Сервис расчета рыночной стоимости жилой недвижимости г. Москвы

Документация

Хакатон ЛЦТ, 6 задача, октябрь/ноябрь 2022г.

**NskTeam**

Плотникова Александра

Белова Надежда

# Оглавление

[**Оглавление**](#_4yzxqeddxqza) **2**

[**Описание сервиса**](#_4t80lsza8j5x) **3**

[Общая идея](#_sv0u1ag0gcx6) 3

[Ссылки](#_otqs22xbcx5y) 3

[**Структура сервиса**](#_q0yj536axjvl) **4**

[Схема сервиса](#_gs0tr3zco62e) 4

[Модули](#_pdb0pvse74p) 4

[Интерфейс и usability](#_a47llnyw6q9a) 5

[Файлы проекта](#_rgeq95ud5n1d) 6

[**Технологии**](#_79axmtxoqoa2) **7**

[**Устройство модулей (детали и принципы)**](#_mfml3v4kmb26) **9**

[1. Автоподгрузка объявлений о продаже квартир](#_y4ouzx7jbmdv) 9

[Схема](#_qotco4hbw61b) 9

[Общий алгоритм работы](#_7ohe3cmsthds) 9

[Где найти в коде](#_3mfow2jll2r) 12

[Первичное наполнение системы данными:](#_rk30v12t60sn) 13

[2. Подбор аналогов для эталона](#_flh8gx95ts6) 13

[Схема](#_cuihftl69vcj) 13

[Общий алгоритм работы](#_njbp99qp6ube) 13

[Где найти в коде](#_wis3anrsvnxs) 14

[3. Расчет стоимости эталона и остальных квартир из пула](#_rgemo9y3kh2s) 16

[Схема](#_klu3xrvlqcxm) 16

[Общий алгоритм работы](#_vgnrq2kkzw06) 16

[Где найти в коде](#_36qz1r1vtn8b) 17

[4. Machine Learning](#_nltizww84ynu) 18

[Схема работы](#_7ephuw2w3oin) 18

[О модели](#_3zp1131ziysb) 18

[Где найти в коде и среди рабочих файлов](#_pbxinp39i1ne) 20

[**Инструкция для страницы Home (остальные - очевидные).**](#_lra9v4s77var) **21**

[**Ограничения**](#_miy44ltgf1if) **22**

[**Команда NskTeam**](#_n881oaji1vcb) **23**

# 

# Описание сервиса

Сервис расчета рыночной стоимости недвижимости г.Москвы автоматизирует процесс расчета стоимости однотипных квартир, находящихся в собственности города, в зависимости от их индивидуальных параметров. Сервис ускорит процедуру выдачи экспертных заключений о стоимости квартир и позволит избежать сделок, при которых городское имущество продается по цене ниже рыночной.

## Общаяидея

пользователь подает на вход сервиса файл формата xlsx со списком квартир и их параметров согласно ТЗ с отмеченным эталонным объектом. Сервис подбирает несколько актуальных объявлений о продаже квартир в Москве, близких по расположению к эталону, дает возможность гибко корректировать подбор аналогов с помощью фильтра и исключать точечно аналоги из расчета, рассчитывает рыночную цену на основании подобранных объектов и с учетом корректировок согласно ТЗ (Приложение 1, 2).

Также пользователь может выгрузить результаты расчета в виде xlsx файла, и запустить процесс обновления данных в системе об актуальных предложениях на рынке недвижимости.

Помимо сказанного выше, сервис отдельно предлагает свой вариант рыночной цены эталонного объекта, рассчитанный искусственным интеллектом на базе алгоритма Catboost Regression.

Актуальные объявления о продаже квартир подкачиваются в систему следующим образом: автоматически (по нажатию кнопки)система подгружает последние объявления. Эти объявления сразу идут в работу для поиска аналогов квартир для расчета рыночной стоимости эталонного объекта. Подробное описание механизма можно прочесть в документации ниже.

Также сервис хранит в табличной форме все ранее проведенные оценки объектов. Пользователь может посмотреть последние архивные данные на странице с архивом и выкачать экселевский файл с архивом.

## Ссылки

Прототип: <http://178.170.192.124:8501/>

GitHub: <https://github.com/sasa2111/nskteam_6_hackaton>

# Структура сервиса

Сервис состоит из одного докер-контейнера, в котором содержатся все файлы с кодом используемых модулей, таблица с архивом вычислений, таблица с базой используемых объявлений, ML-модель, конфигурационный файл для интерфейса.

## Схема сервиса

**Docker container**

Модуль расчета рыночной цены эталона по ТЗ (.py)

Модуль подбора аналогов для эталона (.py)

Модуль расчета рыночной цены всего пула

Архив расчетов

(Табл. excel)

ML-модель альтернативного расчета цены (pickle)

База объявлений (Табл. excel)

Модуль автоподгрузки новых объявлений (.py)

**Web-интерфейс (py)**

Специальная почта для проекта

Рассылки объявлений с сайта Яндекс. Недвижимость с различными фильтрами по параметрам

файл xlsx с пулом и отмеченным эталоном

файл xlsx с оценкой пула и данными об аналогах (+корректировки, удельные цены факт и по расчетам, общие цены)

## Модули

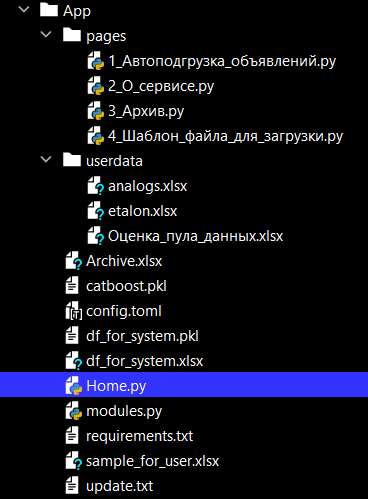
1. Модуль расчета рыночной стоимости по ТЗ: 2 питоновских модуля (функции), которые срабатывают на Шаге 3 в сервисе после выполнения пользователем определенных действий + нажатие кнопки. Один модуль рассчитывает цену для эталона по аналогам. Другой модуль оценивает пул объектов на основе оценки эталона.
2. Модуль подбора аналогов для эталона: выдает список подходящих объявлений в соответствии с эталоном и указанным пользователем фильтрами (радиус, отклонение по площади, тип этажа и т.д.)
3. Модуль автоподгрузки новых объявлений: питоновская функция (модуль), который по нажатию пользователем кнопки добавляет в базу объявлений самые свежие объявления, которые стали доступны системе. Модуль не обращается напрямую к сайтам объявлений. Модуль обращается только к специально заведенной для проекта электронной почте, на которую приходят рассылки объявлений с сайта Яндекс.недвижимость, и расшифровывает письма - превращает их в новые элементы базы объявлений.
4. ML-модель расчета цены: заранее обученная модель, которая для любого объявления выдает свой прогноз цены на основании рыночных данных. Это происходит на шаге 3, там же где идет оценка по ТЗ. В итоге система выдает 2 оценки - по ТЗ и оценку искусственного интеллекта.
5. База объявлений: pkl-файл, содержащий таблицу объявлений. Используется для подбора аналогов, пополняется каждый раз, когда пользователь запускает обновление базы. Базу можно скачать в формате xlsx.
6. Архив оценок. В специальном разделе сервиса можно выкачать весь архив и посмотреть последние расчеты прямо на странице.

## Интерфейс и usability

1. Главная страница (Home), содержащая в себе весь основной путь пользователя оформлена в виде трех ключевых шагов-блоков. Пользователь идет от шага к шагу, в конце загружает файл с оценками пула квартир.
   1. Шаг 1: загрузка пула и поиск координат эталонного объекта.
   2. Шаг 2: подбор аналогов
   3. Шаг 3: расчет стоимости пула квартир по ТЗ и с помощью ИИ.
2. Дополнительный функционал сервиса расположен на дополнительных страницах:
   1. Автоподгрузка новых объявлений
   2. Выгрузка архива оценок
   3. Информация о сервисе
   4. Возможность скачать шаблон для загрузки нового файла
3. Весь интерфейс сервиса сделан на Streamlit (библиотека Python).
4. Все таблицы в сервисе интерактивные:
   1. можно менять ширину полей мышкой,
   2. можно проматывать таблицы вправо-влево (есть ползунок),
   3. можно сортировать таблицы по значениям полей - кликом мыши по названию поля).
5. Интерактивные карты:
   1. можно приближать/удалять скроллом или плюсиком в углу карты - чтобы видеть дома, улицы в нужном масштабе.
   2. можно перемещаться по карте с помощью мыши.
6. удобный интерфейс фильтров для подбора аналогов: можно по параметру набирать нужные значения характеристик, есть подсказки.
7. Загрузка и выгрузка файлов в формате xlsx
8. Минималистичный дизайн (шаблонное решение streamlit, можно несильно кастомизировать)

## Файлы проекта

Папка App:



* Home.py - главная страница сервиса. В этом файле написан ее функционал и интерфейс.
* папка pages - содержит 4 файла .py с второстепенными страницами сервиса:
  + 1\_Автоподгрузка\_объявлений.py - функционал и интерфейс одноименного раздела
  + 2\_О\_сервисе.py - функционал и интерфейс одноименного раздела
  + 3\_Архив.Py - функционал и интерфейс одноименного раздела
  + 4\_Шаблон\_файла\_для\_загрузки.py - функционал и интерфейс одноименного раздела
* modules.py - набор ключевых модулей сервиса, которые вызываются в Home.py и в файле 1\_Автоподгрузка\_объявлений.py
* catboost.pkl - ML-модель для альтернативной оценки рыночной стоимости квартир иск.интеллектом
* config.toml - конфигурационный файл для настройки внешнего вида сервиса с помощью streamlit. В docker-контейнере находится в определенной папке .streamlit
* requirements.txt - список библиотек, которые инсталлируются в докер-образ.
* update.txt - хранит дату и время последнего обновления базы объявлений.
* sample\_for\_user.xlsx - файл, который выгружается в разделе с Шаблоном для загрузки, и который отображается на странице Home как пример.
* df\_for\_system.pkl - сериализованный датасет с базой объявлений для поиска аналогов (хранит датасет в точности таким, каким мы его сформировали).
* df\_for\_system.xlsx - эксель-копия сериализованного датасета для выгрузки пользователем в разделе с Автоподгрузки.
* Archive.xlsx - файл с архивом оценок для выгрузки пользователем
* Папка userdata - хранит 3 последних пользовательских файла, которые каждый раз перезаписываются системой в процессе взаимодействия с пользователем по основному алгоритму на странице Home.

Также на одном уровне с папкой App:

* Dockerfile - для разворачивания образа Docker
* Readme.txt - Инструкция по разворачиванию докер-образа локально и запуску контейнера с приложением.

# Технологии

|  |  |
| --- | --- |
| Язык программирования: | python |
| Интерфейс (фронт): | библиотека streamlit |
| Контейнеризация: | Docker (Прототип развернут на SberCloud) |
| Автоподгрузка объявлений: | Специально заведенная для проекта почта на mail.ru,  + настроенные специальным образом подписки на рассылки объявлений на сайте Яндекс.недвижимость,  + python: получение данных с почты и расшифровка - библиотеки imaplib, email, base64, dill, pandas, time, geopy |
| ML-модель: | CatboostRegressor |
| Архив и база объявлений: | excel, pickle (dill) |
| Использованные библиотеки Python: | pandas, datetime, time, imaplib, email, base64, streamlit, geopy, catboost, dill |

# Устройство модулей (детали и принципы)

## Автоподгрузка объявлений о продаже квартир

### Схема

**Яндекс.Недвижимость**

Рассылка. Фильтр 1

Рассылка. Фильтр 2

Рассылка. Фильтр 3

.

.

.

.

.

.

.

Рассылка. Фильтр 26

**Спец. почта на mail.ru**

(sbor\_dannyh\_msk@mail.ru)

**Модуль автоподгрузки объявлений:**

запускается по запросу пользователя

смотрит непрочитанные письма

открывает их (отмеч. прочитанными)

в каждом письме 1-5 объявлений - вытаскивает данные

расшифровывает данные о квартирах

подкачивает координаты по адресам (широта, долгота)

добавляет данные в общую таблицу

**База объявлений**

Модуль автоподгрузки объявлений запускается по нажатию кнопки пользователем.

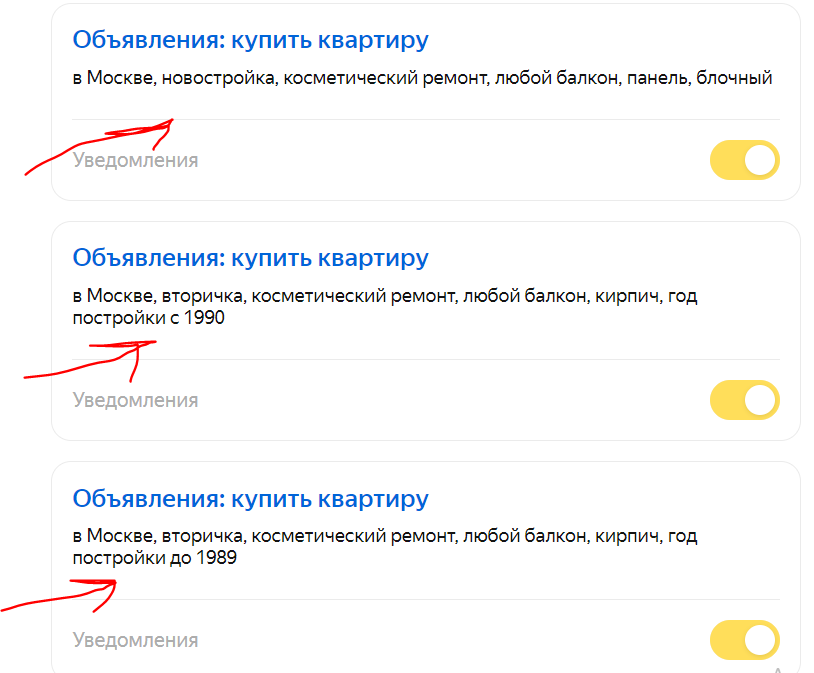
При запуске он подключается к почте по специальному ключу для приложений, заходит в определенную папку, куда ежедневно попадает рассылка от Яндекс.недвижимости, пробегает по всем непрочитанным письмам в папке, считывая из них данные о квартирах и отмечая прочитанными.

### Общий алгоритм работы

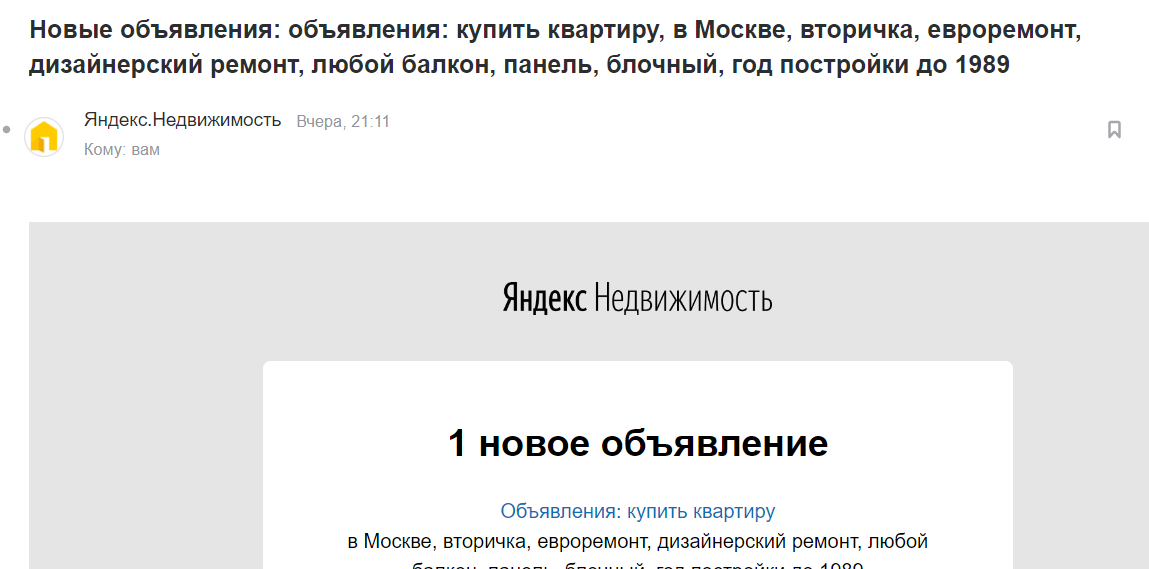
При расшифровке писем модуль автоподгрузки часть данных получает из тела письма, а часть данных из темы письма.

1. в теме письма указаны параметры фильтрации, по которым настроена данная рассылка: материал стен, сегмент, состояние ремонта. см. **Изображение1.**
2. в теле письма указана информация о конкретных квартирах из данной рассылки: адрес, число комнат, этаж квартиры и общая этажность дома, время до метро, ближайшая станция метро, удельная и общая цены. См. Изображения 2,3.
3. В данной версии модуля используется 26 видов различных фильтров/рассылок с сайта Яндекс.недвижимость, что позволяет получать подробные данные о новых объявлениях. Также настроена максимально частая рассылка по всем фильтрам, поскольку в 1 письмо яндекс ставит от 1 до 5 объявлений, а все остальные объявления, которые скопились за время с предыдущего письма, можно посмотреть по указанной ссылке в письме. За счет максимально частых уведомлений система получает максимальное число данных о новых объявлениях. Яндекс отправляет письма по несколько раз в день, и большинство из них содержат менее 5 объявлений, что позволяет предположить, что наша система охватывает бОльшую часть данных по указанным фильтрам. Ежедневно - по несколько десятков объявлений приходит на почту.

**Изображение 1:** скриншот с настройками рассылок на Яндекс.Недвижимость.



**Изображение 2**: Тема письма в почте о рассылке с фильтрами.

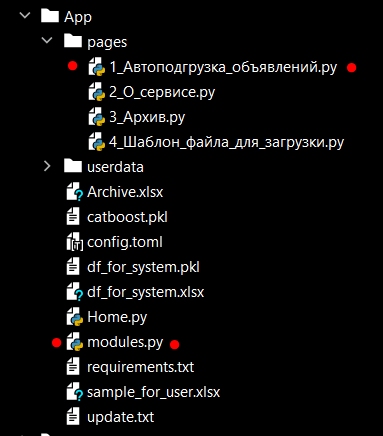


**Изображение 3**: информация об объявлении в письме из рассылки.



1. После расшифровки писем, сервис автоподгрузки объявлений, подкачивает к ним данные о широте и долготе по адресам с помощью python библиотеки geopy и потом собирает все полученные объявления в общую таблицу, которая присоединяется к тем данным, которые уже были ранее в системе. Дубли удаляются (если вдруг в почте кто-то отметил непрочитанным лишнее письмо, оно считается заново и потом как дубль уйдет).

### Где найти в коде

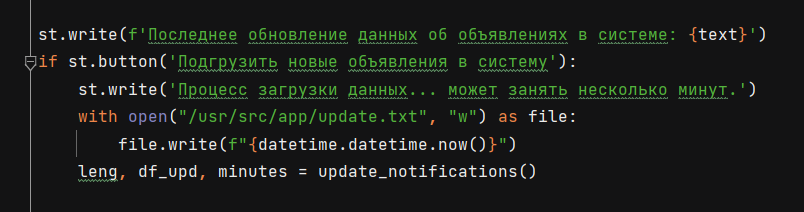


Файл ./modules.py:

* функция mail\_preparation: обрабатывает одно письмо (расшифровывает и собирает данные об объявлениях из письма в таблицу).
* функция update\_notifications: обработка всех непрочитанных писем в почте и объединения их в одну таблицу, использует в работе функцию mail\_preparation.

Файл ./pages/1\_Автоподгрузка\_объявлений.py:

* импортирует update\_notifications
* при нажатии кнопки “Подгрузить новые объявления в систему” запускает процесс парсинга



### Первичное наполнение системы данными:

Для первичного наполнения системы данными использовались выгрузки с ЦИАНА.

1600 объявлений (циан дает возможность в экселе выгрузить 200 объявлений, сделана 8 раз выгрузка по фильтрам: студия, 1к-квартира, 2к-квартира и тд), эти объявления имеют в себе информацию в том числе о размере кухонь, с помощью Pandas адаптированы к тому же формату, к которому приводятся объявления, полученные парсингом из почты, и сейчас используются при работе сервиса.

Итого на текущий момент в системе более 2,5 тыс квартир из открытых источников с учетом постоянной подгрузки объявлений с Яндекс.недвижимости. Все объявления взяты после 24.10.2022г.

## 2. Подбор аналогов для эталона

### Схема

Эталон (строка из таблицы пула) + координаты

Настройки фильтров на шаге 2 (радиус, отклонение по кв.м., ….)

Модуль поиска аналогов

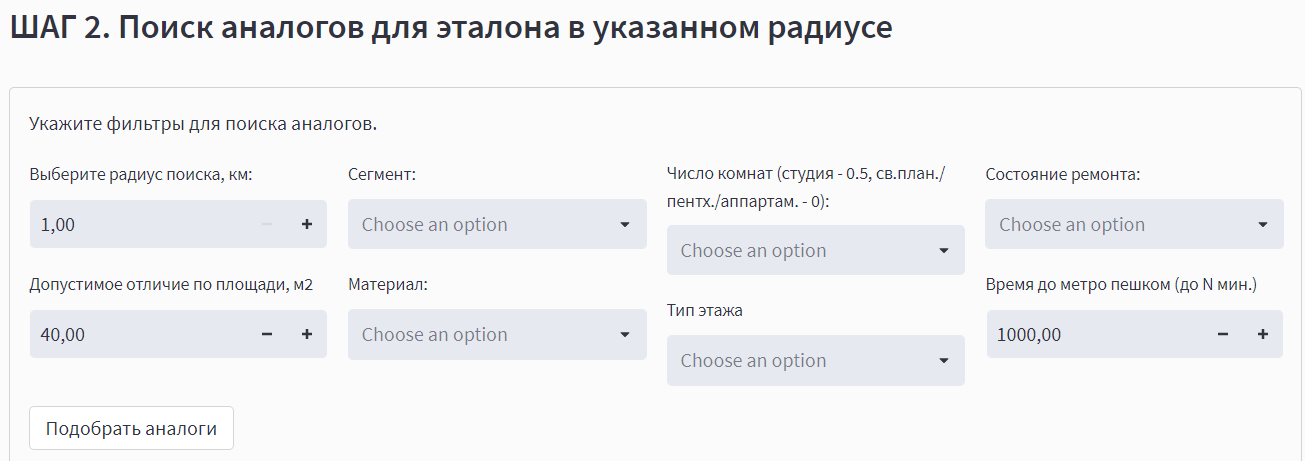
подбор аналогов из общего датасета по фильтрам в нужном радиусе

вывод таблицы аналогов

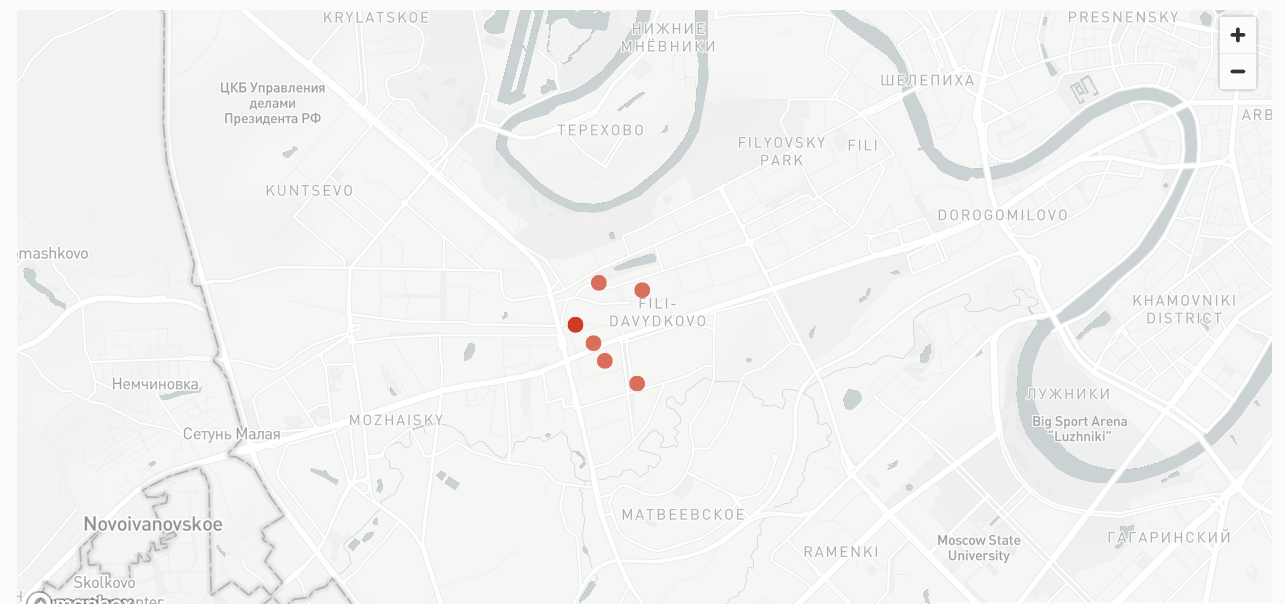
вывод точек на карте

### Общий алгоритм работы

1. Пользователь на Шаге 2 настраивает фильтры: радиус поиска, отклонение по площади, тип этажа и т.д. Если в поле пользователь не выбрал ничего, то при поиске это поле не учитывается.

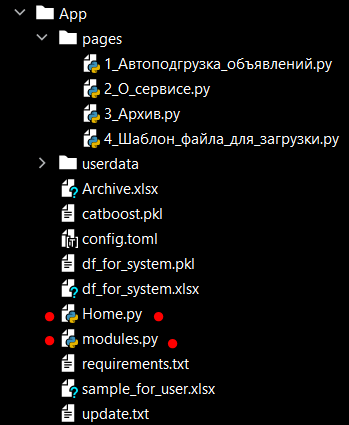


1. При клике “подобрать аналоги”, сервис вызывает функцию find\_analogs, которая из базы объявлений выбирает подходящие варианты.
2. сервис выводит на экране список подобранных аналогов со всеми их параметрами (таблицу можно проматывать вправо)
3. сервис выводит по таблице карту с нанесенными на нее точками - координаты аналогов и эталона. Более темные точки означают, что по этому адресу расположено сразу несколько квартир из подобранных аналогов. Карта интерактивная, можно скроллить (или приближать плюсиком), перемещать в разные стороны мышью.



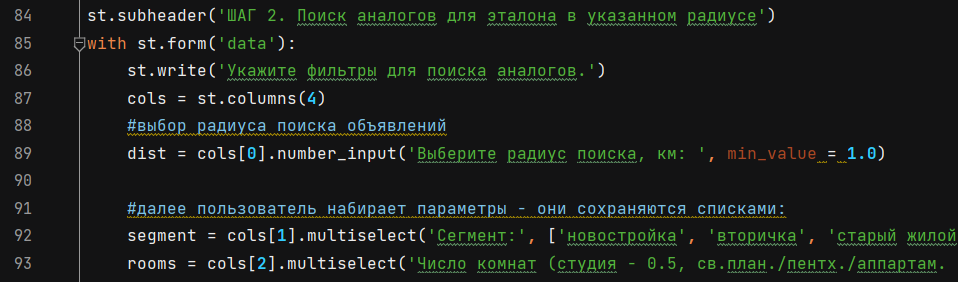
#### 

### Где найти в коде

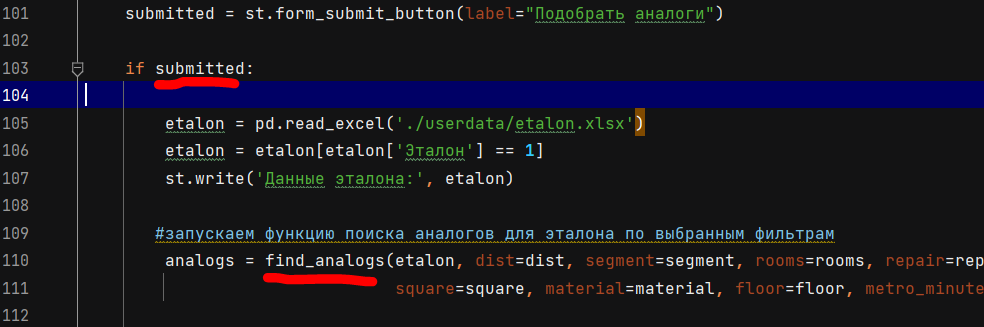


Файл Home.py:

* настройка фильтров на шаге 2:



* запуск поиска аналогов по нажатию кнопки “Подобрать аналоги”



Файл modules.py:

* функция find\_analogs(...) - отбирает подходящие объявления из общего списка

## 3. Расчет стоимости эталона и остальных квартир из пула

### Схема

Эталон

Пул

Аналоги

список id аналогов для исключения из расчета

Расчет стоимости

ФИО

Дополнение списка аналогов и эталона спец. параметрами для расчета корректировок

Расчет корректировок и удельных цен аналогов

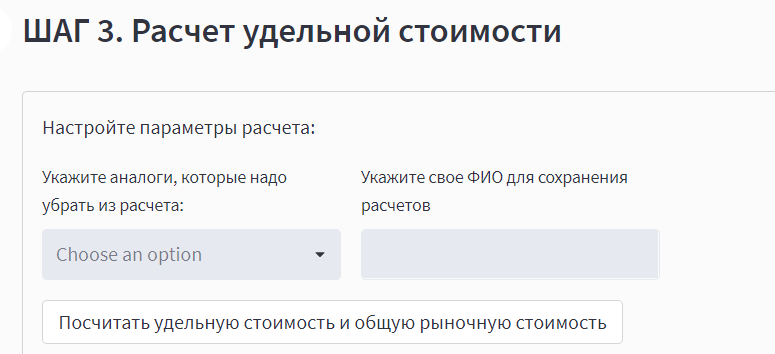
Расчет корректировок и удельных цен объектов пула

Расчет цен объектов пула с помощью ИИ (доп.оценка)

Вывод итоговых таблиц на экран: каждый объект имеет 2 разных оценки - по ТЗ и по расчетам ИИ.

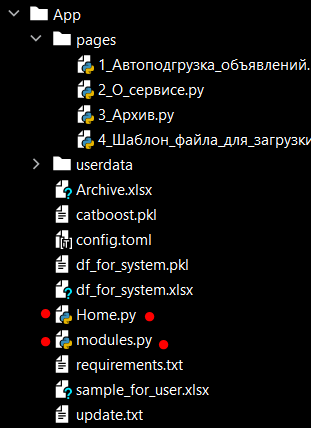
### Общий алгоритм работы

1. После шага 2 в системе есть список аналогов и эталон.
2. На шаге 3 пользователь выбирает номера аналогов, которые хочет исключить из расчета и указывает свое ФИО для будущей таблицы.



1. по нажатию кнопки “Посчитать удельную стоимость и общую рыночную стоимость” запускается расчет:
   1. Сервис отсекает отмеченные пользователем элементы списка аналогов
   2. К новому списку добавляются параметры, влияющие на расчет корректировок (диапазон площади квартиры, кухни и тд по таблицам ТЗ).
   3. далее для каждого из аналогов система считает корректировки и удельную цену согласно ТЗ - вызывает функцию count\_analog\_unitprice, собирает все расчеты по аналогам в общую таблицу.
   4. считается средняя удельная цена аналогов и добавляется в таблицу эталона как удельная цена
   5. далее, аналогично, для каждого элемента пула запускается функция для расчета удельных цен всего пула - count\_pool\_unitprice. И все расчеты собираются в одну таблицу с элементами пула, корректировками, удельными ценами и общими ценами.
   6. В таблицы с аналогами и пулом добавляются данные о ФИО и дате расчета.
   7. Запускается ML-алгоритм для альтернативного расчета стоимости квартир пула искусственным интеллектом Catboost, обученным заранее.
      1. формируется таблица на основе данных пула с другими обозначениями полей и значениями категориальных признаков для того, чтобы ИИ смог эти данные понять.
      2. запускается модель ИИ catboost.pkl и выдает прогноз стоимости квартир пула.
      3. прогноз ИИ записывается в общую таблицу с пулом данных, корректировками, расчетами по ТЗ., в последнем столбце ‘ML\_predict\_price’
   8. Система выводит на экран информацию о расчетах (таблицы можно проматывать вправо, чтобы видеть все корректировки и расчеты.

### Где найти в коде

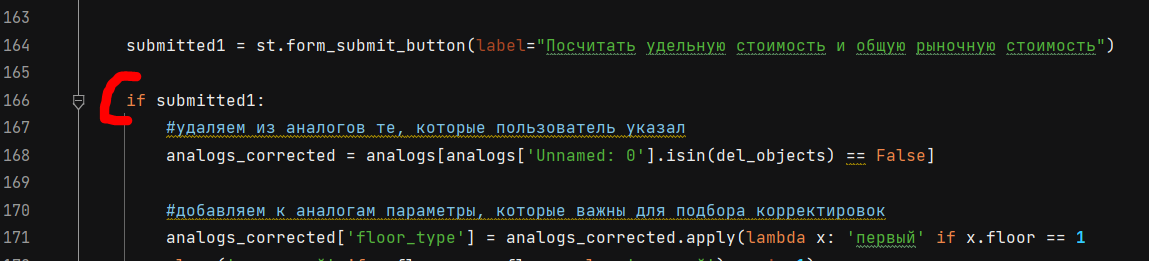


Файл modules.py:

* функция count\_analog\_unitprice - считает корректировки для аналогов и удельные цены
* Функция count\_pool\_unitprice - считает корректировки для объектов пула и удельные цены

Файл Home.py:

* Запуск всего процесса по вышеприведенной схеме после нажатия кнопки:



## 4. Machine Learning

### Схема работы

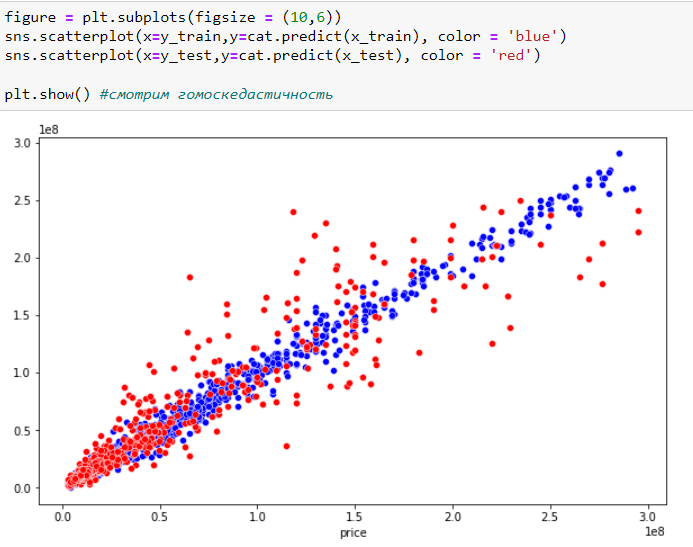
На шаге 3 после расчета стоимостей по ТЗ запускается дополнительный расчет силами ИИ.

1. данные пула адаптируются к удобному для ИИ виду.
2. модель catboost.plk делает прогноз стоимостей квартир пула
3. в таблицу с пулом данных, последним столбиком добавляется оценка рыночной стоимости, рассчитанная моделью. Если промотать таблицу вправо - можно ее увидеть рядом с остальными расчетами.

### О модели

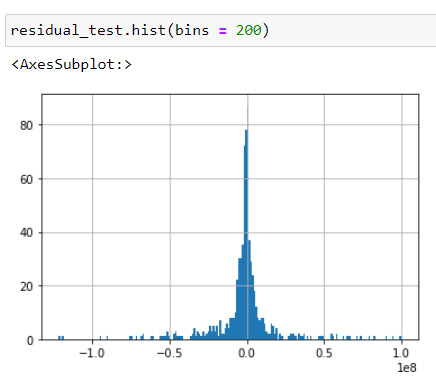
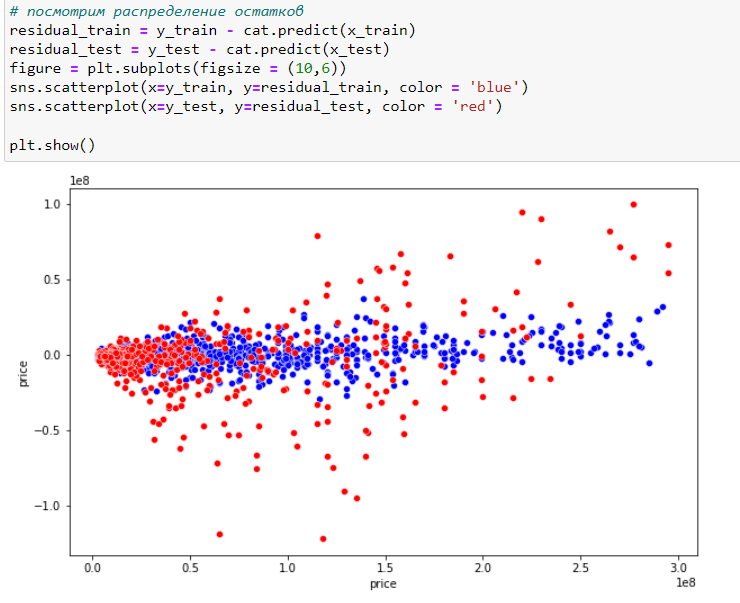
1. Модель обучена на имеющихся данных (более 2300 объявлений стоимостью до 300 млн в г.Мск), собранных в период с 24.10 по 04.11.2022 г.
2. Качество модели: r2-score = 86% на тестовой выборке, и 98% на обучающей выборке. То есть, для новых данных можно ожидать точность около 86%.
3. Пайплайн для регулярного переобучения модели в сервисе по причине нехватки времени не сделан. Но можно достаточно быстро его добавить - по аналогии с автоподгрузкой объявлений можно сделать страницу с переобучением модели, где будет по нажатию кнопки запускаться пайплайн переобучения модели на более свежих данных, которые скопились в сервисе. И свежеобученная модель будет сохраняться в формате pkl с тем же именем catboost.pkl.
4. Модель Catboost выбрана в сравнении с линейной регрессией, регрессией LASSO(L1-регуляризация) и Ridge-регрессией (L2-регуляризация) как показавшая лучшую метрику R2-score (на ~12% выше на тестовой выборке), и наиболее приличный график, отражающий уровень гомоскедастичности (соотношение реальных данных на тестовой выборке и прогнозных).

Гомоскедастичность (см. график): видим, что дисперсия остатков довольно стабильна, хотя есть несколько выбросов по середине. Для тестовой выборки (красный цвет), дисперсия больше, чем для тренировочной, но в целом можно говорить о ее постоянстве, что важно для оценки качества модели. Для других моделей результаты говорили скорее о гетероскедастичности, т.к. их графики расходились в ширину довольно сильно, не показывали стабильности по дисперсии.



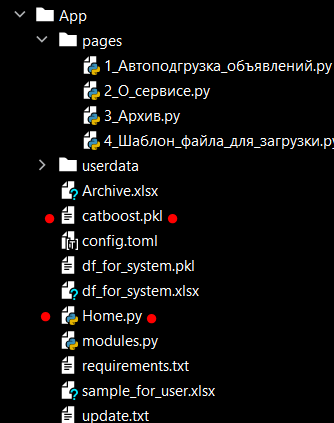
Также видим, что модель для квартир стоимостью до 100 млн.руб. ведет себя намного стабильнее и качественнее, т.к. в этом сегменте намного больше выборка для обучения и тестов. Более 2000 квартир выборки - имеют стоимость до 100 млн.

Распределение остатков близко к нормальному. Правый график - на тестовой выборке.



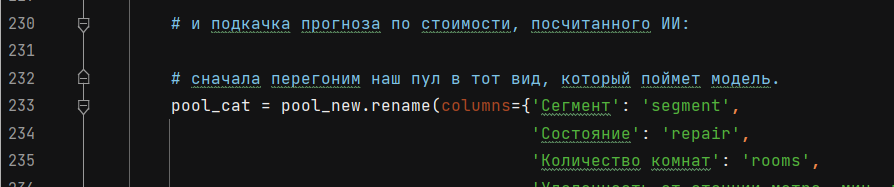
На таких показателях качества была выбрана модель Catboost. Другие алгоритмы показали себя намного хуже.

### Где найти в коде и среди рабочих файлов

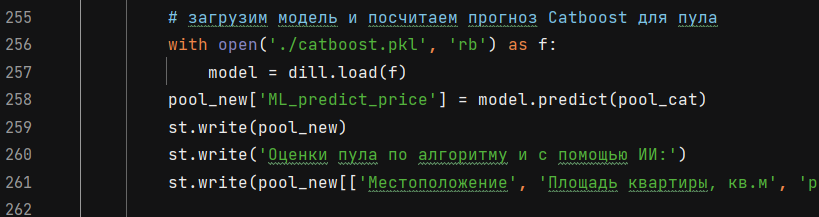


В файле Home.py:

* На шаге 3 адаптация данных:



* потом обращение к модели и прогноз:



# Инструкция для страницы Home (остальные - очевидные).

На главной странице Home - функционал основного алгоритма.

1. Пользователь заходит на сервис <http://178.170.192.124:8501/>
2. Сначала надо выполнить Шаг1:
   1. загружаем файл (соответствующий шаблону, который можно скачать в разделе Шаблон файла для загрузки. В файле первый столбик обязательно “Эталон”, в нем одна единица, остальные - нули. Названия полей должны быть, как в шаблоне.
   2. Появится кнопка “ Найти координаты эталона” - жмем. Сервис напишет, что координаты найдены и покажет на карте, либо скажет, что не нашел и даст рекомендации.
3. Шаг 2.
   1. выбираем фильтры для поиска аналогов. радиус можно указать от 1 км. Отличие по площади по умолчанию 40 м, это довольно мало для 1 км на текущей базе в 2,5 тыс объявлений, можно взять побольше, или радиус увеличить.
   2. жмем “подобрать аналоги” - видим список, сколько найдено, изучаем, скроллим вправо, смотрим на карте (можно скроллом удалить, чтоб побольше территорию увидеть и улицы, сдвинуть карту вправо-влево-вверх-вниз мышью).
   3. Если аналогов мало - меняем фильтры и жмем на ту же кнопку еще раз, пока не устроит результат по их числу или качеству.
4. Шаг 3.
   1. Указываем какие аналоги лишние в списке номерами (номера есть в таблице аналогов на шаге 2)
   2. Указываем свое имя для сохранения в файл.
   3. жмем “посчитать удельную и общую рыночную стоимость”.
   4. получаем 5 таблиц:
      1. аналоги и корректировки + перерасчет их удельных цен (промотать вправо таблицу)
      2. эталон с расчетом удельной цены и общей
      3. пул с корректировками, удельными ценами, общими ценами и оценкой ИИ - надо промотать таблицы вправо, чтоб это все увидеть.
      4. Краткая таблица: Оценки пула по алгоритму и с помощью ИИ
      5. Если все устраивает - скачиваем файл. Если не устраивает, возвращаемся на шаги выше, корректируем фильтры.

# Ограничения

1. В текущей версии сервис (MVP) не рассчитан на многопользовательское применение. Для масштабирования на нескольких пользователей (возможность корректно обрабатывать несколько запросов разных пользователей в 1 момент) необходимы доработки. Если в текущей версии одновременно к сервису подключатся несколько пользователей, сервис может выдать ошибку или перепутать данные одного пользователя с другим. Также в текущей версии нет системы разграничения ролей и доступов, паролей и логинов. Для этого нужны доработки.
2. В текущей версии нет возможности прямо в системе внести изменения в рассчитанные автоматически корректировки по таблицам из приложения 2 в ТЗ. Можно исключать объекты из расчета точечно, можно гибко настраивать фильтры подбора аналогов, но в сами корректировки внести изменения в рамках работы сервиса в MVP нельзя. Но при загрузке файла все корректировки можно увидеть в нем и при желании провести некоторый пересчет вручную.
3. Ограничения дизайна. Фронт сделан с помощью streamlit, которая позволяет кастомизировать внешний вид сервиса в рамках ее шаблонных минималистичных решений (ключевые цвета можно менять - фон, цвет кнопок, вторичный цвет для полей, некрупные корректировки расположения элементов, иконочки, фавикон).
4. Подгрузка новых объявлений происходит по нажатию кнопки. Для настройки автоподгрузки в определенный час (ночью например) необходимы доработки. Рекомендуется хотя бы раз в сутки запускать процесс подгрузки, т.к. за сутки скапливается довольно много данных, которые требуют времени на расшифровку и подгрузку в сервис.
5. Выбранный способ парсинга данных о квартирах не может подгружать данные о площади кухонь. В Яндекс.Недвижимости нельзя настроить диапазоны площадей кухонь, там можно указать лишь “от стольки-то метров”, что для наших задач не информативно, т.к. от 6 метров - это может быть и 25 метров. Поэтому, система сама предполагает при расчете корректировок, что кухня - это около 20% от общей площади квартиры. Но в итоговой таблице видно, что сервис данные не подгрузил.
6. Также данный способ парсинга данных вытаскивает только те квартиры, у которых есть балкон или лоджия. Все фильтры настроены таким образом, что в них попадают только квартиры с балконом или лоджией. Такое решение принято в связи с тем, что при установке фильтров нет возможности выбрать квартиры точно без балкона. Там можно выбрать балкон, лоджию, или неважно. Вычленить квартиры 100% без балкона и лоджии невозможно. Поэтому фильтры настроены так, чтобы в объявлениях стопроцентно присутствовали или балкон или лоджия, что позволит точно знать при расчете корректировок, какой коэффициент брать. Будет уверенность, что в аналогах это поле верно заполнено. Хотя и будут отсечены все варианты без лоджий и балконов.
7. Модель искусственного интеллекта для корректной работы сервиса должна регулярно переобучаться, т.к. цены меняются. Хотя бы раз в месяц. В текущей версии сервиса не настроен пайплайн переобучения по причине нехватки времени.. Но это можно настроить в будущем. Пайплайн может также запускаться по нажатию кнопки (или с определенной периодичностью) - адаптировать данные из общей таблицы объявлений, потом на них обучаться и сохранять свежую модель.
8. Почта для автоподгрузки объявлений заведена специально для проекта, но сейчас она привязана к номеру телефона участника команды. Для передачи сервиса в эксплуатацию нужно поменять номер, привязанный к почте на номер кого-то из сотрудников компании-заказчика. Также рассылки на эту почту идут с аккаунта участника команды, привязанного к телефону, нужно поменять телефон в аккаунте и прочие данные, либо завести отдельный аккаунт, сделать на нем аналогичные подписки на почту, а в текущем аккаунте их отключить.

# Команда NskTeam

1. **Плотникова Александра**, [sasa2111@mail.ru](mailto:sasa2111@mail.ru), телеграм @\*\*\*\*, +7 \*\*\* \*\*\* \*\*\*\*
   * Код на питоне (фронт, бэк, контейнеризация),
     1. проектирование и создание сервиса,
     2. реализация алгоритма расчета по ТЗ к задаче 6,
     3. ML - сбор данных для датасета, обработка, анализ данных, подбор лучшего алгоритма на основе качества прогнозирования - catboost,
     4. упаковка в контейнер докер: написание докерфайла, разворачивание образа, создание контейнера.
     5. идея метода автоподгрузки данных и его реализация (с помощью рассылок и выделенной почты),
   * выгрузка образа и запуск контейнера на сберклауд для демонстрации прототипа,
   * тестирование и отладка,
   * составление документации,
   * составление презентации.
2. **Белова Надежда**, почта \*\*\*\*\*, 8 9\*\*\*\*\*(телеграм привязан к номеру)
   * генерация идей
   * сбор данных,
   * тестирование готового сервиса,
   * поиск статей по парсингу для изучения вопроса.