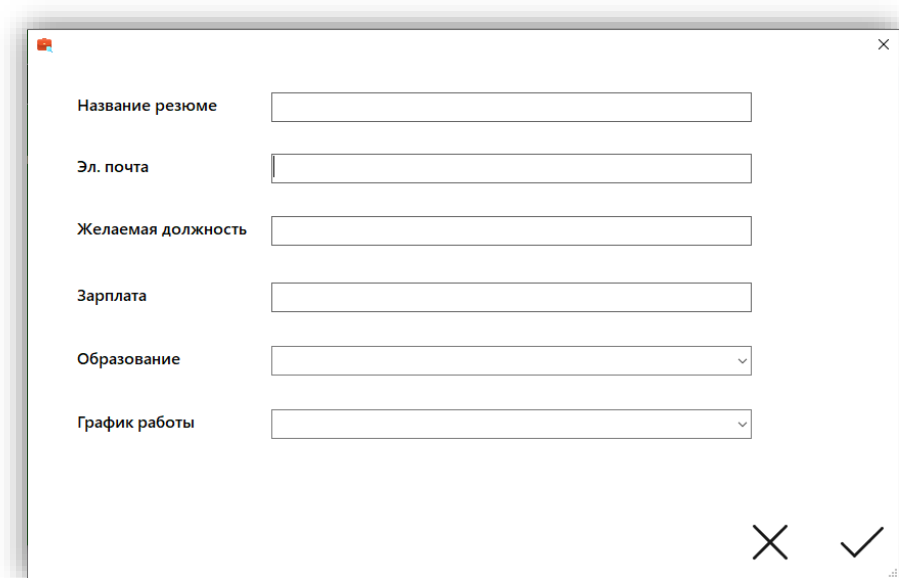
### **Входные данные**

Входные данные заполняются с помощью клавиатуры в специально введённые для этого поля на форме в программном модуле.

В качестве примера в данном программном продукте рассмотрим форму «AddResumeForm» (Рисунок 3), со следующими полями для ввода данных:

- Название резюме
- Эл. Почта
- Желаемая должность
- Зарплата
- Образование
- График работы
- Профессиональные Навыки И Знания

После ввода данных и нажатия соответствующей кнопки программа проверяет корректность введенных в текстовые поля данных.



The image shows a screenshot of a software window titled "Форма добавления резюме" (Resume Addition Form). The window has a standard Windows-style title bar with a close button (X) in the top right corner. Inside the window, there are six input fields arranged vertically, each with a label to its left:

- Название резюме (Resume Name): A text input field.
- Эл. почта (Email): A text input field.
- Желаемая должность (Desired Position): A text input field.
- Зарплата (Salary): A text input field.
- Образование (Education): A dropdown menu with a small downward arrow on the right.
- График работы (Work Schedule): A dropdown menu with a small downward arrow on the right.

At the bottom right of the window, there are two large, dark buttons: one with a cross (X) symbol and one with a checkmark (✓) symbol, likely for canceling or confirming the operation.

Рисунок 3 - Форма добавления резюме

В случае, если проверки пройдены успешно, то добавленная информация отражается в таблице dataGridView на главной форме приложения (PerCabForm/PerCabRabForm) (рисунок 4). Также она добавляется в соответствующую таблицу базы данных.

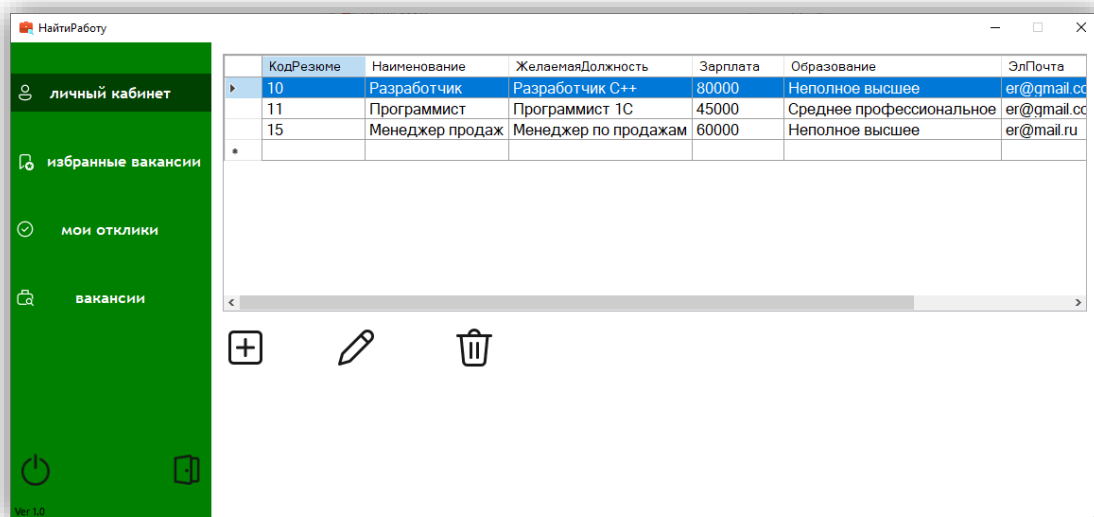


Рисунок 4 - Главная форма программного модуля

**Вывод:** были описаны алгоритмы работы созданного программного модуля и вспомогательной базы данных. Также было описано правильное функционирование программы и изложен способ организации входных данных.

#### 4. ТЕСТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ

Тестирование программного модуля можно произвести в три этапа:

1. **Тестирование устойчивости.** В ходе выполнения данной группы тестов проверялась устойчивость программы при вводе некорректных или пустых данных в форму добавления документа.

2. **Тестирование функциональности.** В ходе выполнения данной группы тестов была проверена правильность выполнения функций, заявленных в техническом задании.

3. **Тестирование приемлемости.** В ходе выполнения данной группы тестов выполнялась проверка удовлетворения способов использования созданного программного продукта требованиям задания (время, за которое приложение реагирует и отвечает на команды пользователя, понятность и доступность интерфейса, устойчивость и скорость вычислительного процесса).

Таблица с результатами данных видов тестирования представлена ниже.

*Таблица 1 – Результаты тестирования*

№	Вид тестирования	Результат	Комментарий
1	Тестирование устойчивости	Успешно	Группа тестов пройдена без обнаружения ошибок
2	Тестирование функциональности	Успешно	Группа тестов пройдена без обнаружения ошибок
3	Тестирование приемлемости	Успешно	Группа тестов пройдена без обнаружения ошибок

Для того чтобы наглядно показать данные, внесённые в таблицу 1, ниже содержится описание выполнения одного из каждой группы тестов, указанных выше.

### Группа 1. Тестирование устойчивости

Для показа теста данной группы был выполнен тест ввода пустых данных в поля «Логин» и «Пароль» формы авторизации программного модуля. Для этого необходимо выполнить следующие действия:

- Открыть форму авторизации программы (Рисунок 5);
- Оставить данную форму без данных и нажать кнопку «Войти»;
- В результате выполненных действий получить сообщение о том, что необходимые поля пусты (Рисунок 6);

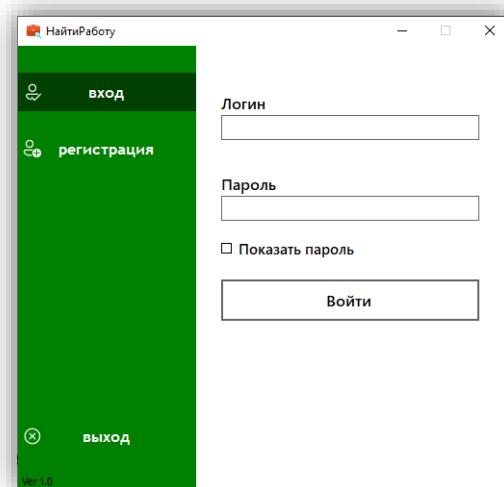


Рисунок 5 - Форма авторизации программного модуля

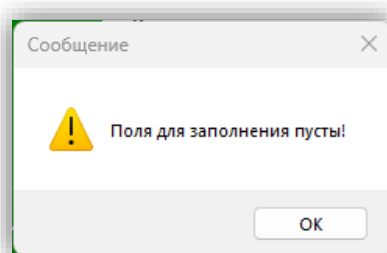


Рисунок 6 - Сообщение о пустых данных

Ожидаемый результат – программный модуль выдаст предупреждение о вводе «пустых данных».

Фактический результат – выдает предупреждение о вводе «пустых данных» на рисунке 6.

Тестирование выполнено успешно.

## **Группа 2. Тестирование функциональности**

Для данной группы было выполнено тестирование функции добавления вакансии на главной форме пользователя «Работодатель» программного модуля. Для этого необходимо выполнить следующие действия:

- Открыть форму авторизации программного модуля (Рисунок 5);
- На главной форме пользователя «Работодатель» перейти на вкладку «Личный кабинет» (Рисунок 7);
- Нажать на кнопку в виде плюса;
- В открывшемся окне ввести необходимые данные в соответствующие поля (Рисунок 8);
- Нажать на кнопку в виде плюса и дождаться вывода соответствующего сообщения (Рисунок 9);

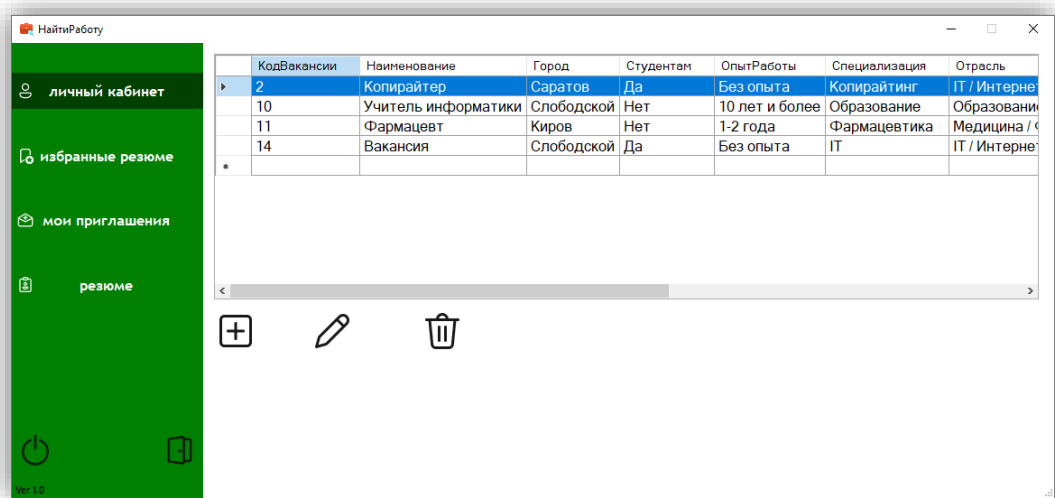


Рисунок 7 - Главная форма пользователя «Работодатель»

Название вакансии:

Город:

Студентам: ☐

Опыт работы:

Специализация:

График работы:

Зарплата:

Образование:

Отрасль:

Buttons: X, ✓

Рисунок 8 - Окно добавления вакансии

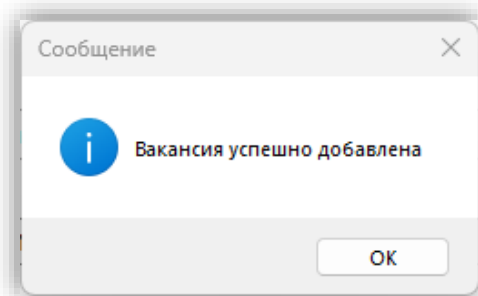


Рисунок 9 - Сообщение об успешном добавлении данных



Ожидаемый результат – появляется сообщение об успешном добавлении вакансии в базу данных

Фактический результат - появляется сообщение об успешном добавлении вакансии в базу данных (Рисунок 9)

Тестирование выполнено успешно.

### **Группа 3. Тестирование приемлемости**

Для данной группы было выполнено тестирование программного модуля на скорость восстановления его работы после сбоя, связанного с базой данных.

Чтобы имитировать отключение базы данных в неожиданный момент, был выбран способ отключения службы SQL Server (SQLEXPRESS), которая отвечает за хранение и обработку данных, управление доступом к ним и обеспечение быстрой транзакционной обработки.

Для этого необходимо выполнить следующие действия:

- Открыть программный модуль;
- Во время выполнения случайных действий в модуле отключить базу данных путём остановки службы;
- Дождаться появления сообщения об ошибке (Рисунок 10);

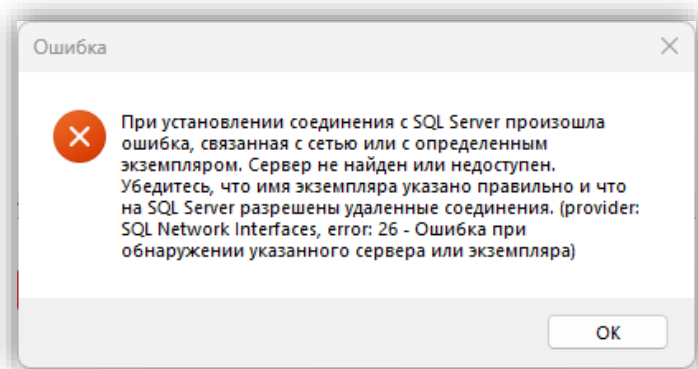


Рисунок 10 - Сообщение об ошибке при работе программного модуля

- После появления сообщения об ошибке вновь запускаем службу без необходимости перезапуска программного модуля;

Ожидаемый результат – время, затрачиваемое на восстановление нормального функционирования после сбоя, не превысило времени, указанного в техническом задании на разработку программного модуля.

Фактический результат - время, затрачиваемое на восстановление нормального функционирования после сбоя, не превысило времени, указанного в техническом задании на разработку программного модуля.

Тестирование пройдено успешно.

**Вывод:** были описаны этапы тестирования программного модуля, создана таблица, содержащая результаты проведения каждого этапа тестирования и описаны тесты из каждой заявленной группы с целью их демонстрации на практике.

## **ЗАКЛЮЧЕНИЕ**

Спроектированный программный модуль для учёта и подбора вакансий центра занятости населения дает возможность пользователю просматривать, добавлять, изменять, удалять резюме и вакансии, а также откликаться на них.

Цель курсовой работы заключалась в разработке программного модуля для учёта и подбора вакансий для пользователя. Поставленная цель была реализована в полном объёме.

В ходе выполнения курсовой работы были реализованы следующие задачи:

- Проведён анализ предметной области;
- Создано техническое задание на разработку программного модуля;
- Описаны алгоритмы и функционирование программы;
- Проведено тестирование программного продукта;

Все задачи были выполнены вовремя и в соответствии с поставленным руководителем курсового проекта заданием. Перед началом разработки программного продукта нами были изучены аналоги разрабатываемого модуля. Также было разработано техническое задание, в котором описаны назначение программы, её основной функционал, минимальный состав технических средств, необходимых для её функционирования и т.д.

Для упрощения ознакомления с модулем разработчика, который будет изменять созданный программный продукт, были описаны используемые алгоритмы и функционирование программы.

После внесения каждой новой функции во время разработки проходило тестирование программного продукта на наличие ошибок. В случае обнаружения ошибок, каждая из них была незамедлительно устранена. Во время тестовой эксплуатации готового программного модуля было проведено дополнительное тестирование с целью исключения незамеченных в процессе разработки недостатков и недочётов, которые могли быть обнаружены во время возможной эксплуатации конечным пользователем.

Разработанный программный продукт имеет удобный, ориентированный на обычного пользователя интерфейс и простую структуру. Всё это облегчает и упрощает работу с программным продуктом, что является несомненным плюсом для тех, кто будет с ним работать.

Таким образом, поставленные задачи решены в полном объеме. Все заявленные цели были достигнуты.

## СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77. Единая система программной документации. Виды программ и программных документов, введ. 01.01.1978. – г. Москва: Изд-во стандартов, 1980. – 4 с.
2. ГОСТ 19.201-78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению, введ. 01.01.1980, г.Москва: Изд-во стандартов, 1988. - 3 с.
3. Албахари, Б. С# 7.0. Справочник. Полное описание языка / Б. Албахари, Д. Албахари. – М.: Вильямс, 2018. – 174 с.
4. Документация по С#. Начало работы, руководства, справочная информация [Электронный ресурс]. – <https://docs.microsoft.com/ru-ru/dotnet/csharp/>
5. Кнут Д. Искусства программирования на ЭВМ, Том 3. Сортировка и поиск, 2-е изд. / Д. Кнут. – М.: Вильямс, 2018. – 832 с.
6. Пахомов, Б. С# для начинающих. / Б. Пахомов. – СПб.: БХВ-Петербург, 2014. – 432 с.
7. Рихтер, Д. CLR via С#. Программирование на платформе Microsoft.NET Framework 4.5 на языке С# / Д. Рихтер. – СПб.: Питер, 2017. – 896 с.
8. Стасышин, В. М. Базы данных: технологии доступа : учеб. пособие для СПО / В. М. Стасышин, Т. Л. Стасышина. — 2-е изд., испр. и доп. — М. : Издательство Юрайт, 2018. — 164 с.
9. Маркин, А. В. Программирование на sql в 2 ч. Часть 2 : учебник и практикум для бакалавриата и магистратуры / А. В. Маркин. — М. : Издательство Юрайт, 2019. — 292 с.

10. Создание и удаление таблиц [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/3.2.php>
11. Добавление данных. Команда Insert [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/4.1.php>
12. Выборка данных. Команда Select [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/4.2.php>
13. Сортировка. ORDER BY [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/4.3.php>
14. Фильтрация. WHERE [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/4.5.php>
15. Обновление данных. Команда UPDATE [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/4.7.php>
16. Удаление данных. Команда DELETE [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/4.8.php>
17. Операторы фильтрации [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/4.6.php>
18. Хранимые процедуры [Электронный ресурс]. – Режим доступа: <https://metanit.com/sql/sqlserver/11.1.php>
19. Элемент Button C# [Электронный ресурс]. – Режим доступа: <https://metanit.com/sharp/windowsforms/4.1.php>
20. Элемент MaskedTextBox C# [Электронный ресурс]. – Режим доступа: <https://metanit.com/sharp/windowsforms/4.4.php>
21. Элементы RadioButton и CheckBox C# [Электронный ресурс]. – Режим доступа: <https://metanit.com/sharp/windowsforms/4.5.php>
22. Элемент ComboBox C# [Электронный ресурс]. – Режим доступа: <https://metanit.com/sharp/windowsforms/4.7.php>

23. DateTimePicker и MonthCalendar C# [Электронный ресурс]. – Режим доступа: <https://metanit.com/sharp/windowsforms/4.15.php>

24. Элемент PictureBox C# [Электронный ресурс]. – Режим доступа: <https://metanit.com/sharp/windowsforms/4.16.php>

25. Окно сообщения MessageBox C# [Электронный ресурс]. – Режим доступа: <https://metanit.com/sharp/windowsforms/4.19.php>

26. ChatGPT-4o [Электронный ресурс]. – Режим доступа: [Chat GPT: нейросеть на русском языке. Онлайн-сервис в России](#)

# **ПРИЛОЖЕНИЯ**



## **Приложение 1. Ссылка на программу**

Скачать программу вы можете по ссылке ниже:

<https://github.com/Ra1den19/KursovayaRabota/tree/main>

## Приложение 2. Исходный код программы

### AddResumeForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class AddResumeForm : Form
    {
        int userId = AuthForm.UserId;

        public AddResumeForm()
        {
            InitializeComponent();
            toolTip_add.SetToolTip(pictureBox_addresume, "Добавить");
            toolTip_cancel.SetToolTip(pictureBox_cancel, "Закрыть");
        }

        private void pictureBox_cancel_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void pictureBox_addresume_Click(object sender, EventArgs e)
        {
            try
```

```

{
    string grafik = comboBox_grafik.SelectedItem.ToString();

    string name = textBox_name.Text;

    string mail = textBox_mail.Text;

    string dol = textBox_dol.Text;

    string salary = textBox_sal.Text;

    string edu = comboBox_edu.SelectedItem.ToString();

    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"insert into Резюме(КодПользователя, Наименование, ЖелаемаяДолжность,
Зарплата, Образование, ЭлПочта, ГрафикРаботы) values('{userId}', '{name}', '{dol}', '{salary}', '{edu}', '{mail}',
'{grafik}'); select КодРезюме, Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта,
ГрафикРаботы from Резюме where КодПользователя = '{userId}'";

    SqlCommand com = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(com);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    DialogResult res = MessageBox.Show("Резюме успешно опубликовано", "Сообщение",
MessageBoxButtons.OK, MessageBoxIcon.Information);

    textBox_name.Clear();

    textBox_mail.Clear();

    textBox_dol.Clear();

    textBox_sal.Clear();
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
}
}

private void pictureBox_cancel_MouseEnter(object sender, EventArgs e)
{
    pictureBox_cancel.BackColor = Color.Silver;
}

private void pictureBox_cancel_MouseLeave(object sender, EventArgs e)
{

```

```
        pictureBox_cancel.BackColor = Color.White;
    }

    private void pictureBox_addresume_MouseEnter(object sender, EventArgs e)
    {
        pictureBox_addresume.BackColor = Color.Silver;
    }

    private void pictureBox_addresume_MouseLeave(object sender, EventArgs e)
    {
        pictureBox_addresume.BackColor = Color.White;
    }
}
```

## AddVacancyForm.cs

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace НайтиРаботу

{

    public partial class AddVacancyForm : Form

    {

        int userId = AuthForm.UserId;

        public AddVacancyForm()

        {

            InitializeComponent();

            toolTip_add.SetToolTip(pictureBox_addvacancy, "Добавить");

            toolTip_cancel.SetToolTip(pictureBox_cancel, "Закрыть");

        }

        private void pictureBox_addvacancy_Click(object sender, EventArgs e)

        {

            try

            {

                string name = textBox_name.Text;
```

```

string city = textBox_city.Text;

bool student = checkBox_stud.Checked;

string exp = comboBox_exp.SelectedItem.ToString();

string spec = textBox_spec.Text;

string graph = comboBox_graph.SelectedItem.ToString();

string salary = textBox_sal.Text;

string edu = comboBox_edu.SelectedItem.ToString();

string otrasl = comboBox_otr.SelectedItem.ToString();


SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

if (checkBox_stud.Checked == true)
{
    string query = $"insert into Вакансии(КодПользователя, Наименование, Город, Студентам,
ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование) values('{userId}', '{name}',
'{city}', 'Да', '{exp}', '{spec}', '{graph}', '{otrasl}', '{salary}', '{edu}'); select КодВакансии, Наименование, Город,
Студентам, ОпытРаботы, Специализация, Отрасль, Зарплата, Образование, ГрафикРаботы from Вакансии
where КодПользователя = '{userId}'";

    SqlCommand com = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(com);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    DialogResult res = MessageBox.Show("Вакансия успешно добавлена", "Сообщение",
MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else if (checkBox_stud.Checked == false)
{
    string query = $"insert into Вакансии(КодПользователя ,Наименование, Город, Студентам,
ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование) values('{userId}', '{name}',
'{city}', 'Нет', '{exp}', '{spec}', '{graph}', '{otrasl}', '{salary}', '{edu}'); select КодВакансии, Наименование, Город,
Студентам, ОпытРаботы, Специализация, Отрасль, Зарплата, Образование, ГрафикРаботы from Вакансии
where КодПользователя = '{userId}'";

```

```

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        DialogResult res = MessageBox.Show("Вакансия успешно добавлена", "Сообщение",
        MessageBoxButtons.OK, MessageBoxIcon.Information);

    }

}

catch (Exception ex)

{

    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);

}

}

private void AddVacancyForm_Load(object sender, EventArgs e)

{

    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"select КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
    Специализация, Отрасль, Зарплата, Образование, ГрафикРаботы from Вакансии where КодПользователя =
    '{userId}'";

    SqlCommand com = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(com);

}

private void pictureBox_cancel_Click(object sender, EventArgs e)

{

    this.Close();

}

```

```
private void pictureBox_cancel_MouseEnter(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_cancel.BackColor = Color.Silver;
```

```
}
```

```
private void pictureBox_cancel_MouseLeave(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_cancel.BackColor = Color.White;
```

```
}
```

```
private void pictureBox_addvacancy_MouseEnter(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_addvacancy.BackColor = Color.Silver;
```

```
}
```

```
private void pictureBox_addvacancy_MouseLeave(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_addvacancy.BackColor = Color.White;
```

```
}
```

```
}
```

```
}
```



## AuthForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace НайтиРаботу
{
    public partial class AuthForm : Form
    {
        public static int UserId;

        public AuthForm()
        {
            InitializeComponent();
        }

        private void auth_button_Click(object sender, EventArgs e)
        {
            try
            {
                string login = textBox_login.Text;
                string password = textBox_password.Text;
                if(textBox_login.Text != "" && textBox_password.Text != "")
                {
                    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
                    string query = $"select * from Пользователи where Логин = '{login}' and Пароль = '{password}'";

                    SqlCommand com = new SqlCommand(query, con);
                    con.Open();
                    SqlDataReader rd = com.ExecuteReader();
```

```

        if (rd.HasRows)
        {
            while (rd.Read())
            {
                UserId = (int)rd["КодПользователя"];
                string role = rd["Роль"].ToString();
                if (role == "Соискатель")
                {
                    ClientForm cf = new ClientForm();
                    cf.ShowDialog();

                    textBox_login.Clear();
                    textBox_password.Clear();
                }
                else if (role == "Работодатель")
                {
                    RabotodatelForm rf = new RabotodatelForm();
                    rf.ShowDialog();

                    textBox_login.Clear();
                    textBox_password.Clear();
                }
            }
        }
        else
        {
            DialogResult res = MessageBox.Show("Неверный логин или пароль", "Ошибка",
            MessageBoxButtons.OK, MessageBoxIcon.Error);

        }
        else
        {
            DialogResult res = MessageBox.Show("Поля для заполнения пусты!", "Сообщение",
            MessageBoxButtons.OK, MessageBoxIcon.Warning);

        }
    }
    catch (Exception ex)
    {

```

```
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void checkBoxPas_CheckedChanged(object sender, EventArgs e)
{
    if(checkBoxPas.Checked == true)
    {
        textBox_password.UseSystemPasswordChar = false;
        checkBoxPas.Text = "Скрыть пароль";
    }
    else if(checkBoxPas.Checked == false)
    {
        textBox_password.UseSystemPasswordChar = true;
        checkBoxPas.Text = "Показать пароль";
    }
}
}
```

## ClientForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class ClientForm : Form
    {
        public ClientForm()
        {
            InitializeComponent();
            toolTip_exit.SetToolTip(pictureBox_closeapp, "Закрыть программу");
            toolTip_logout.SetToolTip(pictureBox_logout, "Выйти из учетной записи");
        }

        private void percab_button_Click(object sender, EventArgs e)
        {
            try
            {
                percab_button.BackColor = Color.FromArgb(0, 64, 0);
                favvacancies_button.BackColor = Color.Green;
                myotkliki_button.BackColor = Color.Green;
                vacancies_button.BackColor = Color.Green;
                PerCabForm pc = new PerCabForm();
                pc.TopLevel = false;
                pc.FormBorderStyle = FormBorderStyle.None;
                pc.Dock = DockStyle.Fill;
                panel1.Controls.Clear();
```

```

        panel1.Controls.Add(pc);
        pc.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void vacancies_button_Click(object sender, EventArgs e)
{
    try
    {
        percab_button.BackColor = Color.Green;
        favvacancies_button.BackColor = Color.Green;
        myotkliki_button.BackColor = Color.Green;
        vacancies_button.BackColor = Color.FromArgb(0, 64, 0);
        VacanciesForm vf = new VacanciesForm();
        vf.TopLevel = false;
        vf.FormBorderStyle = FormBorderStyle.None;
        vf.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(vf);
        vf.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void myotkliki_button_Click(object sender, EventArgs e)
{
    try

```

```

    {
        percab_button.BackColor = Color.Green; ;
        favvacancies_button.BackColor = Color.Green;
        myotkliki_button.BackColor = Color.FromArgb(0, 64, 0);
        vacancies_button.BackColor = Color.Green;
        MyOtklikiForm of = new MyOtklikiForm();
        of.TopLevel = false;
        of.FormBorderStyle = FormBorderStyle.None;
        of.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(of);
        of.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void favvacancies_button_Click(object sender, EventArgs e)
{
    try
    {
        percab_button.BackColor = Color.Green;
        favvacancies_button.BackColor = Color.FromArgb(0, 64, 0);
        myotkliki_button.BackColor = Color.Green;
        vacancies_button.BackColor = Color.Green;
        FavVacanciesForm fv = new FavVacanciesForm();
        fv.TopLevel = false;
        fv.FormBorderStyle = FormBorderStyle.None;
        fv.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(fv);
        fv.Show();
    }
}

```

```
        catch (Exception ex)
        {
            DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }
```

```
private void pictureBox_closeapp_Click(object sender, EventArgs e)
{
    DialogResult res = MessageBox.Show("Закрыть программу?", "Сообщение",
MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    if (res == DialogResult.Yes)
    {
        Application.Exit();
    }
}
```

```
private void pictureBox_logout_Click(object sender, EventArgs e)
{
    DialogResult res = MessageBox.Show("Выйти из учетной записи?", "Сообщение",
MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    if (res == DialogResult.Yes)
    {
        this.Close();
    }
}
```

```
private void percab_button_MouseEnter(object sender, EventArgs e)
{
    percab_button.ForeColor = Color.Red;
}
```

```
private void percab_button_MouseLeave(object sender, EventArgs e)
{
    percab_button.ForeColor = Color.White;
}
```

```
private void favvacancies_button_MouseEnter(object sender, EventArgs e)
{
    favvacancies_button.ForeColor = Color.Red;
}
```

```
private void favvacancies_button_MouseLeave(object sender, EventArgs e)
{
    favvacancies_button.ForeColor = Color.White;
}
```

```
private void myotklike_button_MouseEnter(object sender, EventArgs e)
{
    myotklike_button.ForeColor = Color.Red;
}
```

```
private void myotklike_button_MouseLeave(object sender, EventArgs e)
{
    myotklike_button.ForeColor = Color.White;
}
```

```
private void vacancies_button_MouseEnter(object sender, EventArgs e)
{
    vacancies_button.ForeColor = Color.Red;
}
```

```
private void vacancies_button_MouseLeave(object sender, EventArgs e)
{
    vacancies_button.ForeColor = Color.White;
}
```

```
private void ClientForm_Load(object sender, EventArgs e)
{
    try
    {
        percab_button.BackColor = Color.FromArgb(0, 64, 0);
        favvacancies_button.BackColor = Color.Green;
    }
}
```



```

        myotkliki_button.BackColor = Color.Green;
        vacancies_button.BackColor= Color.Green;
        PerCabForm pc = new PerCabForm();
        pc.TopLevel = false;
        pc.FormBorderStyle = FormBorderStyle.None;
        pc.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(pc);
        pc.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void pictureBox_closeapp_MouseEnter(object sender, EventArgs e)
{
    pictureBox_closeapp.BackColor = Color.FromArgb(0, 192, 0);
}

```

```

private void pictureBox_closeapp_MouseLeave(object sender, EventArgs e)
{
    pictureBox_closeapp.BackColor = Color.Green;
}

```

```

private void pictureBox_logout_MouseEnter(object sender, EventArgs e)
{
    pictureBox_logout.BackColor = Color.FromArgb(0, 192, 0);
}

```

```

private void pictureBox_logout_MouseLeave(object sender, EventArgs e)
{
    pictureBox_logout.BackColor = Color.Green;
}

```



## **DataBaseConfig.cs**

```
using System;

using System.Collections.Generic;

using System.Data.Sql;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace НайтиРаботу
{
    public static class DataBaseConfig
    {
        public static string ConnectionString { get; private set; }


        static DataBaseConfig()
        {
            string serverName = Environment.MachineName + @"\SQLEXPRESS";
            string databaseName = "НайтиРаботу";

            ConnectionString = $"Data Source={serverName};Initial Catalog={databaseName};Integrated
Security=True";
        }
    }
}
```

## EditResumeForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class EditResumeForm : Form
    {
        int userId = AuthForm.UserId;

        public EditResumeForm()
        {
            InitializeComponent();

            toolTip_confirm.SetToolTip(pictureBox_confirm, "Подтвердить");
            toolTip_cancelorclose.SetToolTip(pictureBox_cancel, "Закрыть");
        }

        private void pictureBox_cancel_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void EditResumeForm_Load(object sender, EventArgs e)
        {
            try
            {
                SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

                string query = $"select КодРезюме, Наименование, ЖелаемаяДолжность, Зарплата,
Образование, ЭлПочта, ГрафикРаботы from Резюме where КодПользователя = '{userId}'";
```

```

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void ОтображениеДанных_SelectionChanged(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.SelectedRows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

            label_id.Text = selectedRow.Cells["КодРезюме"].Value.ToString();

            textBox_name.Text = selectedRow.Cells["Наименование"].Value.ToString();

            textBox_dol.Text = selectedRow.Cells["ЖелаемаяДолжность"].Value.ToString();

            textBox_sal.Text = selectedRow.Cells["Зарплата"].Value.ToString();

            comboBox_edu.SelectedItem = selectedRow.Cells["Образование"].Value.ToString();

            textBox_mail.Text = selectedRow.Cells["ЭлПочта"].Value.ToString();

            comboBox_grafik.SelectedItem = selectedRow.Cells["ГрафикРаботы"].Value.ToString();
        }
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void pictureBox_confirm_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.SelectedRows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];
            int id = Convert.ToInt32(selectedRow.Cells["КодРезюме"].Value);

            string newValue1 = textBox_name.Text;
            string newValue6 = textBox_dol.Text;
            string newValue7 = textBox_sal.Text;
            string newValue8 = comboBox_edu.SelectedItem.ToString();
            string newValue3 = textBox_mail.Text;
            string newValue9 = comboBox_grafik.SelectedItem.ToString();

            string query = $"update Резюме set Наименование = '{newValue1}', ЖелаемаяДолжность = '{newValue6}', Зарплата = '{newValue7}', Образование = '{newValue8}', ЭлПочта = '{newValue3}', ГрафикРаботы = '{newValue9}' where КодРезюме = '{id}'; select КодРезюме, Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Резюме where КодПользователя = '{userId}'";

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
            SqlCommand com = new SqlCommand(query, con);
            SqlDataAdapter adapter = new SqlDataAdapter(com);
            DataTable dt = new DataTable();
            adapter.Fill(dt);
            ОтображениеДанных.DataSource = dt;
        }
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

private void pictureBox_cancel_MouseEnter(object sender, EventArgs e)
{
    pictureBox_cancel.BackColor = Color.Silver;
}

```

```
}
```

```
private void pictureBox_cancel_MouseLeave(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_cancel.BackColor = Color.White;
```

```
}
```

```
private void pictureBox_confirm_MouseEnter(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_confirm.BackColor = Color.Silver;
```

```
}
```

```
private void pictureBox_confirm_MouseLeave(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_confirm.BackColor = Color.White;
```

```
}
```

```
}
```

```
}
```

## EditVacancyForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class EditVacancyForm : Form
    {
        int userId = AuthForm.UserId;

        public EditVacancyForm()
        {
            InitializeComponent();
            toolTip_confirm.SetToolTip(pictureBox_confirm, "Подтвердить");
            toolTip_cancel.SetToolTip(pictureBox_cancel, "Закрыть");
        }

        private void pictureBox_confirm_Click(object sender, EventArgs e)
        {
            try
            {
                if (ОтображениеДанных.SelectedRows.Count > 0)
                {
                    DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];
                    int id = Convert.ToInt32(selectedRow.Cells["КодВакансии"].Value);
                    string newValue1 = textBox_name.Text;
                    string newValue2 = textBox_city.Text;
                    string newValue4 = comboBox_exp.SelectedItem.ToString();
                }
            }
        }
    }
}
```



```

string newValue5 = textBox_spec.Text;

string newValue6 = comboBox_graph.SelectedItem.ToString();

string newValue7 = comboBox_otr.SelectedItem.ToString();

string newValue8 = textBox_sal.Text;

string newValue9 = comboBox_edu.SelectedItem.ToString();


SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

if (checkBox_stud.Checked == true)
{
    string query = $"update Вакансии set Наименование = '{newValue1}', Город = '{newValue2}', Студентам = 'Да', ОпытРаботы = '{newValue4}', Специализация = '{newValue5}', ГрафикРаботы = '{newValue6}', Отрасль = '{newValue7}', Зарплата = '{newValue8}', Образование = '{newValue9}' where КодВакансии = '{id}'; select КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация, Отрасль, Зарплата, Образование, ГрафикРаботы from Вакансии where КодПользователя = '{userId}'";

    SqlCommand com = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(com);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}

else if (checkBox_stud.Checked == false)
{
    string query = $"update Вакансии set Наименование = '{newValue1}', Город = '{newValue2}', Студентам = 'Нет', ОпытРаботы = '{newValue4}', Специализация = '{newValue5}', ГрафикРаботы = '{newValue6}', Отрасль = '{newValue7}', Зарплата = '{newValue8}', Образование = '{newValue9}' where КодВакансии = '{id}'; select КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация, Отрасль, Зарплата, Образование, ГрафикРаботы from Вакансии where КодПользователя = '{userId}'";

    SqlCommand com = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(com);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}

}

}

catch (Exception ex)
{

```

```

        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);

    }

}

private void pictureBox_cancel_Click(object sender, EventArgs e)
{
    this.Close();
}

private void EditVacancyForm_Load(object sender, EventArgs e)
{
    try
    {
        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        string query = $"select КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
        Специализация, Отрасль, Зарплата, Образование, ГрафикРаботы from Вакансии where КодПользователя =
        '{userId}'";

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void ОтображениеДанных_SelectionChanged(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.SelectedRows.Count > 0)
        {

```

```

DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];
label_id.Text = selectedRow.Cells["КодВакансии"].Value.ToString();
textBox_name.Text = selectedRow.Cells["Наименование"].Value.ToString();
textBox_city.Text = selectedRow.Cells["Город"].Value.ToString();
var cellValue = selectedRow.Cells["Студентам"].Value?.ToString();
if (cellValue == "Да")
{
    checkBox_stud.Checked = true;
}
else if (cellValue == "Нет")
{
    checkBox_stud.Checked = false;
}
comboBox_exp.SelectedItem = selectedRow.Cells["ОпытРаботы"].Value.ToString();
textBox_spec.Text = selectedRow.Cells["Специализация"].Value.ToString();
comboBox_graph.SelectedItem = selectedRow.Cells["ГрафикРаботы"].Value.ToString();
comboBox_otr.SelectedItem = selectedRow.Cells["Отрасль"].Value.ToString();
textBox_sal.Text = selectedRow.Cells["Зарплата"].Value.ToString();
comboBox_edu.SelectedItem = selectedRow.Cells["Образование"].Value.ToString();

    }
}
catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

private void pictureBox_cancel_MouseEnter(object sender, EventArgs e)
{
    pictureBox_cancel.BackColor = Color.Silver;
}

private void pictureBox_cancel_MouseLeave(object sender, EventArgs e)
{

```

```
        pictureBox_cancel.BackColor = Color.White;
    }

    private void pictureBox_confirm_MouseEnter(object sender, EventArgs e)
    {
        pictureBox_confirm.BackColor = Color.Silver;
    }

    private void pictureBox_confirm_MouseLeave(object sender, EventArgs e)
    {
        pictureBox_confirm.BackColor = Color.White;
    }
}
}
```

## FavResumesForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class FavResumesForm : Form
    {
        int userId = AuthForm.UserId;

        public FavResumesForm()
        {
            InitializeComponent();
            toolTip_delfav.SetToolTip(pictureBox_delffromfav, "Убрать из избранного");
        }

        private void button_otklik_Click(object sender, EventArgs e)
        {
            try
            {
                if (ОтображениеДанных.Rows.Count > 0)
                {
                    DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];
                    int id = Convert.ToInt32(selectedRow.Cells["КодРезюме"].Value);

                    string query = $"insert into Приглашения(КодПользователя, КодРезюме) values('{userId}', '{id}')";
                    select ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество, Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from ИзбранныеРезюме inner join Резюме on ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where ИзбранныеРезюме.КодПользователя = '{userId}";

                    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
```

```

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

private void pictureBox_delfromfav_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

            int id = Convert.ToInt32(selectedRow.Cells["КодРезюме"].Value);

            string query = $"delete from ИзбранныеРезюме where КодРезюме = '{id}'; select
ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество, Наименование, ЖелаемаяДолжность, Зарплата,
Образование, ЭлПочта, ГрафикРаботы from ИзбранныеРезюме inner join Резюме on
ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join Пользователи on Резюме.КодПользователя =
Пользователи.КодПользователя where ИзбранныеРезюме.КодПользователя = '{userId}'";

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            SqlCommand com = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(com);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }
    }

    catch (Exception ex)
    {

```

```

        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);

    }

}

private void FavResumesForm_Load(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"select ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество,
Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from
ИзбранныеРезюме inner join Резюме on ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join
Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where
ИзбранныеРезюме.КодПользователя = '{userId}'";

    SqlCommand command = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(command);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}

private void comboBox_sort_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        if (comboBox_sort.SelectedIndex == 0)
        {
            string query = $"select ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество,
Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from
ИзбранныеРезюме inner join Резюме on ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join
Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where
ИзбранныеРезюме.КодПользователя = '{userId}' order by Зарплата";

            SqlCommand command = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(command);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }

        else if (comboBox_sort.SelectedIndex == 1)
    }
}

```

```

        {
            string query = $"select ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество,
Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from
ИзбранныеРезюме inner join Резюме on ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join
Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where
ИзбранныеРезюме.КодПользователя = '{userId}' order by Образование";

            SqlCommand command = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(command);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }

        else if (comboBox_sort.SelectedIndex == 2)
        {
            string query = $"select ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество,
Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from
ИзбранныеРезюме inner join Резюме on ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join
Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where
ИзбранныеРезюме.КодПользователя = '{userId}' order by ГрафикРаботы";

            SqlCommand command = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(command);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void comboBox_filteredu_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        string index = comboBox_filteredu.SelectedItem.ToString();

        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
    }
}

```



```
        string query = $"select ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество,
Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from
ИзбранныеРезюме inner join Резюме on ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join
Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where
ИзбранныеРезюме.КодПользователя = '{userId}' and Образование = '{index}'";
```

```
        SqlCommand command = new SqlCommand(query, con);
```

```
        SqlDataAdapter adapter = new SqlDataAdapter(command);
```

```
        DataTable dt = new DataTable();
```

```
        adapter.Fill(dt);
```

```
        ОтображениеДанных.DataSource = dt;
```

```
    }
```

```
    catch (Exception ex)
```

```
    {
```

```
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
```

```
    }
```

```
}
```

```
private void comboBox_filtergraph_SelectedIndexChanged(object sender, EventArgs e)
```

```
{
```

```
    try
```

```
    {
```

```
        string index = comboBox_filtergraph.SelectedItem.ToString();
```

```
        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
```

```
        string query = $"select ИзбранныеРезюме.КодРезюме, Фамилия, Имя, Отчество,
Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from
ИзбранныеРезюме inner join Резюме on ИзбранныеРезюме.КодРезюме = Резюме.КодРезюме inner join
Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where
ИзбранныеРезюме.КодПользователя = '{userId}' and ГрафикРаботы = '{index}'";
```

```
        SqlCommand command = new SqlCommand(query, con);
```

```
        SqlDataAdapter adapter = new SqlDataAdapter(command);
```

```
        DataTable dt = new DataTable();
```

```
        adapter.Fill(dt);
```

```
        ОтображениеДанных.DataSource = dt;
```

```
    }
```

```
    catch (Exception ex)
```

```
    {
```

```
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
```

```
    }
```

```
}
```

```
private void pictureBox_delfromfav_MouseEnter(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_delfromfav.BackColor = Color.Silver;
```

```
}
```

```
private void pictureBox_delfromfav_MouseLeave(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_delfromfav.BackColor = Color.White;
```

```
}
```

```
}
```

```
}
```

## FavVacanciesForm.cs

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Reflection;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace НайтиРаботу

{

    public partial class FavVacanciesForm : Form

    {

        int userId = AuthForm.UserId;

        public FavVacanciesForm()

        {

            InitializeComponent();

            toolTip_delfav.SetToolTip(pictureBox_delfromfav, "Убрать из избранного");

        }

        private void FavVacanciesForm_Load(object sender, EventArgs e)

        {

            try

            {

                SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

                string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where ИзбранныеВакансии.КодПользователя = '{userId}'";

                SqlCommand com = new SqlCommand(query, con);

                SqlDataAdapter adapter = new SqlDataAdapter(com);

                DataTable dt = new DataTable();

                adapter.Fill(dt);
```

```

        ОтображениеДанных.DataSource = dt;
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void comboBox_filterotrasli_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        string otrasl = comboBox_filterotrasli.SelectedItem.ToString();

        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город,
        Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии
        inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
        ИзбранныеВакансии.КодПользователя = '{userId}' and Отрасль = '{otrasl}'";

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void pictureBox_delfromfav_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {

```

```

        DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

        int id = Convert.ToInt32(selectedRow.Cells["КодВакансии"].Value);

        string query = $"delete from ИзбранныеВакансии where КодВакансии = '{id}'; select
ИзбранныеВакансии.КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация,
ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии inner join ИзбранныеВакансии on
Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where ИзбранныеВакансии.КодПользователя =
'{userId}'";

        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
}
}

private void button_otklik_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

            int id = Convert.ToInt32(selectedRow.Cells["КодВакансии"].Value);

            string query = $"insert into Отклики(КодПользователя, КодВакансии) values('{userId}',
'{id}'); select ИзбранныеВакансии.КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии inner join
ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
ИзбранныеВакансии.КодПользователя = '{userId}'";

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            SqlCommand com = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(com);

            DataTable dt = new DataTable();

```

```

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

private void comboBox_sort_SelectedIndexChanged(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    if (comboBox_sort.SelectedIndex == 0)
    {
        string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город,
        Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии
        inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
        ИзбранныеВакансии.КодПользователя = '{userId}' order by Зарплата";

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

    else if (comboBox_sort.SelectedIndex == 1)
    {
        string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город,
        Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии
        inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
        ИзбранныеВакансии.КодПользователя = '{userId}' order by ОпытРаботы";

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

```

```

else if (comboBox_sort.SelectedIndex == 2)
{
    string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город,
Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии
inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
ИзбранныеВакансии.КодПользователя = '{userId}' order by ГрафикРаботы";

    SqlCommand command = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(command);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}

else if (comboBox_sort.SelectedIndex == 3)
{
    string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город,
Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии
inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
ИзбранныеВакансии.КодПользователя = '{userId}' order by Отрасль";

    SqlCommand command = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(command);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}
}

private void comboBox_filter_SelectedIndexChanged(object sender, EventArgs e)
{
    string index = comboBox_filter.SelectedItem.ToString();

    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город, Студентам,
ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии inner join
ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
ИзбранныеВакансии.КодПользователя = '{userId}' and ОпытРаботы = '{index}'";

    SqlCommand command = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(command);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}

```

```
}
```

```
private void comboBox_filtergraph_SelectedIndexChanged(object sender, EventArgs e)
{
    string index = comboBox_filtergraph.SelectedItem.ToString();

    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город, Студентам,
ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии inner join
ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
ИзбранныеВакансии.КодПользователя = '{userId}' and ГрафикРаботы = '{index}'";

    SqlCommand command = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(command);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}
```

```
private void checkBox_stud_CheckedChanged(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    if (checkBox_stud.Checked == true)
    {
        string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город,
Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии
inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
ИзбранныеВакансии.КодПользователя = '{userId}' and Студентам = 'Да'";

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
    else
    {
        string query = $"select ИзбранныеВакансии.КодВакансии, Наименование, Город,
Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии
inner join ИзбранныеВакансии on Вакансии.КодВакансии = ИзбранныеВакансии.КодВакансии where
ИзбранныеВакансии.КодПользователя = '{userId}'";

        SqlCommand command = new SqlCommand(query, con);
```



```

        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        ОтображениеДанных.DataSource = dt;
    }
}

private void pictureBox_delfromfav_MouseEnter(object sender, EventArgs e)
{
    pictureBox_delfromfav.BackColor = Color.Silver;
}

private void pictureBox_delfromfav_MouseLeave(object sender, EventArgs e)
{
    pictureBox_delfromfav.BackColor = Color.White;
}
}
}

```

## MainForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class MainForm : Form
    {
        public MainForm()
        {
            InitializeComponent();

            private void exit_button_Click(object sender, EventArgs e)
            {
                DialogResult res = MessageBox.Show("Закрыть программу?", "Сообщение",
                MessageBoxButtons.YesNo, MessageBoxIcon.Question);
                if (res == DialogResult.Yes)
                {
                    Application.Exit();
                }
            }

            private void auth_button_Click(object sender, EventArgs e)
            {
                try
                {
                    auth_button.BackColor = Color.FromArgb(0, 64, 0);
                    reg_button.BackColor = Color.Green;
                }
            }
        }
    }
}
```

```

        AuthForm au = new AuthForm();
        au.TopLevel = false;
        au.FormBorderStyle = FormBorderStyle.None;
        au.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(au);
        au.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void reg_button_Click(object sender, EventArgs e)
{
    try
    {
        reg_button.BackColor = Color.FromArgb(0, 64, 0);
        auth_button.BackColor = Color.Green;
        RegForm reg = new RegForm();
        reg.TopLevel = false;
        reg.FormBorderStyle = FormBorderStyle.None;
        reg.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(reg);
        reg.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void MainForm_Load(object sender, EventArgs e)
{
    try
    {
        auth_button.BackColor = Color.FromArgb(0, 64, 0);
        reg_button.BackColor = Color.Green;
        AuthForm au = new AuthForm();
        au.TopLevel = false;
        au.FormBorderStyle = FormBorderStyle.None;
        au.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(au);
        au.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void auth_button_MouseEnter(object sender, EventArgs e)
{
    auth_button.ForeColor = Color.Red;
}

```

```

private void auth_button_MouseLeave(object sender, EventArgs e)
{
    auth_button.ForeColor = Color.White;
}

```

```

private void reg_button_MouseEnter(object sender, EventArgs e)
{
    reg_button.ForeColor = Color.Red;
}

```

```
private void reg_button_MouseLeave(object sender, EventArgs e)
{
    reg_button.ForeColor = Color.White;
}

private void exit_button_MouseEnter(object sender, EventArgs e)
{
    exit_button.ForeColor = Color.Red;
}

private void exit_button_MouseLeave(object sender, EventArgs e)
{
    exit_button.ForeColor = Color.White;
}
}
```

## MyInvitesForm.cs

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace НайтиРаботу

{

    public partial class MyInvitesForm : Form

    {

        int userId = AuthForm.UserId;

        public MyInvitesForm()

        {

            InitializeComponent();

            toolTip_delete.SetToolTip(pictureBox_delete, "Отменить приглашение");

        }

        private void MyInvitesForm_Load(object sender, EventArgs e)

        {

            try

            {

                SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

                string query = $"select Приглашения.КодРезюме, Фамилия, Имя, Отчество, Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Резюме inner join Приглашения on Резюме.КодРезюме = Приглашения.КодРезюме inner join Пользователи on Резюме.КодПользователя = Пользователи.КодПользователя where Приглашения.КодПользователя = '{userId}'";

                SqlCommand com = new SqlCommand(query, con);

                SqlDataAdapter adapter = new SqlDataAdapter(com);

                DataTable dt = new DataTable();

                adapter.Fill(dt);

                ОтображениеДанных.DataSource = dt;
```

```

    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void textBox_find_TextChanged(object sender, EventArgs e)
{
    try
    {
        string find = textBox_find.Text.Trim();

        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        string query = $"select Приглашения.КодРезюме, Фамилия, Имя, Отчество, Наименование,
ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Резюме inner join Приглашения
on Резюме.КодРезюме = Приглашения.КодРезюме inner join Пользователи on Резюме.КодПользователя =
Пользователи.КодПользователя where Приглашения.КодПользователя = '{userId}' and Наименование like
'%{find}%'";

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void pictureBox_delete_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {

```

```

        DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

        int id = Convert.ToInt32(selectedRow.Cells["КодРезюме"].Value);

        string query = $"delete from Приглашения where КодРезюме = '{id}'; select
Приглашения.КодРезюме, Фамилия, Имя, Отчество, Наименование, ЖелаемаяДолжность, Зарплата,
Образование, ЭлПочта, ГрафикРаботы from Резюме inner join Приглашения on Резюме.КодРезюме =
Приглашения.КодРезюме inner join Пользователи on Резюме.КодПользователя =
Пользователи.КодПользователя where Приглашения.КодПользователя = '{userId}'";

        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;

    }

}

catch (Exception ex)

{

    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);

}

}

private void pictureBox_delete_MouseEnter(object sender, EventArgs e)

{

    pictureBox_delete.BackColor = Color.Silver;

}

private void pictureBox_delete_MouseLeave(object sender, EventArgs e)

{

    pictureBox_delete.BackColor = Color.White;

}

}

}

```



## MyOtklikiForm.cs

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace НайтиРаботу

{

    public partial class MyOtklikiForm : Form

    {

        int userId = AuthForm.UserId;

        public MyOtklikiForm()

        {

            InitializeComponent();

            toolTip_delete.SetToolTip(pictureBox_delete, "Отменить отклик");

        }

        private void MyOtklikiForm_Load(object sender, EventArgs e)

        {

            try

            {

                SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

                string query = $"select Отклики.КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии inner join Отклики on Вакансии.КодВакансии = Отклики.КодВакансии where Отклики.КодПользователя = '{userId}'";

                SqlCommand com = new SqlCommand(query, con);

                SqlDataAdapter adapter = new SqlDataAdapter(com);

                DataTable dt = new DataTable();

                adapter.Fill(dt);

                ОтображениеДанных.DataSource = dt;

            }

            catch { }

        }

    }

}
```

```

    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void textBox_find_TextChanged(object sender, EventArgs e)
{
    try
    {
        string find = textBox_find.Text.Trim();

        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        string query = $"select Отклики.КодВакансии, Наименование, Город, Студентам,
        ОпытРаботы, Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии inner join
        Отклики on Вакансии.КодВакансии = Отклики.КодВакансии where Отклики.КодПользователя = '{userId}'
        and Наименование like '%{find}%'";

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void pictureBox_delete_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

```

```

        int id = Convert.ToInt32(selectedRow.Cells["КодВакансии"].Value);

        string query = $"delete from Отклики where КодВакансии = '{id}'; select
Отклики.КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация, ГрафикРаботы,
Отрасль, Зарплата, Образование from Вакансии inner join Отклики on Вакансии.КодВакансии =
Отклики.КодВакансии where Отклики.КодПользователя = '{userId}'";

        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

private void pictureBox_delete_MouseEnter(object sender, EventArgs e)
{
    pictureBox_delete.BackColor = Color.Silver;
}

private void pictureBox_delete_MouseLeave(object sender, EventArgs e)
{
    pictureBox_delete.BackColor = Color.White;
}
}
}

```

## PerCabForm.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class PerCabForm : Form
    {
        int userId = AuthForm.UserId;

        public PerCabForm()
        {
            InitializeComponent();

            toolTip_add.SetToolTip(pictureBox_addresume, "Добавить");
            toolTip_edit.SetToolTip(pictureBox_editresume, "Изменить");
            toolTip_del.SetToolTip(pictureBox_deleteresume, "Удалить");
        }

        private void pictureBox_addresume_Click(object sender, EventArgs e)
        {
            AddResumeForm ar = new AddResumeForm();

            ar.ShowDialog();
        }

        private void pictureBox_editresume_Click(object sender, EventArgs e)
        {
            EditResumeForm er = new EditResumeForm();
```

```

        er.ShowDialog();
    }

    private void PerCabForm_Load(object sender, EventArgs e)
    {
        try
        {
            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            string query = $"select КодРезюме, Наименование, ЖелаемаяДолжность, Зарплата,
Образование, ЭлПочта, ГрафикРаботы from Резюме where КодПользователя = '{userId}'";

            SqlCommand com = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(com);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }
        catch (Exception ex)
        {
            DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }

    private void pictureBox_deleteresume_Click(object sender, EventArgs e)
    {
        try
        {
            if (ОтображениеДанных.Rows.Count > 0)
            {
                DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

                int id = Convert.ToInt32(selectedRow.Cells["КодРезюме"].Value);

                string query = $"delete from Резюме where КодРезюме = '{id}'; select КодРезюме,
Наименование, ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Резюме where
КодПользователя = '{userId}'";

                SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

                SqlCommand com = new SqlCommand(query, con);

                SqlDataAdapter adapter = new SqlDataAdapter(com);

```

```

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

```

```

private void pictureBox_addresume_MouseEnter(object sender, EventArgs e)
{
    pictureBox_addresume.BackColor = Color.Silver;
}

```

```

private void pictureBox_addresume_MouseLeave(object sender, EventArgs e)
{
    pictureBox_addresume.BackColor = Color.White;
}

```

```

private void pictureBox_editresume_MouseEnter(object sender, EventArgs e)
{
    pictureBox_editresume.BackColor = Color.Silver;
}

```

```

private void pictureBox_editresume_MouseLeave(object sender, EventArgs e)
{
    pictureBox_editresume.BackColor = Color.White;
}

```

```

private void pictureBox_deleteresume_MouseEnter(object sender, EventArgs e)
{
    pictureBox_deleteresume.BackColor = Color.Silver;
}

```

```
private void pictureBox_deleteresome_MouseLeave(object sender, EventArgs e)
{
    pictureBox_deleteresome.BackColor = Color.White;
}
}
```

## PerCabRabForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class PerCabRabForm : Form
    {
        int userId = AuthForm.UserId;

        public PerCabRabForm()
        {
            InitializeComponent();

            toolTip_add.SetToolTip(pictureBox_addvacancy, "Добавить");
            toolTip_edit.SetToolTip(pictureBox_editvacancy, "Изменить");
            toolTip_del.SetToolTip(pictureBox_deletevacancy, "Удалить");
        }

        private void PerCabRabForm_Load(object sender, EventArgs e)
        {
            try
            {
                SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

                string query = $"select КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация, Отрасль, Зарплата, Образование, ГрафикРаботы from Вакансии where КодПользователя = '{userId}'";

                SqlCommand com = new SqlCommand(query, con);

                SqlDataAdapter adapter = new SqlDataAdapter(com);

                DataTable dt = new DataTable();
```



```

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void pictureBox_addvacancy_Click(object sender, EventArgs e)
{
    AddVacancyForm avf = new AddVacancyForm();
    avf.ShowDialog();
}

private void pictureBox_deletevacancy_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];
            int id = Convert.ToInt32(selectedRow.Cells["КодВакансии"].Value);

            string query = $"delete from Вакансии where КодВакансии = '{id}'; select КодВакансии,
Наименование, Город, Студентам, ОпытРаботы, Специализация, Отрасль, Зарплата, Образование,
ГрафикРаботы from Вакансии where КодПользователя = '{userId}'";

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
            SqlCommand com = new SqlCommand(query, con);
            SqlDataAdapter adapter = new SqlDataAdapter(com);
            DataTable dt = new DataTable();
            adapter.Fill(dt);
            ОтображениеДанных.DataSource = dt;
        }
    }
    catch (Exception ex)
    {

```

```
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
```

```
    }
}
```

```
private void pictureBox_editvacancy_Click(object sender, EventArgs e)
{
    EditVacancyForm ev = new EditVacancyForm();
    ev.ShowDialog();
}
```

```
private void pictureBox_addvacancy_MouseEnter(object sender, EventArgs e)
{
    pictureBox_addvacancy.BackColor = Color.Silver;
}
```

```
private void pictureBox_addvacancy_MouseLeave(object sender, EventArgs e)
{
    pictureBox_addvacancy.BackColor = Color.White;
}
```

```
private void pictureBox_editvacancy_MouseEnter(object sender, EventArgs e)
{
    pictureBox_editvacancy.BackColor = Color.Silver;
}
```

```
private void pictureBox_editvacancy_MouseLeave(object sender, EventArgs e)
{
    pictureBox_editvacancy.BackColor = Color.White;
}
```

```
private void pictureBox_deletevacancy_MouseEnter(object sender, EventArgs e)
{
    pictureBox_deletevacancy.BackColor = Color.Silver;
}
```

```
private void pictureBox_deletevacancy_MouseLeave(object sender, EventArgs e)
{
    pictureBox_deletevacancy.BackColor = Color.White;
}
}
```

## Program.cs

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;


namespace НайтиРаботу
{
    internal static class Program
    {
        /// <summary>
        /// Главная точка входа для приложения.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new MainForm());
        }
    }
}
```

## **RabotodateIForm.cs**

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class RabotodateIForm : Form
    {
        public RabotodateIForm()
        {
            InitializeComponent();
            toolTip_exit.SetToolTip(pictureBox_closeapp, "Закрыть программу");
            toolTip_logout.SetToolTip(pictureBox_logout, "Выйти из учетной записи");
        }

        private void pictureBox_closeapp_Click(object sender, EventArgs e)
        {
            DialogResult res = MessageBox.Show("Закрыть программу?", "Сообщение",
            MessageBoxButtons.YesNo, MessageBoxIcon.Question);

            if (res == DialogResult.Yes)
            {
                Application.Exit();
            }
        }

        private void percab_button_Click(object sender, EventArgs e)
        {
            try
            {

```

```

        percab_button.BackColor = Color.FromArgb(0, 64, 0);
        favresumes_button.BackColor = Color.Green;
        myotkliki_button.BackColor = Color.Green;
        resumes_button.BackColor = Color.Green;
        PerCabRabForm pcr = new PerCabRabForm();
        pcr.TopLevel = false;
        pcr.FormBorderStyle = FormBorderStyle.None;
        pcr.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(pcr);
        pcr.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void pictureBox_logout_Click(object sender, EventArgs e)
{
    DialogResult res = MessageBox.Show("Выйти из учетной записи?", "Сообщение",
    MessageBoxButtons.YesNo, MessageBoxIcon.Question);
    if (res == DialogResult.Yes)
    {
        this.Close();
    }
}

private void percab_button_MouseEnter(object sender, EventArgs e)
{
    percab_button.ForeColor = Color.Red;
}

private void percab_button_MouseLeave(object sender, EventArgs e)
{

```

```
    percab_button.ForeColor = Color.White;
}
```

```
private void favresumes_button_MouseEnter(object sender, EventArgs e)
{
    favresumes_button.ForeColor = Color.Red;
}
```

```
private void favresumes_button_MouseLeave(object sender, EventArgs e)
{
    favresumes_button.ForeColor = Color.White;
}
```

```
private void myotklike_button_MouseEnter(object sender, EventArgs e)
{
    myotklike_button.ForeColor = Color.Red;
}
```

```
private void myotklike_button_MouseLeave(object sender, EventArgs e)
{
    myotklike_button.ForeColor = Color.White;
}
```

```
private void resumes_button_MouseEnter(object sender, EventArgs e)
{
    resumes_button.ForeColor = Color.Red;
}
```

```
private void resumes_button_MouseLeave(object sender, EventArgs e)
{
    resumes_button.ForeColor = Color.White;
}
```

```
private void resumes_button_Click(object sender, EventArgs e)
{
    try
```

```

    {
        percab_button.BackColor = Color.Green;
        favresumes_button.BackColor = Color.Green;
        myotkliki_button.BackColor = Color.Green;
        resumes_button.BackColor = Color.FromArgb(0, 64, 0);
        ResumesForm rf = new ResumesForm();
        rf.TopLevel = false;
        rf.FormBorderStyle = FormBorderStyle.None;
        rf.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(rf);
        rf.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void myotkliki_button_Click(object sender, EventArgs e)
{
    try
    {
        percab_button.BackColor = Color.Green;
        favresumes_button.BackColor = Color.Green;
        myotkliki_button.BackColor = Color.FromArgb(0, 64, 0);
        resumes_button.BackColor = Color.Green;
        MyInvitesForm mi = new MyInvitesForm();
        mi.TopLevel = false;
        mi.FormBorderStyle = FormBorderStyle.None;
        mi.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(mi);
        mi.Show();
    }
}

```



```

        catch (Exception ex)
        {
            DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        }
    }

```

```

private void favresumes_button_Click(object sender, EventArgs e)
{
    try
    {
        percab_button.BackColor = Color.Green;
        favresumes_button.BackColor = Color.FromArgb(0, 64, 0);
        myotklike_button.BackColor = Color.Green;
        resumes_button.BackColor = Color.Green;
        FavResumesForm fr = new FavResumesForm();
        fr.TopLevel = false;
        fr.FormBorderStyle = FormBorderStyle.None;
        fr.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(fr);
        fr.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
    }
}

```

```

private void RabotodatelForm_Load(object sender, EventArgs e)
{
    try
    {
        percab_button.BackColor = Color.FromArgb(0, 64, 0);
        favresumes_button.BackColor = Color.Green;
    }
}

```

```

        myotkliki_button.BackColor = Color.Green;
        resumes_button.BackColor = Color.Green;
        PerCabRabForm pcr = new PerCabRabForm();
        pcr.TopLevel = false;
        pcr.FormBorderStyle = FormBorderStyle.None;
        pcr.Dock = DockStyle.Fill;
        panel1.Controls.Clear();
        panel1.Controls.Add(pcr);
        pcr.Show();
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```

private void pictureBox_closeapp_MouseEnter(object sender, EventArgs e)
{
    pictureBox_closeapp.BackColor = Color.FromArgb(0, 192, 0);
}

```

```

private void pictureBox_closeapp_MouseLeave(object sender, EventArgs e)
{
    pictureBox_closeapp.BackColor = Color.Green;
}

```

```

private void pictureBox_logout_MouseEnter(object sender, EventArgs e)
{
    pictureBox_logout.BackColor = Color.FromArgb(0, 192, 0);
}

```

```

private void pictureBox_logout_MouseLeave(object sender, EventArgs e)
{
    pictureBox_logout.BackColor = Color.Green;
}

```



## RegForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class RegForm : Form
    {
        public RegForm()
        {
            InitializeComponent();

            private void button_reg_Click(object sender, EventArgs e)
            {
                try
                {
                    string fam = textBox_fam.Text;
                    string im = textBox_im.Text;
                    string ot = textBox_ot.Text;
                    string Date = date.Value.ToShortDateString();
                    string nom = maskedTextBox_nom.Text;
                    string login = textBox_login.Text;
                    string password = textBox_password.Text;
                    string role = comboBox_role.SelectedItem.ToString();

                    if (textBox_fam.Text != "" && textBox_im.Text != "" && maskedTextBox_nom.Text != "" &&
                        textBox_login.Text != "" && textBox_password.Text != "")
```

```

        {
            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            string query = $"insert into Пользователи(Логин, Пароль, Роль, Фамилия, Имя, Отчество,
ДатаРождения, НомерТелефона) values('{login}', '{password}', '{role}', '{fam}', '{im}', '{ot}', '{Date}', '{nom}')";

            con.Open();

            SqlCommand com = new SqlCommand(query, con);

            com.ExecuteNonQuery();

            DialogResult res = MessageBox.Show("Вы успешно зарегистрированы", "Сообщение",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        else
        {
            DialogResult res = MessageBox.Show("Пожалуйста, заполните необходимые поля",
"Сообщение", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
    }
}
}

```

## ResumesForm.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Reflection;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace НайтиРаботу
{
    public partial class ResumesForm : Form
    {
        int userId = AuthForm.UserId;

        public ResumesForm()
        {
            InitializeComponent();

            toolTip_addfav.SetToolTip(pictureBox_addfav, "Добавить в избранное");
        }

        private void button_otklik_Click(object sender, EventArgs e)
        {
            try
            {
                if (ОтображениеДанных.Rows.Count > 0)
                {
                    DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

                    int id = Convert.ToInt32(selectedRow.Cells["КодРезюме"].Value);

                    string query = $"insert into Приглашения(КодПользователя, КодРезюме) values('{userId}', '{id}')";
                    select КодРезюме, Фамилия, Имя, Отчество, Наименование, ЖелаемаяДолжность, Зарплата,
                    Образование, ЭлПочта, ГрафикРаботы from Пользователи inner join Резюме on
                    Пользователи.КодПользователя = Резюме.КодПользователя";

                    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);
```

```

        SqlCommand com = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

private void pictureBox_addfav_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

            int id = Convert.ToInt32(selectedRow.Cells["КодРезюме"].Value);

            string query = $"insert into ИзбранныеРезюме(КодПользователя, КодРезюме)
values('{userId}', '{id}')"; select КодРезюме, Фамилия, Имя, Отчество, Наименование, ЖелаемаяДолжность,
Зарплата, Образование, ЭлПочта, ГрафикРаботы from Пользователи inner join Резюме on
Пользователи.КодПользователя = Резюме.КодПользователя";

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            SqlCommand com = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(com);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```
    }  
}
```

```
private void comboBox_sort_SelectedIndexChanged(object sender, EventArgs e)  
{  
    try  
    {  
        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);  
        if (comboBox_sort.SelectedIndex == 0)  
        {  
            string query = $"exec SortResumeBySalary";  
            SqlCommand command = new SqlCommand(query, con);  
            SqlDataAdapter adapter = new SqlDataAdapter(command);  
            DataTable dt = new DataTable();  
            adapter.Fill(dt);  
            ОтображениеДанных.DataSource = dt;  
        }  
        else if (comboBox_sort.SelectedIndex == 1)  
        {  
            string query = $"exec SortResumeByEducation";  
            SqlCommand command = new SqlCommand(query, con);  
            SqlDataAdapter adapter = new SqlDataAdapter(command);  
            DataTable dt = new DataTable();  
            adapter.Fill(dt);  
            ОтображениеДанных.DataSource = dt;  
        }  
        else if (comboBox_sort.SelectedIndex == 2)  
        {  
            string query = $"exec SortResumeByGraph";  
            SqlCommand command = new SqlCommand(query, con);  
            SqlDataAdapter adapter = new SqlDataAdapter(command);  
            DataTable dt = new DataTable();  
            adapter.Fill(dt);  
            ОтображениеДанных.DataSource = dt;  
        }  
    }  
}
```



```

        catch (Exception ex)
        {
            DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
        }
    }

    private void comboBox_filteredu_SelectedIndexChanged(object sender, EventArgs e)
    {
        try
        {
            string index = comboBox_filteredu.SelectedItem.ToString();

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            string query = $"select КодРезюме, Фамилия, Имя, Отчество, Наименование,
ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Пользователи inner join Резюме
on Пользователи.КодПользователя = Резюме.КодПользователя where Образование = '{index}'";

            SqlCommand command = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(command);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }
        catch (Exception ex)
        {
            DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
        }
    }

    private void comboBox_filtergraph_SelectedIndexChanged(object sender, EventArgs e)
    {
        try
        {
            string index = comboBox_filtergraph.SelectedItem.ToString();

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            string query = $"select КодРезюме, Фамилия, Имя, Отчество, Наименование,
ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Пользователи inner join Резюме
on Пользователи.КодПользователя = Резюме.КодПользователя where ГрафикРаботы = '{index}'";

```

```

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void ResumesForm_Load(object sender, EventArgs e)
{
    try
    {
        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        string query = $"select КодРезюме, Фамилия, Имя, Отчество, Наименование,
ЖелаемаяДолжность, Зарплата, Образование, ЭлПочта, ГрафикРаботы from Пользователи inner join Резюме
on Пользователи.КодПользователя = Резюме.КодПользователя ";

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

private void pictureBox_addfav_MouseEnter(object sender, EventArgs e)
{
    pictureBox_addfav.BackColor = Color.Silver;
}

```

```
}
```

```
private void pictureBox_addfav_MouseLeave(object sender, EventArgs e)
```

```
{
```

```
    pictureBox_addfav.BackColor = Color.White;
```

```
}
```

```
}
```

```
}
```

## VacanciesForm.cs

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace НайтиРаботу

{

    public partial class VacanciesForm : Form

    {

        int userId = AuthForm.UserId;

        public VacanciesForm()

        {

            InitializeComponent();

            toolTip_addfav.SetToolTip(pictureBox_addfav, "Добавить в избранное");

        }

        private void pictureBox_addfav_Click(object sender, EventArgs e)

        {

            try

            {

                if (ОтображениеДанных.Rows.Count > 0)

                {

                    DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

                    int id = Convert.ToInt32(selectedRow.Cells["КодВакансии"].Value);

                    string query = $"insert into ИзбранныеВакансии(КодПользователя, КодВакансии)

values('{userId}', '{id}')"; select КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация,

ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии";

                    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

                    SqlCommand com = new SqlCommand(query, con);
```

```

        SqlDataAdapter adapter = new SqlDataAdapter(com);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
}

catch (Exception ex)
{
    DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

private void button_otklik_Click(object sender, EventArgs e)
{
    try
    {
        if (ОтображениеДанных.Rows.Count > 0)
        {
            DataGridViewRow selectedRow = ОтображениеДанных.SelectedRows[0];

            int id = Convert.ToInt32(selectedRow.Cells["КодВакансии"].Value);

            string query = $"insert into Отклики(КодПользователя, КодВакансии) values('{userId}',
            '{id}')"; select КодВакансии, Наименование, Город, Студентам, ОпытРаботы, Специализация, ГрафикРаботы,
            Отрасль, Зарплата, Образование from Вакансии";

            SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

            SqlCommand com = new SqlCommand(query, con);

            SqlDataAdapter adapter = new SqlDataAdapter(com);

            DataTable dt = new DataTable();

            adapter.Fill(dt);

            ОтображениеДанных.DataSource = dt;
        }
    }

    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}

```

```
}
```

```
private void VacanciesForm_Load(object sender, EventArgs e)
{
    try
    {
        SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

        string query = "select КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
        Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии";

        SqlCommand cmd = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(cmd);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
    catch (Exception ex)
    {
        DialogResult res = MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
    }
}
```

```
private void comboBox_sort_SelectedIndexChanged(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    if (comboBox_sort.SelectedIndex == 0)
    {
        string query = "exec SortBySalary";

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
    else if (comboBox_sort.SelectedIndex == 1)
    {

```

```

        string query = "exec SortByExp";
        SqlCommand command = new SqlCommand(query, con);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        ОтображениеДанных.DataSource = dt;
    }
    else if (comboBox_sort.SelectedIndex == 2)
    {
        string query = "exec SortByGraph";
        SqlCommand command = new SqlCommand(query, con);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        ОтображениеДанных.DataSource = dt;
    }
    else if (comboBox_sort.SelectedIndex == 3)
    {
        string query = "exec SortByOtrasl";
        SqlCommand command = new SqlCommand(query, con);
        SqlDataAdapter adapter = new SqlDataAdapter(command);
        DataTable dt = new DataTable();
        adapter.Fill(dt);
        ОтображениеДанных.DataSource = dt;
    }
}

private void comboBox_filterexp_SelectedIndexChanged(object sender, EventArgs e)
{
    string index = comboBox_filterexp.SelectedItem.ToString();
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"select КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
    Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии where ОпытРаботы =
    '{index}'";

    SqlCommand command = new SqlCommand(query, con);
    SqlDataAdapter adapter = new SqlDataAdapter(command);

```

```

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }

private void comboBox_filterotrasli_SelectedIndexChanged(object sender, EventArgs e)
{
    string index = comboBox_filterotrasli.SelectedItem.ToString();

    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"select КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
    Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии where Отрасль = '{index}'";

    SqlCommand command = new SqlCommand(query, con);

    SqlDataAdapter adapter = new SqlDataAdapter(command);

    DataTable dt = new DataTable();

    adapter.Fill(dt);

    ОтображениеДанных.DataSource = dt;
}

private void checkBox_stud_CheckedChanged(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    if (checkBox_stud.Checked == true)
    {
        string query = "exec Student";

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);

        DataTable dt = new DataTable();

        adapter.Fill(dt);

        ОтображениеДанных.DataSource = dt;
    }
    else
    {
        string query = "select КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
        Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии";

        SqlCommand command = new SqlCommand(query, con);

        SqlDataAdapter adapter = new SqlDataAdapter(command);
    }
}

```



```

        DataTable dt = new DataTable();
        adapter.Fill(dt);
        ОтображениеДанных.DataSource = dt;
    }
}

private void comboBox_filtergraph_SelectedIndexChanged(object sender, EventArgs e)
{
    string index = comboBox_filtergraph.SelectedItem.ToString();
    SqlConnection con = new SqlConnection(DataBaseConfig.ConnectionString);

    string query = $"select КодВакансии, Наименование, Город, Студентам, ОпытРаботы,
    Специализация, ГрафикРаботы, Отрасль, Зарплата, Образование from Вакансии where ГрафикРаботы =
    '{index}'";

    SqlCommand command = new SqlCommand(query, con);
    SqlDataAdapter adapter = new SqlDataAdapter(command);
    DataTable dt = new DataTable();
    adapter.Fill(dt);
    ОтображениеДанных.DataSource = dt;
}

private void pictureBox_addfav_MouseEnter(object sender, EventArgs e)
{
    pictureBox_addfav.BackColor = Color.Silver;
}

private void pictureBox_addfav_MouseLeave(object sender, EventArgs e)
{
    pictureBox_addfav.BackColor = Color.White;
}
}

```

### **Приложение 3. Руководство оператора**

Функциональным назначением программы является учет и подбор вакансий.

Программа должна обеспечивать возможность выполнения нижеперечисленных функций:

- Добавление вакансии
- Добавление резюме
- Изменение вакансии
- Изменение резюме
- Удаление вакансии
- Удаление резюме
- Возможность отклика на вакансию
- Возможность отклика на резюме
- Добавление вакансии в избранное
- Добавление резюме в избранное
- Удаление вакансии из избранного
- Удаление резюме из избранного

Климатические условия эксплуатации программного обеспечения должны удовлетворять требования к условиям эксплуатации, предъявляемым к техническим средствам в части условий их эксплуатации.

В состав технических средств персонального компьютера должен входить:

- a) Процессор с тактовой частотой не менее 1 ГГц;
- b) Оперативная память не менее 1 Гб;
- c) Клавиатура;
- d) Мышь;
- e) Устройство для хранения данных не менее 200 Мб свободного пространства;
- f) Монитор с разрешением экрана не менее 1280x768;

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7/10/11, Microsoft SQL Server 2014.

Конечный пользователь должен обладать навыками работы с графическим пользовательским интерфейсом операционной системы.

## Выполнение программы

Для запуска программного продукта необходимо запустить исполняемый файл «НайтиРаботу» с расширением .exe. В случае успешного запуска программы, пользователю будет представлено окно авторизации и регистрации (Рисунок 11).

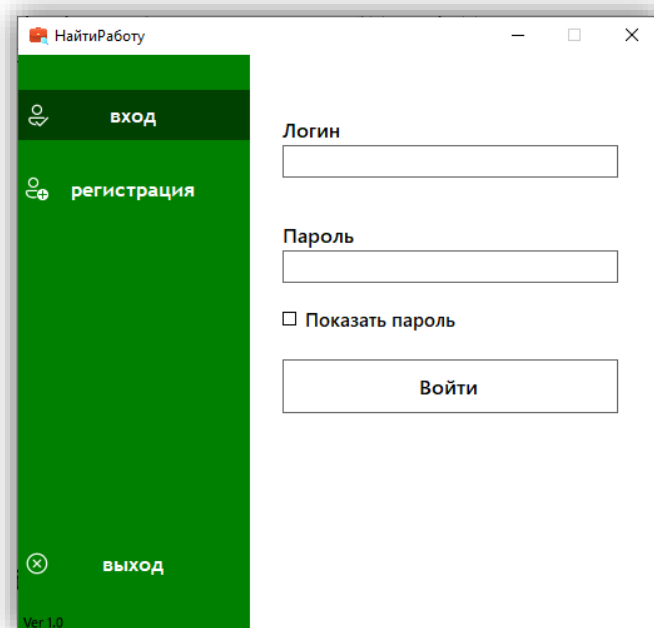


Рисунок 11 - Окно авторизации и регистрации

## Работодатель

Для добавления вакансии необходимо перейти на вкладку «Личный кабинет» и нажать на зелёный плюсики (Рисунок 12). Откроется окно заполнения данных, где необходимо ввести необходимые данные (Рисунок 13).

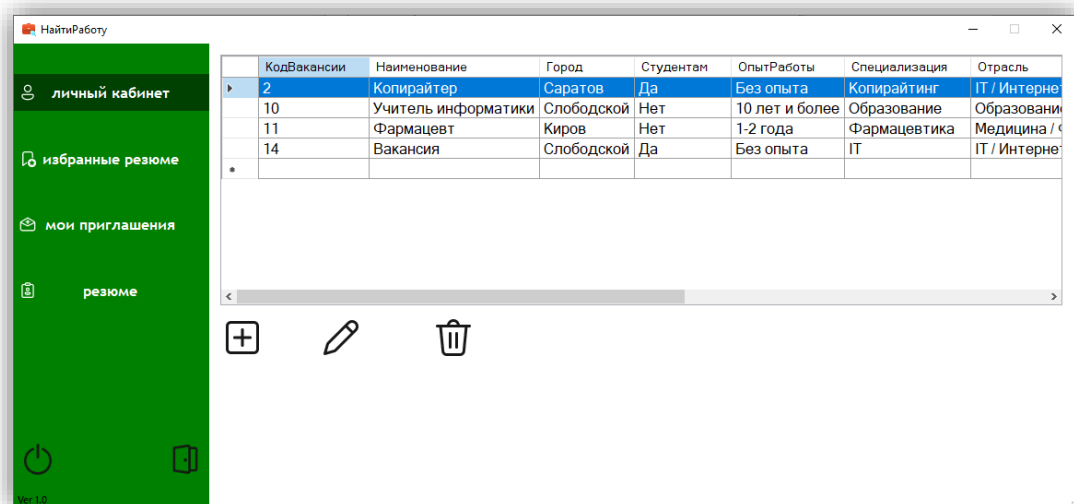


Рисунок 12 - Вкладка «Личный кабинет»

Название вакансии

Город

Студентам ☐

Опыт работы

Специализация

График работы

Зарплата

Образование

Отрасль

✕ ✓

Рисунок 13 - Окно заполнения данных

Для изменения вакансии нажмите на значок редактирования на вкладке «Личный кабинет» (Рисунок 12). После этого откроется окно редактирования вакансии, на которой вы сможете изменить данные вакансии (Рисунок 14).

КодВакансии	Наименование	Город	Студентам	ОпытРаботы	Специализация	Отрасль
2	Копирайтер	Саратов	Да	Без опыта	Копирайтинг	IT / Интернет /
10	Учитель информатики	Слободской	Нет	10 лет и более	Образование	Образование /
11	Фармацевт	Киров	Нет	1-2 года	Фармацевтика	Медицина / Фар
14	Вакансия	Слободской	Да	Без опыта	IT	IT / Интернет /

Наименование:

Город:

Студентам: ☐

Опыт работы:

Специализация:

График работы:

Отрасль:

Зарплата:

Образование:

✕ ✓

Рисунок 14 - Окно изменения вакансии

Для удаления вакансии необходимо нажать на значок удаления на вкладке «Личный кабинет», предварительно выбрав вакансию в списке выше (Рисунок 12).

На вкладке «Избранные резюме» отображаются понравившиеся пользователю резюме (Рисунок 15). Также пользователь может отправлять приглашения по избранным резюме, сортировать и фильтровать их или удалять из избранного.

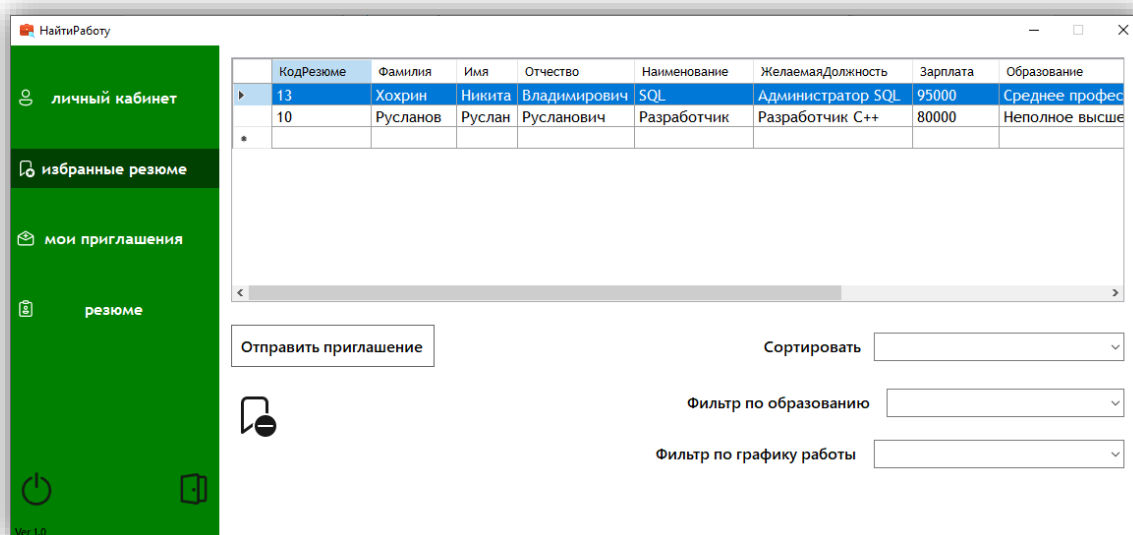


Рисунок 15 - Вкладка «Избранные резюме»

На вкладке «Мои приглашения» отображаются резюме, на которые пользователь отправил приглашение (Рисунок 16). Пользователь также может осуществлять поиск резюме, либо же отменить приглашение.

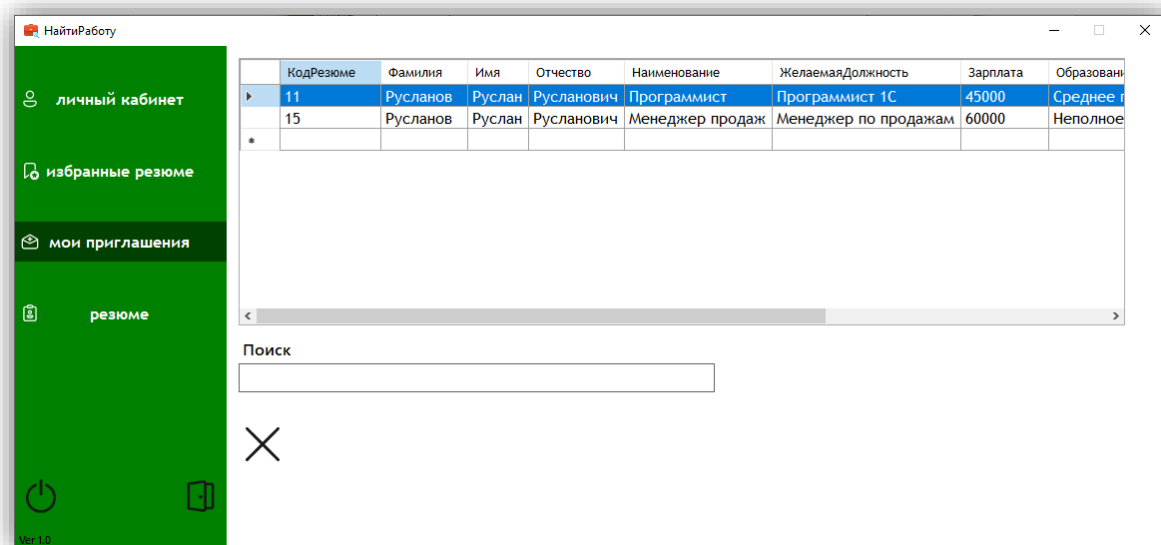


Рисунок 16 - Вкладка «Мои приглашения»

На вкладке «Резюме» пользователю отображаются доступные резюме, на которые он может отправить приглашение или добавить резюме в избранное (Рисунок 17). Также есть возможность сортировки и фильтрации резюме.

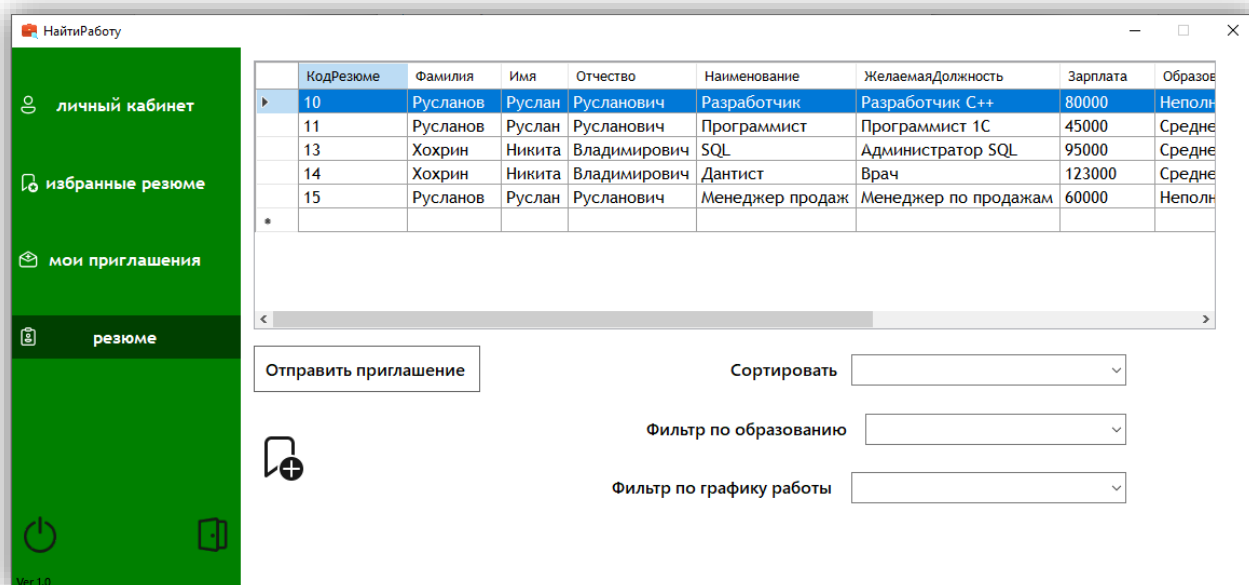


Рисунок 17 - Вкладка «Резюме»



## Соискатель

На вкладке «Личный кабинет» отображаются активные резюме пользователя (Рисунок 18). Чтобы добавить резюме, необходимо нажать на кнопку добавления (*Зелёный плюсик*), после чего откроется окно добавления резюме (Рисунок 19). В данном окне необходимо ввести данные в соответствующие поля и нажать кнопку подтверждения.

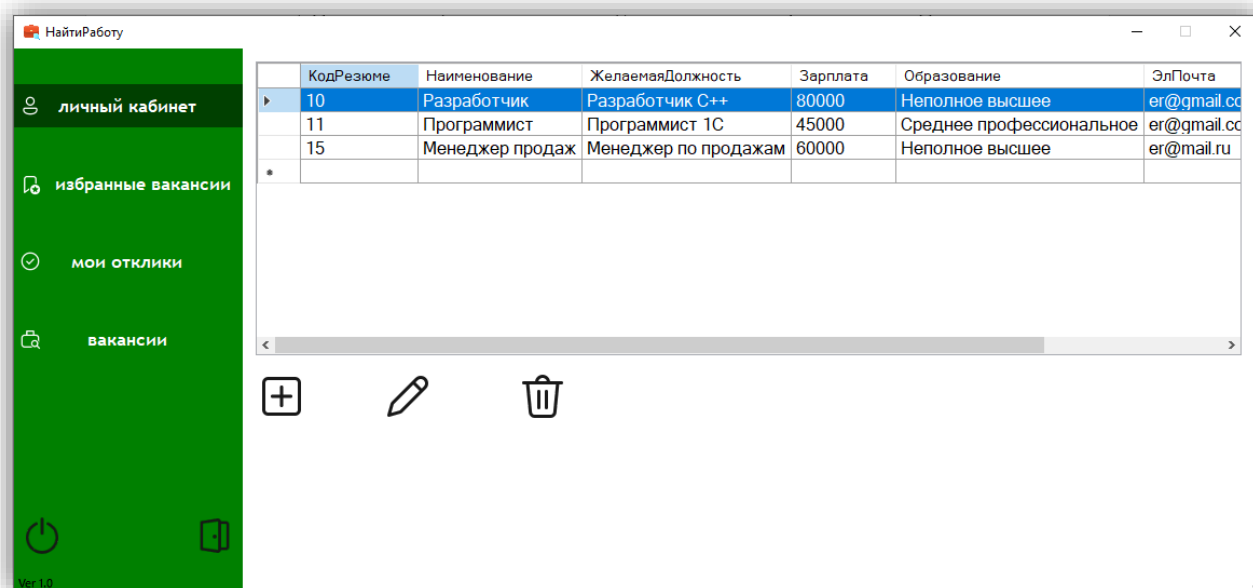


Рисунок 18 - Вкладка «Личный кабинет»

The image shows a software window titled 'Окно добавления резюме' (Add Resume Window). It contains six input fields for user data: 'Название резюме' (Resume Name), 'Эл. почта' (Email), 'Желаемая должность' (Desired Position), 'Зарплата' (Salary), 'Образование' (Education), and 'График работы' (Work Schedule). The last two fields are dropdown menus. At the bottom right, there are two buttons: a cancel button with an 'X' icon and a confirm button with a checkmark icon. The window has a standard title bar with a close button in the top right corner.

Рисунок 19 - Окно добавления резюме

Для изменения существующих резюме необходимо нажать на кнопку редактирования (Рисунок 18). В открывшемся окне необходимо ввести новые данные, предварительно выбрав резюме в списке (Рисунок 20).

КодРезюме	Наименование	ЖелаемаяДолжность	Зарплата	Образование	ЭлПочта
10	Разработчик	Разработчик C++	80000	Неполное высшее	er@gmail.com
11	Программист	Программист 1С	45000	Среднее профессиональное	er@gmail.com
15	Менеджер продаж	Менеджер по продажам	60000	Неполное высшее	er@mail.ru
*					

Название резюме:

Эл. почта:

Желаемая должность:

Зарплата:

Образование:

График работы:

✕ ✓

Рисунок 20 - Окно редактирования резюме

Для удаления резюме необходимо нажать на кнопку удаления (Рисунок 18).

На вкладке «Избранные вакансии» отображаются понравившиеся пользователю вакансии (Рисунок 21), на которые он может откликнуться или убрать из избранного. Также есть возможность сортировки и фильтрации вакансий.

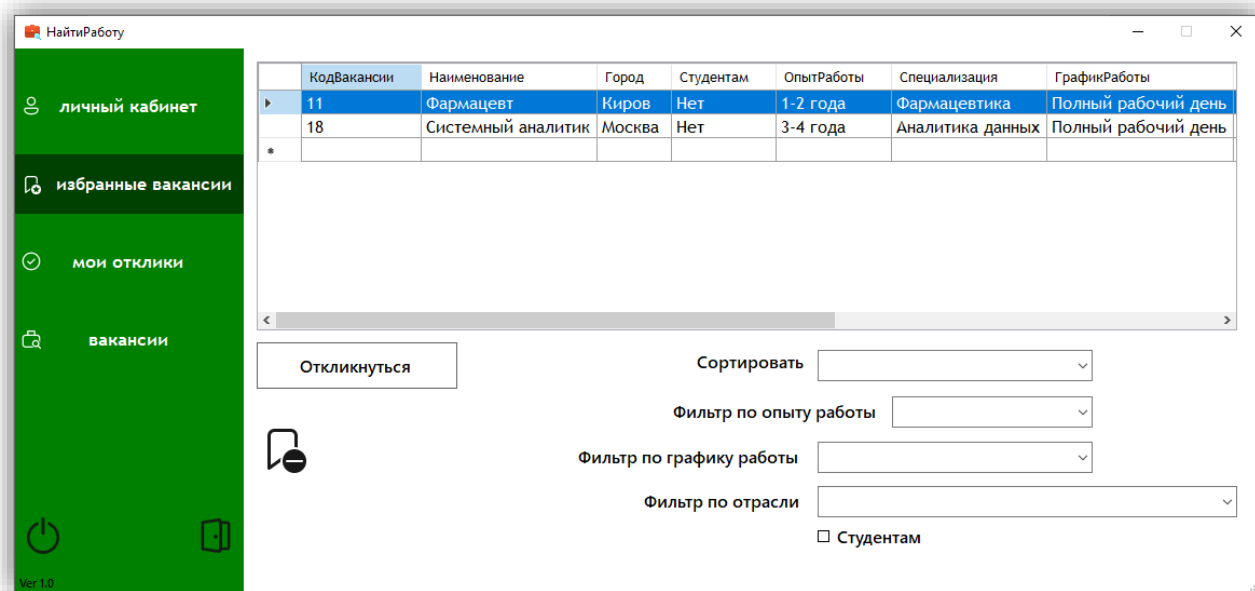


Рисунок 21 - Вкладка «Избранные вакансии»

На вкладке «Мои отклики» отображаются вакансии, на которые откликнулся пользователь (Рисунок 22). Также есть возможность отмены отклика на вакансию и поиска вакансий.

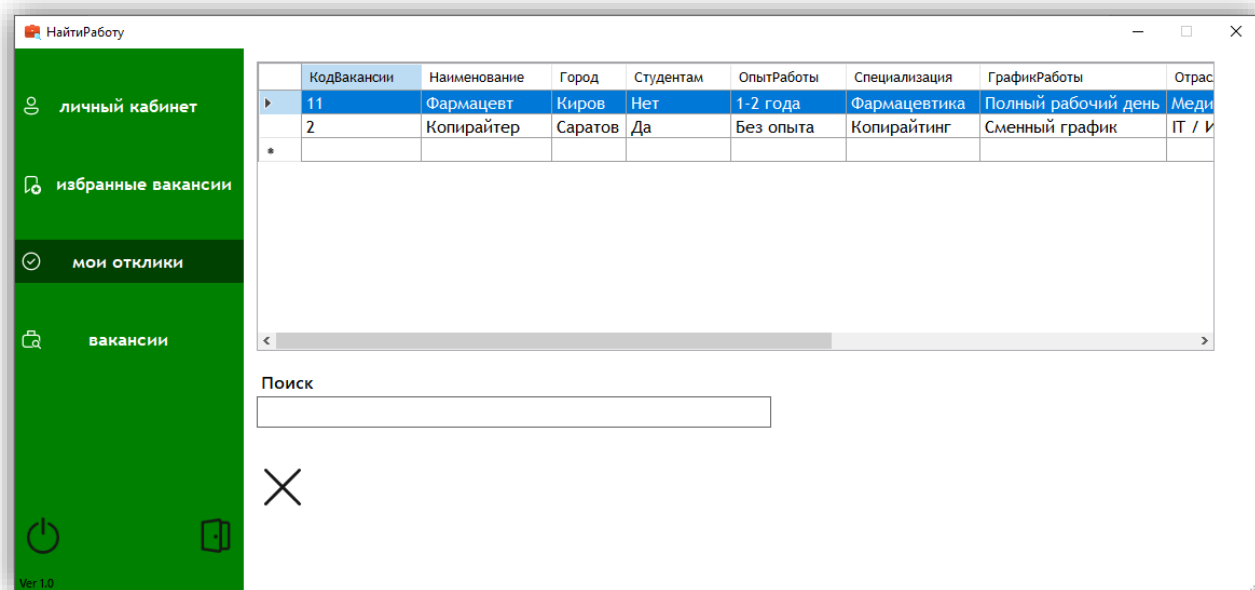


Рисунок 22 - Вкладка «Мои отклики»

На вкладке «Вакансии» отображаются доступные вакансии, на которые пользователь может откликнуться или добавить в избранное (Рисунок 23). Также есть возможность сортировки и фильтрации вакансий.

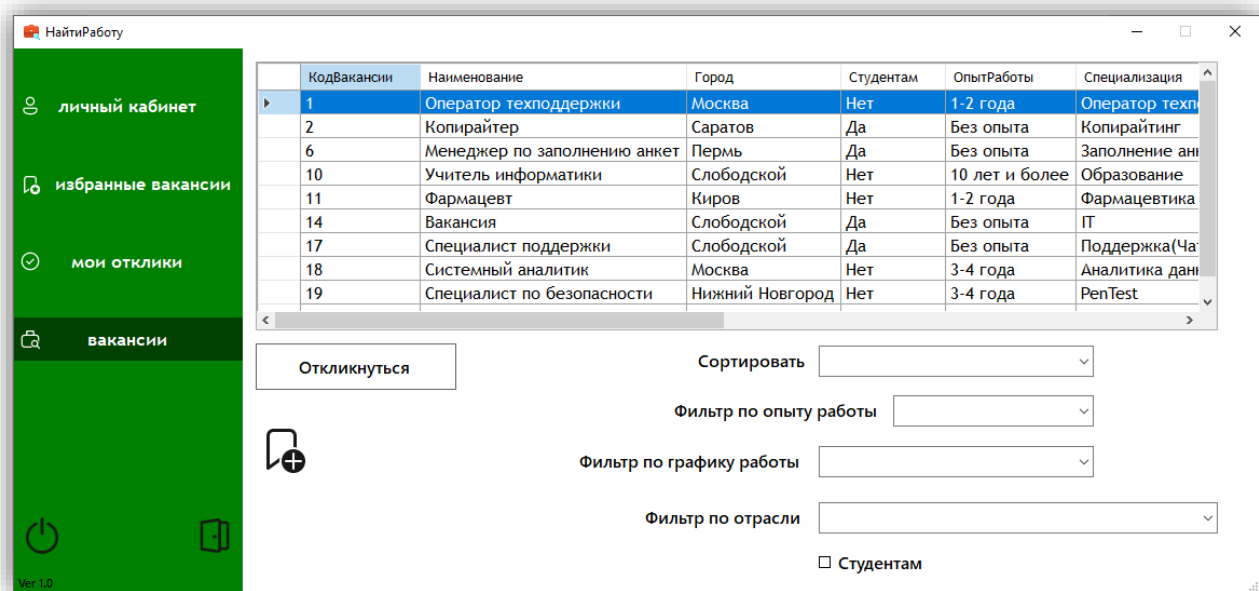


Рисунок 23 - Вкладка «Вакансии»