БУ ВО ХАНТЫ-МАНСИЙСКОГО АВТОНОМНОГО ОКРУГА - ЮГРЫ «СУРГУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Кафедра информатики и вычислительной техники

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**Информационная система для сбора данных используемых при выборе места постройки заведения общественного питания «CE-helper»**

Направление: 09.03.02 «Информационные системы и технологии»

Профиль: «Информационные системы и технологии»

Квалификация: бакалавр

|  |  |
| --- | --- |
|  | Студент гр. 607-91:  Хисматов Эльмир Зиннурович \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Научный руководитель:  к.ф.-м.н., доцент кафедры ИВТ  Лысенкова Светлана Александровна \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Допущен к защите:  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.  зав. кафедры ИВТ, к.т.н., доцент  Федоров Дмитрий Алексеевич \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Сургут, 2023

**РЕФЕРАТ**

Работа содержит 43 страницы, 14 рисунков, 4 таблицы, 23 используемых источника, 6 приложений.

**Информационная система для сбора данных используемых при выборе места постройки заведения общественного питания «CE-helper»**

Ключевые слова: веб-приложение, заведение общественного питания, сайт, модель предметной области, BPMN диаграмма, use-case диаграмма, спецификация прецедентов, диаграмма классов, диаграмма компонентов, информационная система.

Актуальность работы заключается в том, что разрабатываемая система может помочь в следующих аспектах: помощь в принятии решений, экономия времени и ресурсов, улучшение точности анализа, анализ конкурентной среды, повышение качества обслуживания клиентов.

Целью выпускной квалификационной работы является проектирование и разработка информационной системы для сбора данных, которые помогут более быстро и точно определить наилучшее место для постройки заведения общественного питания, а также сэкономить время и ресурсы.

В ходе данной работы была изучена предметная область, рассмотрены аналоги разрабатываемой информационной системы, поставлены цель и задачи разработки системы, осуществлено проектирование системы, разработана информационная система.

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

|  |  |  |
| --- | --- | --- |
| Термин | Расшифровка | Синоним |
| API | Application Programming  Interface | Интерфейс прикладного программирования |
| ИС | Информационная система |  |
| CE | Catering Establishment | Заведение общественного питания |
| МПО | модель предметной области |  |
| ПО | Программное обеспечение | Софт, железо, программа |
| UI | User Interface | Пользовательский интерфейс |
| BPMN | Business Process Model and Notation | Модель бизнес-процесса и нотация |
| БД | База данных |  |
| CSS | Cascading style sheets | каскадные таблицы стилей |
| HTML | Hypertext markup protocol | протокол передачи гипертекста |
| UML | unified modeling language | универсальный язык моделирования |
| SQL | Structured Query Language | Язык запросов к базам данных |
| ПК | Персональный компьютер |  |
| JSON | JavaScript Object Notation | Json файл, текстовый формат обмена данными |

**ОГЛАВЛЕНИЕ**

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ 2](#_Toc137385856)

[ВВЕДЕНИЕ 6](#_Toc137385857)

[ГЛАВА 1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И ОБЗОР АНАЛОГОВ 8](#_Toc137385858)

[1.2. Обзор аналогов 10](#_Toc137385859)

[1.2.1. Google Maps 10](#_Toc137385860)

[1.2.2. Яндекс Карты 11](#_Toc137385861)

[1.2.3. 2 GIS 12](#_Toc137385862)

[ГЛАВА 2. ТРЕБОВАНИЯ К СИСТЕМЕ 15](#_Toc137385863)

[2.1. Функциональные требования к информационной системе 15](#_Toc137385864)

[2.2. Требования к информационному обеспечению информационной системы 15](#_Toc137385865)

[2.3. Требования к лингвистическому обеспечению системы 15](#_Toc137385866)

[2.4. Требования к программному обеспечению системы 16](#_Toc137385867)

[2.5. Требования к техническому обеспечению системы 17](#_Toc137385868)

[2.6. Требования к методическому обеспечению 18](#_Toc137385869)

[2.7. Требования к математическому обеспечению 18](#_Toc137385870)

[ГЛАВА 3. ПРОЕКТИРОВАНИЕ СИСТЕМЫ 20](#_Toc137385871)

[3.1. Таблица SWOT-анализа 20](#_Toc137385872)

[3.2. Концептуальная модель предметной области 21](#_Toc137385873)

[3.3. Use case диаграмма 23](#_Toc137385874)

[3.4. Описание прецедентов 24](#_Toc137385875)

[3.5. Моделирование процесса сбора данных и получения процента 26](#_Toc137385876)

[3.6. Диаграмма классов 27](#_Toc137385877)

[3.7. Диаграмма компонентов 32](#_Toc137385878)

[ГЛАВА 4. РЕАЛИЗАЦИЯ СИСТЕМЫ. 33](#_Toc137385879)

[4.1. Страницы системы 33](#_Toc137385880)

[4.2. Основные функции системы 36](#_Toc137385881)

[4.3. Запрос через API и запрос к базе данных. 38](#_Toc137385882)

[ЗАКЛЮЧЕНИЕ 39](#_Toc137385883)

[СПИСОК ЛИТЕРАТУРЫ 41](#_Toc137385884)

[ПРИЛОЖЕНИЕ 1. Модуль event\_reverse\_geocode.js 44](#_Toc137385885)

[ПРИЛОЖЕНИЕ 2. Модуль init.py 48](#_Toc137385886)

[ПРИЛОЖЕНИЕ 3. Модуль main.py 49](#_Toc137385887)

[ПРИЛОЖЕНИЕ 4. Модуль zapros\_people.py 50](#_Toc137385888)

[ПРИЛОЖЕНИЕ 5. Модуль coord\_repository.py 51](#_Toc137385889)

[ПРИЛОЖЕНИЕ 6. Модуль menu.js 53](#_Toc137385890)

# ВВЕДЕНИЕ

За последние несколько лет количество заведений общественного питания увеличилось на заметное значение [1] и вынуждает участников ресторанного бизнеса применять различные средства, которые делятся как на технические и аппаратурные, так на и людские.

Все это приводит к увеличению затрат при выборе места постройки заведения общественного питания. Разрабатываемая информационная система призвана оптимизировать этот процесс, позволяя быстро собирать основные необходимые данные и вычисляя предполагаемый процент успешности нового заведения. Но чтобы ускорить этот процесс нужны современные средства и методы. Поэтому было решено в качестве основы системы выбрать систему API от Яндекс Карты. Данный вид системы позволяет создать у себя карту необходимой местности и через выданный ключ обрабатывать запросы, выдавая запрашиваемые данные. Все данные будут использоваться в формуле, которая подсчитает примерный процент успешности нового заведения общественного питания.

Основными пользователями системы будут владельцы бизнеса в области общественного питания. Ведь именно им нужно быстро и оперативно собирать необходимую информацию, чтобы лучше понимать области города, которые могут подойти для постройки нового заведения. И разрабатываемая информационная система сможет в этом помочь.

Целью данной работы является проектирование и разработка информационной системы для сбора информации, используемой при выборе места постройки заведения общественного питания.

Чтобы разработать правильно работающую систему, ставятся следующие задачи, которые определяют шаги по разработке данной системы:

1. Изучить предметную область и построить её модель;
2. Определить аналоги и провести их анализ;
3. Описать все виды обеспечения;
4. Спроектировать информационную систему. Чтобы правильно это реализовать, необходимо поставить следующие задачи:
   1. Составить таблицу SWOT анализа;
   2. Построить диаграмму вариантов использования use case;
   3. Описать возможные прецеденты работы с программой;
   4. Реализовать поведенческую модель в нотации BPMN;
   5. Построить диаграммы классов и компонентов.
5. Разработать информационную систему. Для правильной реализации составляется следующий список задач:
   1. Определить все необходимые модули программы;
   2. Разработать код;
   3. Разработать сайт под систему;
   4. Реализовать на сайте работу всех модулей.

В качестве программного средства проектирования диаграмм используется редактор диаграмм и блок-схем diagrams.net.

Целью разрабатываемой информационной системы является ускорение процесса сбора информации, используемой при выборе места для постройки заведения общественного питания.

Апробация работы. Основные положения данной работы докладывались на IX Всероссийской конференции молодых ученых «Наука и инновации XXI века» (г. Сургут, 2023 г.), а также на XXVII Открытой региональной студенческой научной конференции имени Г.И. Назина «Наука 60-й параллели» (г. Сургут, 2023 г.).

# ГЛАВА 1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ И ОБЗОР АНАЛОГОВ

Предприятие общественного питания — сферу гостеприимства и общественного питания, которая охватывает различные виды предприятий, где можно приобрести готовую еду и напитки на месте или с собой.

Заведения общественного питания могут быть представлены различными форматами: кафе, рестораны, бары, фаст-фуды, кофейни, пекарни, кондитерские и т.д. Каждый из них имеет свои особенности и функциональные возможности, а также связан с определенной культурой питания и общественными традициями.

Организация работы заведений общественного питания требует от владельцев и управляющих предприятий знаний и навыков в области гастрономии, клиентоориентированности, управления персоналом, финансового планирования и маркетинга. Кроме того, они должны следить за соблюдением требований по безопасности пищевых продуктов, санитарно-гигиенических норм и правил обслуживания.

Современный рынок общепита характеризуется высокой конкуренцией и постоянным изменением вкусовых предпочтений потребителей, что требует от участников отрасли быстрой адаптации к новым трендам и инновационным технологиям.

Областью использования информационной системы «CE-helper» является заведения общественного питания, такие, как кафе, столовые, рестораны, бары. Данные заведения играют заметную роль в организации отдыха населения. Посетители приходят в кафе или ресторан не только для того чтобы вкусно поесть, но и для того, чтобы провести деловую встречу, приятный вечер в кругу друзей, отметить какое-либо событие в личной жизни, и просто отдохнуть. Общественное питание, являясь составной частью социальной сферы, в силу своей специфики затрагивает интересы практически всех социальных групп населения. В рыночных условиях общественное питание становится все более дифференцированным, создает различные формы предоставления своих услуг, адаптированных к возникающим в потреблении этих услуг инновациям.

# 1.2. Обзор аналогов

Как таковых прямых аналогов у разрабатываемой ИС не было найдено, однако имеются косвенные аналоги. Проведем их обзор.

## 1.2.1. Google Maps

Google Maps [2] — это онлайн-карта, разработанная компанией Google, которая позволяет пользователям искать места и получать информацию о них, создавать маршруты и планировать поездки. С помощью Google Maps можно найти местоположение организаций и учреждений, таких как рестораны, гостиницы, магазины, банки и медицинские учреждения, а также искать общественный транспорт, автомобильные маршруты и пешеходные маршруты.

Google Maps использует технологию геолокации, чтобы определять местоположение пользователя и показывать ему карту его окрестностей. Кроме того, на карте можно увидеть информацию о трафике, оценки и отзывы пользователей об организациях, фотографии и видео, а также другую полезную информацию, которая может быть полезна при планировании поездок.

Google Maps также предоставляет функции для создания и редактирования карт, добавления мест и организаций, а также для обмена информацией с другими пользователями. Кроме того, Google Maps может быть интегрирована с другими сервисами Google, такими как Google Street View, Google Earth и Google My Business.

Google Maps является одним из самых популярных сервисов компании Google и широко используется во всем мире для планирования поездок, поиска мест и организаций, а также для обмена информацией с другими пользователями.

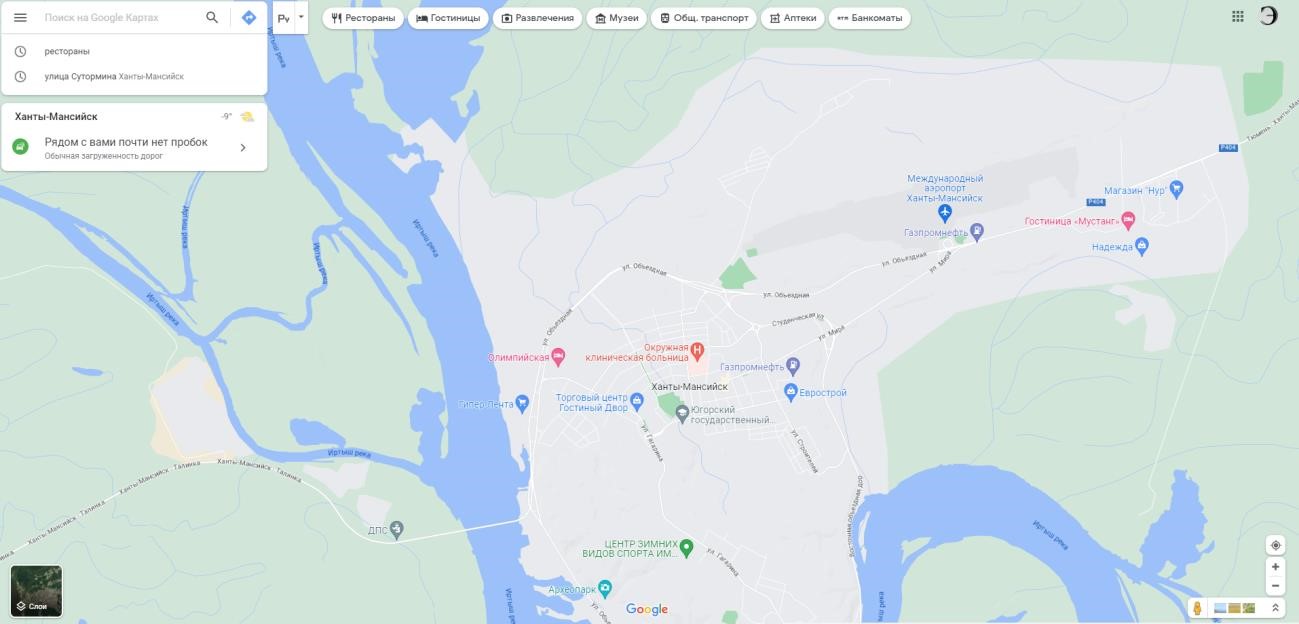


Рис. 1. Google Maps

## 1.2.2. Яндекс Карты

Яндекс Карты [3] – это онлайн-карта, разработанная компанией Яндекс, которая позволяет пользователям искать места и получать информацию о них, создавать маршруты и планировать поездки. С помощью Яндекс Карт можно найти местоположение организаций и учреждений, таких как рестораны, гостиницы, магазины, банки и медицинские учреждения, а также искать общественный транспорт, автомобильные маршруты и пешеходные маршруты.

Яндекс Карты используют технологию геолокации, чтобы определять местоположение пользователя и показывать ему карту его окрестностей. Кроме того, на карте можно увидеть информацию о трафике, оценки и отзывы пользователей об организациях, фотографии и видео, а также другую полезную информацию, которая может быть полезна при планировании поездок.

Яндекс Карты также предоставляют функции для создания и редактирования карт, добавления мест и организаций, а также для обмена информацией с другими пользователями. Кроме того, Яндекс Карты могут быть интегрированы с другими сервисами Яндекса, такими как Яндекс.Навигатор, Яндекс.Погода и Яндекс.Метро.

Яндекс Карты являются одним из самых популярных сервисов компании Яндекс и широко используются в России и других странах СНГ для планирования поездок, поиска мест и организаций, а также для обмена информацией с другими пользователями.

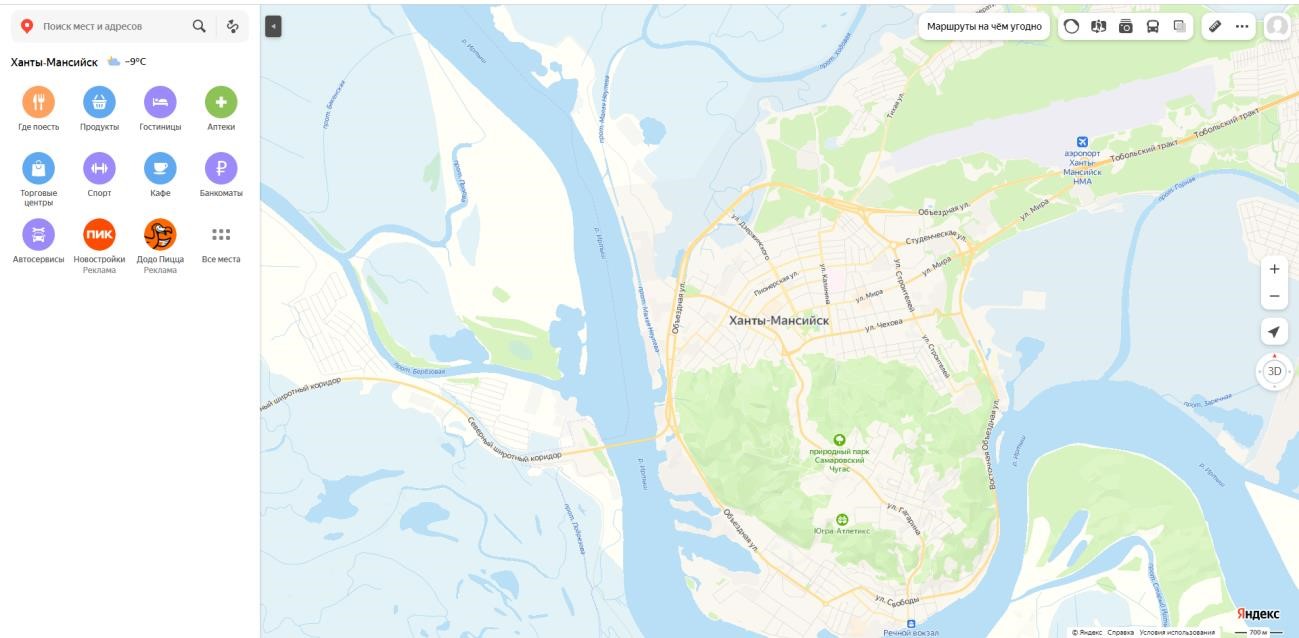


Рис. 2. Яндекс Карты

## 1.2.3. 2 GIS

2GIS [3] — это онлайн-карта и сервис поиска информации о местах и организациях, разработанный компанией DoubleGIS. 2GIS позволяет пользователям искать информацию об организациях, таких как рестораны, магазины, банки, аптеки и медицинские учреждения, а также получать информацию о государственных учреждениях и других объектах.

2GIS использует технологию геолокации, чтобы определять местоположение пользователя и показывать ему карту его окрестностей. Кроме того, на карте можно увидеть информацию о трафике, оценки и отзывы пользователей об организациях, фотографии и видео, а также другую полезную информацию, которая может быть полезна при планировании поездок.

2GIS также предоставляет функции для создания и редактирования карт, добавления мест и организаций, а также для обмена информацией с другими пользователями. Кроме того, 2GIS может быть интегрирована с другими сервисами DoubleGIS, такими как 2GIS.Маршруты и 2GIS.Недвижимость.

2GIS является одним из самых популярных сервисов поиска информации о местах и организациях в России и странах СНГ. Сервис отличается высокой точностью и детализацией карт, а также наличием обширной базы данных об организациях и местах.



Рис. 3. 2 GIS

Теперь определив список косвенных аналогов, построим таблицу сравнения с разрабатываемой ИС.

*Табл. 1. Обзор аналогов*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерий** | **CE-helper** | **Google Maps** | **Яндекс Карты** | **2 GIS** |
| Простота интерфейса | + | + | + | + |
| Фильтр по типу заведения | + | - | + | + |
| Количество людей | + | - | - | - |
| Одновременный сбор нескольких типов данных | + | - | - | - |
| Формула успешности | + | - | - | - |

Проведем анализ выбранных критериев:

Простота интерфейса – показывает насколько пользователь сможет быстро найти то что ему нужно при использовании приложения.

Фильтр по типу заведения – подразделяет поиск по предлагаемым типам заведения (например, кафе или ресторан).

Количество людей – показывает среднее количество людей, проходящих по выбранной улице за текущую дату.

Одновременный сбор нескольких типов данных – возможность собрать и показать пользователю несколько выбранных им данных на одном экране.

Формула успешности – вычисление по формуле при сборе данных процента успешности выбранного заведения общественного питания.

Определив все критерии и проанализировав их наличие в других системах и разрабатываемой системе, была составлена таблица, которая показала, что у «CE-helper» имеется больше преимуществ перед выбранными аналогами. Разрабатываемая ИС как сочетает в себе основные критерии аналогов, такие как фильтр по заведениям и простоту интерфейса. Но главное, что «CE-helper» включает в себя и остальные категории.

# ГЛАВА 2. ТРЕБОВАНИЯ К СИСТЕМЕ

## 2.1. Функциональные требования к информационной системе

К информационной системе для сбора данных предъявляются следующие функциональные требования:

* Выбор типа заведения при помощи фильтра;
* Показ основных собранных данных;
* Показ результата успешности по формуле;
* Хранение оставленных отзывов в БД.

## 2.2. Требования к информационному обеспечению информационной системы

Источниками информации для ИС является система API от Яндекс Карты и Авито.

Входными данными являются выбор места и посылаемые запросы от системы, с учетом выбора типа заведения.

Выходными данными является процент успешности заведения.

# 2.3. Требования к лингвистическому обеспечению системы

Лингвистическое обеспечение информационной системы для сбора данных используемых при постройки заведения общественного питания включает в себя следующие языки программирования и технологии:

* На этапе проектирования программного обеспечения с целью создания проектной документации применяется универсальный язык моделирования UML, на основе которого планируется разработать следующие диаграммы:
* модель предметной области,
* use-case диаграмма,
* спецификация прецендентов;
* Описание бизнес-процесса происходит на универсальном языке BPMN;
* Диаграмма классов;
* Диаграмма компонентов;
* «SQLite DB», версии 3.12.2. Это встраиваемая система управления базами данных в которой хранятся данные о количестве людей, необходимые для правильной работы системы.

# 2.4. Требования к программному обеспечению системы

Для выполнения системой заявленных функций требуется наличие дополнительного ПО.

База данных для информационной системы реализована в СУБД SQLite DB, в качестве сервера фреймворк Flask.

Требования к ПО клиентской части системы:

1. Операционная система:

* Windows 7 или более поздняя версия;
* Linux: Ubuntu 10.04 или более поздняя версия;
* Mac OS X 10.6 или более поздняя версия;

1. Браузер с поддержкой JavaScript:

* Google Chrome;
* Mozilla Firefox;
* Яндекс браузер;
* Microsoft Edge.

Требования к ПО, установленному на серверном аппаратном обеспечении:

1. Операционная система:
   * Windows 7 или более поздняя версия;
   * Linux: Ubuntu 10.04 или более поздняя версия;
   * Mac OS X 10.6 или более поздняя версия.
2. Веб-сервер:

* Flask версии 2.3.2 и выше;
* SQLite DB.

Для создания клиентской и серверной части системы используются:

* Фреймворк Flask для программирования серверной части системы, использующий язык Python.
* Visual Studio Code - текстовый редактор, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации.

## 2.5. Требования к техническому обеспечению системы

Для выполнения своих поставленных функций, система предъявляет следующие требования к ПО пользователя:

1. Операционная система: Windows 7 или выше, Linux, MacOS.
2. Процессор: с поддержкой архитектуры х86 или х64, с тактовой частотой не менее 1 ГГц.
3. Оперативная память: не менее 2 Гб.
4. Свободное место на жестком диске: не менее 500 мб.
5. Браузер: Google Chrome, Mozilla Firefox, Microsoft Edge или Safari.
6. Наличие подключения к интернету с достаточной скоростью для работы с API. Также для работы с конкретным API могут потребоваться дополнительные библиотеки или пакеты программного обеспечения, которые необходимо установить в системе.
7. Клавиатура. Механическое устройство ввода текстовой информации.
8. Мышь. Механическое устройство управления курсором и отдачи различных команд компьютеру, путем взаимодействия с различными графическими элементами интерфейса: кнопки, ползунки, контекстные меню и т.д.
9. Монитор. Устройство оперативной визуальной связи с пользователем и интерпретации результатов работы системы, поддерживает возможность вывода цветного изображения

## 2.6. Требования к методическому обеспечению

При разработке данной информационной системы и создании документации на неё, необходимо руководствоваться следующими нормативными документами:

* ГОСТ 34.602-2020. Техническое задание на создание

автоматизированной системы; [8]

* ГОСТ Р 59793–2021. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания; [9]
* ГОСТ 34.201-2020. Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплексность и обозначение документов при создании автоматизированных систем. [10]

## 2.7. Требования к математическому обеспечению

В разрабатываемой ИС используется следующая формула:

F = (k\*w1 + p\*w2 + o\*w3 + a\*w4) + 50%.

Обозначения используемых данных в формуле:

w1,w2,w3,w4 - весовые коэффициенты, k - конкуренты, p - количество людей, o - организации и постройки, a - аренда.

Обоснование используемой формулы:

Для системы была выбрана линейная формула, с использованием весовых коэффициентов, так как все получаемые данные имеют разное влияние на конечный результат.

Распределение весовых коэффициентов:

W1 для k. Берется значение (-1), так как каждый конкурент либо переманивает клиентов, либо передает их, если не может удовлетворить их потребности.

W2 для p. Берется значение (0,015) так как от количества людей зависит то, будут ли у CE посетители. Значение дробное, так как плотность населения в Сургуте составляет больше 1000 человек на квадратный километр. Коэффициент не равен (0,01), так как каждый проходящий человек в теории может стать клиентом заведения, а шанс такого события равен пятидесяти процентам, поэтому добавляется (0,005).

W3 для o. Берется значение (0,5) так как все организации привлекают людей, однако шанс того, что они станут клиентами равен пятидесяти процентам.

W4 для a. Берется значение (-0,0001) так как средняя цена аренды помещения под кафе или другое заведение составляет примерно от 1000 до 1500 рублей за квадратный метр. А площадь помещения необходимая даже для среднего размера заведения составляет примерно 100 квадратных метров, то выходит, что средняя цена аренды варьируется от 100000 рублей. Поэтому необходимо, чтобы конечный результат обработки этого показателя был адекватным в соответствии с другими данными, нужно уменьшить его до показаний других видов используемых данных.

Добавление 50% обуславливается тем, что любое новое заведение имеет 50% шанс на успех, без предварительного анализа. То есть шанс того, что постройка будет пользоваться популярностью равен шансу того, что заведение придется закрыть спустя некоторое время.

# ГЛАВА 3. ПРОЕКТИРОВАНИЕ СИСТЕМЫ

## 3.1. Таблица SWOT-анализа

SWOT-анализ – это метод комплексной оценки факторов, напрямую и косвенно влияющих на бизнес.

SWOT расшифровывается так:

Strengths – сильные стороны компании,

Weakness – слабые стороны компании, Opportunities – возможности внешней среды,

Threats – угрозы внешней среды.

В таблице 2 представлен SWOT-анализ разрабатываемой ИС.

Табл. 2. SWOT-таблица

|  |  |  |
| --- | --- | --- |
|  | Возможности | Угрозы |
| Внешние | Использование API у известной компании, может повысить надежность продукта в лице потребителей | Использование API у большинства информационногогеографических серверов является платным, а бесплатные версии ограниченны, что чревато задержками в работе. |
| Внутренние | Появление нового вида анализа геоданных. | Сложности в сборе данных, если они недоступны в общем доступе. |

# 3.2. Концептуальная модель предметной области

Диаграмма предметной области занимает центральное место в проектировании системы, показывая ее структуру. Диаграмма представляет собой абстракции предметной области и их связи (статические отношения), существующие между ними.

На концептуальной модели предметной области отображены основные сущности и связи между ними (рис.4.)

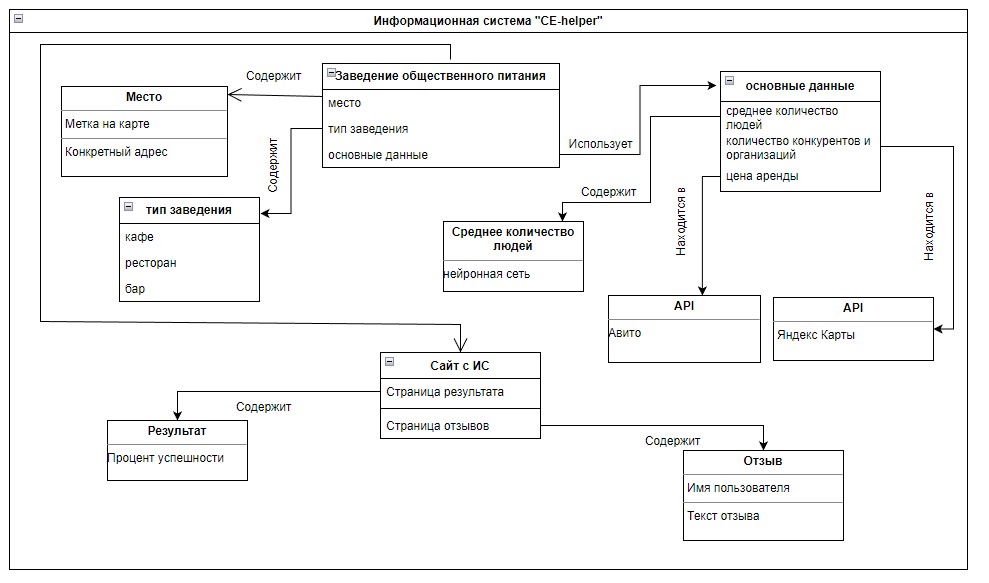


Рис.4. Концептуальная модель предметной области

На (рис. 4) изображены следующие сущности:

* «Место» - содержит в себе адрес или метку, которая показывает примерный адрес;
* «Тип заведения» - позволяет выбрать тип заведения;
* «Заведение общественного питания» - содержит в себе несколько сущностей;
* «Основные данные» - показывает собираемые и используемые данные;
* «API» - содержит в себе выбранные системы API;
* «Среднее количество людей» - содержит в себе метод сбора данных по данной сущности;
* «Сайт с ИС» - содержит в себе страницу с результатом и страницу отзывов;
* «Результат» - показывает результат на основе основных данных;
* «Отзыв» - содержит в себе имя пользователя и текст отзыва.

## 3.3. Use case диаграмма

Use Case – это сценарная техника описания взаимодействия. С помощью Use Case может быть описано и пользовательское требование, и требование к взаимодействию систем, и описание взаимодействия людей и компаний в реальной жизни [12].

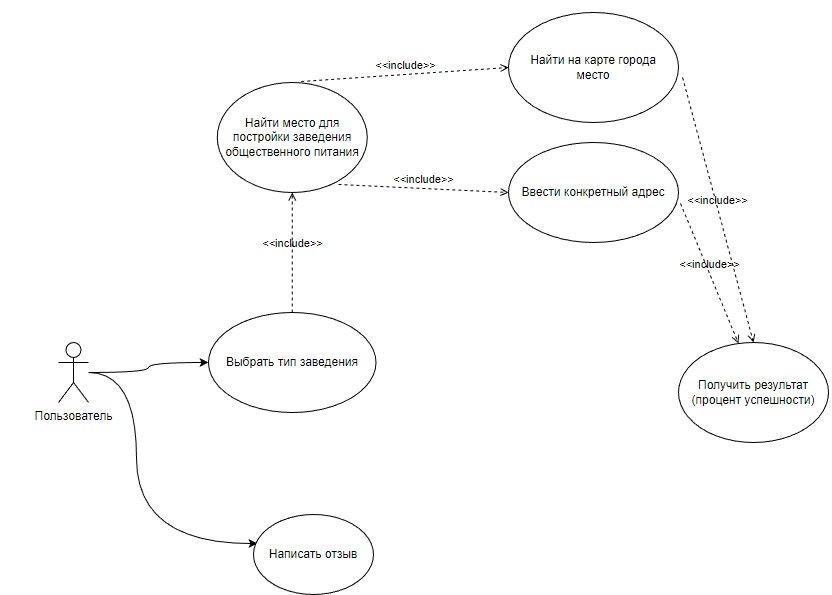


Рис. 5. Use case диаграмма

На рис. 5 представлены возможности пользователя при работе с системой.

## 3.4. Описание прецедентов

**Название прецедента**:

Поиск места для постройки заведения общественного питания

**Основной исполнитель**:

Пользователь системы

**Заинтересованные лица и их требования**:

1. Пользователь. Требует показать процент успешности постройки заведения в выбранном им месте.
2. Владелец сервиса. Хочет получать достоверные сведения о собираемых данных от Яндекс Карты.

**Предусловия**:

Пользователь выбрал место для постройки методом «клика» по карте.

**Постусловия**:

Пользователь получил удовлетворивший его процент в результате.

**Основной успешный сценарий**:

1. Пользователь зашел на сайт.
2. Пользователь выбрал тип заведения.
3. Пользователь выбрал место постройки путем ввода определенного адреса или «клика» по карте.
4. Веб-сервис выдает результат в виде процента.
5. Пользователя устраивает результат.
6. Пользователь оставляет или не оставляет отзыв.
7. Пользователь покидает сайт.

**Основной неуспешный сценарий**:

1. Пользователь зашел на сайт.
2. Пользователь выбрал тип заведения.
3. Пользователь выбрал место постройки путем ввода определенного адреса или «клика» по карте.
4. Веб-сервис выдает результат в виде процента.
5. Пользователя не устраивает результат.
6. Пользователь оставляет или не оставляет отзыв.
7. Пользователь покидает сайт.

**Сценарий с не обновлённой информацией**:

1. Пользователь зашел на сайт.
2. Пользователь выбрал тип заведения.
3. Пользователь выбрал место постройки путем ввода определенного адреса или «клика» по карте.
4. Веб-сервис выдает результат в виде процента.
5. Пользователя устраивает результат.
6. Пользователь проверяет полученную информацию в реальности.
7. Данные не совпадают.
8. Пользователь не удовлетворен результатом.
9. Пользователь оставляет или не оставляет отзыв.

## 3.5. Моделирование процесса сбора данных и получения процента

Диаграмма BPMN (англ. Business Process Model and Notation, нотация и модель бизнес-процессов) — система условных обозначений (нотация) и их описания для моделирования бизнес-процессов.

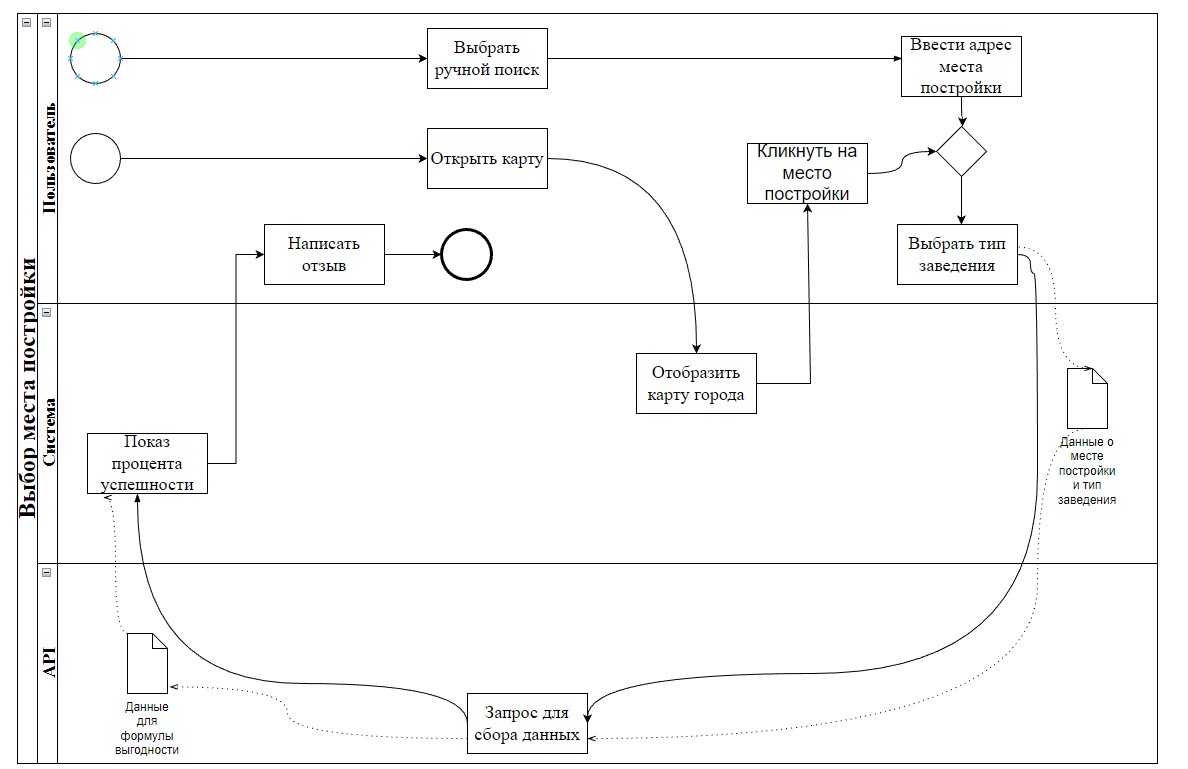


Рис. 6. BPMN диаграмма

На данной диаграмме (рис. 6) изображено моделирование процесса получение результата с процентом успешности заведения общественного питания.

## 3.6. Диаграмма классов

Для того чтобы представить статическую структуру модели системы используется диаграмма классов. Данная диаграмма служит для отображения различных взаимосвязей между отдельными сущностями, такие как объекты и подсистемы, и для описания их внутренней структуры, а также типы отношений между ними.

На рис. 7 представлена диаграмма классов, сущности и типы связи между ними.

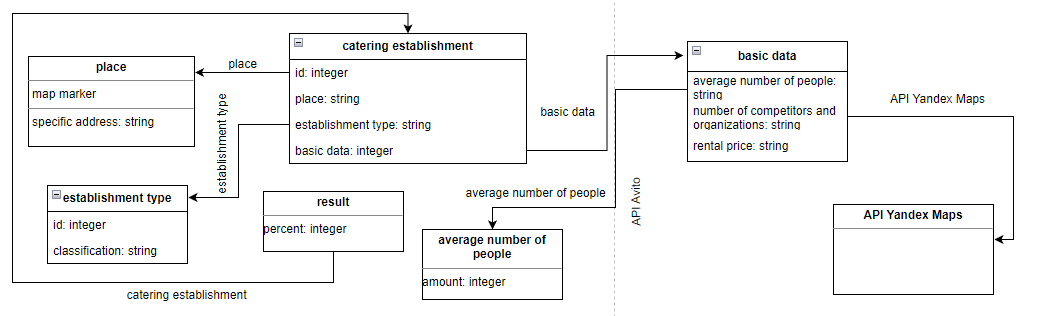


Рис. 7. Диаграмма классов «CE-helper»

На диаграмме отображены следующие сущности: catering establishment, place, establishment type, result, basic data, average number of people, API Yandex Maps. Их описание представлено в таблице (Таблица 3).

Сущности содержат атрибуты. Атрибуты сущностей описаны в таблице (Таблица 4).

*Табл. 3. Описание сущностей в ИС «CE-helper»*

|  |  |
| --- | --- |
| Сущность | Наименование |
| Catering establishment | Заведение общественного питания |
| Place | Место |
| Establishment type | Тип заведения |
| Basic data | Основные данные |
| Average number of people | Среднее количество людей |
| Result | Результат |
| API Yandex Maps | API Яндекс Карты |

Табл. 4. Описание атрибутов сущностей в ИС «CE-helper»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Сущность | Атрибут | Наименование | Тип | Обязательный | Max длина | Ссылка на сущность |
| Catering establishment | id | id | ID | 1 |  |  |
| Establishment type | Тип заведения | STRING | 1 |  | Establishment type |
| Basic data | Основные данные | INTEGER | 1 |  | Basic data |
| Place | Map marker | Метка на карте | - | 1 |  |  |
| Specific address | Конкретны й адрес | STRING | 1 |  |  |
| Establishment type | id | id | INTEGER | 1 |  |  |
| Classification | Тип | STRING | 1 |  |  |
| Сущность | Атрибут | Наименование | Тип | Обязательный | Max длина | Ссылка на сущность |
| Basic data | Average number of people | Среднее количество людей | STRING | 1 |  | Average number of people |
| Number of competitors and organizations | Количество конкурентов и  организаций | STRING | 1 |  | API  Yandex Maps |
| Rental price | Цена аренды | STRING | 1 |  | API Avito |
| Result | Percent | Процент | INTEGER | 1 |  | Catering establishment |
| Average number of people | Amount | Количество | INTEGER | 1 |  |  |
| API Avito | - | - | - | 1 |  |  |
| API  Yandex  Maps | - | - | - | 1 |  |  |

## 3.7. Диаграмма компонентов

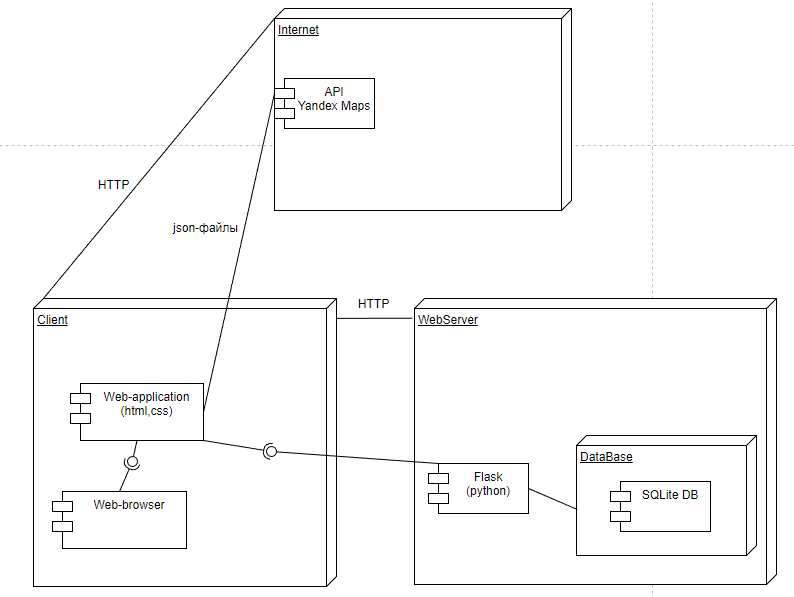


Рис. 8 Диаграмма компонентов

На рис. 8 отображена диаграмма компонентов, которая отображает работу основных элементов системы.

# ГЛАВА 4. РЕАЛИЗАЦИЯ СИСТЕМЫ.

## 4.1. Страницы системы

В данной подглаве будут показаны основные страницы информационной системы.

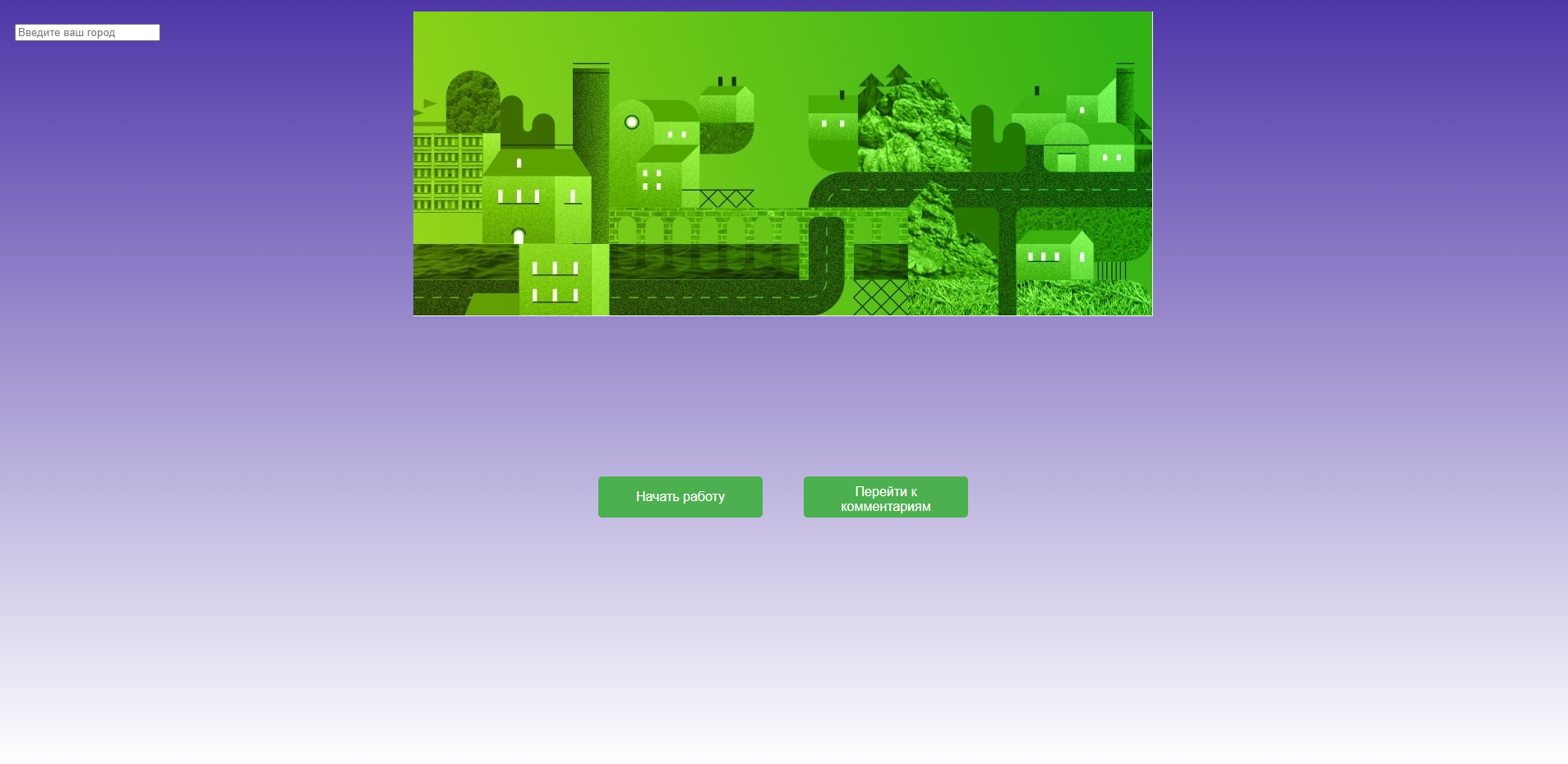


Рис. 9. Главная страница сайта

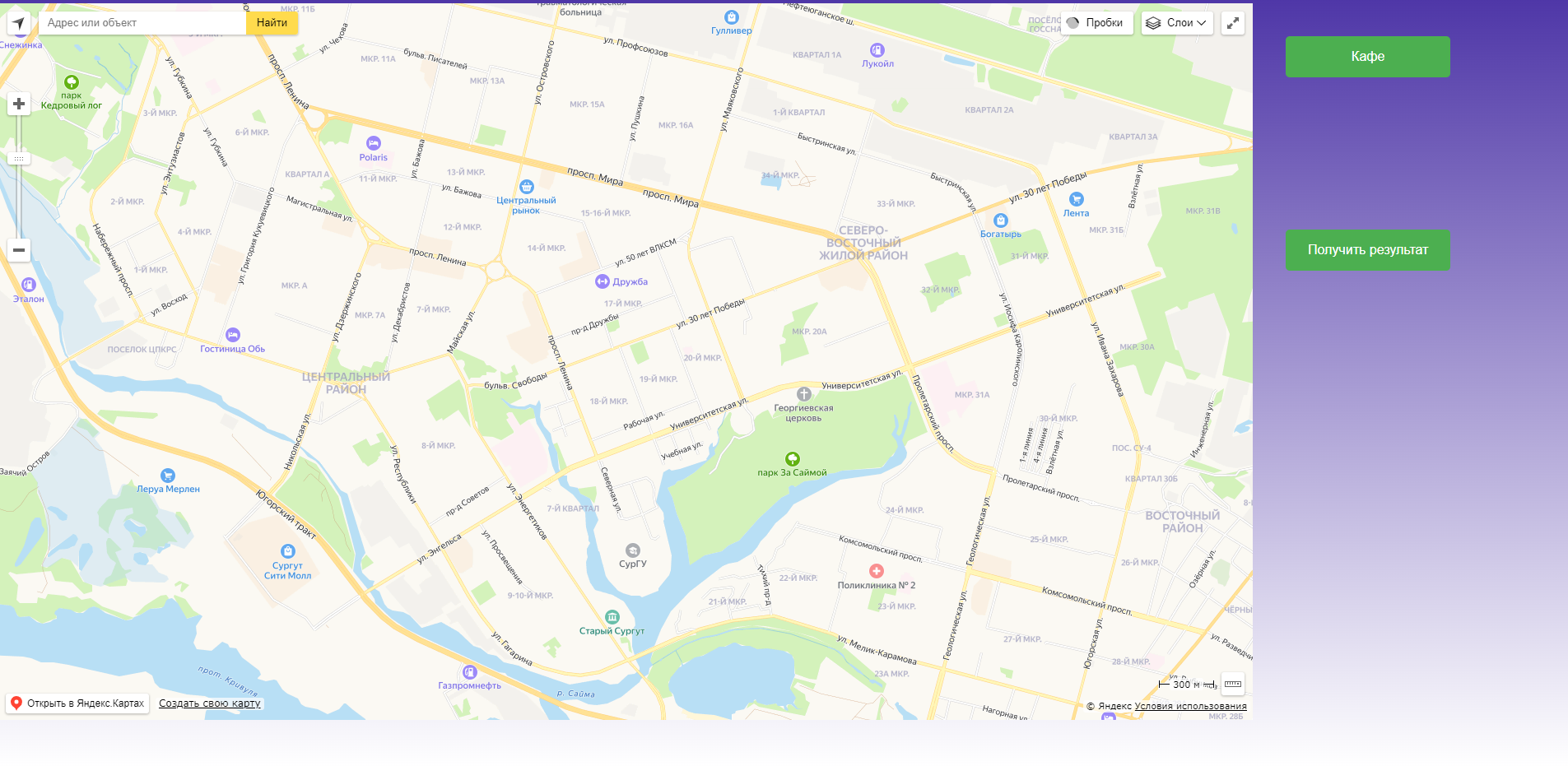


Рис. 10. Рабочая страница сайта

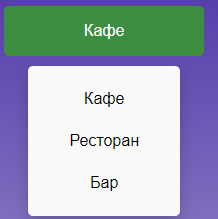


Рис. 11. Выпадающее меню

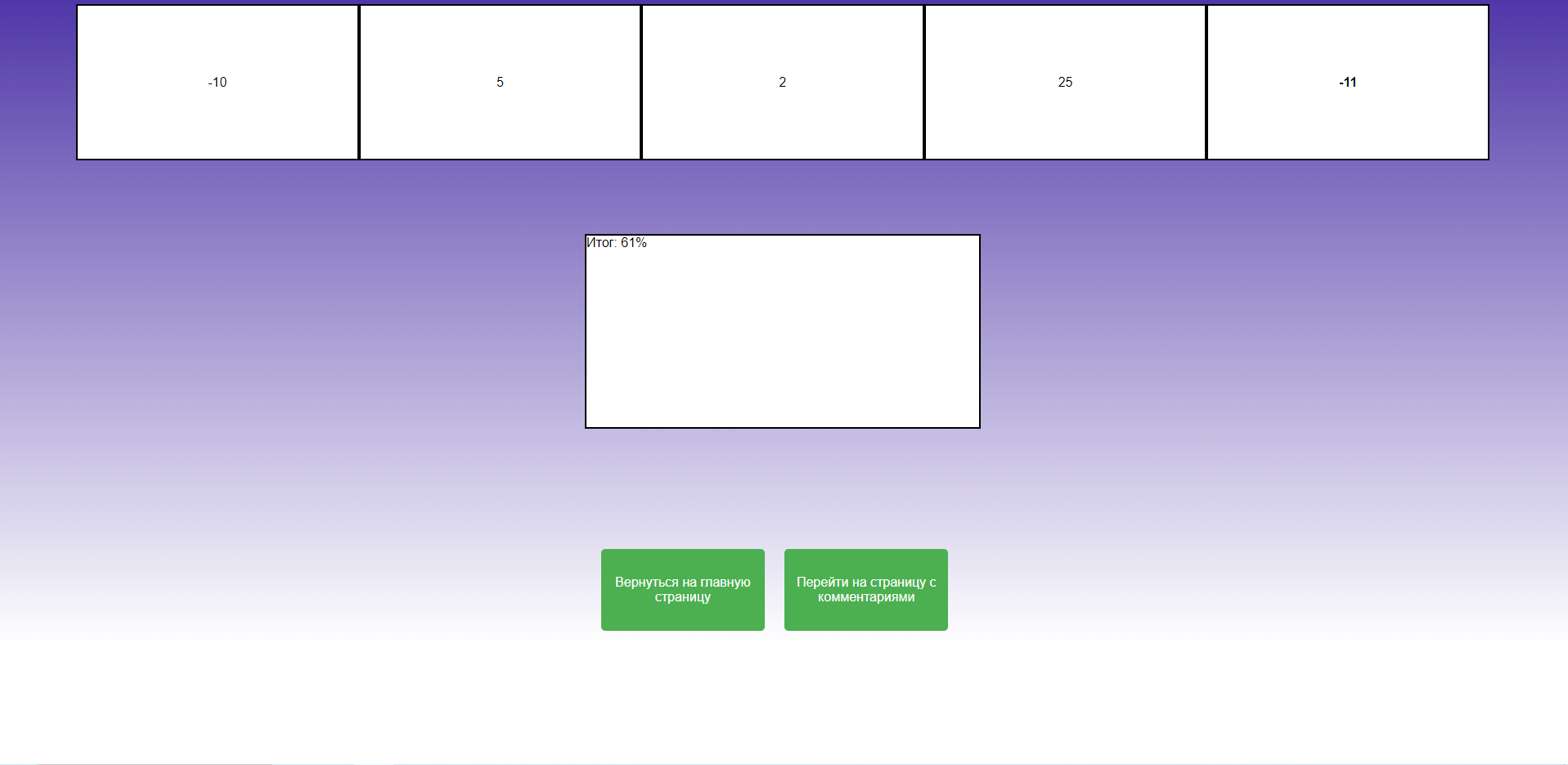


Рис. 12. Страница результата

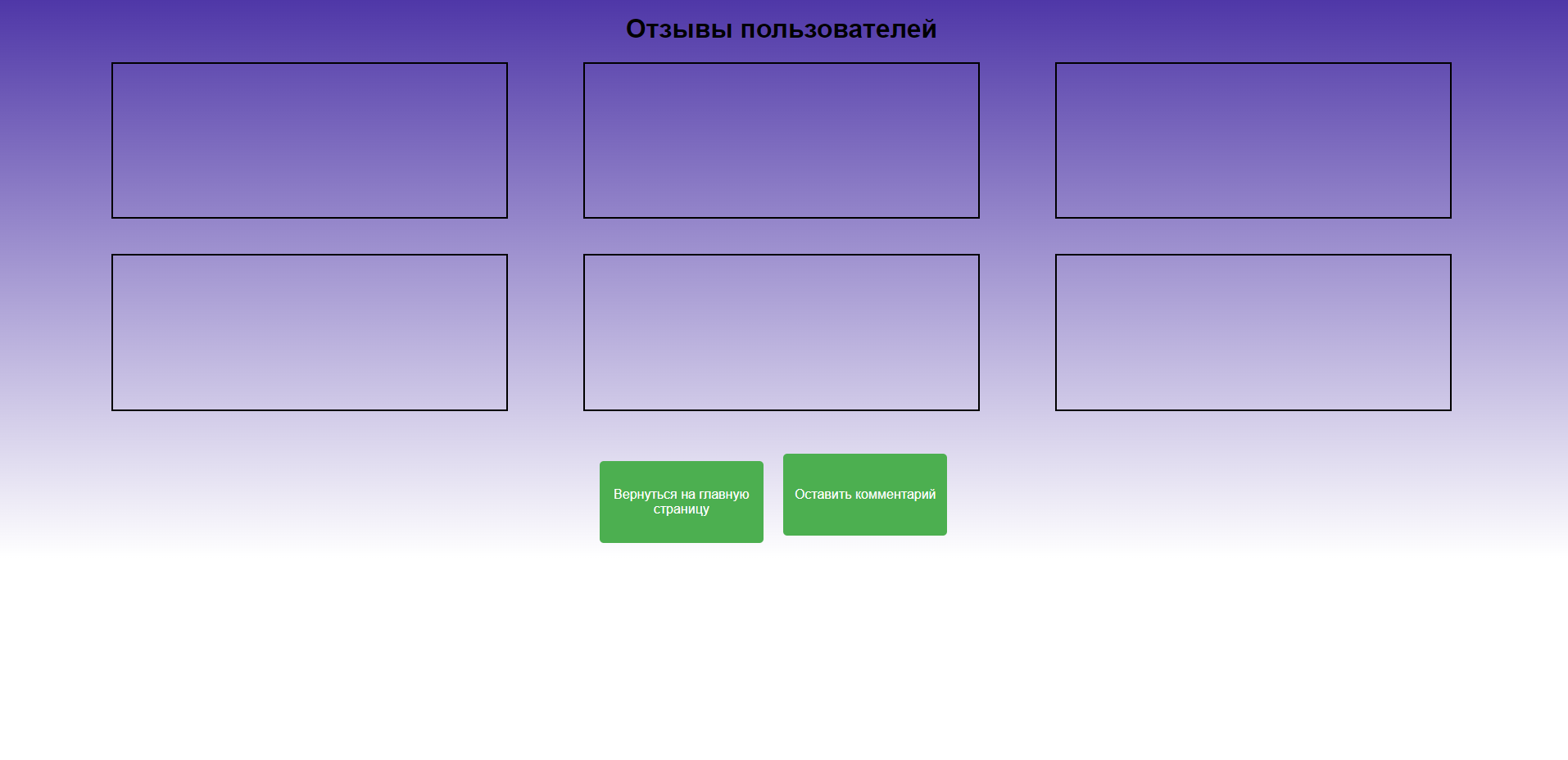


Рис. 13. Страница комментариев

# 4.2. Основные функции системы

На (рис. 9) показана главная страница сайта, на которой можно сделать два действия:

1. Сразу начать работу, нажав на кнопку, перед этим выбрав необходимый город (Пока реализована только работа с городом Сургут);
2. Перейти и посмотреть комментарии, перед тем как воспользоваться веб-приложением.

На (рис.10) представлена рабочая странице сайта подключается карты от Яндекс Карты через систему API. Есть фильтр по заведениям в виде выпадающего меню (рис. 11).

Основной алгоритм работы на рабочей странице представляет собой следующие шаги:

1. Выбрать в фильтр необходимый тип заведения (кафе, ресторан или бар).
2. «Кликнуть» на любое место в городе, тем самым отправив запросы через API в Яндекс Карты и в базу данных.
3. Нажать на кнопку «Получить результат».

На (рис. 12) представлен результат работы системы по следующему адресу: Ханты-Мансийский автономный округ, Сургут, улица Чехова, 10/2.

Всего идет пять основных форм, каждая из которых отображает значения выбранных показателей.

Порядок отображения идет слева направо:

Конкуренты. В данной форме показывается значение в виде количества конкурентов в радиусе одного километра. Радиус обуславливается тем, что человек в большинстве случаев в городской местности преодолевает данное расстояние за десять или пятнадцать минут. И если человеку необходимо поесть где-либо, он ищет ближайшее заведение от себя. Если же до заведения нужно добираться больше, чем двадцать минут, скорее всего он не выберет данное заведение, кроме особых случаев, когда он намеренно собирался в именно то заведение. Также значение уже сразу умножается на выбранный числовой коэффициент.

Организации. В данной форме показывается количество организаций, также расположенных в радиусе одного километра. Если внутри этих зданий отсутствует своя столовая, то рабочий человек во время обеда, будет искать заведение общественного питания. Поэтому строящееся заведение должно быть в радиусе одного километра, чтобы человек имел возможность выбрать именно данное заведение.

Супермаркеты. Также показываются в радиусе одного километра, потому что, данный вид строений, как и организации собирают людей рядом.

Количество людей. Данные по количеству людей были собраны через видеокамеры и занесены в базу данных, которая подключена к системе. Выбор значения количества людей из базы данных зависит от того, какая точка для постройки заведения была выбрана на карте. И ближайшее значение количества определяется наименьшей разницей в полученных и хранимых координатах.

Цена аренды. Также как количество людей, выбирается по наименьшей разнице в координатах.

На (рис. 13) представлены формы, в которых буду оставляться последние из комментариев пользователей. Чтобы написать комментарий регистрация не требуется.

# 4.3. Запрос через API и запрос к базе данных.

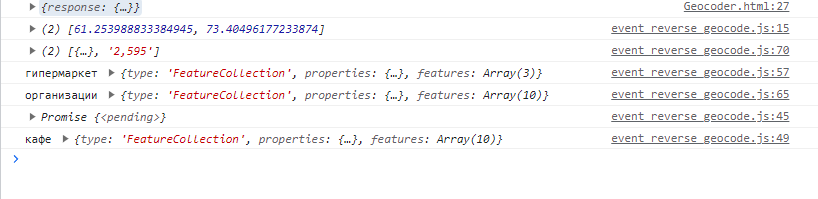


Рис. 14. JSON файл

На (рис. 14) представлен json файл, который представляет собой собранные данные через запросы к API Яндекс Карт, а также выводы из базы данных.

Первая строка после {response: {…}} представляет собой массив из двух чисел, а именно координаты выбранной точки на карте.

Далее идет количество людей ближайшее к выбранной точке, путем вывода значения из базы данных.

Затем идет три коллекции с названиями: гипермаркет, организации и кафе.

Это и есть запросы к API. Каждый из них возвращает коллекцию данных, из которой для работы системы извлекается только количество вернутых объектов.

# ЗАКЛЮЧЕНИЕ

За последние несколько лет количество заведений общественного питания увеличилось на заметное значение [1] и вынуждает участников ресторанного бизнеса применять различные средства, которые делятся как на технические и аппаратурные, так на и людские.

Все это приводит к увеличению затрат при выборе места постройки заведения общественного питания. Разрабатываемая информационная система призвана оптимизировать этот процесс, позволяя быстро собирать основные необходимые данные и вычисляя предполагаемый процент успешности нового заведения. Но чтобы ускорить этот процесс нужны современные средства и методы.

В результате выпускной квалификационной работы были выполнены все поставленные задачи.

Была тщательно изучена предметная область, а также основательно проанализированные существующие аналоги по сбору данных. Описаны все необходимые требования к разрабатываемой системе (функциональное, информационное, лингвистическое, программное, техническое, методическое, математическое). Проведен SWOT- анализ и построены все необходимые модели и диаграммы такие как: модель предметной области, BPMN диаграмма, use-case диаграмма, спецификация прецедентов, диаграмма классов, диаграмма компонентов. Также был разработан пользовательский интерфейс.

Главные задачи, которые должна выполнять система были реализованы, а именно:

1. Запрос через API Яндекс Карты об количество конкурентов, организаций и супермаркетов (гипермаркетов или других крупных сооружений).
2. Вывод данных о количестве людей из базы данных.
3. Отображение процента успешности.

В результате была спроектирована и разработана информационная система по сбору данных, используемых при выборе места постройки заведения общественного питания. Она ускоряет и упрощает процесс сбора необходимых данных, экономя различные виды ресурсов, которые затрачиваются при выборе места постройки.

# СПИСОК ЛИТЕРАТУРЫ

* 1. За 2017-2021 гг численность предприятий общественного питания в России выросла на 2%: с 186,9 тыс до 190,4 тыс. // Busines Stat URL: <https://marketing.rbc.ru/articles/13430/> (дата обращения: 15.10.2022).
  2. Google Maps URL: <https://www.google.com/maps> (дата обращения 31.09.2022).
  3. Яндекс Карты URL: <https://yandex.ru/maps> (дата обращения 31.09.2022).
  4. Ларман К. Применение UML 2.0 и шаблонов проектирования. - 3-е изд. - Диалектика, 2019. - 736 с.
  5. Эрик Э. Предметно-ориентированное проектирование (DDD): структуризация сложных программных систем.: Пер. с англ. - М.: 000 "И.Д. Вильямс", 2011. - 448 с.: ил. - Парал. тит. англ.
  6. Элти Д., Кумбс М. Экспертные системы: концепции и примеры / Д. Элти, М. Кумбс. - М.: Финансы и статистика, 2006. – 236 c.
  7. ГОСТ 34.602-2020 Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы [Электронный ресурс]: Учебно-методические материалы – URL: https://docs.cntd.ru/document/1200181804 (дата обращения: 08.03.2023).
  8. ГОСТ Р 59793–2021 Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания. [Электронный ресурс]: Учебно-методические материалы – URL: https://protect.gost.ru/v.aspx?control=8&baseC=-1&page=0&month=-1&year=-1&search=&RegNum=1&DocOnPageCount=15&id=231287 (дата обращения: 08.03.2023).
  9. ГОСТ 34.201-2020 Информационная технология. Комплекс стандартов на автоматизированные системы. Виды, комплексность и обозначение документов при создании автоматизированных систем. [Электронный ресурс]: Учебно-методические материалы – URL: https://docs.cntd.ru/document/1200181803 (дата обращения: 08.04.2023).
  10. Проектирование информационных систем. Лекции. Модель вариантов использования [Электронный ресурс]: Учебно-методические материалы - Электрон. текстовые дан. – URL: https://sites.google.com/site/anisimovkhv/learning/pris/lecture/tema12 (дата обращения: 08.03.2023).
  11. Шаблоны корпоративных приложений / Мартин Фаулер, Дейвид Райс, Мэттью Фоммел, Эдвард Хайет, Роберт Ми, Рэнди Стаффорд, - 2-е изд. – М.: Вильямс, 2016. – 544 с.
  12. Welcome to Flask — Flask Documentation (2.3.x) // URL: <https://flask.palletsprojects.com/en/2.3.x/> (дата обращения: 20.05.2023).
  13. SQLite Documentation // URL: <https://www.sqlite.org/docs.html> (дата обращения: 20.05.2023).
  14. Страшко М.Т, Ситникова С.Ю. СОВРЕМЕННЫЕ ПОДХОДЫ К СОЗДАНИЮ ВЕБ-САЙТОВ // НАУЧНО-ТЕХНИЧЕСКОЕ И ЭКОНОМИЧЕСКОЕ СОТРУДНИЧЕСТВО СТРАН АТР В XXI ВЕКЕ. - 2022. - №2. - С. 296-300.
  15. Сысолетин, Е. Г. Разработка интернет-приложений : учеб. пособие для вузов / Е. Г. Сысолетин, С. Д. Ростунцев ; под науч. ред. Л. Г. Доросинского. — М. : Издательство Юрайт, 2019. — 90 с
  16. Голицына, О.Л. Информационные системы: Учебное пособие / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - М.: Форум, 2016. - 352 c.
  17. Федотова, Е.Л. Информационные технологии и системы: Уч.пос / Е.Л. Федотова. - М.: Форум, 2018. - 149 c.
  18. Ипатова, Э. Р. Методологии и технологии системного проектирования информационных систем / Э. Р. Ипатова, Ю. В. Ипатов. – Москва : ФЛИНТА, 2021. – 256 c. – ISBN 9785893499780. – URL: http://biblioclub.ru/index.php?page=book\_red&id=79551 (дата обращения: 03.03.2023). – Текст : электронный
  19. Web-технологии ; Составитель: Малышева Е. Н.. – Кемерово : Кемеровский государственный университет культуры и искусств (КемГУКИ), 2014. – 104 c. – URL: http://biblioclub.ru/index.php?page=book\_red&id=275540 (дата обращения: 02.03.2023). – Текст : электронный.
  20. Никулова, Г. А. Web-программирование / Г. А. Никулова, В. Р. Субботин. – Липецк : Липецкий государственный педагогический университет им. П.П. Семенова-Тян-Шанского, 2017. – 58 c. – ISBN 9785885268349. – URL: http://biblioclub.ru/index.php?page=book\_red&id=577452 (дата обращения: 09.03.2023). – Текст : электронный.
  21. Рыбальченко, М. В. Архитектура информационных систем : учебное пособие для вузов : [для студентов, обучающихся по направлению "Информационные системы и технологии"] / М. В. Рыбальченко ; Южный федеральный университет. – Москва : Юрайт, 2019. – 91 с. – (Университеты России). – Текст : непосредственный.
  22. Стасышин, В. М. Проектирование информационных систем и баз данных : учебное пособие / В. М. Стасышин ; В. М. Стасышин ; Новосиб. гос. техн. ун-т. – Новосибирск : Изд-во НГТУ, 2012. – 97, [2] с. – URL: http://library.kuzstu.ru/meto.php?n=178035&type=nstu:common (дата обращения: 16.04.2023). – Текст : электронный
  23. Советов, Б. Я. Информационные технологии : учебник для бакалавров : [для студентов вузов, обучающихся по направлениям подготовки диплом. специалистов "Информатика и вычислительная техника" и "Информационные системы"] / Б. Я. Советов, В. В. Цехановский ; С.- Петерб. гос. электротехн. ун-т. – 6-е изд. – Москва : Юрайт, 2012. – 263 с. – (Бакалавр). – Текст : непосредственный.

# ПРИЛОЖЕНИЕ 1. Модуль event\_reverse\_geocode.js

ymaps.ready(init);

function init() {

    var myPlacemark,

        myMap = new ymaps.Map('map', {

            center: [61.24942013169517, 73.40993995227039],

            zoom: 14

        }, {

            searchControlProvider: 'yandex#search'

        });

    // Слушаем клик на карте.

    myMap.events.add('click', function (e) {

        var coords = e.get('coords');

        console.log(coords);

        // Если метка уже создана – просто передвигаем ее.

        if (myPlacemark) {

            myPlacemark.geometry.setCoordinates(coords);

        }

        // Если нет – создаем.

        else {

            myPlacemark = createPlacemark(coords);

            myMap.geoObjects.add(myPlacemark);

            // Слушаем событие окончания перетаскивания на метке.

            myPlacemark.events.add('dragend', function () {

                getAddress(myPlacemark.geometry.getCoordinates());

            });

        }

        getAddress(coords);

        // const coords = [55.7522, 37.6156]; // Широта и долгота центра поисковой области

        const category = 'кафе'; // Категория организаций для поиска (в данном случае - заведения питания)

        // fetch(https://search-maps.yandex.ru/v1/?apikey=${apiKey}&text=${category}&ll=${coords[1]},${coords[0]}&spn=0.05,0.05&lang=ru&type=biz&results=500&radius=${radius})

        //   .then(response => response.json())

        //   .then(data => {

        //     const features = data.features;

        //     // Обработка полученных объектов

        //   })

        //   .catch(error => console.error(error));

        fetch(`https://search-maps.yandex.ru/v1/?apikey=b774c439-0479-418c-b490-5eceae361a0f&text=${category}&ll=${coords[1]},${coords[0]}&bbox=${coords[1] - 0.0287},${coords[0] - 0.1772}~${coords[1] + 0.0287},${coords[0] + 0.1772}&rspn=1&lang=ru`)

            .then((response) => {

                const filterData = response.json()

                console.log(filterData)

                return filterData;

            })

            .then((data) => {

                console.log('кафе', data);

                localStorage.setItem("Cafe", data.features.length)

            });

        fetch(`https://search-maps.yandex.ru/v1/?apikey=b774c439-0479-418c-b490-5eceae361a0f&text=%D0%B3%D0%B8%D0%BF%D0%B5%D1%80%D0%BC%D0%B0%D1%80%D0%BA%D0%B5%D1%82%20%D0%A1%D1%83%D1%80%D0%B3%D1%83%D1%82&ll=${coords[0]},${coords[1]}&bbox=${coords[1] - 0.0287},${coords[0] - 0.1772}~${coords[1] + 0.0287},${coords[0] + 0.1772}&rspn=1&lang=ru`)

            .then((response) => {

                return response.json();

            })

            .then((data) => {

                console.log('гипермаркет', data);

                localStorage.setItem("Giper", data.features.length)

            });

        fetch(`https://search-maps.yandex.ru/v1/?apikey=b774c439-0479-418c-b490-5eceae361a0f&text=%20%D0%9E%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F%20%D0%A1%D1%83%D1%80%D0%B3%D1%83%D1%82&ll=${coords[0]},${coords[1]}&bbox=${coords[1] - 0.0287},${coords[0] - 0.1772}~${coords[1] + 0.0287},${coords[0] + 0.1772}&rspn=1&lang=ru`)

            .then((response) => {

                return response.json();

            })

            .then((data) => {

                console.log('организации', data);

                localStorage.setItem("Orga", data.features.length)

            });

        axios.post("http://127.0.0.1:5000/coord", { coords })

            .then((response) => {

                console.log(response.data);

                localStorage.setItem("People",response.data[1]);

        })

            .catch(error => {

                console.log(error)

                alert("Ошибка обращения к BackEnd: " + error)

            });

            // peoples.innerText = response.data[1];

        // Создание метки.

        function createPlacemark(coords) {

            return new ymaps.Placemark(coords, {

                iconCaption: 'поиск...'

            }, {

                preset: 'islands#violetDotIconWithCaption',

                draggable: true

            });

        }

        // Определяем адрес по координатам (обратное геокодирование).

        function getAddress(coords) {

            myPlacemark.properties.set('iconCaption', 'поиск...');

            ymaps.geocode(coords).then(function (res) {

                var firstGeoObject = res.geoObjects.get(0);

                myPlacemark.properties

                    .set({

                        // Формируем строку с данными об объекте.

                        iconCaption: [

                            // Название населенного пункта или вышестоящее административно-территориальное образование.

                            firstGeoObject.getLocalities().length ? firstGeoObject.getLocalities() : firstGeoObject.getAdministrativeAreas(),

                            // Получаем путь до топонима, если метод вернул null, запрашиваем наименование здания.

                            firstGeoObject.getThoroughfare() || firstGeoObject.getPremise()

                        ].filter(Boolean).join(', '),

                        // В качестве контента балуна задаем строку с адресом объекта.

                        balloonContent: firstGeoObject.getAddressLine()

                    });

            });

        }

        // Если метка уже создана – просто передвигаем ее.

        if (myPlacemark) {

            myPlacemark.geometry.setCoordinates(coords);

        }

        // Если нет – создаем.

        else {

            myPlacemark = createPlacemark(coords);

            myMap.geoObjects.add(myPlacemark);

            // Слушаем событие окончания перетаскивания на метке.

            myPlacemark.events.add('dragend', function () {

                getAddress(myPlacemark.geometry.getCoordinates());

            });

        }

        getAddress(coords);

        // Создание метки.

        function createPlacemark(coords) {

            return new ymaps.Placemark(coords, {

                iconCaption: 'поиск...'

            }, {

                preset: 'islands#violetDotIconWithCaption',

                draggable: true

            });

        }

        // Определяем адрес по координатам (обратное геокодирование).

        function getAddress(coords) {

            myPlacemark.properties.set('iconCaption', 'поиск...');

            ymaps.geocode(coords).then(function (res) {

                var firstGeoObject = res.geoObjects.get(0);

                myPlacemark.properties

                    .set({

                        // Формируем строку с данными об объекте.

                        iconCaption: [

                            // Название населенного пункта или вышестоящее административно-территориальное образование.

                            firstGeoObject.getLocalities().length ? firstGeoObject.getLocalities() : firstGeoObject.getAdministrativeAreas(),

                            // Получаем путь до топонима, если метод вернул null, запрашиваем наименование здания.

                            firstGeoObject.getThoroughfare() || firstGeoObject.getPremise()

                        ].filter(Boolean).join(', '),

                        // В качестве контента балуна задаем строку с адресом объекта.

                        balloonContent: firstGeoObject.getAddressLine()

                    });

            });

        }

    });

    // var sravnenie=document.getElementById("comparison")

    // sravnenie.onclick(()=>axios.get("http://127.0.0.1:5000/coord/"+coords)

    // .then((response) => onOk(response))

    // .catch(error => {

    //   console.log(error)

    //   alert("Ошибка обращения к BackEnd: " + error)

    // }))

}

# ПРИЛОЖЕНИЕ 2. Модуль init.py

from flask import Flask, redirect, url\_for, request, jsonify

from routes.main import mainpg

from routes.zapros\_people import coords\_bp

# инициализация классов

app = Flask(\_\_name\_\_)

app.register\_blueprint(coords\_bp)

app.register\_blueprint(mainpg)

# запуск приложения

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(debug=True)

# ПРИЛОЖЕНИЕ 3. Модуль main.py

from flask import Blueprint, render\_template

# from coord\_repository inport

mainpg = Blueprint('mainpg', \_\_name\_\_)

@mainpg.route("/")

def route\_test():

    return render\_template("./Mainpage.html")

@mainpg.route("/Geocoder.html")

def route\_test2():

    return render\_template("./Geocoder.html")

@mainpg.route("/results.html")

def route\_test3():

    return render\_template("./results.html")

@mainpg.route("/comments.html")

def route\_test4():

    return render\_template("./comments.html")

@mainpg.route("/Mainpage.html")

def route\_test5():

    return render\_template("./Mainpage.html")

# ПРИЛОЖЕНИЕ 4. Модуль zapros\_people.py

from flask import Blueprint, jsonify, request

from coord\_repository import CoordsRepository

coordsRepository = CoordsRepository()

coords\_bp = Blueprint('coords', \_\_name\_\_)

@coords\_bp.route("/coord", methods=["POST"])

def route\_coords():

    coordinates=request.get\_json()

    nearest\_number = coordsRepository.Coord(coordinates)

    print(nearest\_number)

    print(coordinates)

    response = jsonify(coordinates,nearest\_number)

    response.headers.add('Access-Control-Allow-Origin', '\*')

    return response

# ПРИЛОЖЕНИЕ 5. Модуль coord\_repository.py

import sqlite3

import math

class CoordsRepository:

    def get\_connect(self):

        return sqlite3.connect("./number\_of\_peoples.sqlite")

    def Coord(self, coordinates):

        try:

            connect = self.get\_connect()

            cursor = connect.cursor()

            cursor.execute(

                # """SELECT \* FROM `list` WHERE number = ?""", (coords,))

               """SELECT \* FROM `list`  """,coordinates )

            list = cursor.fetchall()

            print(f"coordinates {coordinates}")

            (ind, nearest\_number) = find\_nearest\_point(list,coordinates)

            print(f"list {list[ind]}")

            # print(f"list {list}")

            return list

        except Exception as error\_list:

            print(f"error\_list {error\_list}")

        finally:

            connect.close()

        return nearest\_number

def find\_nearest\_point(list, coordinates):

    min\_dist = math.inf

    number =-1

    ind =0

    for elem in list:

        el\_coords=elem[2][:-1]

        el\_coords=el\_coords[1:]

        list\_el\_coords = el\_coords.split(', ')

        el\_coords = []

        for a in list\_el\_coords:

            el\_coords.append(float(a))

        cur\_dist = haversine\_distance(el\_coords[1],el\_coords[0],

                                        coordinates['coords'][1],coordinates['coords'][0]

                                            )

        if cur\_dist<min\_dist:

            min\_dist = cur\_dist

            number = elem[1]

        ind = ind +1

    return ind, number

def haversine\_distance(lat1, lon1, lat2, lon2):

        # Константа радиуса Земли в километрах

        R = 6371

        # Преобразование градусов в радианы

        lat1 = math.radians(lat1)

        lon1 = math.radians(lon1)

        lat2 = math.radians(lat2)

        lon2 = math.radians(lon2)

        # Вычисление разности широт и долгот

        dlat = lat2 - lat1

        dlon = lon2 - lon1

        # Вычисление гаверсинуса центрального угла

        a = math.sin(dlat/2)\*\*2 + math.cos(lat1) \* math.cos(lat2) \* math.sin(dlon/2)\*\*2

        c = 2 \* math.asin(math.sqrt(a))

        # Вычисление расстояния по дуге сферы

        d = R \* c

        return d

# ПРИЛОЖЕНИЕ 6. Модуль menu.js

  // Получаем ссылки на элементы списка и кнопку

  var dropdown = document.querySelector(".dropdown");

  var dropdownContent = document.querySelector(".dropdown-content");

  var dropbtn = document.querySelector(".dropbtn");

  // Добавляем обработчик события для клика на кнопке

  dropbtn.addEventListener("click", function() {

    dropdownContent.classList.toggle("show");

  });

  // Добавляем обработчики событий для клика на элементах списка

  var dropdownItems = dropdownContent.querySelectorAll("a");

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

    dropdownItems[i].addEventListener("click", function() {

      // Запоминаем выбранное значение в localStorage

      localStorage.setItem("selectedValue", this.textContent);

      // Обновляем текст на кнопке

      dropbtn.textContent = this.textContent;

      // Скрываем выпадающее меню

      dropdownContent.classList.remove("show");

    });

  }

  // При загрузке страницы проверяем, есть ли сохраненное значение в localStorage

  if (localStorage.getItem("selectedValue")) {

    dropbtn.textContent = localStorage.getItem("selectedValue");

    window.addEventListener('beforeunload', function() {

      localStorage.removeItem('dropdown');

    });

  }