Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное  
образовательное учреждение высшего образования  
«Уфимский государственный авиационный технический университет»

Кафедра ВМиК

Отчёт по лабораторной работе № 4-5

Тема: «Проектирование БД. Проектирование пользовательского интерфейса»

Выполнили:

студенты группы ПРО-330 Т.А. Ишембитов

Д.А. Ишембитов

Проверил:

канд. наук Р.В. Насыров

Уфа – 2022

**Содержание**

[1 Задание 3](#_Toc121258774)

[2 ER диаграмма спроектированной БД 5](#_Toc121258775)

[3 Листинг SQL-скрипта БД 5](#_Toc121258776)

[4 Листинг основных функций для работы с БД 7](#_Toc121258777)

[5 Основные таблицы построенной БД 9](#_Toc121258778)

[6 Основные html страницы сайта 9](#_Toc121258779)

[7 Вывод 13](#_Toc121258780)

[8 Приложения 13](#_Toc121258781)

# Задание

Цель работы:

№4. Ознакомление с основными методами проектирования базы данных (БД).

№5. Ознакомление с методами проектирования пользовательского интерфейса

**Задание:**

**Проектирование БД**

1. На основе материалов по предмету «Базы данных» спроектировать БД приложения для хранения обработанных данных.
2. БД должна содержать таблицу с информацией об обработанных файлах (минимальный набор колонок: название файла, дата обработки).
3. Таблица обработанных файлов должна ограничивать основную таблицу с данными по внешнему ключу.
4. Полученные после обработки классы или категории должны быть также вынесены в отдельную таблицу с ограничением по внешнему ключу. Т.е. должна быть проведена нормализация данных.
5. Написать SQL-скрипт для создания структуры спроектированной БД.
6. Изучить код примера из архива lab4\_example\_app\_python.zip.На основе данного примера реализовать необходимые CRUD (Create, Read, Update, Delete) операции для работы с БД. Загрузить код приложения на GitHub.
7. Проверить выполнение CRUD-операций на тестовой БД (только SQLite).

**Проектирование пользовательского интерфейса.**

Во время выполнения лабораторной работы необходимо описать ожидаемое поведение разрабатываемой системы с точки зрения внешнего по отношению к ней пользователя, то есть осуществить "конструирование" внешних взаимодействий будущей ИС с пользователем без конкретизации его внутреннего устройства.

1. Определить структуру проектируемого пользовательского интерфейса (визуальное оформление, отвечающее за представление информации пользователю; функциональные возможности системы, включающие набор возможностей для эффективного выполнения профессиональной деятельности; техники взаимодействия пользователя с системой + дополнительные функциональные возможности системы) с учетом задачи (лекции, рекомендации Приложение 1, дополнительные функциональные возможности системы (пример) в Приложении 3).

2. Определить стили пользовательского интерфейса (графический (GUI, web-интерфейс (WUI), объектно-ориентированный интерфейс) с учетом задачи (лекции, рекомендации Приложение 1).

3. Определиться с размещением элементов пользовательского интерфейса (кнопки, иконки, выпадающие списки, поля для записи текста и пр.) (Лекции, рекомендации Приложение 1).

4. Написать Требования к интерфейсу пользователя (Пример в приложении 2).

5. Разработать взаимодействие разрабатываемой программы с пользователем: сценарий (можно диаграммой последовательности, диаграммой взаимодействия), экранные формы, набор подсказок (перечисление), и пр.

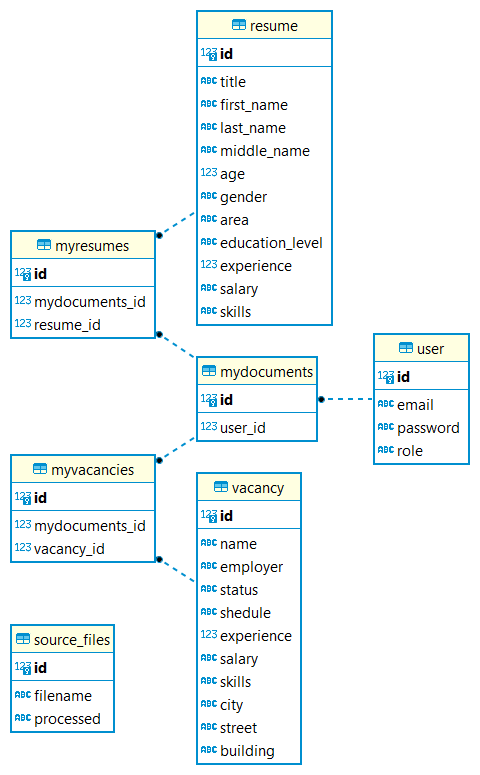
6. Ознакомиться с методическим материалом по базовой верстке вебстраниц.

7. На основе п.1-5 разработать (сверстать) основные html-страницы для вашего приложения, используя CSS-стили и HTML5-модель верстки. Загрузить код html-страниц на GitHub.

**Написать отчет.** Отчет должен содержать:

1. ER-диаграмму спроектированной БД.
2. Листинг SQL-скрипта для создания структуры БД.
3. Листинг основных функций для работы с БД (с пояснениями, что делает каждая функция).
4. Скриншот части содержимого основной таблицы данных и связанных таблиц.
5. Скриншоты основных html-страниц с пояснениями и обозначением разметки базовых HTML5-тегов (аналогично рис. 2.4).

## ER диаграмма спроектированной БД



## Листинг SQL-скрипта БД

create table source\_files (

id integer PRIMARY KEY autoincrement,

filename varchar(255) NOT NULL,

processed datetime

);

create table user (

id integer PRIMARY KEY autoincrement,

email varchar(255) NOT NULL,

password varchar(255) NOT NULL,

role varchar(255) NOT NULL

);

create table mydocuments (

id integer PRIMARY KEY autoincrement,

user\_id integer NOT NULL,

CONSTRAINT fk\_user

FOREIGN KEY (user\_id)

REFERENCES user (id)

ON DELETE CASCADE

);

create table resume (

id integer PRIMARY KEY autoincrement,

title varchar(255) NOT NULL,

first\_name varchar(255) NOT NULL,

last\_name varchar(255) NOT NULL,

middle\_name varchar(255) NOT NULL,

age integer NOT NULL,

gender varchar(255) NOT NULL,

area varchar(255) NOT NULL,

education\_level varchar(255) NOT NULL,

experience integer NOT NULL,

salary varchar(255) NOT NULL,

skills varchar(255) NOT NULL

);

create table vacancy (

id integer PRIMARY KEY autoincrement,

name varchar(255) NOT NULL,

employer varchar(255) NOT NULL,

status varchar(255) NOT NULL,

shedule varchar(255) NOT NULL,

experience integer NOT NULL,

salary varchar(255) NOT NULL,

skills varchar(255) NOT NULL,

city varchar(255) NOT NULL,

street varchar(255) NOT NULL,

building varchar(255) NOT NULL

);

create table myresumes (

id integer PRIMARY KEY autoincrement,

mydocuments\_id integer NOT NULL,

resume\_id integer NOT NULL,

CONSTRAINT fk\_mydocuments

FOREIGN KEY (mydocuments\_id)

REFERENCES mydocuments (id)

ON DELETE CASCADE,

CONSTRAINT fk\_resume

FOREIGN KEY (resume\_id)

REFERENCES resume (id)

ON DELETE CASCADE

);

create table myvacancies (

id integer PRIMARY KEY autoincrement,

mydocuments\_id integer NOT NULL,

vacancy\_id integer NOT NULL,

CONSTRAINT fk\_mydocuments

FOREIGN KEY (mydocuments\_id)

REFERENCES mydocuments (id)

ON DELETE CASCADE,

CONSTRAINT fk\_vacancy

FOREIGN KEY (vacancy\_id)

REFERENCES vacancy (id)

ON DELETE CASCADE

);

## Листинг основных функций для работы с БД

**Пример функции SELECT\*FROM для вывода выборки таблицы source\_file**

# Вывод списка обработанных файлов с сортировкой по дате

def select\_all\_from\_source\_files(connector: StoreConnector):

connector.start\_transaction() # начинаем выполнение запросов

query = f'SELECT \* FROM source\_files ORDER BY processed'

result = connector.execute(query).fetchall()

connector.end\_transaction() # завершаем выполнение запросов

return result

**Пример функции INSERT INTO для заполнения таблицы source\_file**

# Вставка в таблицу обработанных файлов

def insert\_into\_source\_files(connector: StoreConnector, filename: str):

now = datetime.now() # текущая дата и время

date\_time = now.strftime("%Y-%m-%d %H:%M:%S") # преобразуем в формат SQL

connector.start\_transaction()

query = f'INSERT INTO source\_files (filename, processed) VALUES (\'{filename}\', \'{date\_time}\')'

result = connector.execute(query)

connector.end\_transaction()

return result

**Пример функции INSERT INTO для заполнения таблицы vacancy данными из файла**

# Вставка строк из DataFrame в БД

def insert\_rows\_into\_vacancy\_data(connector: StoreConnector, dataframe: DataFrame):

rows = dataframe.to\_dict('records')

connector.start\_transaction()

for row in rows:

connector.execute(f'INSERT INTO vacancy (name, employer, status, shedule, experience, salary, skills, city, street, building) VALUES (\'{row["vacancy\_name"]}\', \'{row["employer\_name"]}\', \'{row["vacancy\_type"]}\', \'{row["vacancy\_schedule"]}\', \'{row["vacancy\_experience"]}\', \'{row["vacancy\_salary"]}\', \'{row["skill\_name"]}\', \'{row["city"]}\', \'{row["street"]}\', \'{row["building"]}\')')

connector.end\_transaction()

**Пример функции DELETE FROM для удаления строк из таблицы source\_file**

# Удаление строк

def delete\_from\_source\_files(connector: StoreConnector):

connector.start\_transaction()

query = f'DELETE FROM source\_files WHERE id<100'

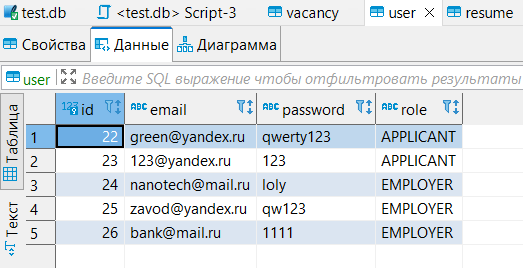
result = connector.execute(query)

connector.end\_transaction()

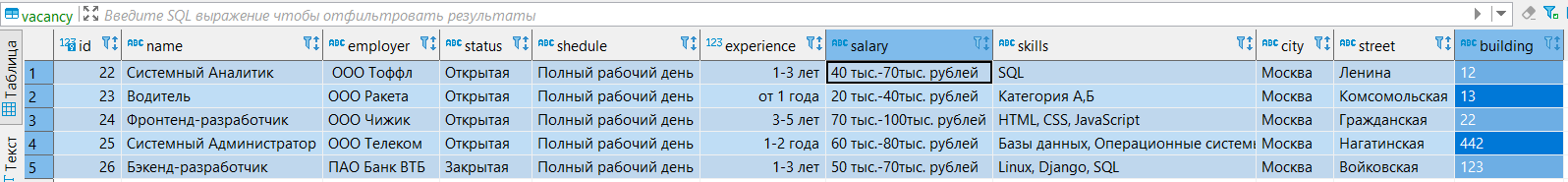
return result

## Основные таблицы построенной БД

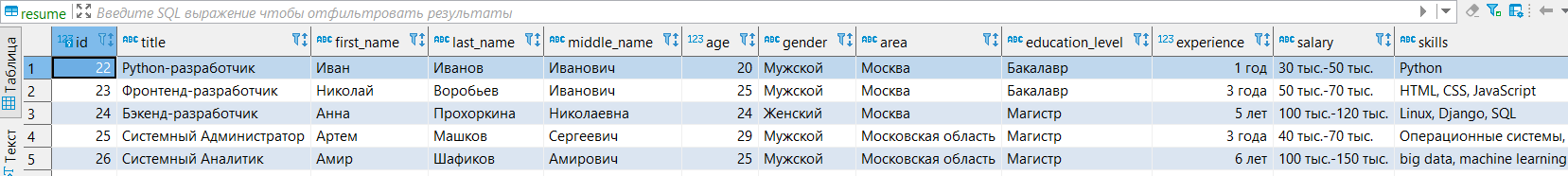
**Таблица user**

****

**Таблица vacancy**

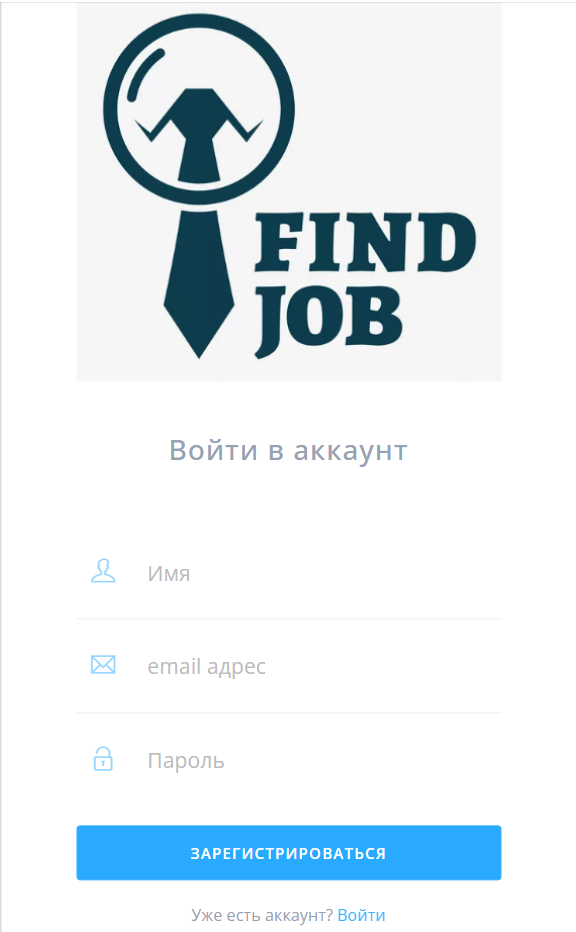
****

**Таблица resume**

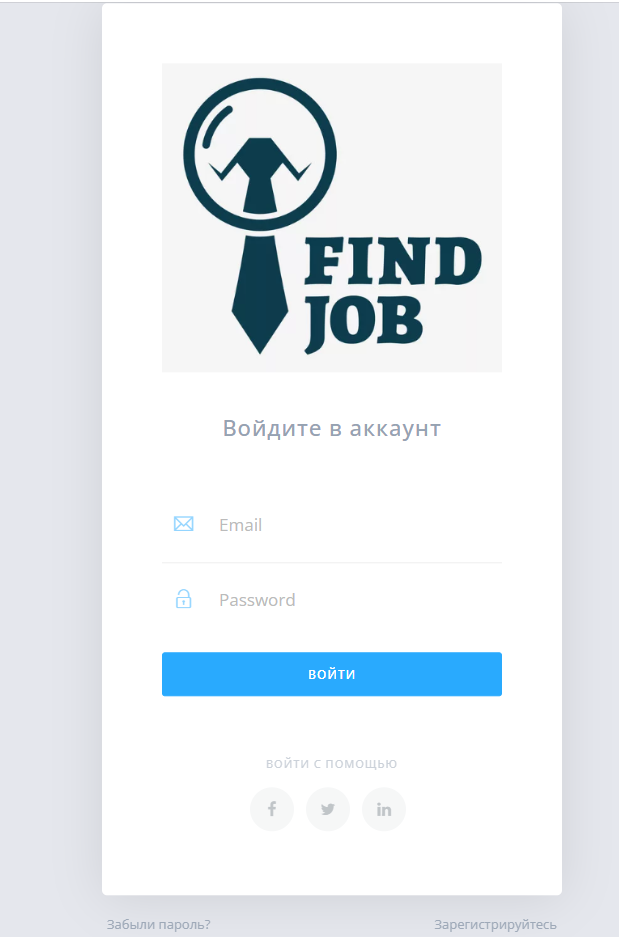
****

## Основные html страницы сайта

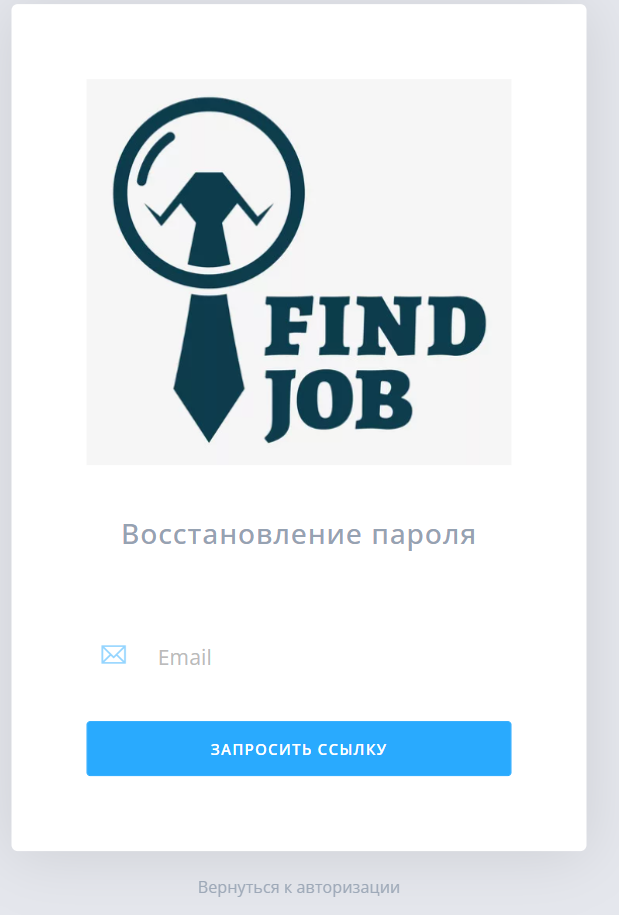
* 1. **Окно регистрации**

****

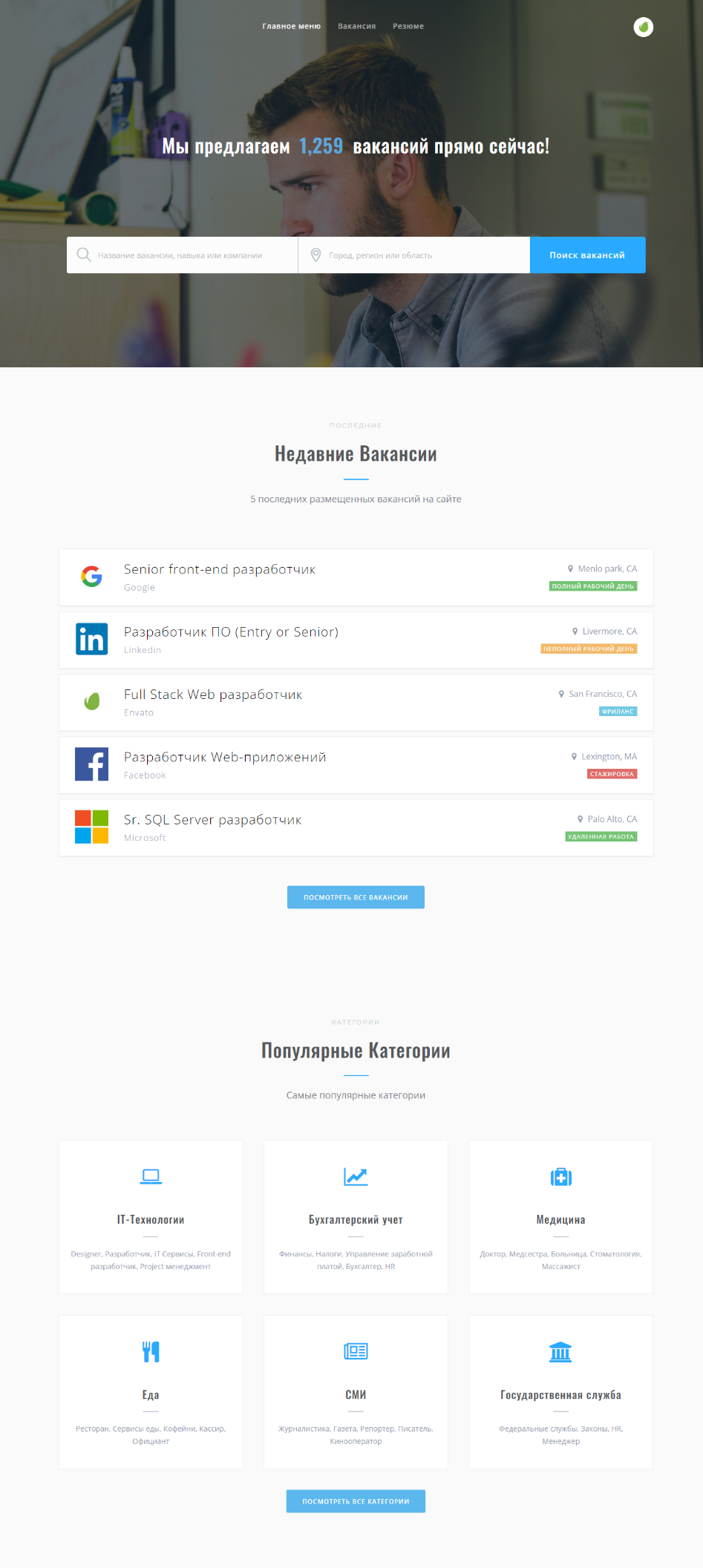
* 1. **Окно авторизации**

****

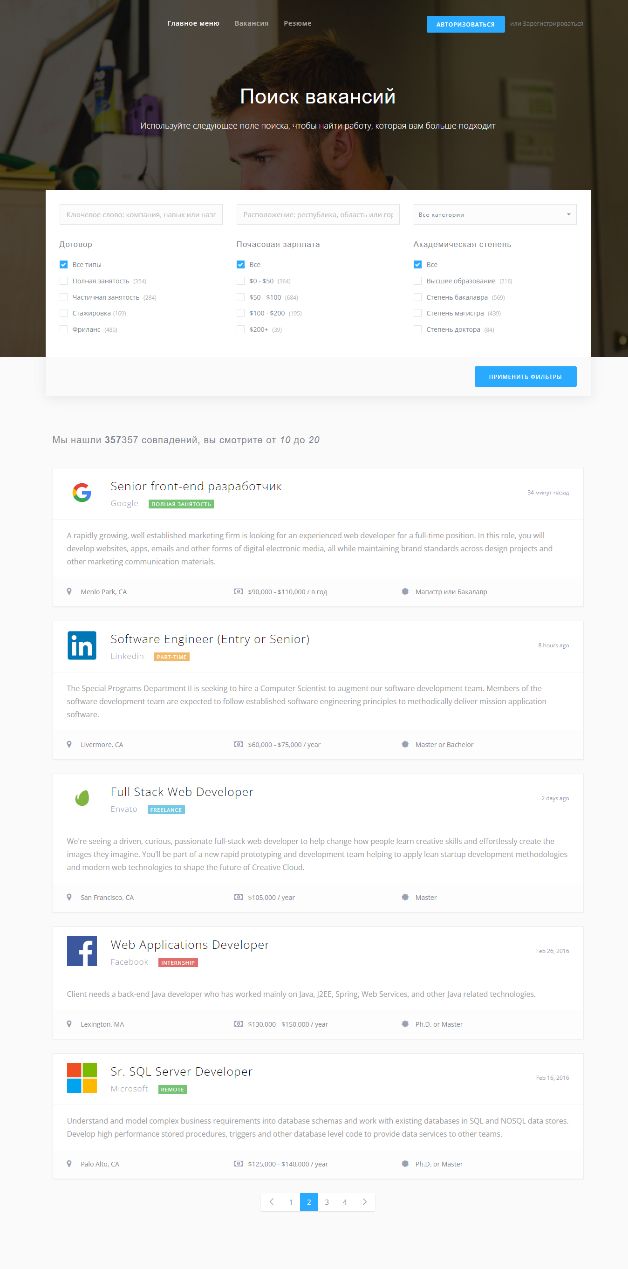
* 1. **Окно восстановления пароля**

****

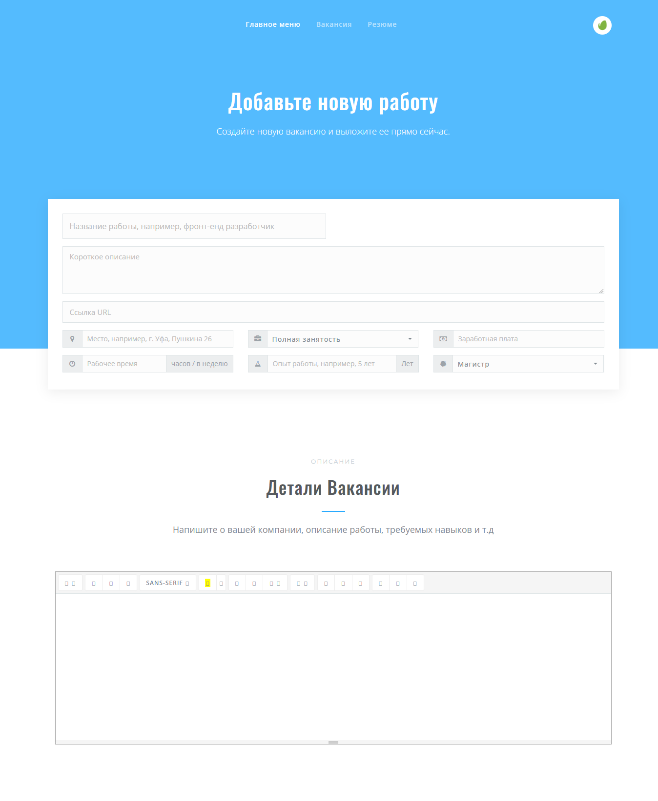
* 1. **Главное меню**



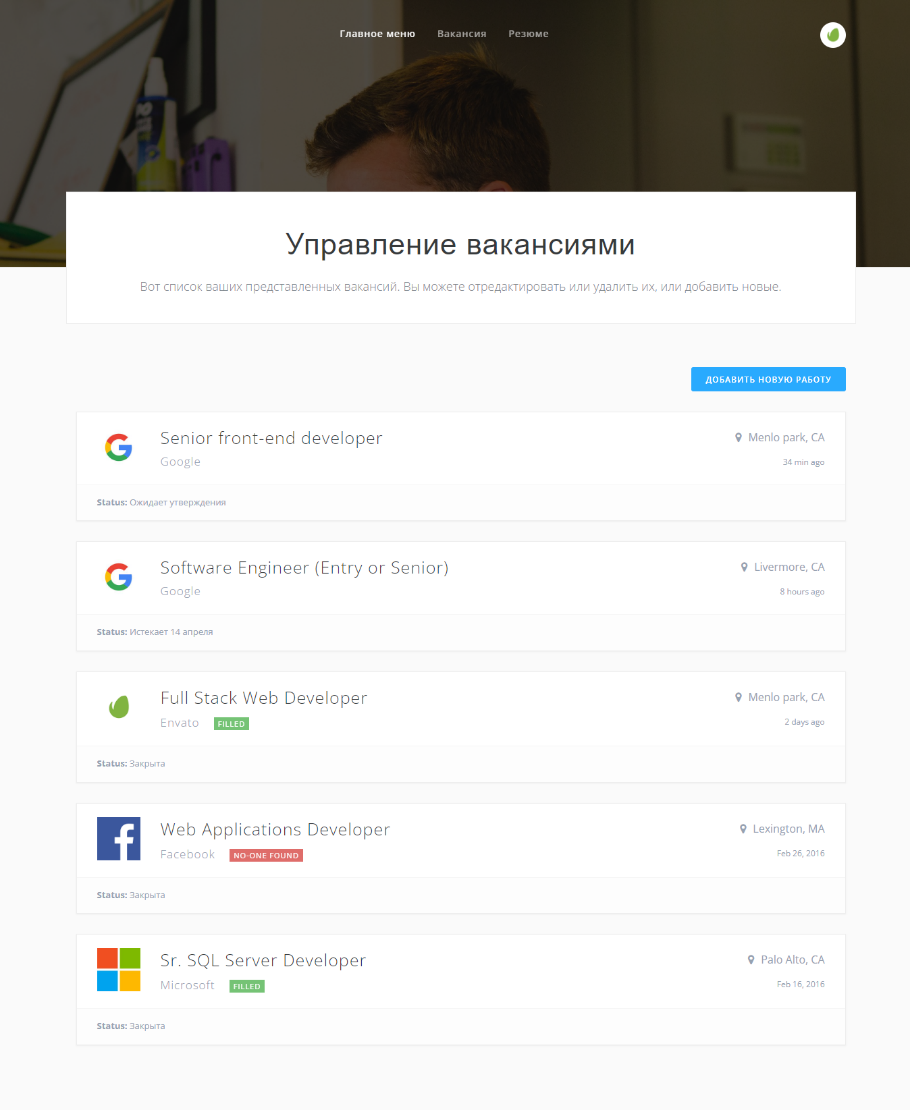
* 1. **Окно поиска вакансий**



* 1. **Окно создания вакансии**



* 1. **Окно управления вакансией**



## Вывод

В ходе лабораторной работы построили базу данных, а также создали основные страницы пользовательского интерфейса.

## Приложения

1. Пример Технического задания [Электронный ресурс] // URL: [170112 Project Record ТЗ ЭЗ в.0.5.pdf - Google Диск](https://drive.google.com/file/d/1oM_kAuUhzq27zA7s4fckEY5oTUMf7eOA/view).
2. Репозиторий GitHub // URL: <https://github.com/Phonker18/PiKPO_Project.git>