ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ ГОРОДА МОСКВЫ
Государственное бюджетное общеобразовательное учреждение города Москвы
«Школа № 1387»
(ГБОУ школа № 1387)

Кейс №5: Сбор и обработка данных температуры

Авторы работы:
Марков Александр Сергеевич
Прошина Евгения Александровна
Аваков Артём Артурович
11 И класс
ГБОУ школа № 1387
Научный руководитель:

Москва, 2020

Оглавление

Анализ технических требований

Задание

Реализовать программный модуль для сбора, хранения и обработки данных с удалённых температур данных

Условия выполнения

Посредством специализированного сервиса, расположенного по адресу http://dt.miet.ru/ppo_it, осуществить сбор данных об уличной температуре в 16 городах. Необходимо использовать показатели датчиков, находящихся в 10

квартирах не менее, чем пяти районов города. Время осуществления – 48 часов реального времени.
Обращение к сервису происходит не реже, чем один раз в 10 минут.
Полученные данные хранятся с помощью реляционной СУБД, реализованной на основе ER-модели.
Взаимодействие с данными осуществляется через пользовательский интерфейс, который по запросу пользователя отображает информацию о температуре в квартирах и на улице в виде графика/диаграммы (см. пункт пользовательский интерфейс)
Пользовательский интерфейс является кроссплатформенным, а также организован в соответствии со стандартами построения UI
Для облегчения работы с вносящимися в программный код изменениями используется система управления версиями git

Пользовательский интерфейс

Согласно регламенту испытаний функционал UI подразумевает выведение следующих данных в виде графика/диаграммы/величины:

- Данные температуры в реальном времени в определённой квартире
- График изменения уличной температуре на протяжении суток реального времени в одном из городов
- График изменения средней температуры в квартирах в одном из городов на протяжении суток реального времени
- График изменения температуры в одной квартире в каждом из городов
- Диаграмма максимальных температур в квартирах в каждом из районов (не менее пяти) одного города

Язык программирования и программные средства

Языки программирования

Учитывая изложенные выше особенности технического задания, было решено, что в качестве основного языка программирования будет использоваться Python.
При выборе важную роль сыграли такие его качества, как минималистичный синтаксиса ядра и богатая стандартная библиотека, позволяющая работать с высокоуровневыми структурами данных и взаимодействовать со многими сетевыми протоколами, в частности HTTP.
Управление БД осуществляется посредством СУБД SQLite ввиду того, что его легко использовать при кроссплатформенном переносе, а также он очень надёжен с точки зрения программного кода.

Сравнение языков программирования

	Python	C++	Java
Опыт работы	+	-	-
Простота синтаксиса и удобство	+	-	-
Производительность	-	+	+
Работа с БД	+	-	-
Динамическая типизация	+	-	-

Таблица 1

Производительность языка не играла важную роль при выборе языка, так как программный продукт используется малым числом лиц и объёмы обрабатываемых данных относительно малы

Модули стандартной библиотеки Python

- модуль requests – инструмент составления HTTP и GET-запросов для взаимодействия с сервером
- модуль time – для работы с реальным/серверным временем
- модуль sqlite3 – кроссплатформенное средство работы с БД

Библиотеки Python

- Matplotlib – осуществляет визуализацию информации из БД в UI
- IPyWidgets – интерактивные HTML виджеты для Jupyter Notebook

Программные средства

Дистрибутив Anaconda предустанавливает инструмент интерактивной разработки Jupyter Notebook, удобный с точки зрения разработки и использования интерфейса, в том числе он поддерживает создание графического интерфейса для пользователя. В качестве IDE используется Spyder из дистрибутива Anaconda. Одна из важных особенностей этой среды разработки – это интеграция с научными библиотеками Python, к примеру Matplotlib

Описание основных этапов разработки

Первый этап

С помощью интернет-ресурсов мы изучили синтаксис и принципы работы описанных выше модулей и библиотек языка Python, СУБД SQLite, а также ознакомились с основами работы в программных средствах Jupyter Notebook и Spyder

Второй этап
Для реализации программного кода выбраны методы структурного программирования. Составлены концепции функций, образующих модуль взаимодействия с сервером и обработкой полученной информации и модуль пользовательского интерфейса и визуализации данных (далее «модуль 1» и «модуль 2» соответственно; см. Алгоритм работы программного продукта)

Третий этап
На сайте https://github.com/ создана система контроля версий продукта, доступная по следующей ссылке: https://github.com/CangCiwei/PredProf. Составлен план выполнения технического задания согласно функциональности продукта.

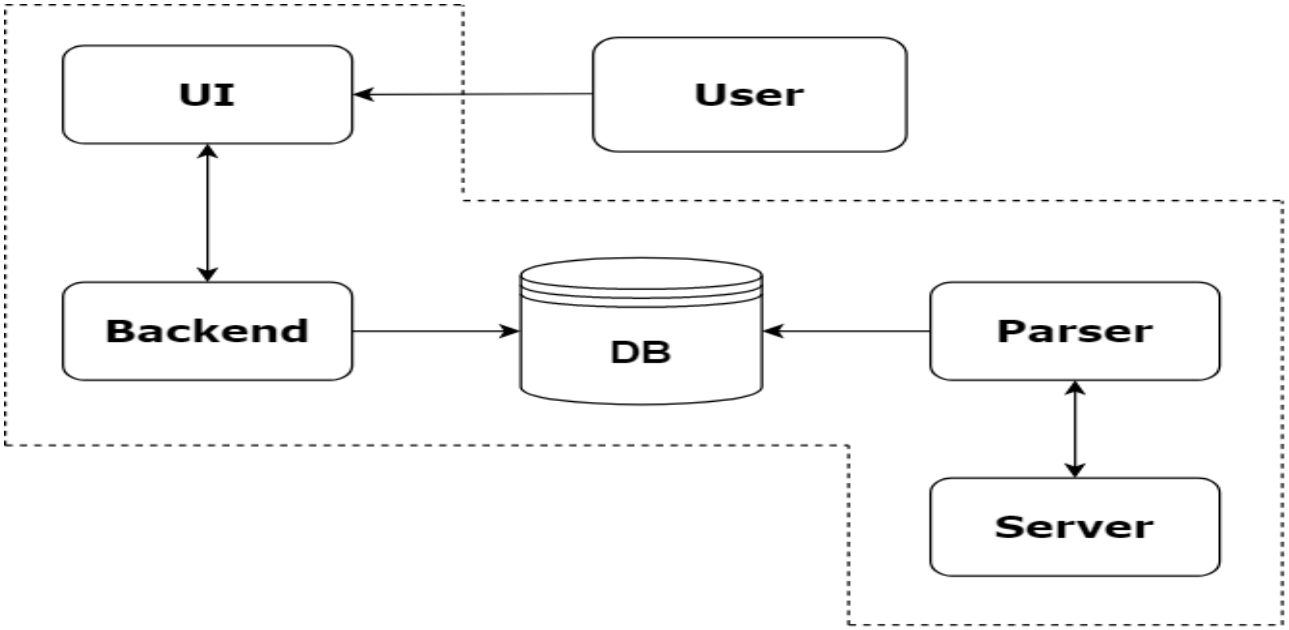
Четвёртый этап
Проектируется UI. Осуществляется написание программного кода для модуля 1 и модуля 2. Код комментируется. Каждая составная функцию тестируется на корректность возвращаемых данных.
Оформляется техническая документация.

Пятый этап
Тестируется функциональность программы. Проводятся необходимые корректировки программного кода (см. Третий этап) и повторные тестирования.

Шестой этап
Производится сбор данных температуры на протяжении 48 часов реального времени. По ним строятся графики, описанные в разделе Пользовательский интерфейс

Структурная и функциональная схема

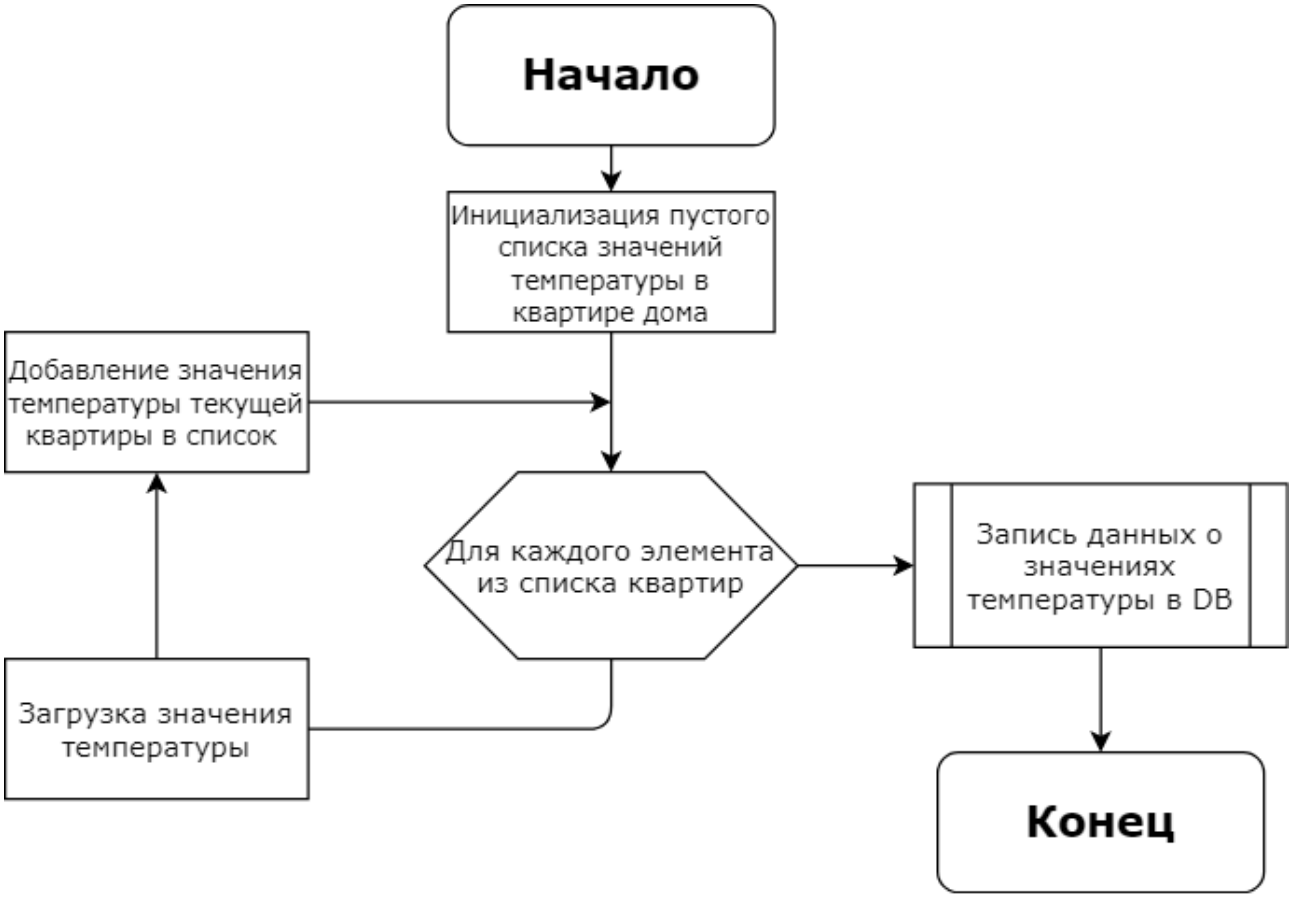
1. Структурная схема работы алгоритма



Функциональные схемы: Parser

2. Функциональная схема: внесения данных уличной температуры в городах

3. Функциональная схема: внесение значений температуры в квартире



Алгоритм работы программного продукта

Пользователь посредством специализированного UI взаимодействует с данными находящимися в DB посредством Backend

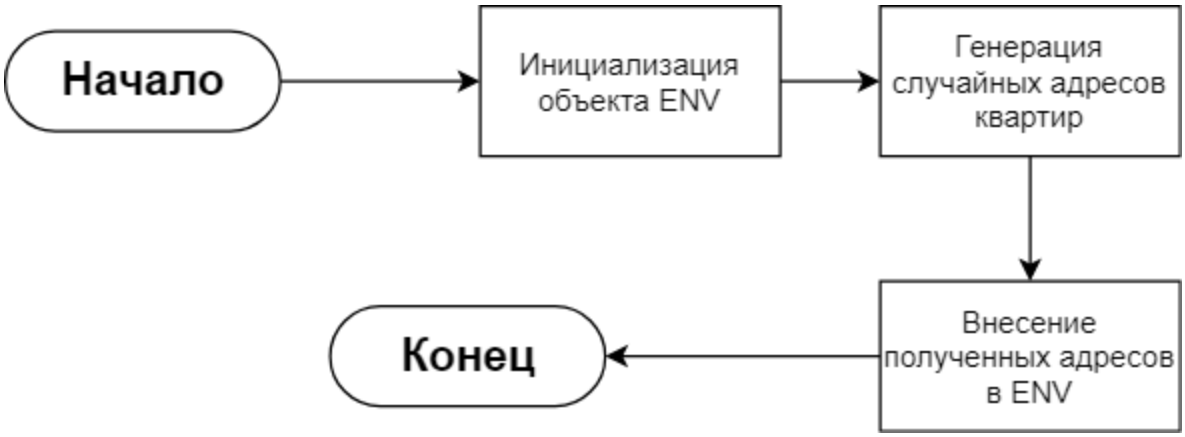
Раздел Backend производит визуальное отображение информации в качестве графиков и диаграмм, которое реализовано с помощью Matplotlib, также здесь осуществлена некоторая обработка информации в соответствии с регламентом, например, определение максимальных значений температуры.

DB хранит информацию, полученную с сервера в виде нескольких таблиц: таблица показаний уличной температуры в городах, таблица показаний температуры в множестве квартир одного города, таблица показаний температуры в одной квартире в каждом из городов, таблица показаний температуры в квартирах в нескольких (5) районах одного города.

Parser производит обработку данных полученных сервиса и передаёт их в DB в соответствующие таблицы. В том числе, с помощью модуля requests здесь

извлекается токен для взаимодействия с сервиса, производится извлечение данных на разных уровнях сервера; создаются «цели» для создания графика «температур множества квартир» и здесь же находится подпрограмма, отвечающая за время сбора данных с сервиса, реализованная с помощью модуля time
Ниже представлены функциональные схемы работы некоторых частей алгоритма:

1. Загрузка «целей» - адресов квартир, из которых будут считываться данные температуры

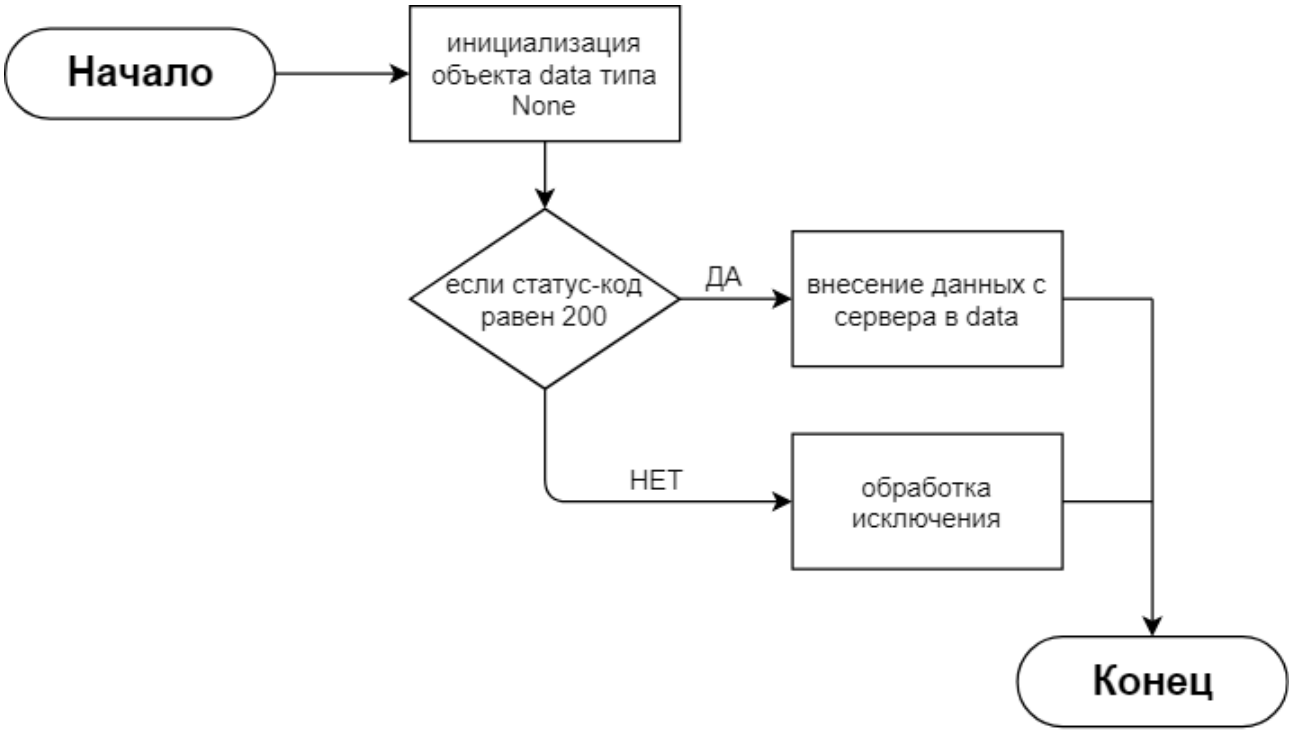


   ENV – это объект типа dict, содержащий список целей targets, url-адрес сервиса, token и время работы программы.

2. Сбор данных о каждой цели

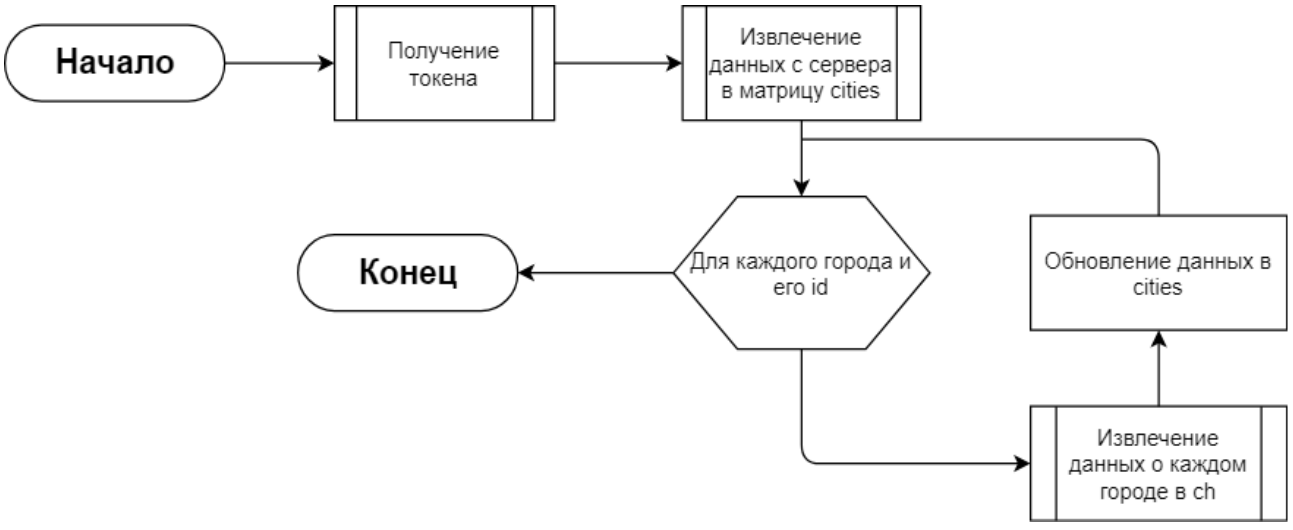
Алгоритм возвращает список показаний температуры в квартирах

3. Извлечение данных с сервера



Алгоритм возвращает data в формате json

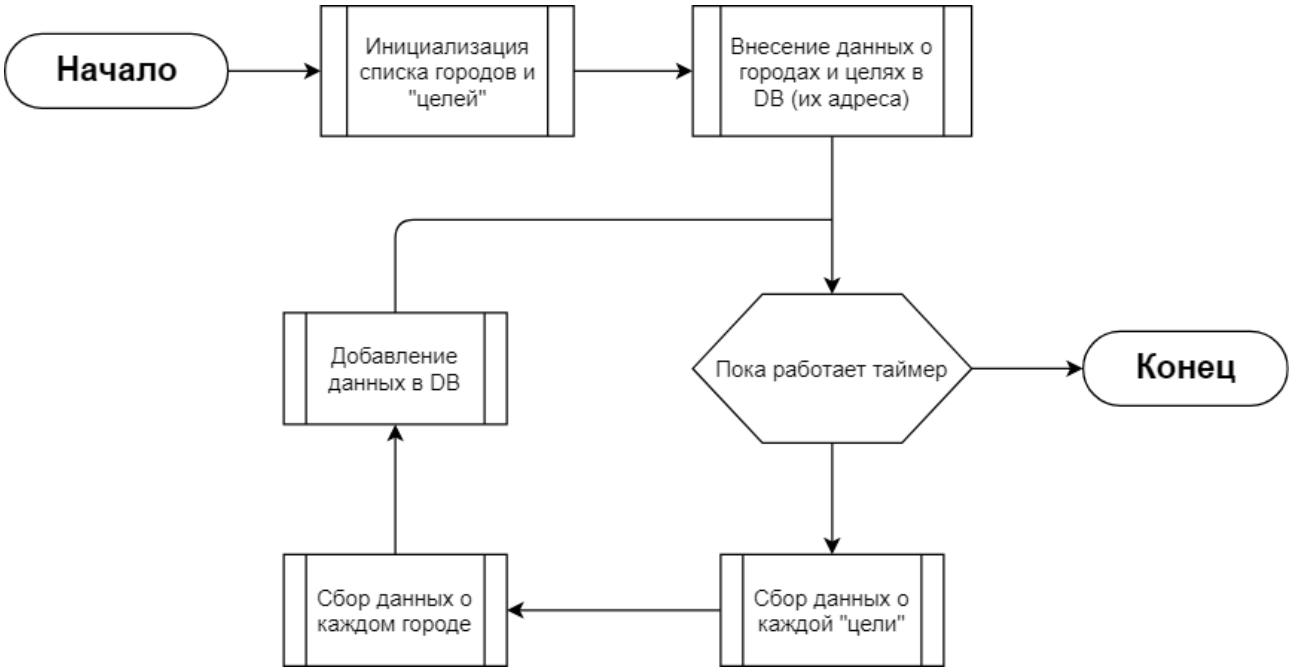
4. Извлечение подробных данных о каждом городе



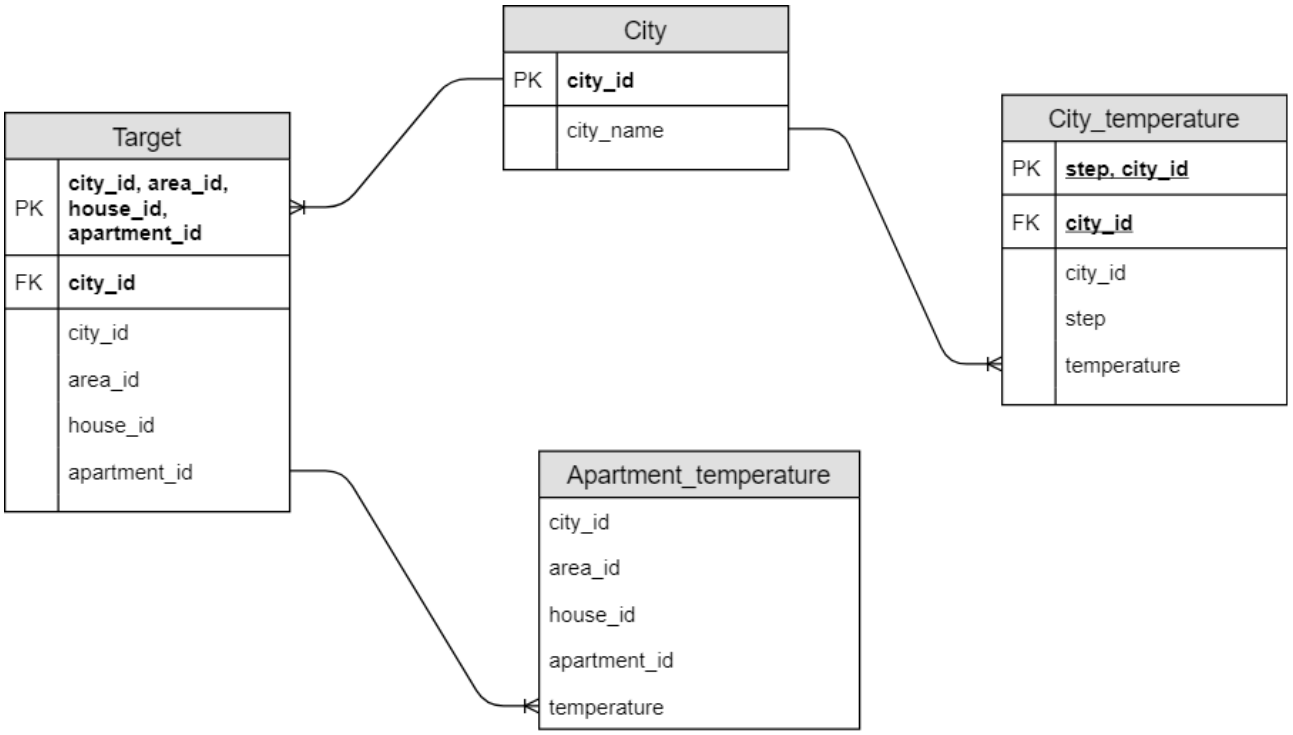
Дополнительно, перед использованием объектов cities и ch они проверяются на корректность (не тип None). Алгоритм возвращает cities

– объект типа dict. Использование этой программы подразумевает постоянные изменения в показаниях температуры

5. Основной алгоритм



ER-модель



Результаты разработки

Реализован программный продукт, позволяющий выводить данные температуры в различных квартирах и регионах. Разработан программный модуль, обеспечивающий взаимодействие с сервисом. Полученная с него информация хранится в реляционной базе данных. Взаимодействие с пользователем происходит через специализированный кроссплатформенный UI.
Выполнены все пункты технического задания, описанные в начале документации.

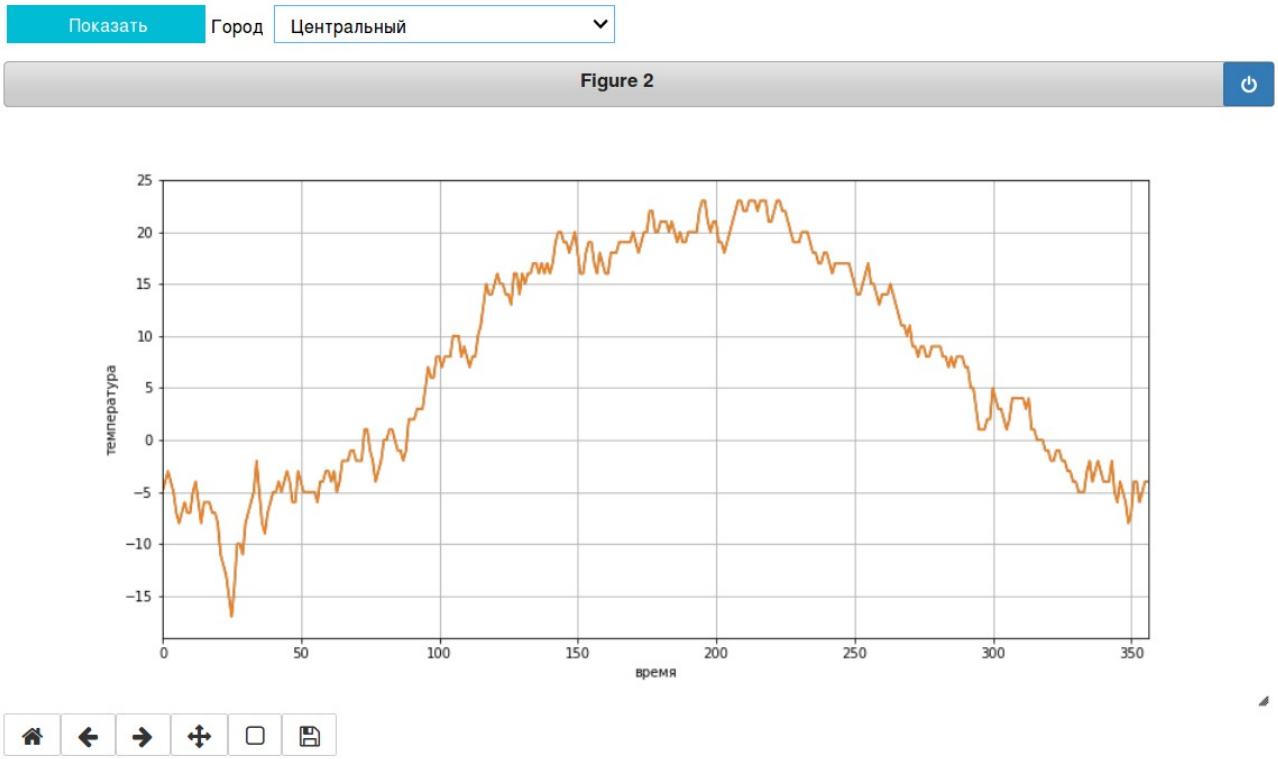
Номера заданий соответствуют порядку представленному в техническом задании

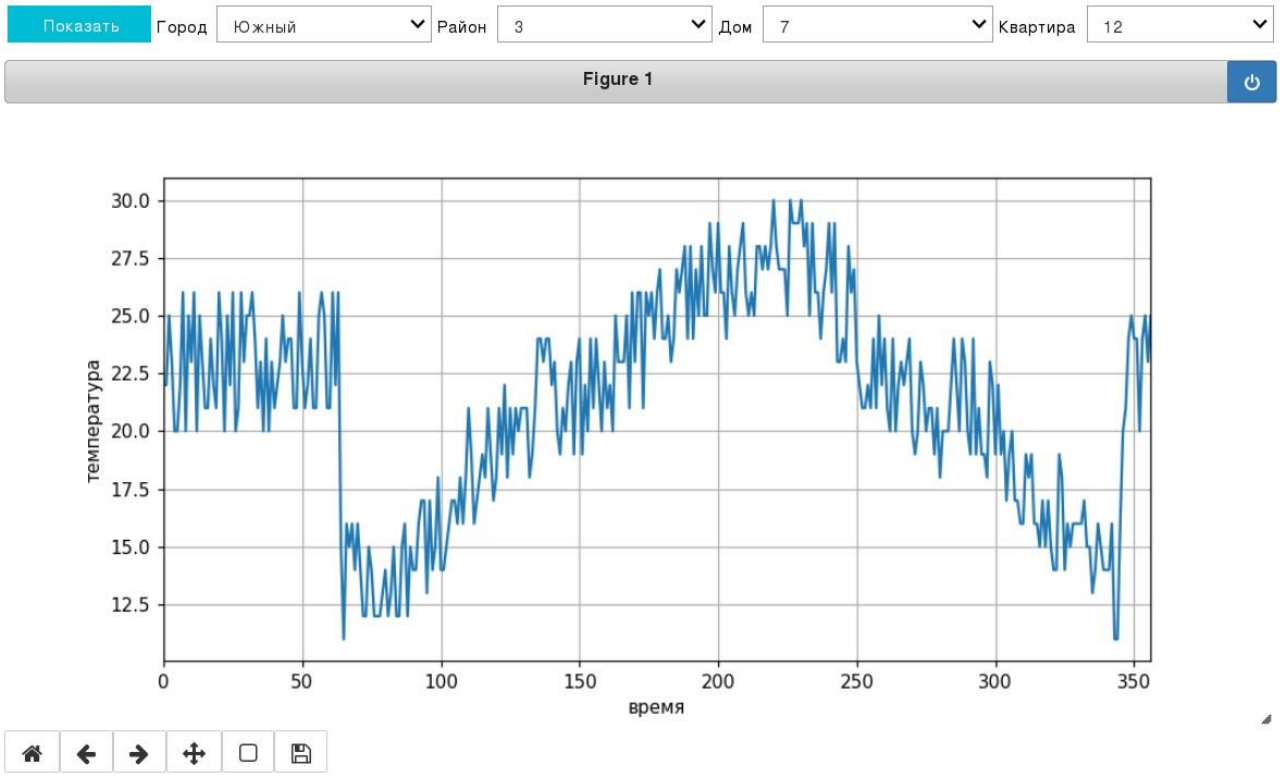
Внешний вид интерфейса

Пример получения данных температуры в реальном времени

23 °C

Примеры графиков





Программный код

Доступен по ссылке:

https://github.com/CangCiwei/PredProf