

# **Отчет по проекту**

## **“Моделирование базы данных и сайта, для работы с ней”**

Подготовила: студентка группы ВВО-19  
Редькина Я.Б.

### **1. Постановка задачи**

Разработать модель “база данных + сайт”, реализующую иллюстрированный именной указатель некоторого множества объектов (Фамилии), имена которых встречаются в нескольких изображениях (отсканированных страницах). Организовать возможность ввода новых объектов и изображений.

### **2. Используемые средства**

HTML, CSS, JavaScript, Python 3, Flask, SQLite3.

### **3. Описание БД**

*Сущности:* люди (таблица Persons), файлы (таблица Files), отчеты (таблица Reports);

*Связи между сущностями:*

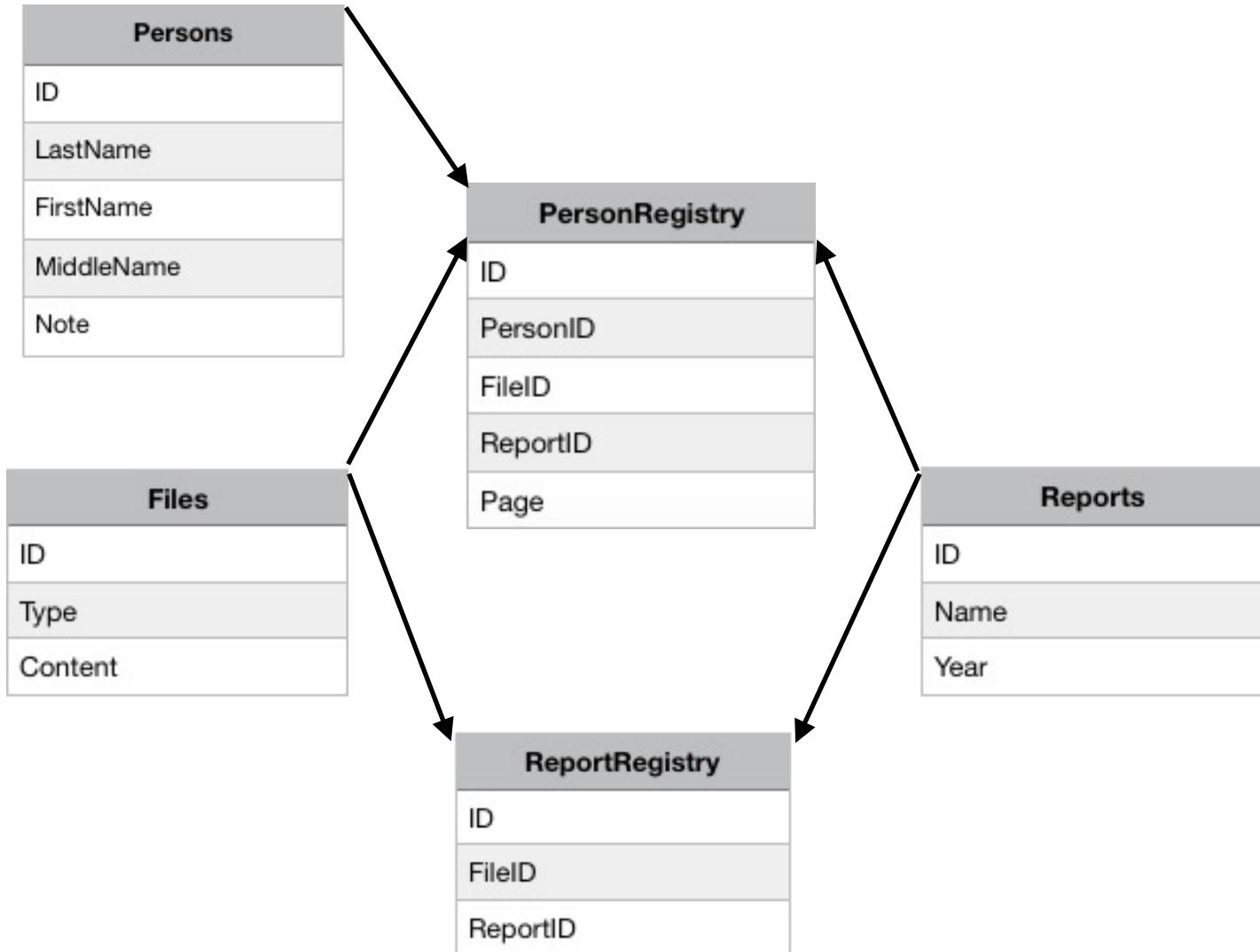
Отчеты - файлы в таблице ReportRegistry через внешние ключи ReportID, FileID;

Мощность связи - один-ко-многим (в одном отчете может быть несколько файлов - например, если изображения в jpg формате).

Люди - Файлы - Отчеты в таблице PersonRegistry через внешние ключи PersonID, FileID, ReportID;

Мощность связи - многие-ко-многим (один человек может встречаться на нескольких страницах, а на странице могут упоминаться несколько людей).

*Таблицы:* схема представлена на рисунке



#### Описание системы:

В таблице Persons хранятся данные о людях (ФИО и дополнительные заметки). Имена могут быть записаны как на латинице, так и на кириллице.

В таблице Files - изображения в двух допустимых форматах: pdf и jpg.

В таблице Reports содержится информация о самих бумажных отчетах (название, год).

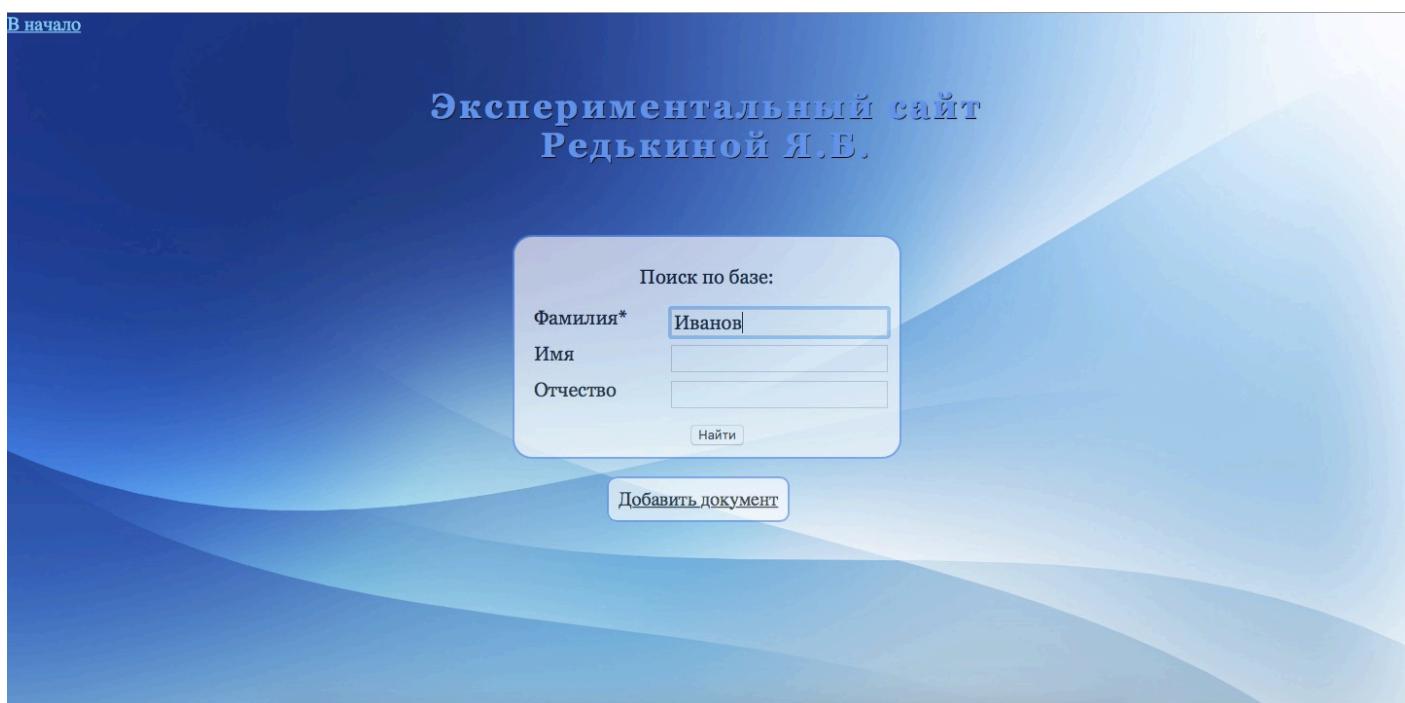
Связь между реальными отчетами и их отсканированными изображениями отражена в таблице ReportRegistry.

Связь между людьми - файлами - отчетами содержится в таблице PersonRegistry. Кроме того, в этой таблице есть поле страница (page), на которой в бумажном варианте встречается ФИО человека. На самих страницах могут быть как одна, так и несколько фамилий. Поэтому в этой таблице реализована связь многие-ко-многим (через внешние ключи-идентификаторы).

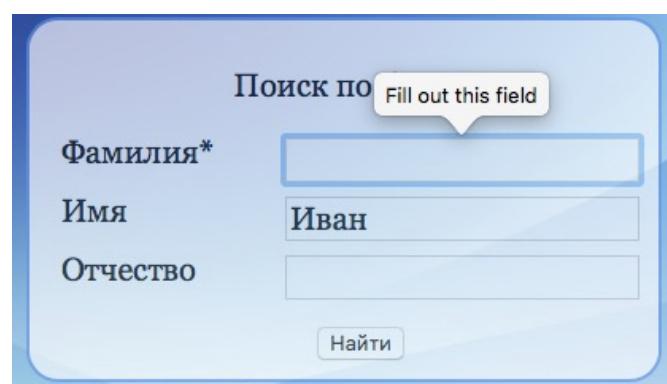
#### 4. Описание сайта

Сайт разработан с помощью веб-фреймворка Flask для Python, а также с помощью языков HTML, CSS, JavaScript.

На стартовой странице организована форма для поиска страниц по введенному имени человека. Поле “Фамилия” является обязательным (проверка осуществляется через возможности HTML). Поля формы “Имя” и “Отчество” позволяют уточнить поиск.



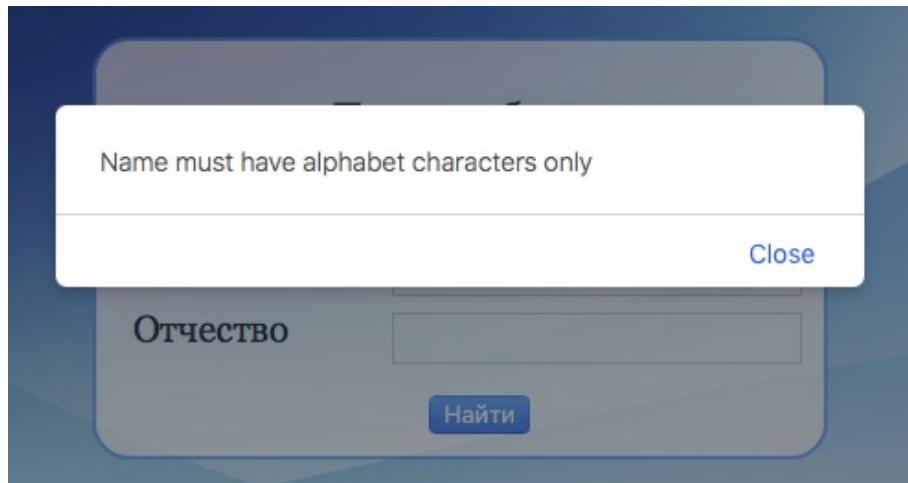
The screenshot shows the search interface on the experimental website. At the top left is a link "В начало". The main title is "Экспериментальный сайт Редькиной Я.Б.". Below the title is a search form titled "Поиск по базе:". It contains three input fields: "Фамилия\*" (Family name\*) with the value "Иванов", "Имя" (Name) with the value "Иван", and "Отчество" (Middle name) with an empty value. A "Найти" (Find) button is located below the fields. Below the search form is a "Добавить документ" (Add document) button.



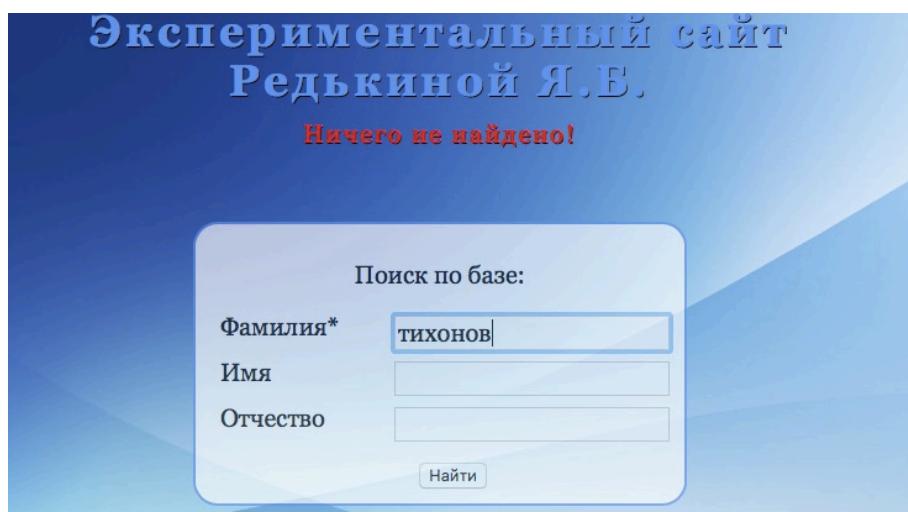
This screenshot shows the same search interface as above, but with a tooltip "Fill out this field" pointing to the "Фамилия\*" (Family name\*) input field, indicating it is a required field.

Регистр не учитывается: можно вводить любым регистром. Поиск будет осуществлен в любом случае.

Кроме того, средствами JavaScript реализована проверка (валидация) введенных данных: ввод ФИО допустим только латиницей или кириллицей, цифры и спецсимволы недопустимы:



Если в базе данных нет искомого человека, то выводится соответствующее сообщение:



Если совпадение найдено (или несколько совпадений по людям) - формируется новая страница с результатом поиска в базе данных: указывается название отчета и страница этого отчета, где упоминается человек:

[В начало](#)

## Экспериментальный сайт Редькиной Я.В.

Результаты поиска: Иванов

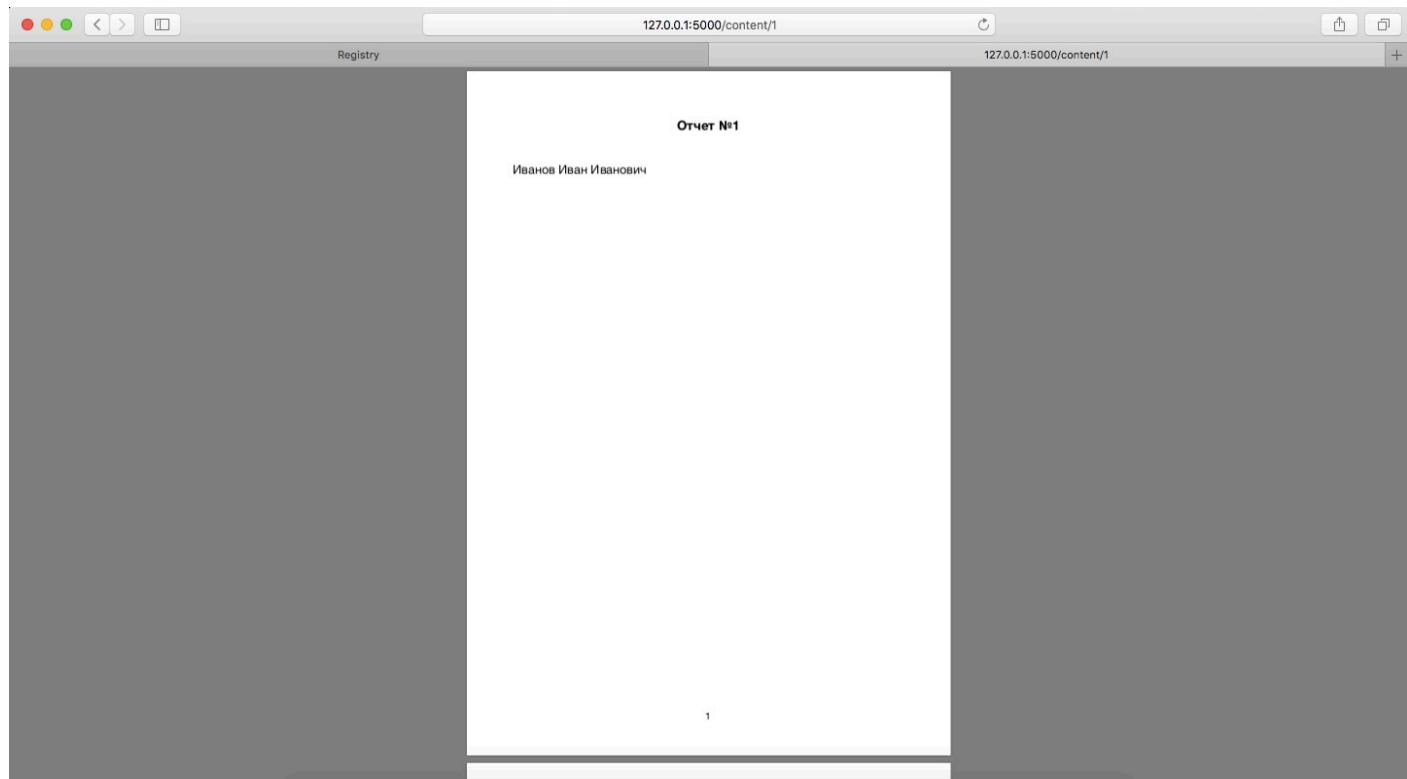
Отчет 1 Стока 1 [Просмотр](#)

Отчет 2 Стока 1 [Просмотр](#)

Отчет 3 Стока 1 [Просмотр](#)

Отчет 2 Стока 3 [Просмотр](#)

Напротив каждой страницы есть ссылка на просмотр самого файла (изображения). Могут быть представлены jpg (одно изображение) или pdf (возможно с несколькими страницами). Открываются файлы всегда в новой вкладке браузера:



На стартовой странице реализована возможность добавить новый документ в базу данных:

[В начало](#)

## Экспериментальный сайт Редькиной Я.Е.

Поиск по базе:

Фамилия\*

Имя

Отчество

→ [Добавить документ](#)

Open "127.0.0.1:5000/upload" in a new tab

После нажатия на кнопку откроется новая страница с формой для заполнения необходимых данных:

[В начало](#)

## Экспериментальный сайт Редькиной Я.Е.

Заполните поля:

Фамилия\*

Имя

Отчество

Примечание

Название отчета\*

Год отчета\*

Страница\*

Файл:\*

no file selected

[Добавить](#)

Обязательные поля отмечены звездочкой \*. При этом обязательность ввода осуществляется средствами HTML, а корректность ввода средствами JavaScript: ФИО проверяется на алфавитные символы, год и страница проверяются на цифровые значения:

Заполните поля:

Фамилия\* Тихонов

Year must be a number

Close

Примечание

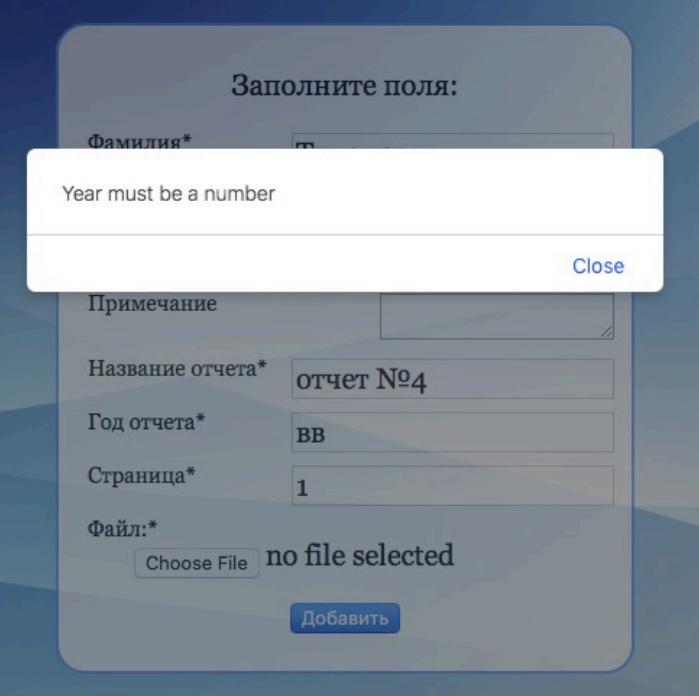
Название отчета\* отчет №4

Год отчета\* 2000

Страница\* 1

Файл:\* Choose File no file selected

Добавить



Кроме того, добавить новую запись не получится без загрузки файла (эта проверка реализуется с помощью Python):

Экспериментальный сайт  
Редькиной Я.Е.

Нет выбранного файла

Заполните поля:

Фамилия\* Тихонов

Имя Алексей

Отчество

Примечание

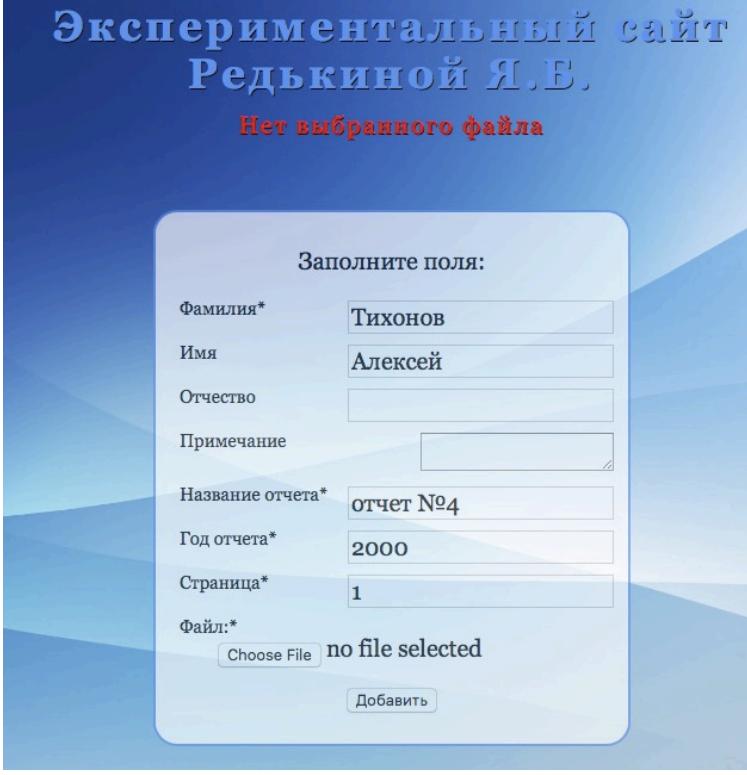
Название отчета\* отчет №4

Год отчета\* 2000

Страница\* 1

Файл:\* Choose File no file selected

Добавить



Если все поля заполнены корректно - будет вызвана соответствующая функция Python, которая настроет связь с базой данных и осуществит через SQL-запросы ввод новых данных в БД.

Результат работы этой функции может быть двух типов: SQL-запросы успешно обработаны и зафиксированы (commit). В этом случае выводится сообщение об успешной операции и осуществляется возврат на стартовую страницу:

[В начало](#)

## Экспериментальный сайт Редькиной Я.Е.

Запись успешно добавлена

Поиск по базе:

Фамилия\*

Имя

Отчество

[Добавить документ](#)

Второй случай - в ходе обработки произошли ошибки, тогда SQL-запросы откатываются назад (rollback) и выводится сообщение о соответствующей ошибке:

[В начало](#)

## Экспериментальный сайт Редькиной Я.Е.

Ошибка выполнения запроса:  
no such table: Person

Заполните поля:

Фамилия\*

Имя

Отчество

Примечание

Название отчета\*

Год отчета\*

Страница\*

Файл:\*

no file selected

Ввести новые данных можно не только через форму сайта в виде одного объекта. Python-функция, осуществляющая ввод оформлена как отдельный скрипт (утилита `insert_batch.py`) и позволяет осуществлять пакетных ввод данных в БД. Для этого нужно передать в качестве параметра список (`list`) соответствующих объектов класса `DBObj` (описание класса есть в основной программе).

## Проведение тестирования

Необходимо установить Python 3, Flask, SQLite3.

Сохранить на диск в некоторую папку все файлы, относящиеся к проекту.

Архив с нимиложен на Яндекс.Диск:

[https://disk.yandex.ru/d/KbpkXtN\\_HHHlnA](https://disk.yandex.ru/d/KbpkXtN_HHHlnA)

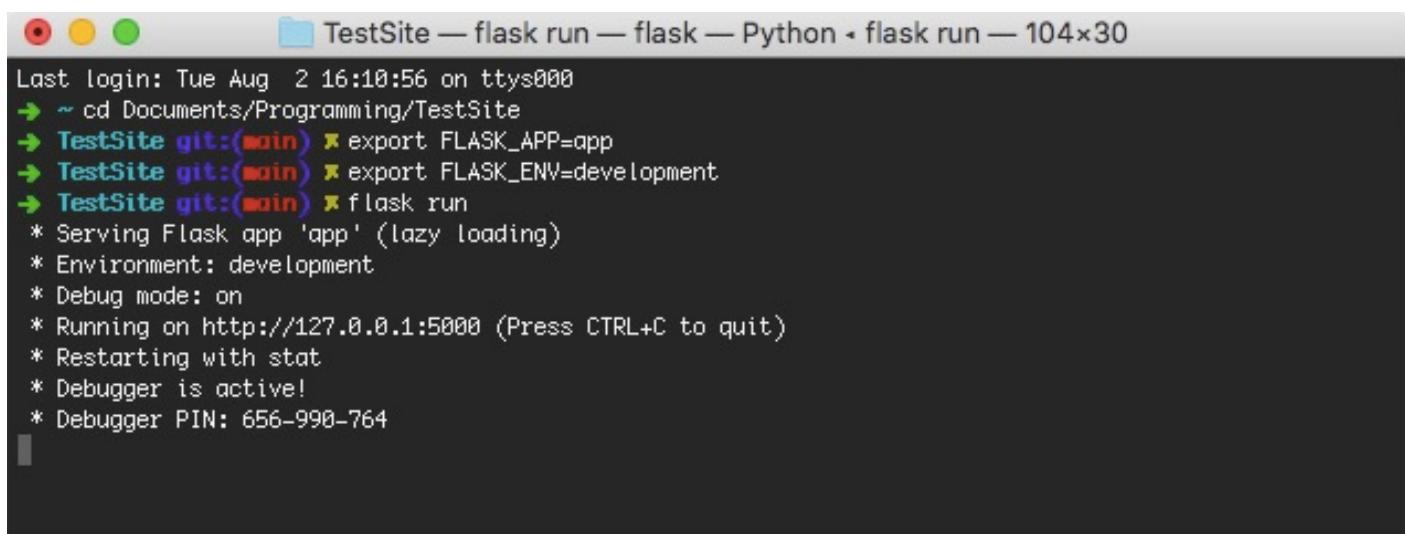
Также проект можно скачать с сайта GitHub.com:

<https://github.com/yanaredkina/Website-project>

Среди файлов будет `init_db.py`. Это скрипт для инициализации БД некоторыми примерами данных.

Далее в командной строке в текущей директории последовательно выполнить:

```
-> python3 init_db.py  
-> export FLASK_APP=app  
-> export FLASK_ENV=development  
-> flask run
```



The screenshot shows a terminal window titled "TestSite — flask run — flask — Python - flask run — 104x30". The window displays the command-line output of running a Flask application. The user has navigated to the directory containing the application code and run the command `flask run`. The terminal shows the Flask application starting up, serving on port 5000, and indicating that the debugger is active with a PIN of 656-990-764.

```
Last login: Tue Aug  2 16:10:56 on ttys000  
→ ~ cd Documents/Programming/TestSite  
→ TestSite git:(main) ✘ export FLASK_APP=app  
→ TestSite git:(main) ✘ export FLASK_ENV=development  
→ TestSite git:(main) ✘ flask run  
* Serving Flask app 'app' (lazy loading)  
* Environment: development  
* Debug mode: on  
* Running on http://127.0.0.1:5000 (Press CTRL+C to quit)  
* Restarting with stat  
* Debugger is active!  
* Debugger PIN: 656-990-764
```

Flask запущен. В браузере по адресу <http://127.0.0.1:5000/> будет стартовая страница сайта.

Можно тестировать!

В базе данных есть документы с такими ФИО:

Петров Петр Петрович  
Романов Роман Романович  
Иванов Иван Иванович  
Сидоров Николай Николаевич  
Кузнецов Виктор Викторович  
Иванов Игорь Игоревич  
Романов Роман Сергеевич

## Итог

В ходе работы над проектом:

- были изучены новые языки: HTML, CSS, JavaScript, SQLite;
- были получены базовые навыки создания сайтов: как статических (чисто на основе HTML), так и динамических (на основе Python);
- был получен начальный опыт веб-дизайна;
- были объединены и применены знания SQL и Python в практическую реализацию системы управления базой данных.