

Доступ к системе

GitHub:

https://github.com/Domnenko-Aleksey/LCT_22

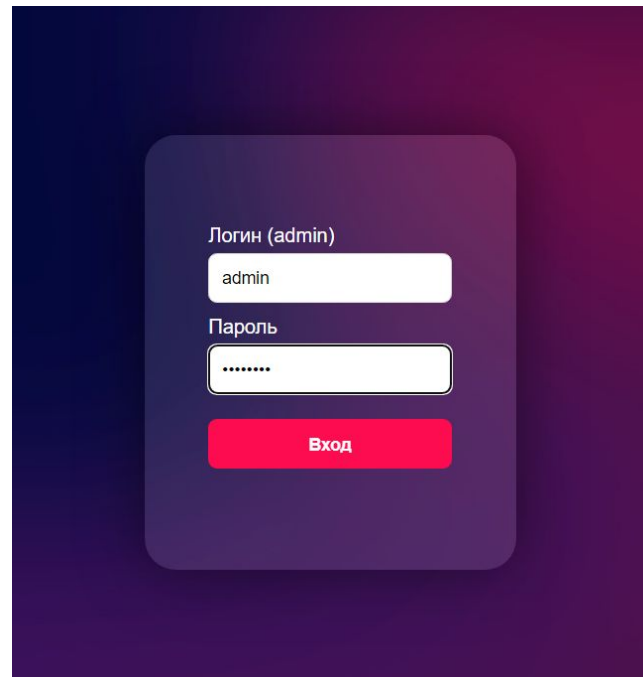
Прототип:

<http://77.222.58.99:9100>

Логин: **admin**

Пароль: **12345678**

@domnenko_a_n - для вопросов

A login form mockup on a dark purple gradient background. The form is a light purple rounded rectangle containing two input fields and a button. The first input field is labeled 'Логин (admin)' and contains the text 'admin'. The second input field is labeled 'Пароль' and contains seven dots. Below the fields is a red button with the text 'Вход' in white.

Логин (admin)

admin

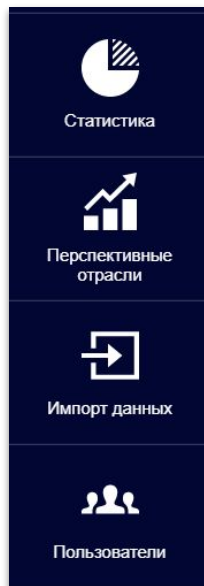
Пароль

.....

Вход

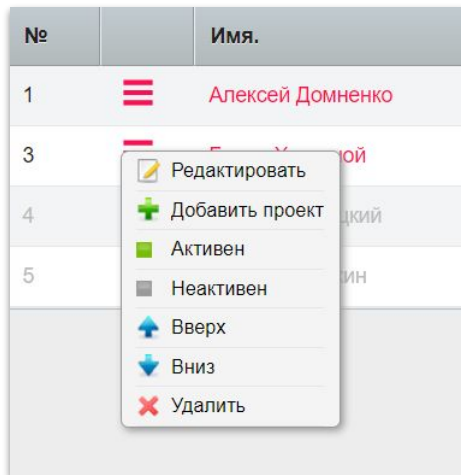
Интерфейс, основные элементы

Компоненты



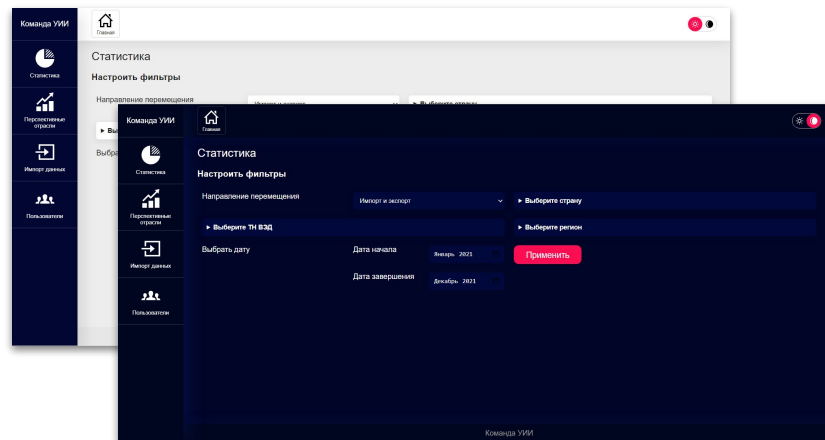
Контекстное меню

активизируется при нажатии на гамбургер - меню



Тема интерфейса

Переключение на светлую / тёмную тему



Стек технологий

Язык программирования (backend) - python

База данных - MySQL (MariaDB), и легко перестраивается на любую из семейства SQL

Frontend - javaScript, HTML5, CSS (flex)

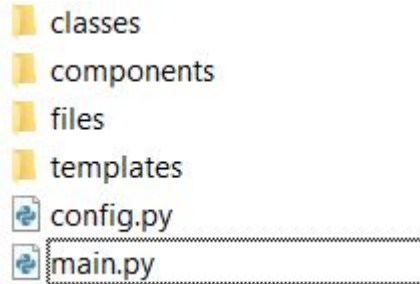
Система выполнена на фреймворке aiohttp, асинхронный HTTP-клиент/сервер.

Шаблонизатор - Jinja2

Система построена на принципах ООП, и принципах «Чистый код» Роберт Мартина

Основные паки:

- classes - классы системы, в основном для работы с БД
- components - рабочие компоненты,
- templates - шаблоны
- files - файлы, данные задания в папке files/data



База данных, таблицы

Сервер: localhost База данных: magnat_rus Таблица: data

ОБЗОР СТРУКТУРА SQL ПОИСК ВСТАВИТЬ ЭКСПОРТ ИМПОРТ ОПЕРАЦИИ СЛЕЖЕНИЕ ТРИГГЕРЫ

СТРУКТУРА ТАБЛИЦЫ СВЯЗИ

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	int		UNSIGNED	Нет	Нет		AUTO_INCREMENT	Изменить Удалить Ещё
2	napr	varchar(8)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
3	period	date			Нет	Нет			Изменить Удалить Ещё
4	nastranapr	varchar(8)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
5	tnved	varchar(32)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
6	tnved_2	varchar(8)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
7	edlsm	varchar(8)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
8	stolm	decimal(14,2)			Нет	Нет			Изменить Удалить Ещё
9	netto	decimal(9,3)			Нет	Нет			Изменить Удалить Ещё
10	kol	decimal(10,2)			Нет	Нет			Изменить Удалить Ещё
11	region_id	int			Нет	Нет			Изменить Удалить Ещё
12	region_s	varchar(255)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё

Консоль

Сервер: localhost База данных: magnat_rus Таблица: regions

ОБЗОР СТРУКТУРА SQL ПОИСК ВСТАВИТЬ ЭКСПОРТ ИМПОРТ ОПЕРАЦИИ СЛЕЖЕНИЕ

СТРУКТУРА ТАБЛИЦЫ СВЯЗИ

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	int		UNSIGNED	Нет	Нет		AUTO_INCREMENT	Изменить Удалить Ещё
2	name	varchar(255)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
3	num	int			Нет	Нет			Изменить Удалить Ещё
4	date_last	datetime			Нет	Нет			Изменить Удалить Ещё

Сервер: localhost База данных: magnat_rus Таблица: users

ОБЗОР СТРУКТУРА SQL ПОИСК ВСТАВИТЬ ЭКСПОРТ ИМПОРТ ОПЕРАЦИИ СЛЕЖЕНИЕ ТРИГГЕРЫ

СТРУКТУРА ТАБЛИЦЫ СВЯЗИ

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	int		UNSIGNED	Нет	Нет		AUTO_INCREMENT	Изменить Удалить Ещё
2	name	varchar(64)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
3	surname	varchar(64)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
4	email	varchar(32)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
5	psw	varchar(64)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
6	text	text	utf8mb4_0900_ai_ci		Да	NULL			Изменить Удалить Ещё
7	image	varchar(32)	utf8mb4_0900_ai_ci		Да	NULL			Изменить Удалить Ещё
8	phone	varchar(128)	utf8mb4_0900_ai_ci		Нет	Нет			Изменить Удалить Ещё
9	date_reg	datetime		Нет	CURRENT_TIMESTAMP		DEFAULT_GENERATED		Изменить Удалить Ещё
10	date_last	datetime		Нет	CURRENT_TIMESTAMP		DEFAULT_GENERATED		Изменить Удалить Ещё
11	ordering	int			Нет	Нет			Изменить Удалить Ещё
12	status	int			Нет	Нет			Изменить Удалить Ещё

Обработка URL

Get запрос разделяется на составляющие в классе CORE и по первому элементу списка **CORE.p[0]** определяется какой компонент будет подгружен,

Файл main.py, функция

```
@aiohttp_jinja2.template('main.html')
async def index(request):
```

ког:

Нижее идёт обработка 404 ошибки, редиректа и ответа на ajax

Авторизация пользователя, хранение сессий осуществляется с помощью **fernet**

```
# Установка сессий
fernet_key = fernet.Fernet.generate_key()
secret_key = base64.urlsafe_b64decode(fernet_key)
setup(app, EncryptedCookieStorage(secret_key))
```

```
# Вызов компонента
functions = {
    '': mainpage.mainpage,
    'users': users.users,
    'import_data': import_data.import_data,
    'stat_data': stat_data.stat_data,
    'promising': promising.promising,
    'predict': predict.predict
}

if (CORE.p[0] not in functions):
    raise web.HTTPNotFound()

# Вызов функции возвращает не False в случае редиректа
r = functions[CORE.p[0]](CORE)

# Проверка и обработка редиректа
if r:
    if 'redirect' in r:
        # Обработка редиректа
        return web.HTTPFound(r['redirect'])
    if 'ajax' in r:
        # Обработка ajax
        return web.HTTPOk(text=r['ajax'])

return {'AUTH': True, 'content': CORE.content, 'head': CORE.getHead()}
```

Компоненты, используемые в работе

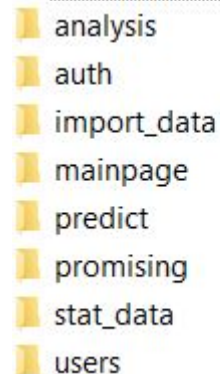
- **auth** - отвечает за авторизацию пользователя, путь в URL: **/auth**
- **Import_data** - загрузка данных в систему, путь в URL: **/import_data/..**
- **mainpage** - главная страница, путь в URL: **/**
- **promising** - поиск перспективных отраслей, путь в URL: **/promising/..**
- **stat_data** - получение статистических данных. путь в URL: **/stat_data/..**
- **users** - управление пользователями. путь в URL: **/users/..**

Файлы js, css компонентов размещены в папках
/templates/название_компонента

Общие js. css файлы находятся в папке
/templates/general

В том числе файл, **pdf.js**. формирующий pdf документ “на лету”

DAN.js, DAN.css - содержат общие функции, например ajax, модальные окна, flex и др.



Компонент stat_data

Отвечает за вывод компонента статистики:

stat_data.py - роутер url запроса для компонента

mainpage.py - главная страница компонента, скрин справа

get_stat_ajax.py - обрабатывает ajax запрос, посылаемый скриптом

/templates/stat_data/js/mainpage.js

использует класс запросов к БД

/classes/GetData.py

Все файлы содержат подробные комментарии, переменные названы удобно для понимания

Компонент stat_data, результат работы

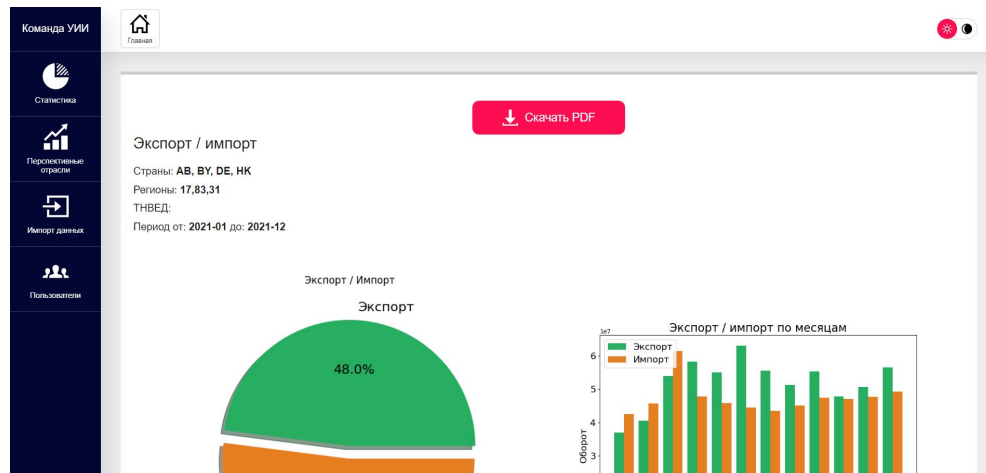
Результат работы файла
`/components/stat_data/get_stat_ajax.py`

Ответ в формате json содержит
HTML данные, которые после выполнения
запроса размещаются на странице.

Для удобства просмотра - включается
прокрутка к блоку результатов вывода.

Pdf файл формируется на лету с помощью

`/templates/general/js/pdf.js`



Компонент promising

Отвечает за вывод компонента “перспективные отрасли”.

promising.py - роутер url запроса для компонента

mainpage.py - главная страница компонента, скрин справа

get_promising_ajax.py - обрабатывает ajax запрос, посылаемый скриптом

/templates/promising/js/mainpage.js

использует класс запросов к БД

/classes/GetData.py

Все файлы содержат подробные комментарии, переменные названы удобно для понимания

№	ТНВЭД	Сумма
1	8419899890	166833888.00
2	8704239108	28537439.00
3	8477200000	24101838.00
4	8429519900	21724220.00
5	2617900000	19448346.00

Компонент promising, файл get_promising_ajax.py

Наши данные могут содержать пропуски (не за все месяцы есть операции по конкретному tnved для конкретного региона, поэтому сначала восстановим пропущенные месяцы с нулевыми значениями.

1. Сформируем диапазон `gam` `d_delta = (d_end.year - d_start.year) * 12 + d_end.month - d_start.month + 1`
2. Сформируем уникальные tnved используемые в отфильтрованных данных из БД
3. Сгруппируем данные по ключам "tnved", "period" и посчитаем сумму стоимости для групп

```
grouped = df.groupby(["tnved", "period"])['stoim'].sum()
```

4. В цикле для каждого tnved подсчитаем коэффициенты линейной регрессии и проверим на функцию "поиска свехрновых"

```
coef = round(lin_reg(sum_list)*100, 2)
```

```
supernova = get_supernovae(sum_list)
if supernova:
    supernova_list.append([t, supernova])
```

5. Линейная регрессия вернёт нам коэффициенты, отберём положительные и отсортируем по убыванию.
6. Функция свехрновых сравнивает срезы массивов за первые месяцы (все) и последние 3. `if a[:-3].sum() == 0 and a[-3:].sum() > 0:`

```
# Линейная регрессия
def lin_reg(lst):
    # Нормализуем данные
    d = np.array(lst).reshape(-1, 1)
    data = normalize(d)
    x = np.arange(len(lst)).reshape(-1, 1)

    reg = LinearRegression().fit(x, data)
    coef = reg.coef_[0][0]
    return coef
```

Компонент import_data

Отвечает за вывод компонента “перспективные отрасли”.

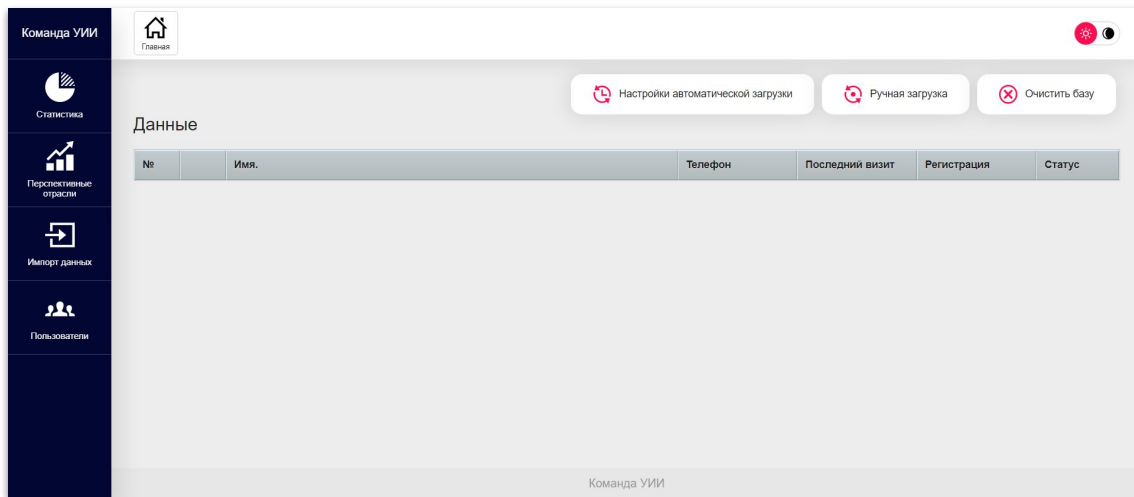
import_data.py - роутер url запроса для компонента

mainpage.py - главная страница компонента, скрин справа

clear_data_ajax.py - очистка данных

load_data_ajax.py - загрузка данных возвращает данные в формате json, отправляемые файлом
/templates/import_data/js/mainpage.js

Все файлы содержат подробные комментарии, переменные названы удобно для понимания



Загрузка и удаление данных

Надо помнить, что при нажатии на кнопку процесс загрузки может затянуться на час, т.к. у нас около 5 миллиона строк, которые надо обработать и загрузить в базу данных, при этом на сервер крутятся ещё около 400 сайтов. Поэтому просьба проверять загрузку данных только после проверки всех прочих компонентов. Код доступа программисты найдут в комментариях к JavaScript коду, который отвечает за ajax загрузку данных.

Загрузка происходит по мере обхода папок. После загрузки каждой папки система возвращает номер текущей папки и общее количество папок, ниже код js и python, ответственный за порционную работу.

```
// Загрузка данных ajax
load_data_ajax() {
    let form = new FormData()
    DAN.ajax('/import_data/load_data_ajax', form, function(data) {
        console.log(data)
        DAN.$('items_progress_count').innerHTML = data.num_sum
        DAN.$('items_progress_current').innerHTML = data.num_current
        DAN.$('items_progress').setAttribute('max', data.num_sum)
        DAN.$('items_progress').value = data.num_current

        if (data.num_current < data.num_sum) {
            IMPORT_DATA.load_data_ajax()
        } else {
            DAN.$('modal_h1').innerHTML = 'Загрузка завершена'
        }
    })
},
```

```
20 # Перебираем все папки
21 dir_list = []
22 for p in path_list:
23     dir_path = dir + '/' + p
24     if os.path.isdir(dir_path):
25         dir_list.append(p)
26
27 IMPORTDATA.dir_list = dir_list # Не читаем в терминале русские буквы => utf-8
28 IMPORTDATA.num_sum = len(dir_list)
29 IMPORTDATA.num_current = 0
30
31 # Текущий path обрабатываемого файла
32 file_path = dir + '/' + IMPORTDATA.dir_list[IMPORTDATA.num_current] + '/DATTSVT.csv'
33 if os.path.isfile(file_path):
34     IMPORTDATA.setData(file_path)
35
36 # Условие следующего хода
37 IMPORTDATA.num_current += 1
38 if IMPORTDATA.num_current > IMPORTDATA.num_sum:
39     IMPORTDATA.initial() # Сбрасываем настройки импорта, импорт окончен
40
41 answer = {'answer': 'success', 'num_sum': IMPORTDATA.num_sum, 'num_current': IMPORTDATA.num_current}
42 return {'ajax': json.dumps(answer)}
```

Загрузка данных

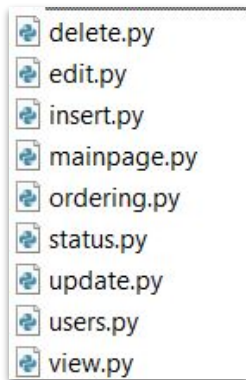
The screenshot displays a development environment with three main components:

- Code Editor (Left):** Shows the `load_data_progress()` function in `mainpage.2.js`. The function constructs an HTML form with a progress bar and a message area. The progress bar is initialized with `max=""` and `value=""`.
- Terminal Window (Bottom Left):** Shows the execution of a Python script `/import_data/import_data.py`. The output indicates a successful request and authentication, with a warning about mixed types in the DataFrame.
- Web Browser (Right):** Shows a loading dialog titled "Загрузка данных" (Loading data) with a progress bar. The console output shows the response from the server, indicating a successful request and the current progress of the data loading process.

Компонент users

Вызов действий осуществляется из контекстного меню

Назначение файлов говорит само за себя, Трудностей с пониманием кода не возникнет

The screenshot shows a web application interface. On the left is a dark blue sidebar with the text 'Команда УИИ' at the top. Below it are icons and labels for 'Главная', 'Статистика', 'Перспективные отрасли', 'Импорт данных', and 'Пользователи'. The main area has a light gray background. At the top right of the main area is a settings icon. Below the sidebar, there is a red button with a plus icon and the text 'Добавить пользователя'. Below this is the title 'Пользователи'. A table follows with columns: '№', 'Имя.', 'Телефон', 'Последний визит', 'Регистрация', and 'Статус'. The table contains five rows of user data. A context menu is open over the first row, showing options: 'Редактировать', 'Добавить проект', 'Активен', 'Неактивен', 'Вверх', 'Вниз', and 'Удалить'. At the bottom of the main area, the text 'Команда УИИ' is repeated.

№	Имя.	Телефон	Последний визит	Регистрация	Статус
1	Алексей Домненко	+7 9276 925 921	2022-10-29 18:33:09	2022-10-29 18:33:09	1
3	Борис Хуторной	+7 999 158 57 42	2022-11-05 19:32:35	2022-11-05 19:32:35	1
4		+7 985 997 8592	2022-11-05 19:38:52	2022-11-05 19:38:52	0
5		+7 925 062 5853	2022-11-05 19:39:41	2022-11-05 19:39:41	0