Программный модуль для кейса №5 “Сбор и обработка данных температуры”

Здравствуйте! Мы, команда “Солнышки”, представляем вам нашу работу по кейсу №5   
“Сбор и обработка данных температуры”.

Мы ставили перед собой задачу создать веб-приложение, в котором бы реализовывались функции из технического задания кейса, а именно получения данных от датчиков, хранение, обработка и наглядное представление собранных данных в виде графиков и диаграмм.

Для реализации всех поставленных задач мы вывели следующие Технические требования к оборудованию:

* Возможность создания полнофункционального веб-приложения,

Так как приложение должно быть кроссплатформенным, то удачнее всего было бы сделать именно веб приложение, которое открывается с любого устройства.

* Возможность делать HTTP-запросы API,

Так как взаимодействие с сервером происходит посредством HTTP-запросов, а получение данных от сервиса осуществляется с помощью GET-запросов.

* Возможность хранения информации,

Так как для сбора и хранения данных о температуре за 24 часа реального времени нужно иметь удобную структуру

* Возможность создания диаграмм и графиков

Для удобства представления обработанных данных

Чтобы реализовать все поставленные задачи, мы решили использовать язык программирования Python и фреймворк Django, а также другие библиотеки.

так как:

* Синтаксис и структура языка просты для понимания, написания и чтения, что облегчает процесс работы;
* Есть большое количество полезных библиотек и расширений языка, которые можно легко использовать благодаря унифицированному импорту и удобным интерфейсам. Они создадут каркас приложения и облегчат разработку.
* Всё является объектами в смысле ООП, поэтому работа удобна

Мы смогли подобрать нужные нам **библиотеки и фреймворки**, чтобы реализовать заявленный функционал, а именно:

* **Django** - свободный фреймворк для создания веб-приложений, с помощью которого мы создали само приложение.
  + Удобное использование;
  + Быстрота создания веб-приложений;
  + Удачная комплектация, наличие необходимых инструментов;
  + Масштабируемость;
  + Безопасность и защита от инъекций SQL, кросс-сайт подлогов, clickjacking’а и кросс-сайтового скриптинга
* **Requests** - стандартный инструмент для составления HTTP-запросов.
  + Облегчает трудоемкий процесс создания запросов
* **Sqlite3** - встраиваемая кроссплатформенная СУБД.
  + Поддерживает достаточно полный набор команд SQL.
  + Хранит данные, которые легко получить для обработки.
* **Matplotlib** - библиотека для визуализации данных двумерной (2D) графикой.
  + Удобно создавать графики и диаграммы.
  + Создаёт изображения
* **NumPy** - библиотека с открытым исходным кодом для языка, дающая возможность обрабатывать массивы данных.
  + Ускоряет работу с массивами в несколько раз
  + Оптимизирован для работы с MatPlotLib.
* **Shutil** – модуль, содержащий набор функций высокого уровня для обработки файлов, групп файлов, и папок.

И всё это находится в открытом доступе!

Использовав всё это, мы разработали наше веб-приложение. Его структурная и функциональная схема выглядит так:

\*рассказ\*

Суть работы приложения можно представить в виде workflow-диаграммы

\*ещё рассказ\*

Как уже отмечалось, для хранения мы использовали базу данных sqlite. Это ER-модель нашей базы данных. В таблицах хранятся данные о месте сбора (город или город-район-дом-квартира) и значения температуры, которые представлены в виде строки с разделителем “\*”. Связь с базой данных осуществляется с помощью моделей в Django.