МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ГЕОДЕЗИИ И КАРТОГРАФИИ

(МИИГАиК)

Кафедра ВТиАОАИ

(КАФЕДРА ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И АВТОМАТИЗИРОВАННОЙ ОБРАБОТКИ АЭРОКОСМИЧЕСКОЙ ИНФОРМАЦИИ)

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

**Интерактивная карта с анализом возможных точек размещения хостелов**

**Выполнил:** студент ФПКиФ группы ИСиТ III-1б

Иванов Андрей Дмитриевич

**Преподаватель:** Груздев Сергей Сергеевич

Оглавление

[Введение 3](#_Toc12994137)

[Актуальность работы 3](#_Toc12994138)

[Используемые инструменты 3](#_Toc12994139)

[Источники данных 3](#_Toc12994140)

[Обзор проекта 4](#_Toc12994141)

[Алгоритм работы 6](#_Toc12994142)

# 

# Введение

Работа заключается в создании интерактивной карты, которая позволит принять решение об открытии объекта типа “хостел” предпринимателям, на основе данных о существующих конкурентах, возможных объектах недвижимости со всей необходимой информацией (метраж, цена аренды), а также данных о существующих конкурентах и изолиниях пешей доступности различных участков от метро.

# Актуальность работы

Актуальность работы заключается в том, что в связи с новым законом, который выходит в ноябре 2019 года, будет запрещено открытие хостелов в жилых домах (кроме первых этажей), что повлечет за собой необходимость в поиске новых помещений для ведения своего бизнеса для многих предпринимателей. Данная интерактивная карта создана с учетом этого закона, все точки с объявлениями на карте отобраны таким образом, что в данных объектах недвижимости можно открывать хостелы, также учтена информация о существующих конкурентах и доступности метро.

# Используемые инструменты

Для разработки интерактивной карты использовался JavaScript, а также библиотека leaflet.js (https://leafletjs.com), язык разметки HTML и стилей для него CSS. В качестве ГИС-сервера и тайл-сервера подложки карты были использованы сервисы HERE (<https://xyz.here.com>). Для обработки данных использовался пакет QGIS.

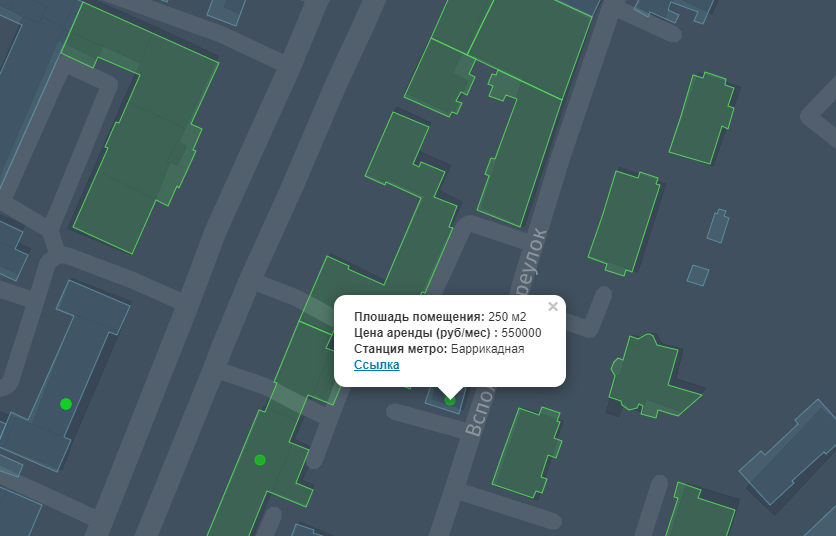
# Источники данных

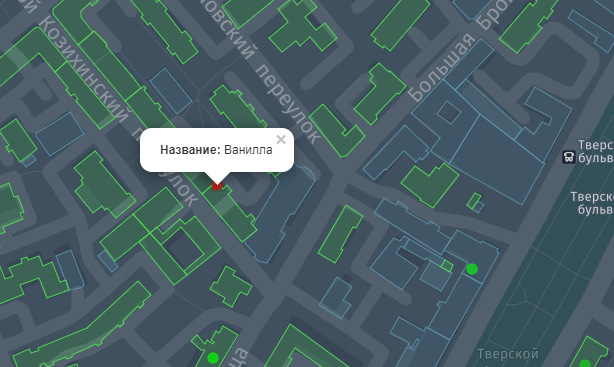
* Данные о расположении хостелов на территории Москвы – портал открытых данных города Москвы - <https://data.mos.ru>
* Данные (геометрия и местоположение) строений на территории Москвы – портал открытых данных города Москвы - <https://data.mos.ru>
* Данные о статусе помещений (жилое/нежилое) – открытые данные Реформа ЖКХ - <https://www.reformagkh.ru/opendata>
* Расчёт изолиний пешей доступности от заданных точек (в нашем случае станций метро) – сервисы HERE - <https://xyz.here.com>

# Обзор проекта

1. **Общий вид**



1. **Информация при клике на объект недвижимости**
2. **Информация о хостелах конкурентах**

****

1. **Изолинии доступности от метро**

****

# Алгоритм работы

Создаем саму карту с необходимыми нам параметрами (приближение, позиция) в окне браузера, используя Leaflet, подложку для карты берем с тайл-сервера HERE.

let M = {

'AppId': '63vR5UbOum9b6qxMqHQw',

'AppCode': 'bx\_Fo6ynpxHtnkl6MND3sg',

'Lat': 55.751,

'Lng': 37.620,

'Zoom': 12,

'TileLayerUrl': {}

}

M.TileLayerUrl = (style) => `https://2.base.maps.api.here.com/maptile/2.1/maptile/newest/${style}/{z}/{x}/{y}/512/png8?app\_id=${M.AppId}&app\_code=${M.AppCode}&ppi=320&lg=rus`

const map = L.map('map').setView([M.Lat, M.Lng], M.Zoom)

L.tileLayer(M.TileLayerUrl('reduced.night')).addTo(map)

Данные, после обработки в Qgis (Геокодирование, очистка от ненужных полей), были переконвертированы в GeoJson, а затем залиты на гис-сервер. Доступ к данным осуществляется по api, с использованием уникального access token каждого набора данных с помощью функции fetch:

fetch(hostel\_data).then(res => {

res.json().then(data => {

hostel\_layer = L.circleMarker(data, {

style: geojsonMarkerOptions,

onEachFeature: onEachFeature

})

hostels = data;

})

})

Каждый набор данных сохраняется в слой для того, чтобы в дальнейшем можно было их переключать, а также стилизуется.

Получаем изолинии пешей доступности при помощи нескольких запросов к API, в теле запроса мы отправляем долготу и широту точки (станции метро), а также время (в секундах), которое показывает необходимую нам пешую доступность. В ответе мы получаем геометрию изолиний.

function drawLines() {

for (var i = 0; i < metro.length; i++) {

var currentFeature = metro[i];

var curLong = currentFeature.Longitude;

var curLat = currentFeature.Latitude

fetch(gatherIsolines(curLat, curLong, 600)).then(res => {

res.json().then(data => {

let splitArray = (array, part) => {

var tmp = []

for (var i = 0; i < array.length; i += part) {

tmp.push(array.slice(i, i + part));

}

return tmp;

}

var polygon = L.polygon(splitArray(data.response.isoline[0].component[0].shape, 2), {

color: '#fff'

})

polyLayers.push(polygon)

polygon = 0;

})

for(layer of polyLayers) {

drawnItems.addLayer(layer);

}

})

}

На нужные нам фичи (недвижимость, хостелы) вешаем event listener на клик, для отображения необходимой нам информации

function onCianFeature(feature, layer) {

layer.bindPopup(`<strong> Плошадь помещения: </strong> ${feature.properties.area} м2<br/>

<strong> Цена аренды (руб/мес) : </strong> ${feature.properties.price} <br/>

<strong> Станция метро: </strong> ${feature.properties.metro}<br/>

<strong> <a href=${feature.properties.link}>Ссылка</a>`

)

layer.on('click', function(e) {

});

}

Далее создаем всю необходимую разметку для отображения данных, используя HTML, вешаем event listener на чекбоксы для включения и выключения слоев.