UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Big Data Management

Landing Zone

Project 1 Report

April 7, 2024

Authors:
Dilbar Isakova
Shofiyyah Nadhiroh

Supervisor:
Sergi Nadal

Contents

1 Project Overview

As a part of our BDMA Joint Project, we are developing an innovative app designed to connect people in Barcelona, including locals, exchange students, and professionals relocating to the city. By matching users with companions sharing similar interests and facilitating participation in various local activities, the app aims to foster genuine friendships and a welcoming atmosphere for newcomers, enriching the Barcelona experience for everyone involved.

Additionally, our app introduces a unique feature where users can post and view recommendations for various places, including attractions, restaurants, and more. This allows users to discover place recommendations tailored to their interests, further enhancing their engagement with the city and its community.

Our app leverages advanced tools such as Google Maps APIs, web-scraped Meetup Platform data, and predictive analytics to boost engagement and foster a sense of community in Barcelona. This document details the architecture designed to achieve our custom, interactive experience.

1.1 Scope of the Project

Our project is dedicated to developing a recommendation system that helps people find exciting places near them, based on what they like and what's currently popular. We use information from a website called Meetup.com to keep up with the latest trends and hobbies that people are interested in. This helps us make sure our suggestions are both fun and timely. We also use Google Maps to help find places that are close by, making it easier for users to visit these recommended spots. Our goal is to make a system that is easy to use and helps people discover new things in their area.

2 Data Source

Meetup JSON

```
{"_id": {"$oid": "65f8d6b76c9cc986cb744712"},
  "title": "Coffee Walk & Beach Sunset Hangout to Make New Friends",
  "hosted_by": "Mary G.",
  "event_time": "Friday, April 5, 20246:30 PM to 8:30 PM CEST",
  "gmaps_link": "https://www.google.com/maps/search/?api=1&query=41.39655%2C%202.194162",
  "vanue_location": "Itnig Cafe",
  "vanue_location_detail": "C. de Pujades, 100, Barcelona, CT",
  "description": "FRI, APR 5, 6:30 PM. Coffee Walk & Beach Sunset Hangout to Make New Friends! Ladies of Barcelona, ages 18-29, you're invited to join us to get coffee to-go at Itnig Cafe and then walk to the beach to enjoy the sunset and some snacks!,
  "topics": ["Coffee","Social","Make New Friends","Picnics","Cafe Lovers"]}
```

User data for post recommendation JSON

```
{"time": "2011-12-30",
  "text": "You need to pay for the internet (2 euro / half-hour or 6 euros per day),breakfast is pretty dull, but location is really close to the metro station (L3 line - Mundet). Staff was friendly and helpful!",
  "rating": 4,
  "place_name": "ALIMARA",
  "place_category": "accommodation"}
```

Google Map APIs JSON

```
{"business_status" : "OPERATIONAL",
  "formatted_address" : "Parc de Montjuic, Sants-Montjuic, 08038 Barcelona, Spain",
  "geometry" :{"location":{
                "lat" : 41.3686304,
                "lng" : 2.1598500},
  "icon" : "https://maps.gstatic.com/mapfiles/place_api/icons/v1/png_71/museum-71.png",
  "icon_background_color" : "#13B5C7",
  "icon_mask_base_uri" : "https://maps.gstatic.com/mapfiles/place_api/icons/v2/museum_pinlet",
  "name" : "Joan Miro Foundation",
```

2.1 System Architecture

We implemented the master-slave architecture as it is designed to handle vast amounts of data by distributing the data and computation across multiple machines. This setup allows Hadoop to process big data tasks in a parallel and fault-tolerant manner. The architecture of our system as shown in the below diagram.

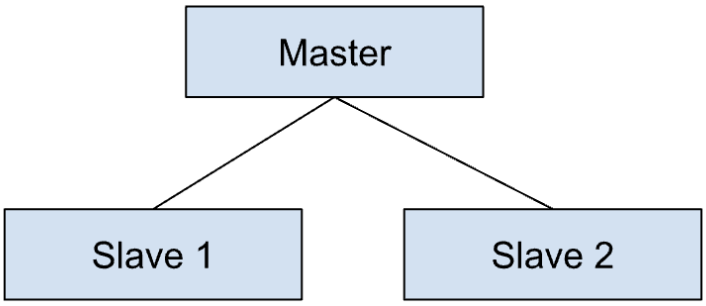


Figure 1: System Architecture

3 Data Collectors

In our project, we utilized varied data collection methods to enhance user experience and analytics. Employing web scraping, API requests, and user interactions, we compiled datasets from Meetup, Google Maps, and user content. This approach enabled access to diverse information, supporting our project's objectives.

Meetup Platform: Our initial phase involved collecting data from the Meetup platform via web scraping. This process enabled us to extract public information about events, group activities, and member interests. Utilizing Python scripts and tools like Beautiful Soup, we automated the extraction of data, ensuring efficiency and accuracy. This dataset provided us with a deep understanding of community engagement and social trends.

Google Maps API: We utilized the Google Maps API to gather detailed information on Barcelona's restaurants, museums and gyms. This process involved:

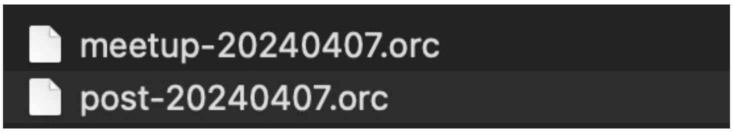
- API Key Acquisition: By setting up a project in the Google Cloud Console and enabling necessary APIs, we obtained an API key, granting our application access to Google Maps services.
- API Requests: Utilizing the Places API, we crafted targeted requests to extract restaurant data based on keywords, ensuring the inclusion of relevant parameters like location and search radius.
- Data Extraction: The responses, in JSON format, provided us with essential details such as names, addresses, and contacts of restaurants, which we then filtered and organized for our project needs. This streamlined approach allowed us to efficiently collect valuable location-based data, enhancing our project's location-based services and analytics.

User Data for Post Recommendations: Finally, we collect user data directly from our platform using a NoSQL database, where users contribute posts about their interests and activities. This approach allows us to tailor our recommendations more accurately, as we base them on the specific preferences and interactions of our users. By analyzing this collected data, we develop algorithms that deliver personalized content, significantly enhancing the overall user experience by making our suggestions more relevant and engaging.

4 Landing Zone

4.1 Temporal Landing

In our system, we use two Python files, **'load_post_data.py'** and **'load_meetup_data.py'**, to handle data collection. These scripts are designed to retrieve information locally. **'load_post_data.py'** is responsible for loading user-generated content, capturing insights into user interests and behaviors from posts they share on our platform. Similarly, **'load_meetup_data.py'** gathers data related to current trends and events from local Meetup information. This process ensures that our recommendations are both personalized and up-to-date. The collected data is temporarily stored, then moved to a more permanent storage solution at the end of each day, ensuring our system is ready for fresh data the next day. This approach allows us to maintain an efficient and dynamic data handling mechanism, essential for delivering relevant and engaging user experiences.

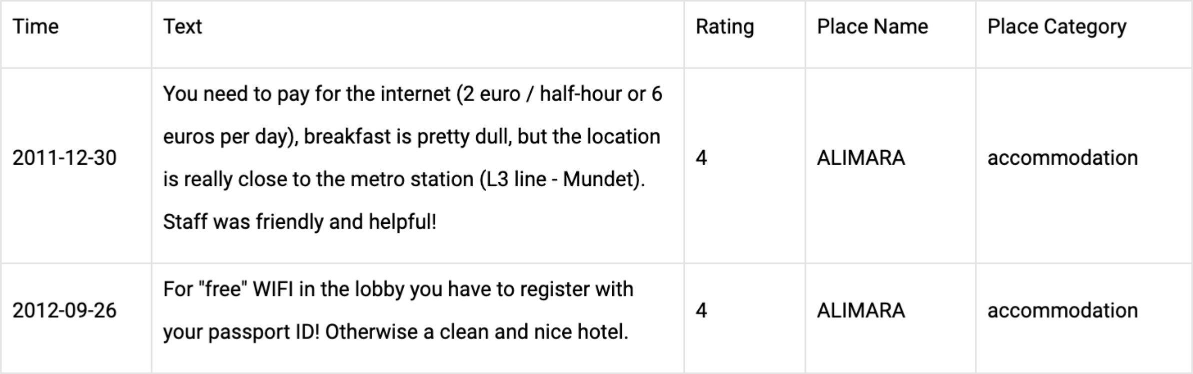


meetup-20240407.orc
post-20240407.orc

Figure 2: Temporarily stored data

4.2 Data Format

Following the conversion from JSON to ORC within the temporary landing zone, the data format appears as follows:



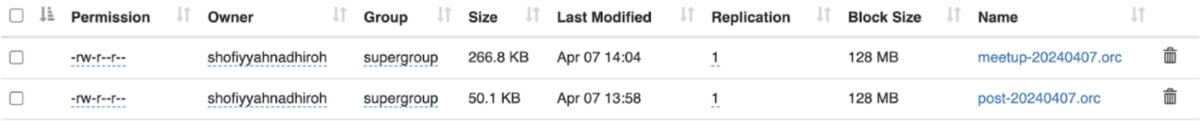
Time	Text	Rating	Place Name	Place Category
2011-12-30	You need to pay for the internet (2 euro / half-hour or 6 euros per day), breakfast is pretty dull, but the location is really close to the metro station (L3 line - Mundet). Staff was friendly and helpful!	4	ALIMARA	accommodation
2012-09-26	For "free" WIFI in the lobby you have to register with your passport ID! Otherwise a clean and nice hotel.	4	ALIMARA	accommodation

Figure 3: Data Format Appearance

The figure presented illustrates the data format specifically for User Data related to Post Recommendations. This format was selected for display due to its resemblance to the data formats used for Meetup and Google Maps data.

4.3 Persistent Landing

At the end of each day following temporal landing, we receive two orc files in HDFS. These files are then saved in HDFS as persistent landings. And this is how it looks in the HDFS, as illustrated below:



□	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
□	-rw-r--r--	shofiyyahnadhiroh	supergroup	266.8 KB	Apr 07 14:04	1	128 MB	meetup-20240407.orc
□	-rw-r--r--	shofiyyahnadhiroh	supergroup	50.1 KB	Apr 07 13:58	1	128 MB	post-20240407.orc

Figure 4: HDFS

In our pursuit of developing a robust and scalable data management solution for our business, we have decided to adopt the Hadoop Distributed File System (HDFS) and convert our JSON files into ORC format. The rationale behind these choices is rooted in the technical strengths and business benefits they offer.

HDFS is an integral component of our data strategy, primarily due to its scalability and reliability. It is designed to store vast amounts of data and provides quick access, which is essential as our customer base grows. This scalability ensures that as we onboard more customers, we can effortlessly expand our storage capabilities without compromising on performance. Moreover, HDFS's ability to serve a large number of clients simultaneously by adding more machines to the cluster aligns perfectly with our expectation of growing client traffic. This aspect of HDFS ensures that our system can scale horizontally, adapting to increased workload by simply incorporating more nodes into the cluster.

The decision to use HDFS also stems from its high reliability. By maintaining multiple copies of data and automatically redeploying processing logic in the event of a failure, HDFS guarantees data integrity and system stability. This feature is crucial for us, ensuring that our service remains uninterrupted and reliable, instilling confidence in our clients regarding the robustness of our system.

Another pivotal reason for choosing HDFS is its synergy with the MapReduce framework, which we plan to implement in the upcoming phase. Since HDFS is the file system component of Hadoop, upon which MapReduce operates, using HDFS allows us to leverage this powerful framework efficiently. Our goal is to use MapReduce to process vast datasets by distributing the computation across multiple nodes, significantly enhancing our ability to calculate all matching resources and tasks within a single time unit.

Transitioning our data storage format to ORC from JSON is another strategic move aimed at optimizing our data processing capabilities. ORC's efficient data compression and faster read capabilities mean that our data-intensive operations become much more efficient, reducing the time and computational resources needed for data processing.

5 The BPMN Diagram - AmiGo

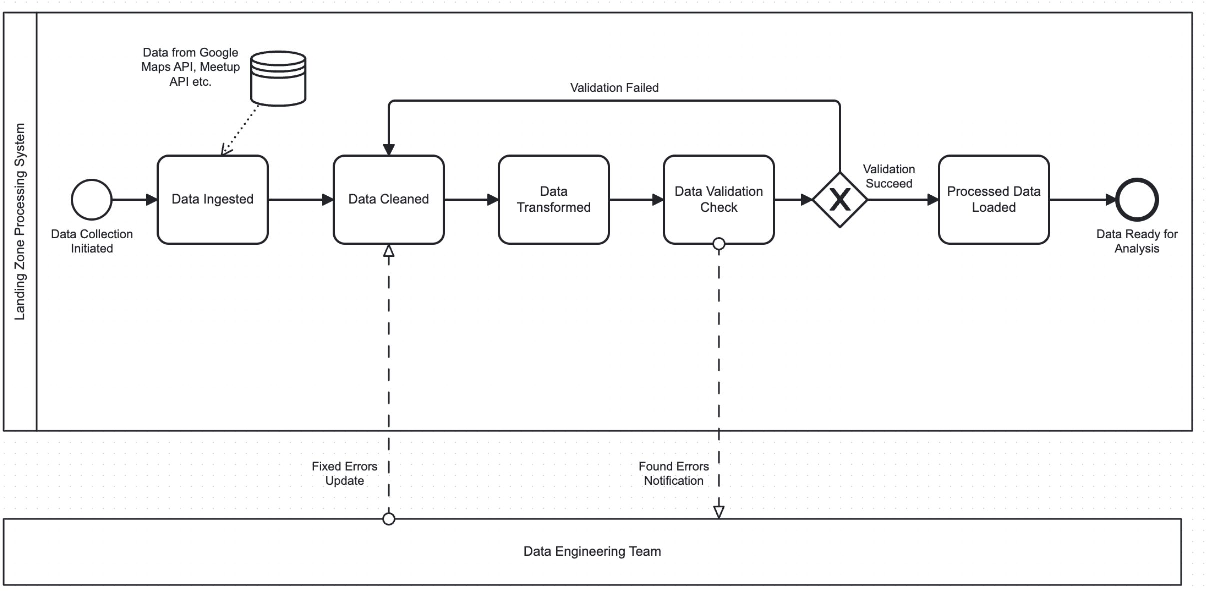


Figure 5: BPMN Diagram

6 Appendix A - Landing Zone Script

load_meetup_data.py

```python
import os
import pandas as pd
from subprocess import PIPE, Popen
import time

timestr = time.strftime("%Y%m%d")

meetup = pd.read_json(f'./data/meetup-{timestr}.json')
meetup.to_orc(f'./data/meetup-{timestr}.orc')

file_name = f'./data/meetup-{timestr}.orc'
hdfs_path = os.path.join(os.sep, 'user', 'shofiyyahnadhiroh', file_name)

put = Popen(["hadoop", "fs", "-put", file_name, hdfs_path], stdin=PIPE, bufsize=-1)
put.communicate()
```

load_post_data.py

```python
import os
import pandas as pd
from subprocess import PIPE, Popen
import time

timestr = time.strftime("%Y%m%d")

post = pd.read_json(f'./data/post-{timestr}.json')
post.to_orc(f'./data/post-{timestr}.orc')

file_name = f'./data/post-{timestr}.orc'
hdfs_path = os.path.join(os.sep, 'user', 'shofiyyahnadhiroh', file_name)

put = Popen(["hadoop", "fs", "-put", file_name, hdfs_path], stdin=PIPE, bufsize=-1)
put.communicate()
```