**Big Data Management**

**Project P1**

**Report by:**

Tania González Puga

Edwin Santiago Delgado Rospigliosi

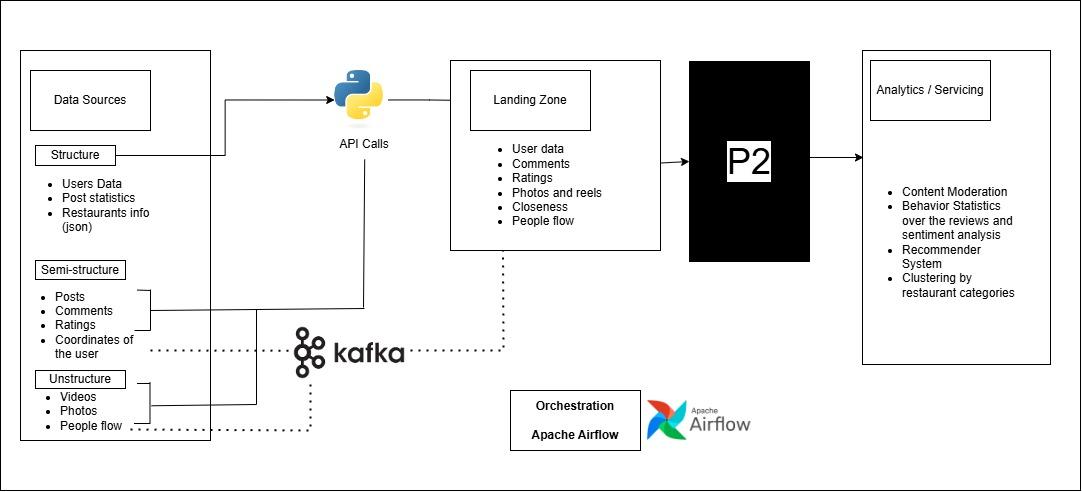
This project investigates the way in which different tools and environments relate to each other in order to be able to use different sources of information efficiently and accurately. In order to have the opportunity to use the largest number of tools and face the largest number of challenges possible, we propose a practical application related to everyday activities: something as simple as finding a place to go out to eat.

* **Contextualizing the problem**

The idea is to get a tool that has the capacity to offer us alternatives to **find a restaurant** based on our preferences and also on the availability in the area, the influx of people and the recommendations of other users in the most up-to-date way possible. We are going to define the domain only for restaurants in Barcelona city.

We chose the proposal based on the fact that we felt the need to develop something that can be used frequently and on a daily basis. And at the same time, we can have information about it. There is data available about shops and restaurants classified by type and rated based on popular reviews. Moreover, there is information on the Internet about the flow of people in most of these places thanks to Google and other platforms. So, all that remains is to try to collect as much of this information as possible and organize and use it, according to our possibilities.

Before going into the details of the design, we know that the project should collect a request from a user, with the type of food they would like to consume and their reference location, based on this information, data should be collected on the restaurants available at a certain distance from the user and deliver suggestions based on the aforementioned criteria, reviews, comments, flow of people and availability of the premises.

* **Architecture Design**

For the final solution, we are distributing the work on the following topics: Orchestration, Data Sources, Data Ingesting, Data Processing, Data Storage and Computing Data. Following the Pipeline design graphic is an explanation with details that we consider necessary to define each function and tool. It is important to note that although we try to ensure that the design is ambitious and that it meets the characteristics that we believe a project of this kind should have, we are prioritizing solutions within our reach, free and functional. Of course, the efficiency and quality of the information could be improved by using perhaps more sophisticated databases and paid API services, and we will discuss in each case what other alternatives could be used.

**Orchestration**

Since this is a project that requires the proper arrangement and flow of data between various processes and platforms, a tool is needed that allows us to integrate all the operations in an orderly and efficient manner. We take into account that there are several alternatives, such as Azure Data Factory or AWS Data Pipeline, but since for other tasks we will use tools from the same family, we opted for Apache Airflow, which among other things will allow us to organize each task, operator and system in a control flow on Python.

**Data Sources:**

The idea is that the project can use both previously structured data in a data warehouse and data that comes in real time. So we organized it in the following way:

**Structure**

Since this is not the objective of the project, it is best to organize the user data in a **data warehouse** beforehand. In addition, the results of processing the data flow received in real time about the restaurants and users that we deem appropriate must be included here.

**Semi-structure or Unstructure**

We cannot collect real-time data directly, since in theory it is information from the restaurants and this project is not focused on interacting with them, but directly with the users. Therefore, to obtain this information it is necessary to resort to other methods and services:

* **Google Maps scraping/Google places API:** Google certainly has the most complete and up-to-date information, and its services complement each other very well, so we can extract comments, ratings, images, videos, locations, flow of people, among others. The APIs can be used for free as long as their data flow limits are not exceeded. In any case, it will be necessary to implement some scraping system to facilitate extraction in certain cases.
* **Yelp Fusion API:** It specializes in collecting reviews and ratings of local businesses. It is also free and many projects use it, so there is a bibliography available.
* **Zomato API:** It offers access to a wide range of restaurant data, including menus, reviews and cuisines. It is free as long as the usage limit is respected.
* **OpenMenu:** It is based on a community-driven platform that allows restaurants to publish their menus and be discovered.
* **Open-data Ajuntament de Barcelona:** offers a dataset with information about the restaurants that are located in the city, and it is updated weekly.

**Data Ingesting**

Even though we have an idea of ​​what our sources of information are, we must take charge of extracting and processing the data that comes in real time. For this task, the most commonly used tool seems to be Apache Kafka, whose use we will explain later. It has a wide range of information and usage examples, so it is also convenient.

**Data Processing**

As we have pointed out previously, data processing is different when it comes from a database or from a source that is updated in real time. Fortunately, there are also different libraries for each process.

* **Batch Processing**

We will use this type of processing with data that does not need to be processed immediately, such as ratings, comments, data from the restaurants, photos, etc. We will use Apache Hadoop, which specializes in processing large volumes of information and can run even if some intermediate processing step fails. However, we must use it in conjunction with Apache Hive, which will allow us to run the SQL for these data sets.

* **Stream Processing**

For data arriving in real time, we will use **Apache Kafka**, which will be in charge of capturing data from the APIs. It specializes in capturing data from different sources and allows us to control it in real time. On the other hand, Apache Spark will allow us to process the data collected in real time through interactive nodes and we can also control it through SQL queries or machine learning processes. As both tools are often used together in the industry, there is a lot of information available on the subject and everything is established using Python.

**Data Storage**

In order to storage data, we will use:

* **Delta Lake**, which is a data lakehouse, in order to store historical trends and analytical insights for posterior ML models to predict recommendations.
* **MongoDB**, to deal with real-time and operational data processing. For example, people flow. (maybe for restaurant info from bcn too?)