FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Scalable BI Cloud Solution

Pedro Junqueira Teixeira Coelho de Melo

DISSERTAÇÃO

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador FEUP: Prof. Luís Paulo Reis

Orientador Externo: Eng° Francisco Capa

July 17, 2020

Resumo

A era digital é cada vez mais afirmativa na sociedade atual, revelando a supremacia tecnológia no quotidiano de cada um de nós. Da Saúde ao Desporto, e da Economia à Política, uma por uma se vai convertendo, progressivamente, aos avanços da virtualização e, de certa forma, realçando-o com orgulho para as demais áreas de saber.
O Mercado da Bolsa foi um dos primeiros a assumir essa necessidade e desde cedo criou e aumentou plataformas acessíveis de todo o mundo. Com isto, os volumes de mercado dispararam e terminologias para tais grandezas tornaram-se banais. Como efeito, o mercado de ações tem hoje em dia uma das maiores, senão a maior, volatilidades associadas o que contribui para imprevisibilidade e incerteza na hora de investir.
De maneira a melhor prever essa imprevisibilidade e incerteza, surge a arquitetura *Business Intelligence*, proporcionando a possibilidade de analisar detalhadamente o mercado e, consequentemente, compreendê-lo melhor.
O presente projeto procura desenvolver uma solução altamente escalável que assente na arquitetura *Business Intelligence* aliada a serviços *cloud*. A solução tem como foco o recurso a um data lake altamente otimizado conjugado com a compressão dos ficheiros nele armazenados, o que permite aplicar os conceitos fundamentais de BI de forma a que maximize a performance e a escalabilidade, e minimize os custos.
A extração diária de dados, desenvolvida em *Python*, a implementação do processo ETL e o armazenamento dos dados num data lake que preencha os requisitos mencionados, tiveram como suporte os recursos *Microsoft Azure*. A análise visual, que suporta com practicidade a solução desenvolvida, foi implementada em *Microsoft Power BI* e consiste em dashboards interativos que permitem verificar com especificidade vários factores que compõem a análise pública de empresas, tal como a performance da própria solução a ser apresentada no presente relatório.

Palavras-chave (Tema): *Mercado da Bolsa, Business Intelligence, Cloud Computing, ETL, Data Warehouse, Data Lake*

Palavras-chave (Tecnologias): *Microsoft Azure, Microsoft Power BI, Python, SQL*

Abstract

The digital age is increasingly affirmative in today's society, revealing the technological supremacy in the daily lives of each of us. From Health to Sports, and from Economy to Politics, one by one it is progressively converting itself to the advances of virtualization and, in a certain way, emphasizing it with pride for the other areas of knowledge.
The Stock Market was one of the first to assume this need and from an early stage it created and increased accessible platforms from all over the world. With this, market volumes skyrocketed and terminologies for such greatness became commonplace. As a result, the stock market today has one of the largest, if not the largest, associated volatilities, which contributes to unpredictability and uncertainty when it comes to investing.
In order to better predict this unpredictability and uncertainty, the architecture *Business Intelligence* arises, providing the possibility of analyzing the market in detail and, consequently, understand it better.
The present project seeks to develop a highly scalable solution that is based on *Business Intelligence* architecture allied with *cloud* services. The solution focuses on the use of a highly optimized data lake combined with the compression of the files stored in it, which allows applying the fundamental concepts of BI in a way that maximizes performance and scalability and minimizes costs.
The daily data extraction, developed in *Python*, the ETL process implementation and the data storage in a data lake that fulfills the mentioned requirements, were supported by *Microsoft Azure* resources. The visual analysis, which practically supports the solution developed, was implemented in *Microsoft Power BI* and consists of interactive dashboards that allow to verify with specificity several factors that make up the public analysis of companies, such as the performance of the solution itself to be presented in this report.

Keywords (Topic): *Stock Market, Business Intelligence, Cloud Computing, ETL, Data Warehouse, Data Lake*

Keywords (Technologies): *Microsoft Azure, Microsoft Power BI, Python, SQL*

Agradecimentos

Agradeço, em primeiro lugar, aos meus pais
por terem tornado este percurso possível e por me fazerem acreditar em todas as minhas ambições.

Agradeço ao Eng° Francisco Capa
pela incansável paciência no acompanhamento deste projeto e por todos os ensinamentos que me passou.

Agradeço ao Professor Luís Paulo Reis
por ter aceite este desafio e por todo o apoio e dedicação ao longo do projeto.

Agradeço à BUSINESSTOFUTURE,
pela oportunidade concedida de ingressar numa equipa de profissionais de excelência, sempre dispostos a ajudar. Um obrigado especial ao Eng° João Pereira pela sua incansável disposição de me apoiar nos obstáculos mais dificeis deste projeto.

Por fim, agradeço a todos os colegas e amigos que estiveram comigo diariamente e que, de alguma, forma contribuiram para o realizar deste projeto com sucesso.

Um obrigado a todos

"In God we trust.
All other must bring data."

W. Edwards Deming

Contents

List of Figures

List of Tables

Acronyms

BI	Business Intelligence
CSV	Comma-Separated Values
DW	Data Warehouse
ETL	Extract, Transform, Load
SA	Staging Area

Chapter 1

Introduction

1.1 Framework

This report is a follow-up to the project developed within the Dissertation discipline (DISS) of the MIEEC - FEUP. This project, developed in partnership with B2F - BUSINESSTOFUTURE, focuses on building a cloud-based Business Intelligence (BI) solution that maximizes scalability. B2F intends to explore knowledge in data processing and analysis through the use of Extract, Load and Transform (**ETL**) technologies in cloud environment. Traditional ETL processes form the backbone of all data analysis tools from a Data Warehouse and this has been the way to process and analyze large amounts of data. However new technologies and features are remodeling the way these techniques are being applied, giving new emphasis on cost reduction and scalability.
In recent years, there has been a significant development in the volume of transactions in the stock market. Bearing in mind the volatile but powerful factor that characterizes the stock market, it becomes valuable to manipulate data related to this colossal market.
In this way, the extraction and treatment of these data leads to the creation of analysis models, which make it possible to make analysis and studies leading to an improvement in the efficiency of the decision-making process.

1.2 Goals

It was proposed a solution ranging from data extraction from the respective data sources, until an analysis demonstrated in interactive dashboards, so that the end user can easily access the information he wants to analyse. Thus, the major goals of this project are:

- Scheduled extraction of data, having stock market websites as sources;
- Allocation of this data to a cloud environment database;
- Data processing processes;
- Ingestion of this data in a scalable and partitioned platform in a cloud environment;

- Development of dashboards and analysis.

Although the project focuses on the BI process and its compliance with the requirement of maximizing scalability, dashboards were also turned into a goal, as a way to analyze the stock market itself and demonstrate in practice the use of data from these processes.

1.3 Approach

In order to obtain the final solution, consisting of a highly scalable data repository and analysis dashboards for the files stored therein, it was necessary to implement several steps.
Briefly, and as a first step, scripts were implemented for Web Scraping (extraction of data from websites), which run every hour, and store the extracted data in Comma-Separated Values (CSV) format, in a cloud. Afterwards, the data is loaded into a database to be submitted to the transformation process. Finally, as soon as this data is in the Data Warehouse, it is exported to the repository, from where the reporting tool extracts the data and implements the final solution.
In Figure 1.1 it is possible to visualize the data flow along the process described as well as understand the approach taken in implementing this project.

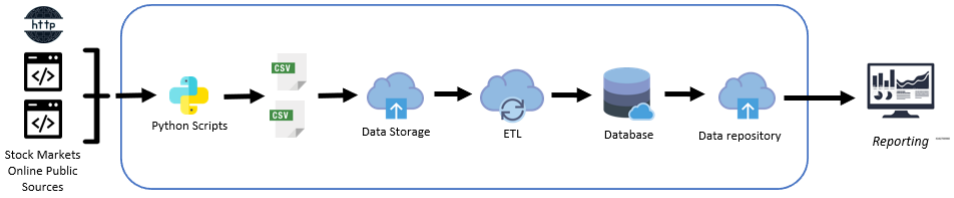

Figure 1.1: Data flow for the approached solution

1.4 Work Plan

The work plan was defined using the sprint methodology. In order to define the execution planning for this project, sequential objectives with stipulated length of time and corresponding sprint have been defined. Table 1.1 shows the plan developed.

Sprint	Week Length	Description
1	2	Effort to understand the Azure cloud platform and the resources to use. Modeling and understanding of the various implementation processes.
2	3	Development and implementation of the mechanism for scheduled data extraction and storage in cloud environment.
3	7	Cloud ETL Development
4	2	Development of dashboards and final analysis using the Reporting tool
5	2	Tests and optimization of the implemented data flow process.
6	4	Results analysis, closure of model and conclusions, and writing of the dissertation.

Table 1.1: Work Plan

1.5 Presentation of the Company

BUSINESSTOFUTURE (B2F) is a Portuguese company founded in 2006 established in Porto. B2F offers services such as business solutions for analysis of data and decision support (Business Intelligence), and also has a sector dedicated to custom development.
The core of the company is the development of BI solutions, as they have best practices in developing this type of solutions and high experience, derived from its extensive market knowledge and its range of large-scale customers.
The company is composed of 26 elements in its totality, with experience in technologies like Microsoft, Cloud, Power BI, among others. The stability in the development of quality solutions and best practices guarantee lasting partnerships with clients such as the Grupo Amorim, Parfois, Sixt, Sonae, STCP, Metro do Porto, JAP, among others.

Figure 1.2: BusinessToFuture logotype

1.6 Structure of the Dissertation

This document is structured in six different and tangible chapters, that describe the stages taken on the development of the project.
Chapter 1 introduces the project and its motivations, giving a major scope of what is this solution and how it was planned to be developed.

Chapter 2 is about the literature review where essential subjects are addressed and described to fully understand the development of the project in question.
An analysis of the proposed solution is made in Chapter 3, where the essential topics to achieve the solution for this project are described, as well as the approach and architecture by the different stages of implementation.
The phased implementation of the solution is described in Chapter 4, containing all the processes included in it that lead to the desired outcome.
Chapter 5 is assigned the presentation of the results obtained from the implementation of the project in question, more specifically, a presentation of the final solution developed, as well as the comparison between the solution proposed and other traditional and alternative solutions.
Finally, Chapter 6 presents conclusions on the outcome of the project, as well as objectives achieved, limitations and future work for this project.

Chapter 2

Literature Review

2.1 Important Concepts

2.1.1 Data

Cambridge dictionary defines data as "information, especially facts or numbers, collected to be examined and considered and used to help decision-making, or information in an electronic form that can be stored and used by a compute"[5]. In technology data could be considered information in form of documents, images, audio clips, software programs, or other types of data, processed by a computer.

2.1.2 Big Data

Big data refers to the large and diverse data sets that grow at ever-increasing rates. In 2012, Harvard Business Review described data exponential growth as "about 2.5 exabytes of data are created each day, and that number is doubling every 40 months or so"[1].
The keywords when talking about big data are **V**olume, **V**ariety and **V**elocity, the three **Vs**.

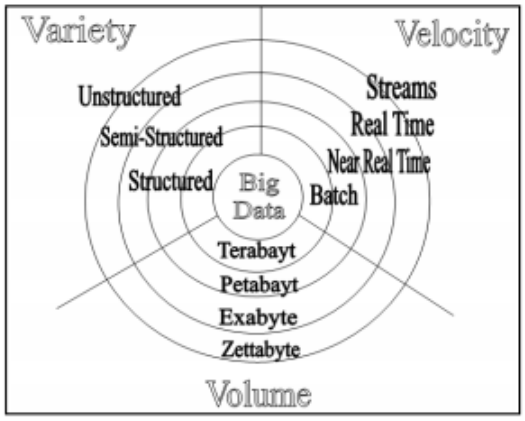

Figure 2.1: The three Vs of big data [1]

Variety is what makes big data really big. All this information comes from multiple sources and can be divided into three types: structured, semi-structured and unstructured. Structured data is already tagged and sorted data whereas unstructured data is random data and difficult to analyze. Semi-structured data becomes the middle term, it "does not conform to fixed fields but contains tags to separate data elements".[1]

Volume represents the size of data. Already crossed the terabytes dimension and in some cases the petabytes (1024 TB). Traditional storage and analysis techniques became outstripped by this grand scale.

Velocity is required to answer the need of manipulating and processing information in this scale. Ideally "big data should be used as it streams into the organization in order to maximize its value."[1]

2.1.3 ETL

ETL stands for extract, transform, load and is the process of extracting data from multiple sources, transform it to suit business needs and load it into a proper database destination, usually a data warehouse.

The first step in **ETL** is the extraction. During this step, data is identified and taken from many different location and because it is not typically possible to pinpoint the exact subset of interest, the volume of data extracted is generally wider than needed to ensure it covers the needs. Sources can be files, spreadsheets, web pages, database tables, etc.

The next step in the **ETL** process is transformation. As mentioned, the data extracted may fit the purpose needs but it may come with unnecessary or in separated sources. So in this step operations such as cleaning, joining, validation or generation of calculated data based on existing values transform the convert the data into the appropriate format and ready for the next and last step.

The final step is simply load the transformed data into a target destination, that may be a database or a data warehouse.

A typical ETL process seems like the one in figure 2.2.

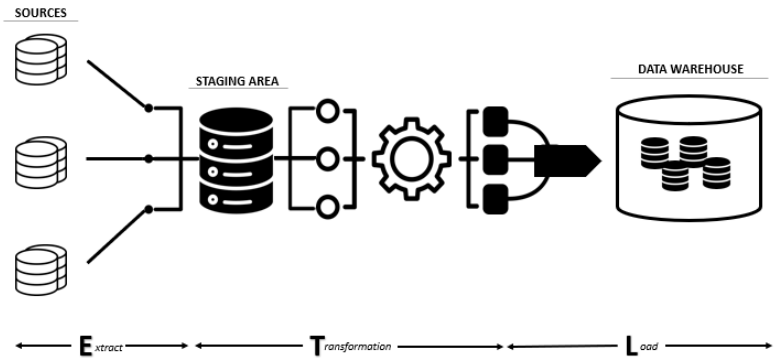

Figure 2.2: Data Flow in an ETL process

2.1.4 Business Intelligence

Technically Business Intelligence (BI) states for extracting, transforming, managing and analyzing business information. It's purpose is to support better business decision making, being a data-driven Decision Support System (DSS). This whole process is based on large data sets, mainly data warehouses, and has the mission of disseminate information and insights across the whole company, from strategic to operational level.
As shown in figure 2.3 a typical BI system consists of four levels of components and a metadata management module. These different levels cooperate with each other to facilitate the basic BI functions.

- **Operational System Level** - business operational systems are mainly online transaction processing (OLTP) systems, supporting daily business operations. Typical OLTP systems are customer order processing systems, financial systems and human resource management systems.
- **Data Aquisition Level** - this level includes three phases: extracting, transforming and loading (ETL). Data is first extracted from OLTP systems that usually produce huge amounts of data. Then is transformed accordingly to transformation rules achieving a clean, unified and aggregated data set. Finally, data is loaded into a central data warehouse. This ETL process is the most fundamental component of a BI system since data quality of all other components mainly relies on it.
- **Data Storage Level** - The data processed by the ETL component is stored in a data warehouse, commonly implemented using a relational database management system (RDMS).
- **Analytics Level** - BI systems support two basic analytical functions: reporting and online analytical processing (OLAP). Through queries into the data warehouse, the reporting function provides managers with different business reports, such as sales reports or product reports. "OLAP allows managers to efficiently browse their business data from different analysis dimensions through slicing, dicing and drilling operations at will"[2]. In adiction there are some other types of analytical applications such as data mining, executive dashboards and costumer relationship management.

As Negash defines "Metadata are special data about other data such as data sources, data warehouse storage, business rules, access authorizations, and how different data is extracted and transformed."[2] It affects the entire BI atmosphere.

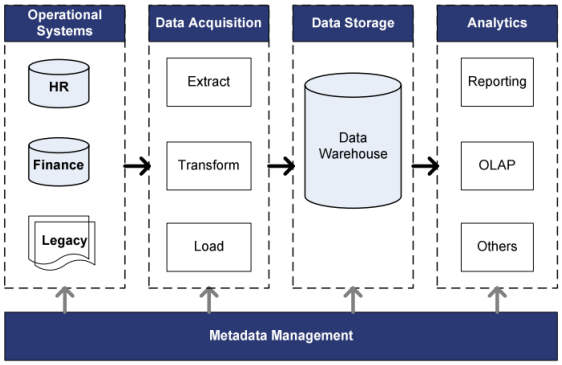

Figure 2.3: Business Intelligence System Architecture [2]

2.1.5 Data Warehouse

"A data warehouse is a type of data management system that is designed to enable and support BI activities"[6]. It centralizes large amounts of data coming from different sources and through it's analytical capabilities allow organizations to derive business insights from this data, supporting better decision making.
A data mart is a subset of the data warehouse and it is specially designed for a particular line of business such as sales, finance, inventory, etc. Figure 2.4 represents the entire data warehouse chain, from uploading data through the operating systems to ETL process and analysis and reports development.

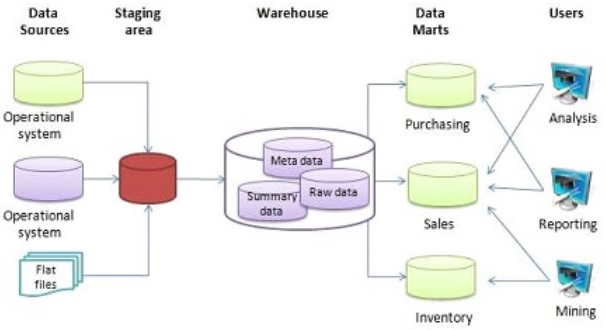

Figure 2.4: The basic architecture of a data warehouse

A data warehouse is a repository for structured, filtered data that has already been processed for a specific purpose. On the other hand, data lake is a vast pool of raw data, which the purpose is not yet defined. Although both are widely used for storing big data, data warehouses users are mainly business professionals while data lakes are most commonly used by data scientists or data specialists since they are often difficult to navigate by those unfamiliar with unprocessed data.

2.2 Existing Scalable Solutions to Storage and Analyse Data

From all over the world companies understand the importance of store their data with a view to perform better experiences to their customers in a more profitable way. Thus, adapting to emerging technologies has been crucial to the survival and success of businesses.

One of the top 3 US-based telecom providers wanted to migrate from Oracle DWH to a Cloudera Hadoop-based data lake to take advantage of advanced analytics. A scalable data storage solution was needed to reduce the cost of data management and analysis, effectively utilize their marketing budget, and increase business revenue. The telecom provider also wanted better insights to grow their B2B lead generation and improve the effectiveness of marketing campaigns and channels.They analyzed the existing data from RDBMS and external systems like Salesforce and Marketo and created a Hadoop data lake on the Cloudera (CDH) cluster for lead generation. Data from all these datasets were mined and mapped to enhance lead generation and conversion.

The same way, academic articles on data storage and analysis fully rely on Hadoop cluster solutions. In 2015 Michał Skuza and Andrzej Romanowski from Lodz University of Technology took advantage of large datasets available from Twitter to implementate a system that predict future stock market indexes prices. Through sentimental analysis - also known as opinion mining refers to a process of extracting information about subjectivity from a textual input - and two training dataset of over three million tweets they predicted if certain stock price is going to rise or drop.

2.3 Cloud Architecture

The National Institute of Standards and Technology (NIST) defines cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."[7]

2.3.1 On-premise vs Cloud

These two approaches operate almost in the opposite way. The main difference between cloud and on-premise is where the hardware and software reside. On-premise is a solution where the organization maintains the entire IT environment locally, which is managed, configured and maintained by the organization itself. However, in a cloud, the IT environment is allocated to servers outside the organization, mainly outsourced and managed by teams responsible for their maintenance and configuration.

	On-premise	Cloud
Deployment	In-house resource deployment and maintenance	Resources hosted and maintained on service provider
Cost	High due to required equipment	Low since it's a no-equipment solution
Control	Retain all data and are fully in control of what happens to it	Ownership of data and access to it are still a struggle between provider and client
Security	Has that extra feeling of security due to data possession but has to ensure security itself	Although encryption applied and risk of data corruption being low, there are records of leaked data, but in very special cases
Agility	Restricted to what was bought initially and restrict access only on local network	Constant improvement is outsourced and easy access everywhere with internet

Table 2.1: On-premises and Cloud Comparison

2.3.2 Cloud Service Models

The following diagram emphasize the number of responsibilities on the server-side (back-end) and it's difference compared to the client-side (front-end):

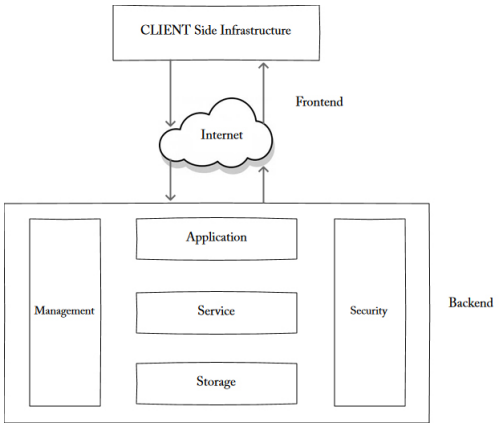

Figure 2.5: Common Cloud Architecture Diagram [3]

As presented in Figure 2.5 "it is the responsibility of the back-end to provide the security of data for cloud users along with the traffic control mechanism"[3], as well as all the software required to establish connection with the server, the middleware.

Cloud computing offers different services based on three delivery models. As described in figure 2.6 they are ordered from top to bottom and are called: ***SaaS***-Software as a Service; ***PaaS***-Platform as a Service; ***IaaS***-Infrastructure as a Service.

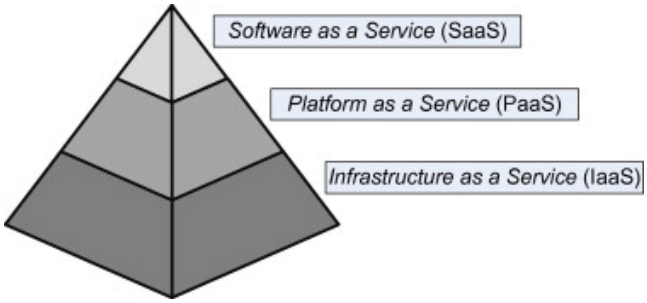

Figure 2.6: Cloud delivery models [4]

SaaS uses the internet to delivery applications and most of them run directly through the web browser, so it does not require any downloads or installations on clients' side. Everything is managed by the provider. Some examples are *Google GSuite (Apps)*, *Dropbox*, *Salesforce* and *Microsoft 365*.

PaaS delivers a framework for developers that they can use to build and create customized applications. While the clients can focus on application development, the provider maintain the server, storage and networking. Some examples are *AWS Elastic Beanstalk*, *Google App Engine*, *Microsoft Azure*, *Force.com*, *Open Shift* and *Heroku*.

IaaS offers all computing resources but in a virtual environment so that multiple users can access them. Vendors are responsible for managing data storage, virtualization, servers and networking, while users manage applications, data and middleware. System Administrators are the main client of ***IaaS***. Some examples are *Amazon EC2*, *Rackspace*.

The figure 2.7 shows the different responsibilities of the three cloud services compared to the On-Premises solution. Filled in blue indicates a management responsibility on the part of the user, while filled in orange indicates an outsourced responsibility.

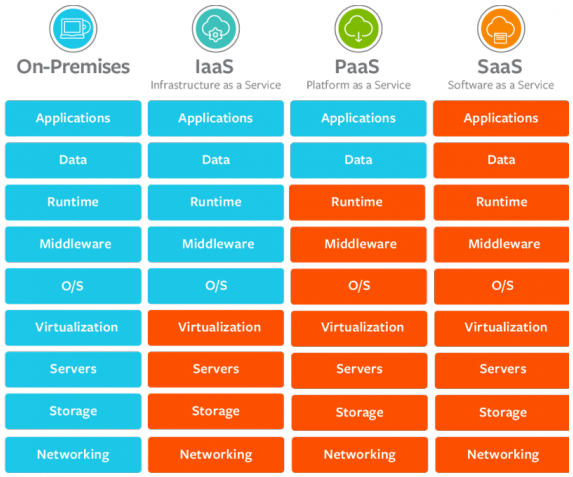

Figure 2.7: Cloud delivery models responsibilities [4]

2.3.3 Cloud Computing Technologies

Environments like *Google Cloud Platform*, *Amazon Web Services* or *Microsoft Azure* include more than one cloud service. In this section will be presented the resource of each of this technologies that could be applied to this project.

2.3.3.1 Google Cloud Platform

Google Cloud Platform is Google's environment of services for compute, storage, networking, big data, internet of things, machine learning, cloud management and security.

Cloud Functions

Cloud Functions was created in July 2018 and it is the event driven serverless compute platform from Google Cloud Platform. It allows to connect with Google Cloud tools but also with third-party services. Google Functions supports code in Node.js 6 and 8 as well as Python, and for the trigger types it supports all commonly used triggers, from HTTP trigger to Cloud Storage and time trigger. Unlike competitors it limits to 1000 functions per project. Also, HTTP functions do not

require an API Gateway and since billing method is based on execution time (GB-sec), number of invocations and how many resources need to be provisioned (GHz-sec), this contributes to lower prices for this scenario.

Cloud Storage
It is the object storage service provided by Google in the Google. This service allies the performance and scalability of Google Cloud Drive with advanced security and sharing capabilities. It is more suitable for enterprises since Google has already a service focused on storage cloud for personal clients. Price depend on storage classes and these are *Standard* for high frequency access, *Nearline* for data accessed less than once a month, *Coldline* for data accessed less than once a quarter and *Archive* for data accessed less than once a year.

Data Fusion
Cloud Data Fusion is the SaaS provided by Google to manage data integration. It is built on top of CDAP source project, an open source framework used to develop data analytics applications, and through its UI it allows to build scalable data integration solutions to clean, prepare, transform and transfer data. There are two editions, the Basic and the Enterprise edition. Billing system is charged depending on what edition is acquired and on how many instances are active.

Cloud SQL
Cloud SQL is the DBaaS provided by Google to set up, maintain, manage and administer relational databases on Google Cloud Platform. It supports *MySQL*, *PostgreSQL* and *SQL Server*. It connects to other Google's services, including Big Query allowing direct query and analysis of the databases. Pricing for Cloud SQL depends on instance type, but commonly its charge calculation is based on CPU and memory allocated, storage and the region where the instance is located.

Data Lake Modernization
On Google's cloud platform, there is no specific product that have been created exclusively for the use of data lakes. However, Cloud Storage can be developed and optimized through the setup features.

2.3.3.2 Amazon Web Services

AWS is a subsidiary of Amazon and it provides on-demand cloud computing platforms to individuals, companies and governments metered on a pay-as-you-go model.

Lambda
Introduced in 2014, AWS Lambda is the event-driven and serverless computing platform provided by Amazon. It runs Node.js, Python, Java, Go, Ruby, C# code and has a built-in feature to custom runtime environment making it possible for developers to run a function in the language of their

choice. User is charged based on the number of requests for the function and its duration, being this duration billed for each 100ms and price depending on the memory allocated for the function.

Amazon S3

Amazon Simple Storage Service is the object storage service by Amazon and it is available in whole world since 2007. It can be employed to store any type of object and allows to store from internet applications, backup and recovery, data archives, hybrid cloud storage, etc.
Amazon S3 price depend on storage class of objects stored and is charged on a GB per month calculation. These storage classes include *S3 Standard* for frequently accessed data, *S3 Standard-Infrequent Access* and *S3 One Zone-Infrequent Access* for less frequently accessed data and *Amazon S3 Glacier* and *Amazon S3 Glacier Deep Archive* for long-term archive and digital preservation. *S3 Glacier Deep Archive* is currently the cheapest storage solution on the market. Additionally, a lifecycle policy can be set and transfer data to a different storage class.

Data Pipeline

Data pipeline is Amazon's IaaS to automate data movement and transformation through the development of data integration projects. Building pipelines to develop ETL processes allow users to obtain more value from data across multiple sources and downstream it to other AWSs such Amazon RDS.

Amazon RDS

Amazon Relational Database Service is a distributed relational database service provided by Amazon Web Services. It was designed to simplify the setup, operation and scaling of relational databases. RDS was released in 2009, supporting only MySQL databases, and upgraded in 2012 and 2013 to support Microsoft SQL Server and PostgreSQL respectively. It has a pay-as-you-go model that depends on what instance the user subscribe and AWS provide a pricing calculator on their RDS website to help combine all variants of the billing system.

Amazon Lake Formation

Amazon Lake Formation makes it possible to build a data lake on several days. The service provides a central point of control from where the user can identify, ingest, clean and transform data from all kind of sources. Also, enforces security policies across multiple services and still manage to acquire new insights. The user operates from one dashboard where it is possible to configure and set up all lifecycle stages and activities. Amazon Lake Formation relies on data stored in Amazon S3 and it is only charged through technologies used inside of it.

2.3.3.3 Microsoft Azure

Function App

Azure Functions is a serverless compute service that lets event-triggered small pieces of code to run without having to explicitly provision or manage infrastructure. These small pieces of code

are called "Functions" and can be triggered by an HTTP request, message receive, changes in data (every time a file is created or modified in the storage) or simply running on a predefined scheduling.

Blob Storage

Azure Blob storage is an object storage solution for the cloud and it is optimized to storage massive amounts of unstructured data.
Unstructured data is data that doesn't adhere to any particular data model or definition meaning it can storage audio, video, images, text files, among many others. Every file inside a Blob storage is called a *"blob"*. Blobs can be added, deleted or edited via programming code or other Azure resources like Data Factory (secção X). Additionally, users or client applications can access objects in Blob storage via HTTP/HTTPS, from anywhere in the world.
A blob storage is branched into three main directories:

- **Storage Account**: private or public account, highly scalable, where all data can be stored and accessed from anywhere in the world over HTTP or HTTPS
- **Container**: virtual environment projected to provide a distributed file system, with folders and files (blobs)
- **Blob**: as mentioned above, is the file itself and can be of any type.

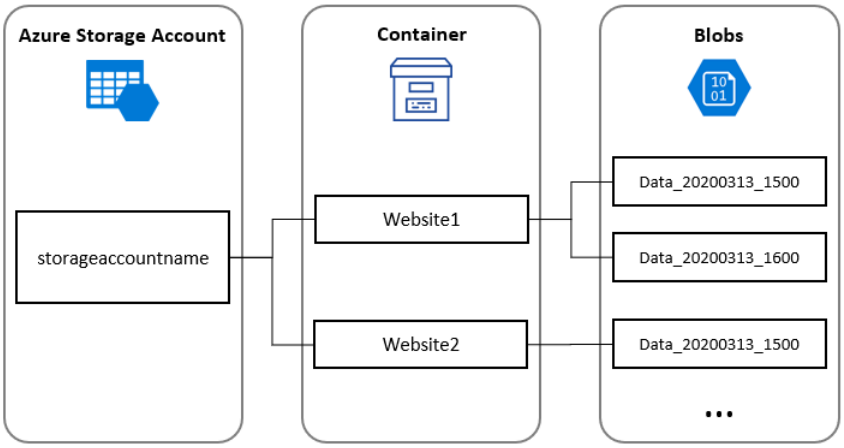

Figure 2.8: Directory Tree in a Blob storage

Data Factory

Azure Data Factory is a cloud service built to support and develop data integration projects. It is the cloud-based ETL and data integration service that allows the creation of data-driven workflows

for orchestrating data movement and transforming data at scale.
Below are listed the main components of this technology and *Figure 3.2* show it's environment:

- **Pipeline** - is a logical grouping of activities and together these activities perform a task, which allow to manage the activities as a set instead of managing each one individually.
- **Activity** - represent a processing step in a pipeline. It can be data movement activity, data transformation activity or control activity.
- **Dataset** - it is the pointer or reference to the data to be used as input or output in an activity.
- **Connection** - it is simply the connection to the data source or sink. This connections are established through a connection string called *linked service*.
- **Trigger** - represent the unit of processing that determines when a pipeline execution needs to be kicked off.

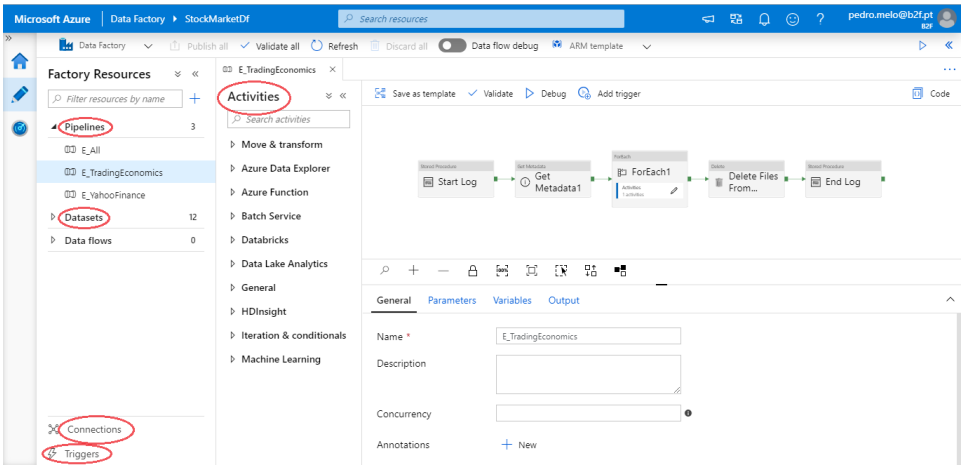

Figure 2.9: Azure Data Factory Main Components

SQL Database

Unlike its competitors, Microsoft offers its database services in separate modules depending on what engine the database in running on. As any other cloud database, management of scalability, setup and availability is part of the provider's work. The key aspect of Azure's SQL Database is the built-in app learner to continuously improve performance, reliability and data protection.

Data Lake Storage Gen 2

Data Lake Storage Gen2 is the result of converging the capabilities of two existing storage services, Azure Blob storage and Azure Data Lake Storage Gen1. From Azure Data Lake Storage Gen1, features like file system semantics, directory, and file level security and scale are combined

with low-cost, tiered storage, high availability/disaster recovery capabilities from Azure Blob storage.
A fundamental aspect of this resource is the addiction of a hierarchical namespace, enabling operations to be more atomic and through this more efficient.

2.4 Stock Market

The stock market is nothing but where investors connect to buy and sell investments, most commonly stocks, which are shares of ownership in a public company. Often, the stock market term refers to one of the major stock market indexes, such as *The Dow Jones Industrial Average* or the *NASDAQ Stock Exchange* being their values a representative of the performance of the entire market. If the stock market index moves lower or higher or if it closes up or down, this means the stocks within the index have either gained or lost value as a whole. ((Although it is harder to track every single stock inside a stock marker index, it is possible to do the same analyses)).
The best way to understand the stock market is to imagine an auction house operating, where buyers and sellers can negotiate and make trades. "In order to raise some money and grow their business, companies list shares of their stock on an exchange through a process called an *initial public offering*, or *IPO*" [8]. After that investors can buy and sell stocks among themselves, determining the supply and demand for each listed stock. With this data, computers algorithms calculate the price investors and traders are wiling to pay, settling a price value for each stock of each company.

2.5 Web Scraping

Ideally data from the internet would be structured and easily readable by computers, but many web pages contain text content that is human-readable but not easily machine-readable. In fact, "web scraping bridges this gap and opens up a new world of data to researchers by automatically extracting structured data sets from human-readable content"[9]. The web scraper finds specific data elements on the web page, transforms them if needed and stores these data as a structured data set."Data like item pricing, stock pricing, different reports, market pricing and product details, can be gathered through web scraping."[10]

2.5.1 Main Steps of Web Scraping

The full process should be focused on extracting targeted information from websites contributions to take effective decisions in business process.

The following explains the four main steps of web scraping:

- **Crawling**: done by the web crawler, "essentially mimics how a web browser operates"[9], browsing through the web pages based on outcome's requirements.

- **Scraping**: the actual process of gathering data or information from pages visited by the crawler.
- **Extracting**: information and useful data is extracted from the full data dictionary collected in the previous step.
- **Formatting**: to make it understandable to the end-user, data need to be delivered in an appropriate and structured format.

2.5.2 Self-Development vs Outsourced Technologies

As well as many other services, web scraping can be in-house developed or outsourced. For reasons of full quality control and requirements accomplishment, outsourced web scraping will be excluded of the options for this project.

In terms of developing a web scraper, does not exist the best programming language. Hence, there are some requisites that can make the developer prefer one to another. However, communication with internet is the real bottleneck here, since it's speed cannot match that of the processor inside the computer.

2.5.3 Web Scraping Applied to Stock Market

It is easy to understand the interest of relating web scraping with the stock market and imagine the diversity of possibilities created with it.
In 2008, Hadi Pouransari and Hamidreza Chalabi [11] used scripts of *Python* programming language to collect information on daily stock market information and earnings calendar data (this last refers to the earnings announcement event which stock price of a company is affected by). Their purpose is to predict whether a given stock will be rising in the following day after earnings announcement, using machine learning techniques.
In 2015, Carol Anne Hargreaves and Chandrika Kadirvel Mani [12] collected data from finance and stock market websites via outsourced web scraping to apply *Principal Component Analysis*, a statistical technique that reduces a large number of inputs of data to a few factors, and from there select winning stocks. The following year, the same authors applied the same outsource web scrape, collecting data to develop a semi-automated stock trading application, based on the machine learning algorithm SVM (*Support Vector Machine*. [13]

2.6 Data Reporting & Visualization

What is the meaning of data if people cannot understand it's significance? That is what data visualization aims to do, so "any effort to help people understand the significance of data by placing it in a visual context"[14], is considered data visualization.
Nowadays, technologies to visualize data insights go far beyond spreadsheets' tables and charts,

displaying data in a much more sophisticated way through geographic and heat maps, infographics, sparklines and detailed bar, pie and fever charts, with interactive and drill-down capabilities.

2.6.1 Reporting Technologies

There are several reference tools in data visualization such as *Power BI*, *Tableau* and *Qlik*, more related to Business Intelligence activities, and programming languages *R* and *Python* commonly used to data analysis. The following table highlights the main differences between three solutions empowered for BI activities.

2.6.1.1 Power BI

Power BI is a business analytic service by Microsoft that allows every end-user with any technological or analytical background to develop reports and dashboard based on imported data. Data can be sourced by on-premises options such as *Excel* or *SQL Server* as well as cloud-based data repositories.
Power BI consists of various components, available in the market separately and can be used exclusively. The common report creation begins in Power BI Desktop where data is adapted and integrated into visuals, creating dashboards. Finishing all visuals creation commonly the developer shares those visualizations through Power BI Service, the cloud based software that allows the developer to build a report using the visuals created using the desktop version. Finally, using Power BI Mobile every user with permission can see the reports developed through any mobile device. Although there are more components inside the *Microsoft Power BI* environment, these are the main and more commonly used ones.

2.6.1.2 Tableau

Tableau is a powerful data visualization tool used in the business intelligence and information technology industry. This tool allow non-technical users without any programming skills to operate and create dashboards and reports easily understandable by any professional at any level of an organization.
The Tableau Product Suite consists of five main products: Tableau Desktop, Tableau Public, Tableau Online, Tableau Server and Tableau Reader as shown in the below diagram.

For clear understanding , these tools can be classified into **Developer tools**, which refer to the tools used to develop of dashboards, charts and report generation, including Tableau Desktop and Tableau Public, and **Sharing tools**, as the name suggests, the purpose of these tools is sharing the products developed by the above developed products and include Tableau Online, Tableau Server and Tableau Reader.

2.6.1.3 Qlik

Qlik is a software company that provides end-to-end platforms to data integration and user-driven business intelligence analytics. The company's main products are **QlikView** and **Qlik Sense**.

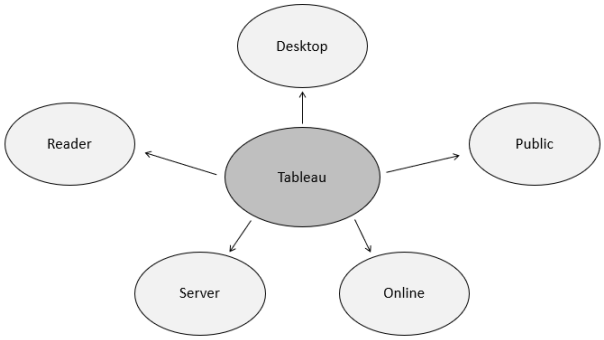

Figure 2.10: Tableau Product Suite

QlikView is more dedicated to deliver prepared visualizations where the user will simply use the charts and dashboards that the developer created however the end-user is much more limited when it comes to creating new visualizations.
On the other hand, Qlik Sense is a tool commonly used when the developer wants the end-user to have the freedom to create a layout of his own and the feeling of self-service data discovery, meaning a much more active and engaged user.

2.6.2 Technologies Comparison

	Power BI	Tableau	Qlik Sense
Price	-Very complete version for free. There are two premium versions called Power BI Pro and Power BI Premium costing 10 and 5 dollars per user and per month, respectively	–Free downloadable version but lacking many features. Advanced version costs 70 dollars per user per month.	– Free version limited to few features to work with. Premium version available for 30 dollars per month per user.
Ease of learning	– Drag and Drop interface, very similar to Excel, with very intuitive dashboard creation	– Although is not as intuitive as it's Microsoft's competitor, still easy to deploy reports and dashboards after some practice.	– Programming knowledge required to obtain the same results.

Continues on the next page

Table 2.2 – continued from previous page

	Power BI	Tableau	Qlik Sense
Add-ons and Features	– It has its own app store featuring a full range of additional features. Besides all dashboarding tools, Power BI has a natural query language tool, allowing users to ask questions directly on data.	– Do not supporting natural query language but users claim to prefer visualization styles.	– Beside not supporting natural query language, very few number of additional features and styles.
Costumer Community and Support	– Since it comes with Microsoft 365 package those who buy any Microsoft's products may be exposed to this tool as well. Also, very effective costumer support.	– Having a history on its own, there is a huge customer community. Also, there is over 150 thousand active costumers in their online community.	– Smaller community since programming skills are required. There are two levels of customer support. Basic, for individuals and available during business hours through telephone support. And Enterprise support, providing support from certified engineers and available 24/7.

Table 2.2: Data Visualization Tools Comparison

In short, there is a meaning behind every number collected and visualizing data brings them to life.

Chapter 3

Proposed Solution

3.1 Solution Overview

In order to get reliable and live information on stock market two online sources were selected: *Yahoo Finance* and *Trading Economics*.
Extraction is based on *Web Scraping* technique through *Python* scripts. These scripts are implemented in an *Azure Function App* that runs every hour between Monday to Friday, and save each web source information into distinct Comma-Separated Values (CSV) files. Every file is saved to a cloud warehouse called *Azure Blob Storage*. After that, and with *Azure Data Factory* as the orchestrator of the ETL process, data flows between tables in *Azure SQL Database* from raw, low meaning data to transformed and high correlated information ready to get insights from.
As soon as the ETL processes end, the *Azure Data Factory* exports these tables to compressed and highly efficient files, in *.parquet* format, and stores them in the *Azure Data Lake Generation 2*. From here another two functions, hosted in the same *Function App*, provide the last fifteen days of data and the data from an on-demand user-triggered event.
Microsoft Power BI has then two fixed pointers to this data, that feed the visualization dashboards.

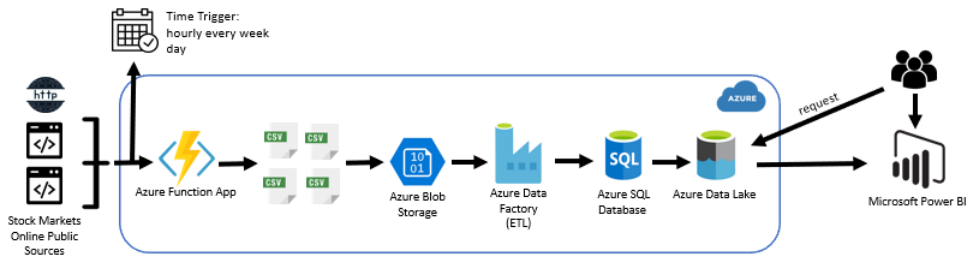

Figure 3.1: System Overview Diagram

3.2 Phased Architecture

In this section the solution is divided into three essential phases and each of the phases following a data flow perspective.

3.2.1 Data extraction from sources

As mentioned, data extraction from the web relies in scripts that deeply explore the source code from each of the two web sources and get the relevant information into structured csv files to store them inside a *Blob Storage's* container.
The scripts are time-triggered every hour of a week day through integrated scheduling of *Function App* resource.

3.2.2 Data transformation and storage

Once the web extraction is done is now time to transform the same data and prepare it to the next step.
This begins the deeply known ETL processes and they occur every weekday at the end of the day, after the last web scraping script runs, relying on *Data Factory*. For the **E**xtract stage all files extracted via *Function App* are imported into tables of the *SQL Database*, more precisely *Staging Area's* tables, where columns are still raw and not processed. After that, SQL scripts transform data types and give more sense to certain columns and values, starting to shape these as final tables, but still inside the **SA** scope. **L**oad is a simple operation of copy these last tables into the respective tables inside the **DW** (Data Warehouse) schema.
With the ETL fully performed, tables are exported to very compressed *.parquet* files and stored inside a *Data Lake Storage* container. This small step is what makes this an innovative and highly efficient project, since *SQL Database*, one of the most expensive resources of a business intelligence process, will only store one day of data, keeping it the simplest and cheapest. On the other hand, store and operate data from *Data Lake Gen2* gives this project another level of performance and scalability.

3.2.3 Data modulation and integration with analysis tool

With the information compressed and stored inside the *Data Lake* is now important to understand how can this files be accessed to create visualizations.
Well, for that there are another two scripts running on *Function App*. One of them runs every workday at 23:45, right after the ETL processes end, and picks the information of that day's *.parquet* files, storing them into *.csv* files on another folder, expiring the oldest data after fifteen days. The other function reads two dates as input given by the user, and performs the exact same operation for each date between those two, creating a possibility of an on-demand analysis.
Visualizations are then created using *Microsoft Power BI* with two fixed directories' pointers (these directories are in the *Data Lake*), one for each analysis explained in the previous paragraph. Every

time these functions run, the user only need to refresh the pointers of Power BI and is ready to analyse the earliest data available.

3.3 Data Model

This section discusses the data model built in this project, as well as an analysis of its content, structure and components.

3.3.1 Fact Tables

The fact table in a dimensional model stores the performance measurements resulting from an organization's business process events. A bedrock principle for dimensional modeling is that a measurement event in the physical world has a one-to-one relationship to a single row in the corresponding fact table, everything else builds from this foundation. Because measurement data is overwhelmingly the largest set of data, it should not be replicated in multiple places and multiple organizational functions around the enterprise.
For this two reasons, under the framework of this project were created two fact table, each to store measurement data from each web page that is scraped.

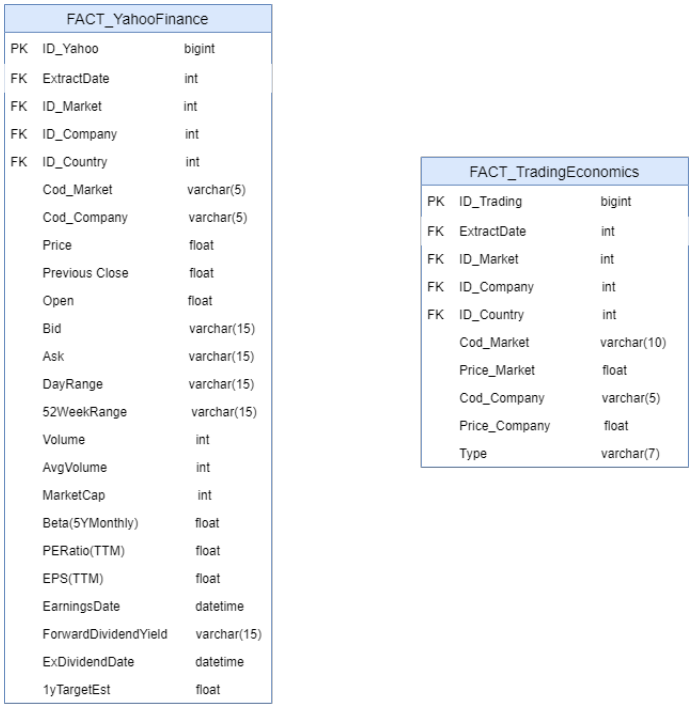

Figure 3.2: Fact Tables of project's Data Model

3.3.2 Dimension Tables

Dimension tables are integral companions to a fact table. The dimension tables contain textual context associated with a business process measurement event. They describe the *"who, what, where, when, how and why"* associated with the event. Dimension tables tend to have fewer rows than fact tables, but can be wide with large text columns. Each one is defined by a single primary key, which serves as the basis for referential integrity with any given fact table to which it is joined. Three specific dimensions were selected in this project: Market, Company and Country. Additionally a Time dimension were created in order to query the measurement event data depending on its extraction date.

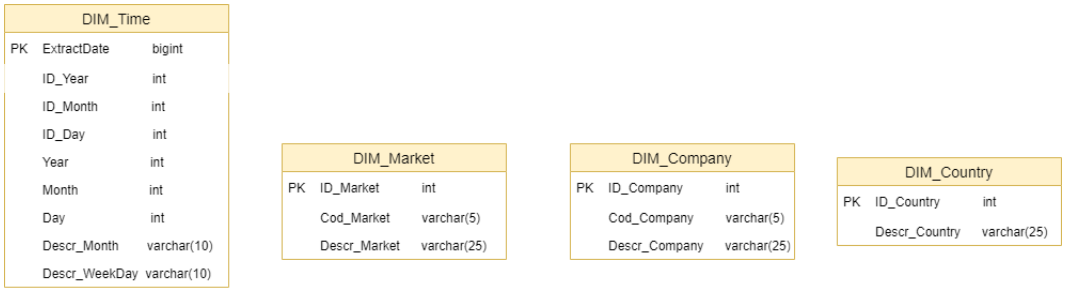

Figure 3.3: Dimension Tables of project's Data Model

3.3.3 Dimensional Model

Each business process is represented by a dimensional model that consists of a fact table containing the event's measurements, surrounded by a halo of dimension tables that contain the textual context of this measures.

This facts and dimensions tables were joined in a *Star Schema* modulation, since the dimensions do not connect with each other. This choice was made based on the simplicity of the visual diagram, making it easy to understand and to scale to another project, but mainly due to the speed and efficiency of the queries when executed in very high-volume tables.

As explained in the previous sub-chapter, fact tables contain the primary keys of each dimension table, otherwise it would be impossible to join them.

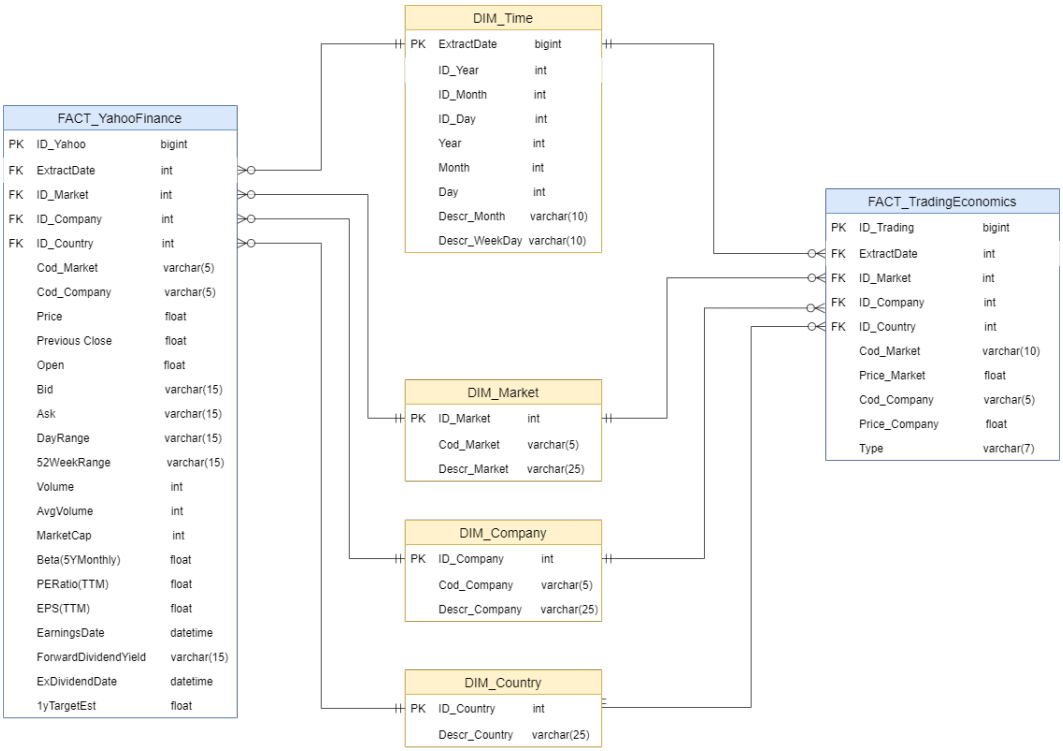

Figure 3.4: Dimensional Model for this specific project

3.4 Budget Estimation

This section aims to estimate the budget needed to host and keep this project running during it's duration, which means full project's duration will be considered as six months and considering the approximation that the project was in full operation during its development.

3.4.1 Azure Resources

All Azure Resources budgets will be summed up and assigned to ***Azure_Budget*** variable.

Function App

Consumption plan is the basic and choosen plan for this resource. It is billed based on number of executions and resource consumption. It includes a monthly free grant of 1 million requests and 400,000 GB-s (Gigabytes per second) of resource consumption, after which it is charged according to Table 3.1.

Meter	Price	Free Grant (p/month)
Execution Time	€0.000014/GB-s	400,000 GB-s

Total Executions	€0.169 per million executions	1 million executions

Table 3.1: Consumption Plan Billing Method

The web scrape functions that run every hour between Monday and Friday sum a total of 24 executions per day each which means 120 executions per week and 480 executions per months, combined represents 960 executions per month. To get the last fifteen days file, a function runs once a day during the same week days, representing 5 executions per week, meaning 20 executions per month. With this calculations it is understandable that approximately 500 executions are fixed during a month. For this reason there are still 999 500 free executions inside the free plan, which are not expected to be exceeded by the on-demand function.
In terms of memory needed to run the functions, the web scrape functions spend in average 100MB per execution, combined, which represents 360 000MB per month that equals to approximately 351GB. The last fifteen days function spend in average 4MB, representing 80MB per month. With a free consumption of 400 000 GB there are still a lot of free consumption to dedicate to on-demands needs.
For these two reasons, pricing of *Azure Function App* resource is considered zero.

$$FA = 0€ \tag{3.1}$$

Blob Storage
There are four tiers which pricing plan and it's calculation is based on. The Hot tier is applicable for most workloads, including this project. The Cool and Archive tiers are for cool or cold data with pricing optimized for lowest GB storage prices. Premium blob storage provides access to block blobs and append blobs with low and consistent latency, with pricing optimized for high transaction rates. Total cost of block blob storage depends on: volume of data stored per month, quantity and types of operations performed, along with any data transfer costs and data redundancy option selected.

	Premium	**Hot**	**Cool**	**Archive**
First 50TB/-Month	€0.1644/GB	€0.0166/GB	€0.00844/GB	€0.00152/GB

Table 3.2: Blob Storage Pricing of Storage

Knowing that the csv files from both sources, combined, have 288KB for each extraction, it is understandable that there are 6912KB new each day meaning new 138 240KB, by other words approximately 0.0009765625GB each month. If data volume created is assumed to be the same each month, the following equation gives the total price for data storage.

$$BS_{storage} = \sum_{n=1}^{6} 0.0166 * 0.0009765625 * n = 0.000340€ \tag{3.2}$$

In terms of operations and data transfer prices Azure Blob Storage charges per 10 000 executions. If both scraping functions combined generate 48 files a day, it means 960 files are written per month and 5760 for the whole project. Additionally, these same files are read in order to integrate the ETL process, so there are 960 read operations per month, meaning the same 5760 for the project's duration.

	Premium	**Hot**	**Cool**	**Archive**
Write operations	€0.0193	€0.0456	€0.0844	€0.1012
Read operations	€0.0016	€0.0037	€0.0085	€5.0598

Table 3.3: Blob Storage Pricing of Operations

According to the table above for the prices per 10 000 operations, the cost of operations within Blob Storage is:

$$BS_{operations} = \frac{5760}{10000} * (0.0456 + 0.0037) = 0.02840€ \tag{3.3}$$

When data is written to a GRS, RA-GRS, GZRS, or RA-GZRS account, the data is replicated to another Azure region. The geo-replication data transfer charge is for bandwidth used to replicate data to the second Azure region. This charge also applies when you change the storage account's replication setting from LRS to GRS or RA-GRS or from ZRS to GZRS or RA-GZRS. Because replication charge is free for LRS option,the one used in this project, costs will be calculated based

only on storage and operations pricing plans.

This means that price for the whole *Blob Storage* resource is equal to:

$$BS = BS_{storage} + BS_{operations} = 0.02874€ \tag{3.4}$$

Data Factory

Data Factory's pricing is based on two charging types: orchestration charge and execution charge. In the orchestration charge, standard pricing stands for €0.844 / 1,000 activity runs per month. According to the estimations there are 252 activities that run every weekday. The following equation computes Data Factory's pricing in this section for the project duration:

$$DF_O = \frac{252 * 5 * 4}{1000} * 0.844 * 6 = 25.5226€ \tag{3.5}$$

The execution charge separates the activities into three different types: data movement activities, pipeline activities and external activities.
For the data movement activities pricing is €0.085/hour. So, having 100 copy activities and knowing, after an experimental test, that each one executes in approximately 6 seconds, this cost can be calculated as:

$$DF_{Emov} = 0.211 * \frac{6 * 22 * 100}{3600} * 6 = 4.642€ \tag{3.6}$$

For the external activities only stored procedures are considered. There are 84 stored procedures that run for approximately 15 seconds each, resulting in a cost of:

$$DF_{Eext} = 0.000211 * \frac{15 * 22 * 84}{3600} * 6 = 0.009748€ \tag{3.7}$$

Finally, the pipeline activities, that are charged at €0.005/hour, are 68 and last for approximately 23 seconds each, resulting in a cost of:

$$DF_{Epip} = 0.005 * \frac{68 * 22 * 23}{3600} * 6 = 0.2867€ \tag{3.8}$$

Summing up all *Azure Data Factory* costs:

$$DF = DF_O + DF_{Emov} + DF_{Eext} + DF_{Epip} = 30.4610€ \tag{3.9}$$

SQL Database

For the SQL Database budget is important to understand the focus of this project in efficiency, low cost and high scalabillity. It should be easily understandable that this database will only store data of one day in its tables. This is due to the daily exportation of the tables' contents into files that are stored later inside a data lake.
This helps keep the database at the most basic and cheap tier. The whole range of tiers can be accessed at Microsoft Azure's page.

$$DB = 4,1298 * 6 = 24.7788€ \tag{3.10}$$

Data Lake Generation 2

Azure's Data Lake follows the same tier plans created for the Blob Storage. This project subscribe a Hot tier and LRS replication so, as for the blob storage, Data Lake will only be charged for storage and operations.

For the storage in Data Lake, Microsoft follows the following pricing for the first 50TB of data:

		Hot	Cool	Archive
First Month	50TB/-	€0.0166/GB	€0.0085/GB	€0.0016/GB

Table 3.4: Data Lake Gen 2 Storage Pricing

A full day of information in *parquet* format has approximately 707KB. Multiplying this for 5 days a week and 4 weeks a month, the monthly size of *parquet* generated is near 14 140KB that is equal to 13.8MB. On the other hand, the *csv* files history for the last 15 days has approximately 54MB. The last one is fixed during the whole project's duration but the *parquet* ones are generated and stored monthly so for storage pricing the following expression can be used (with MB already converted to GB):

$$ADLS_{storage} = \sum_{n=1}^{6} 0.0166 * (0.0134765625 * n + 0.052734375GB) = 0.00995€ \tag{3.11}$$

Again, as for the Blob Storage, Data Lake's transaction prices are based in per 10 000 operations, as follows:

		Hot	Cool	Archive
Write Operations		€0.0592	€0.1097	€0.1316

Read tions	Opera-	€0.0048	€0.0110	€6.5778

Table 3.5: Data Lake Gen 2 Transactions Pricing

This system follows a daily routine of writing six *parquet* files, reading these same files. writing them into *csv format* and reading the *csv* through *Power BI*. This means there are twelve writing operations and twelve reading operations a day. Once again, the monthly perspective is 240 operations of each type so transactions cost is:

$$ADLS_{transactions} = [\frac{240}{10000} * 0.0592 + \frac{240}{10000} * 0.0048] * 6 = 0.009216€ \tag{3.12}$$

Total Data Lake's cost can be calculated as:

$$ADLS = ADLS_{storage} + ADLS_{transactions} = 0.01917€ \tag{3.13}$$

It's important to understand that costs don't include the on-demand feature of the system due to it's unpredictable volume generated since it all depends on the amount of data the end-users selects to analyse. However, this is not a critical approximation because the impact on the final cost would not be very relevant.

3.4.2 Microsoft Power BI

This project has a Pro subscription of Microsoft Power Bi. According to the monthly cost of €8.40 this resource has a total cost of:

$$PBI = 8.40 * 6 = 50,4€ \tag{3.14}$$

All tiers and pricings are available in section 2.6.1.1

3.4.3 Total Estimation

The estimation of the total cost of keeping this project running for six months is computed in the table :

Resource	Cost
Function App	€0
Blob Storage	€0.02874
Data Factory	€30.4610
SQL Database	€24.7788
Data Lake Gen2	€0.01917
Microsoft Power BI	€50.4
Total	€105.68

Table 3.6: Total Budget Estimation

Chapter 4

Solution Implementation

In this chapter are presented extremely detailed explanations on implementation of each phase of this project as well as the event registry as a key performance indicator of the ETL process developed.

4.1 Phased Implementation

There are five main stages in this project and this section describes all detailed contents on implementing each of them

4.1.1 Azure resources deployment

First things first, and as predictable to deploy Azure resources was necessary to deploy an *Azure Resource Group* to pool all other resources together. Only then the remaining project resources were deployed, which are, as mentioned previously: *Function App*, *Blob Storage*, *Data Factory*, *SQL Database* and *Data Lake Storage Gen2*. Figure 4.1 illustrate all Azure's resources deployed inside this project's *Azure Resource Group*.

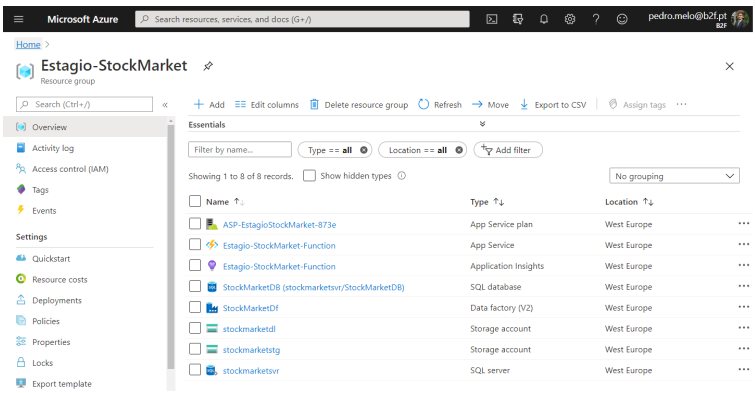

Figure 4.1: Azure Resource Group

4.1.2 Data extraction from sources

Further analysis revealed that each web source had an unique source code so in order to get similar data from web scraping it was mandatory to develop two independent scripts, each for each source. Despite that, the framework of both scripts is extremely identical so it is possible to build a diagram showing every main step of these scripts.

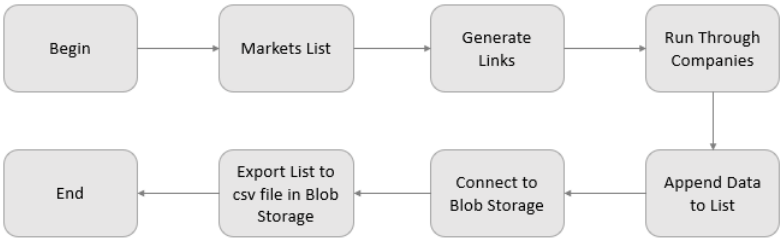

Figure 4.2: Structure of Web Scraping Scripts

First of all, a list with the markets to retain information is given. Each market and each company has an acronym, which is called *ticker* in stock market language. The hyperlink to access to markets or companies page has only one variable and that is the ticker, subsequently, a link that depends on the list of companies' tickers is generated and access to every web pages of interest is guaranteed.
Following this, information is consistently appended to a list. After concluding the connection to the *Blob Storage*, the list mentioned is exported as a *.csv* file and stored inside a container of the resource.
This routine is repeated for each source every hour within weekdays.

4.1.3 Cloud ETL development

The ETL process is the engine that keeps data flowing in an autonomous and efficient way from raw and meaningless data to insightful information. To fully understand an ETL process is important to realize that the Database has the *Staging Area* and *Data Warehouse* schemas. **SA** is meant to keep unstructured or unclassified data, in contrast to **DW** that stores data fully processed and ready to get insights from.
With this in mind, the ETL starts with the **E**xtraction stage that harvests the information from each file generated by the web scrape scripts and stores them into **SA** tables within the database. One example of the execution of the extraction stage is presented in Figure 4.3.

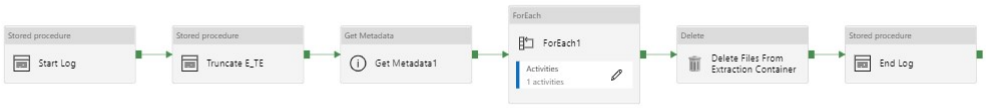

Figure 4.3: Example of Extraction stage in *Data Factory*

The next stage of the ETL includes the **T**ransformation processes. These process data and assign correct data types to the data inside the extraction's SA tables. Since this stage is based on running SQL code to manipulate the data, its pipelines inside *Data Factory* are just executions of stored procedures that run that code. An example of a pipeline that runs a transformation stage is below.

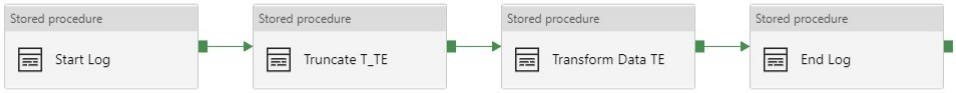

Figure 4.4: Example of Transformation stage in *Data Factory*

With these transformations, changes came in the tables of Facts, mainly on data columns and columns that were previously ranges of numbers, becoming two separate columns so analysis is now more efficient. Figure 4.5 shows how FACT tables turned after **T**ransformation stage.

Ending the ETL is the **L**oad Stage. This stage is as simple as replicate the resulting tables of the previous stage to tables inside the **DW** scope. Also, dimension tables are created based on the same data. Once again this stage is based on running stored procedures in SQL code, which makes Data Factory's pipelines as simplest as possible, as shown in Figure 4.6.

Out of the traditional ETL scope but still present in the innovation that this project seeks to promote, there is a stage that exports all data from the Database into compressed files and stores them inside the *Data Lake Gen2*. To highlight the cost reduction this brings, the pipeline assigned to execute this stage merely identifies the tables in the **DW** schema and, through a single copy activity, exports them to *.parquet* files. So the impact this has in *Azure Data Factory* cost is almost zero.

These examples show only activities and procedures to a specific source of information. In order to make this an autonomous process were developed general pipelines to run an entire stage of the ETL and even a pipeline to run the complete ETL process. These types of pipelines are deployed through activities that call executions on other pipelines, as it's seen in Figure 4.7.

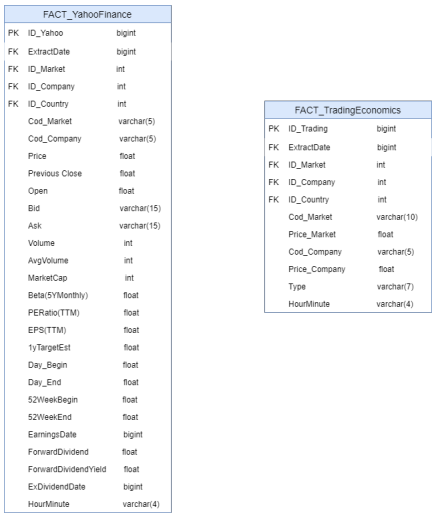

Figure 4.5: Fact Tables after Transformation stage

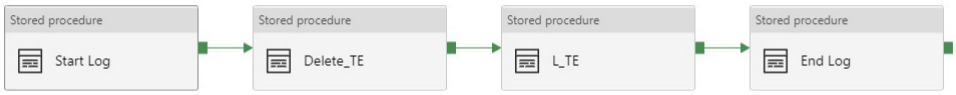

Figure 4.6: Example of Load stage in *Data Factory*

Figure 4.7: Pipeline of the entire ETL in *Data Factory*

4.1.4 On-demand and historic data conversion/usage

Data stored inside *Data Lake Gen2* is highly compressed and, in order to be available to end-user usage, two scripts were developed. One of them has a time-trigger to run every weekday and stores on a fixed file location the last fifteen days of information collected. The other answers to the need of on-demand analysis, providing access to data that is not in this recent historic content.

On-Demand

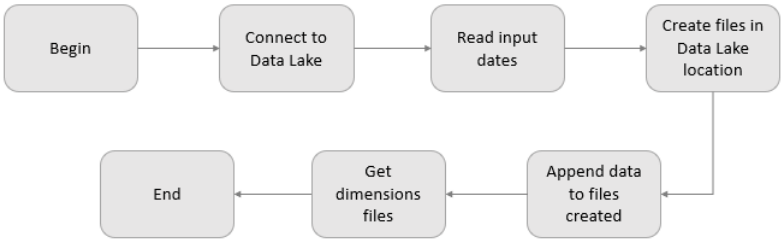

Figure 4.8: Structure of On-Demand script

The on-demand script is triggered through an HTTP call where the start and end date of information the user wants to analyse are inserted. So right after connecting to *Data Lake Gen2*, a range of dates is generated and assigned to a list. *.Csv* files are created on the fixed *"On-Demand"* location and through each iteration of the list mentioned the data read is appended to the respective file.

Last 15 Days

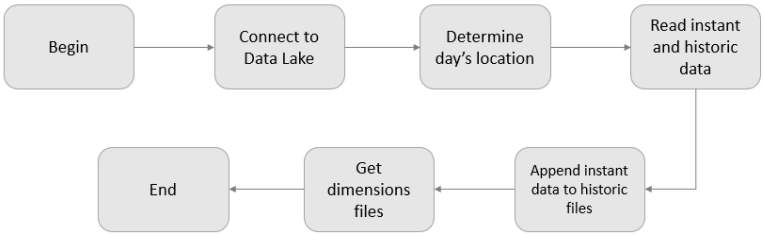

Figure 4.9: Structure of Last 15 Days script

Historical script run every week day at 23:45 with the purpose of append data from that day's *.parquet* to the *.csv* present on the last fifteen days fixed location.
Firstly, the script connects to this fixed location and also determine the day's location on *Data Lake Gen2*. Next it opens all factual files inside both directories in order to append data to the cumulative files. Finally it updates the dimensions files inside the last fifteen day's folder with the same procedure stated above.

4.1.5 Data model implementation

With Data Lake having the data already treated and compressed, the next step is to build the multidimensional model and then the analysis, the latter being the step that will be in direct contact with the end user.
The implementations of the processes involved in this phase were carried out on the PBI platform, which has been connected to the Data Lake in the Azure cloud in order to obtain the data present in

the folders of the respective analyses. Also, for the analysis of the ETL process itself a connection between PBI and the Logs table in the SQL Database was made. The decision of not exporting this table like the others was based on the null impact it has on the budget versus the considerably higher performance needs and number of processes to be developed so that it followed the same procedures than the dimensional and fact tables. This table does not represent part of the data lake solution proposed for this project, sticking only to analysis purpose, that is why this table is not integrated in the dimensional model of Figure 3.4.

After the transformation stage, the tables (in file form) are not suitable for the analysis phase. This phase, which involves the construction of dashboards with the help of data previously extracted and treated, requires a multidimensional model, with links between the different tables that make up the DW.

As an example, when implementing a date filter in a dashboard, it will consist of the data for the year, month and day, which are present in the DIM_Time (check data model in Figure 3.4). However, another chart present in this dashboard that contains, for example, the variation of a company's stock price over time, will not respond to changes in the date filter if there is no link between the DIM_Time table and the table that contains the stock price data.

This need does not arise only in relation to the date, it arises in many other cases, such as in relation to the company names (DIM_Company contains the names of all the companies).

Since the presence of a multidimensional data model was indispensable, the link between the various tables was necessary. These links were made in PBI, and are shown in Figure 4.10.

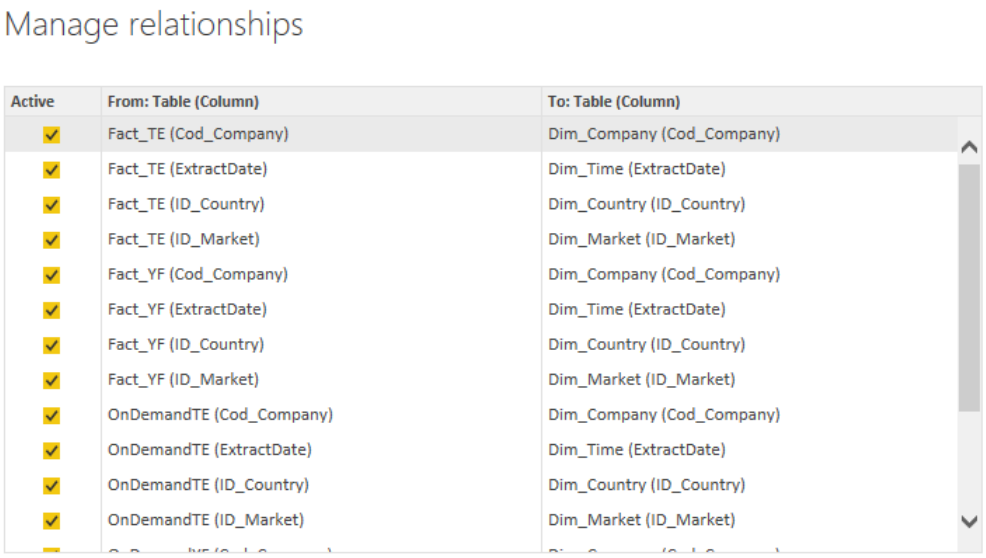

Figure 4.10: Connections between tables of the DW

Having completed this step, it is possible to check the final data model and thus begin the analysis phase. The model can be reviewed (once already shown in Figure 3.4) in Figure 4.11, but this time built in PBI.

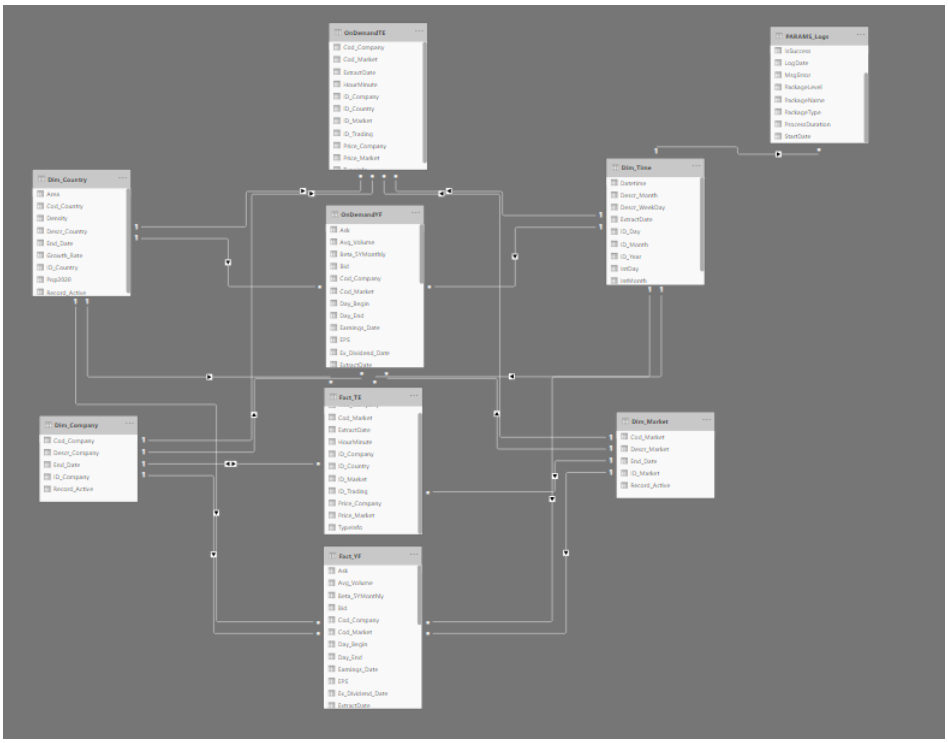

Figure 4.11: Data Model built in PBI

4.2 ETL Evaluation Methodology

In order to evaluate the implemented solution, an event log was developed. The system event log was implemented in the *Azure Data Factory*, more precisely, in the ETL process, and has as purposes of:

- errors detection during the ETL process, as well as knowledge of the pipeline defective.
- give intelligence about the processing time for each pipeline.
- provide ETL performance indicators.

As it was possible to verify previously (Section 4.1.3), the pipelines start and end with procedures for recording events, and these procedures, as soon as executed, fill in a table present in the Azure database, named Logs. Thus, this process provides information regarding the initial and final date of pipeline execution, successful or error completion, among others.

Results Messages

	ETL_ID	ID	PackageName	PackageLevel	StartDate	EndDate	IsSuccess	MsgError	PackageType
544	96	604	L_TradingEconomics	3	2020-06-18 22:19:34.920	2020-06-18 22:20:13.090	1	NULL	L
545	96	605	L_YahooFinance	3	2020-06-18 22:20:18.293	2020-06-18 22:20:36.373	1	NULL	L
546	96	606	ExportToDataLake	2	2020-06-18 22:20:46.093	2020-06-18 22:21:01.327	1	NULL	Export
547	97	607	ETL_All	1	2020-06-19 22:10:02.620	2020-06-19 22:34:30.630	1	NULL	ETL
548	97	608	E_All	2	2020-06-19 22:10:06.460	2020-06-19 22:30:41.913	1	NULL	E
549	97	609	E_DIM_Country	3	2020-06-19 22:10:11.073	2020-06-19 22:10:19.650	1	NULL	E
550	97	610	E_TradingEconomics	3	2020-06-19 22:10:25.997	2020-06-19 22:18:58.483	1	NULL	E
551	97	611	E_YahooFinance	3	2020-06-19 22:19:05.770	2020-06-19 22:30:36.630	1	NULL	E
552	97	612	T_All	2	2020-06-19 22:30:45.993	2020-06-19 22:32:11.663	1	NULL	T
553	97	613	T_TradingEconomics	3	2020-06-19 22:30:50.320	2020-06-19 22:31:09.617	1	NULL	T
554	97	614	T_YahooFinance	3	2020-06-19 22:31:13.520	2020-06-19 22:32:08.180	1	NULL	T
555	97	615	L_All	2	2020-06-19 22:32:15.583	2020-06-19 22:34:05.507	1	NULL	L
556	97	616	L_DIMs	2	2020-06-19 22:32:18.540	2020-06-19 22:33:07.443	1	NULL	L
557	97	617	L_TradingEconomics	3	2020-06-19 22:33:11.760	2020-06-19 22:33:39.960	1	NULL	L
558	97	618	L_YahooFinance	3	2020-06-19 22:33:44.790	2020-06-19 22:33:58.837	1	NULL	L
559	97	619	ExportToDataLake	2	2020-06-19 22:34:10.663	2020-06-19 22:34:26.650	1	NULL	Export

Figure 4.12: Logs table in *Azure SQL Database*

Figure 4.12 shows logs for all pipeline runs. It is possible to understand detailed information for each pipeline run through column *PackageName*. Column *PackageLevel* levels packages from one to three, being three a simple and independent ETL stage, two the complete runs of all pipelines associated with one of the ETL's stages and one the complete run of the whole ETL.
It is still possible to analyse the package type, the time elapsed for each package run and the message of error in case same of them do not end with success.
In order to verify with greater accuracy and ease the results that this mechanism provides, a solution has been implemented in PBI regarding the logs, which is exposed in the Section 5.3.2.

Chapter 5

Analysis

This chapter presents the analysis of the solution proposed compared with traditional and common solutions, the performance of the project as a whole, scoping the processes inside it, along with dashboards that depict possible useful analysis related with stock market analysis and comparison between the data sources.

5.1 Store and connection to Data Lake Gen2 vs SQL Database

The *Azure Data Lake Gen2* is built with focus on high performance and scalability at low cost. Looking at budget section (3.4) is evident that keeping the Data Lake storage active on this project is far cheaper than store into *Azure SQL Database*.
Additionally, the database tier selected is the most basic and cheaper because that is the whole objective of this project but, if the main historic storage was changed to the database itself certainly this tier would not be enough and as time passed the monthly cost of the database would be more and more expensive.
On the other hand, to make this escape from *SQL Database* pricing possible, additional steps are required. Firstly, exportation from the database to compressed files into the Data Lake through *Azure Data Factory* activities. Then, scripts running in *Azure Function App*, decompressing those files and make them available to direct connection into *Power BI*. Only after those two steps is possible to integrate files into visualization tools.
In contrast, if storage was made inside *SQL Database* the same connection would be direct. Obviously, the lead time of this connection is greater the more steps are required before the actual connection but in fact, performance suffers little impact considering that scripts running takes little time and exporting the tables from data warehouse to the data lake takes less than twenty seconds, as it is depicted in Figure 4.12 in section 4.2.

5.2 Store in Data Lake Gen2 vs Blob Storage

Azure Data Lake Gen2 combines the object storage modality of *Azure Blob Storage* with hierarchical file system storage provided by *Azure Data Lake Gen1*, resulting in a multi-modal storage that utilizes file system capabilities for analytical workload, keeping scalability and cost at levels associated with object storage.
Since the concept of folders and destinations in object storage is merely virtual, through URL concatenation, the performance of operations on files might not be as efficiently. Operations like moving files from one destination to another, that involve an object storage based system to call rename operations and delete operations over the URLs of each file, can be extremely more efficient if performed with an hierarchical file system that connects through a Distributed File System endpoint. This outcomes less operations and less computer resources required to accomplish the same result. Also conversely, the hierarchical file system storage supports atomic operations, improving data consistency because the complete operation will be succeeded or not as a unit.
Security is also improved through Role-Based Access Control (**RBAC**), that handle permissions to manage the service itself (storage account resource) and Access Control Lists (**ACLs**) where permissions are given directly to data objects. Because of this two security levels *Data Lake Gen2* provide an extra layer of flexibility when defining permissions and security.
Azure Data Lake Gen2 is briefly a merged version of *Data Lake Gen1* with *Blob Storage*, keeping the performance and security aspects of a file system storage at low price and high scalability, typical from a object storage system.

5.3 Data Visualization Application

As evidenced in the previous chapter, the last step of the implementation phase is based on the construction of an analysis application using extracted and processed stock market data, using Microsoft Power BI (PBI) as the construction tool for this analysis. The home page, depicted in Figure 5.1, redirects for three possible analysis:

- **Last 15 Days** - An ad-hoc updated solution that allows the user to analyse data from the most recent fifteen days without having to run any other request. It consists on *Price Analysis*, one of the key topics since it provides a comparative analysis between the two data sources from which data were extracted in order to check whether, at a given moment, a price of a given company is different from the other source; *Dividend Percentage Analysis*, that allows to see the changes in the dividend that companies pay to the investors and that depend on the current stock price among others; *Market Capitalization Analysis*, indicator that refers to the total dollar market value of a company's outstanding shares of stock; *Volume Analysis*, presenting the amount of stock traded for each day, whether bought or sold; And finally *PE/Ratio Analysis*, this indicator relates the current price of a company's stock with the earning per share indicator, allowing to calculate values the Return on Investment (ROI)

- **On-Demand** - Allows the user to request the dates he pretends to analyse. The process behind this application gather all data requested through the repository (Data Lake) into files to where Power BI has fixed connections and automatically generates the same analysis dashboard mentioned for the previous solution. Again, the ad-hoc feature allows the user to choose the market and the company if interest.

- **ETL Evaluation** - A collection of visuals and metrics that enables a more technical analysis over the ETL process, consisting on a filter by type of ETL's stages that cross information from the table mentioned in Section 4.2 and presents the number stages ran, the successful number and rate of rans, the average time of a run and also a chart showing the time each stage takes to run by day.

Figure 5.1: Home Page of the Application

In order to carry out a proper analysis, the application is presented in two phases. First are presented the pages related to the stock market and then the pages that analyse the ETL performance.

5.3.1 Stock Market Analysis

As mentioned, the analysis of the stock market can be made from the perspective of the last fifteen days or through an on-demand perspective, that is, through a request made beforehand by the end user who provides the data relating to the dates selected by him/her. These pages make it possible to choose both the market and the company to be analysed through dropdown menus in the upper right corner.
As illustrated in Figure 5.2, the price range of the company's stock *Apple* for the whole month of

May is displayed and by passing the mouse over a certain day it is possible to compare the values of the two data sources.

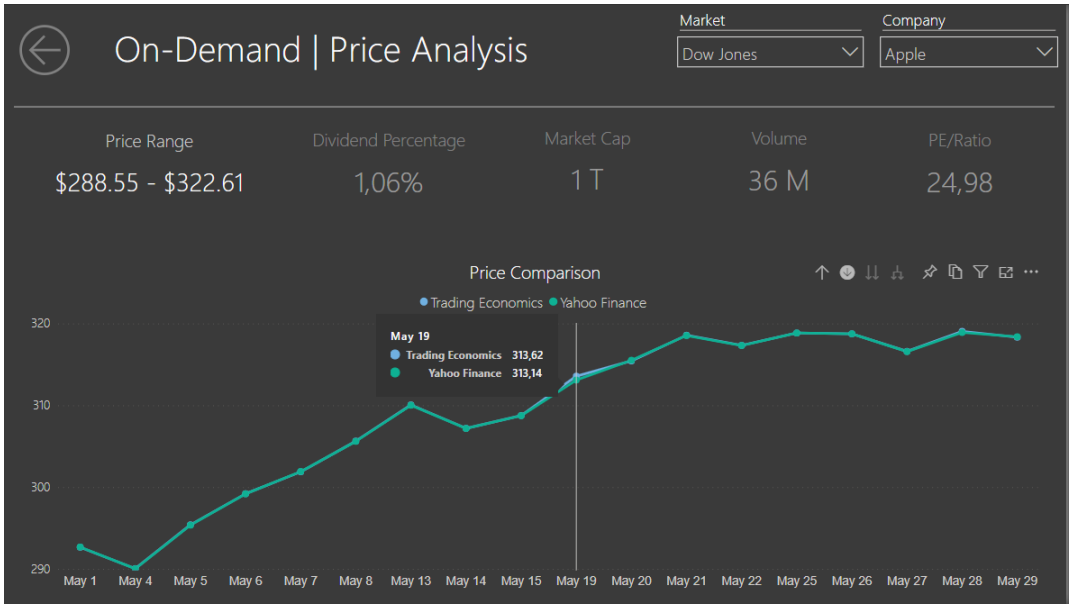

Figure 5.2: Company's Price On-Demand Analysis Dashboard

As an example, the Figure 5.3 shows that clicking on the analysis option of Market Capitalization is possible to change the analysis for the evolution of this same indicator. In this example the macro perspective of analysis was changed to the historical of fifteen days, in order to prove its full functioning.

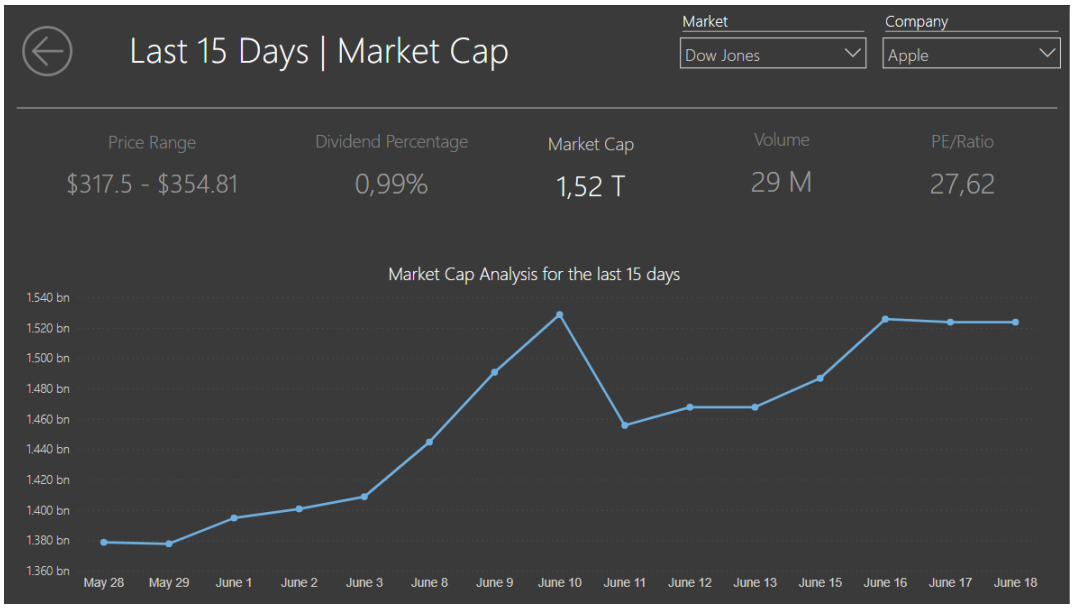

Figure 5.3: Company's Market Cap Analysis Dashboard for the last 15 days

5.3.2 ETL Performance Analysis

As discussed in Section 4.2, a mechanism has been implemented to record information regarding the ETL processes and the process of exporting tables from the *SQL Database* to compressed files in *.parquet* format that are uploaded to the *Data Lake Gen2*.
In Figure 5.4 it is possible to see that applying the filter only to the ETL process as a whole, information is given regarding the success rate in the executions of these pipelines (in the *Azure Data Factory*) and the average time of duration of an entire ETL. The graph shows slight variations that can be justified by the variation of the virtual machines to which the Azure resources resort over time, but which nevertheless has no impact on the optimal performance of the processes.

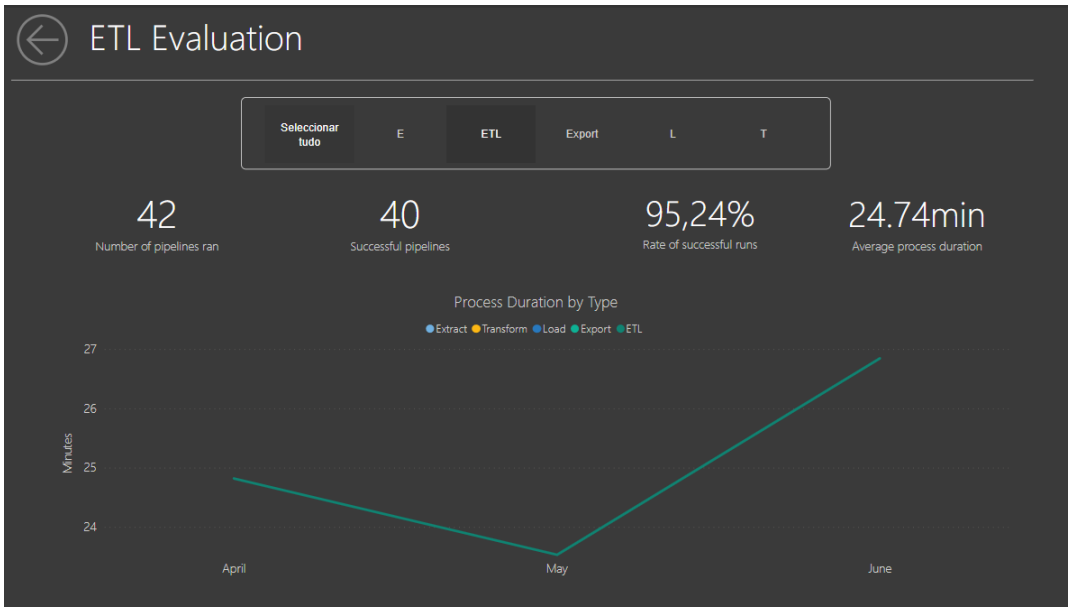

Figure 5.4: ETL Process Evaluation Dashboard

Focusing the interest only on the Extraction processes (**E**), depicted in Figure 5.5, it is interesting to note that the average time of an extraction process is approximately nine and a half minutes. However, it is important to note that this project performs two extraction processes each time the ETL runs, so the extraction processes have an associated time of approximately nineteen minutes. This represents almost the entire duration of the ETL process.

This analysis allows us to conclude that if there is interest in optimizing the ETL process and reducing its average execution time, a good approach is to start by trying to reduce the execution time of the extraction processes.

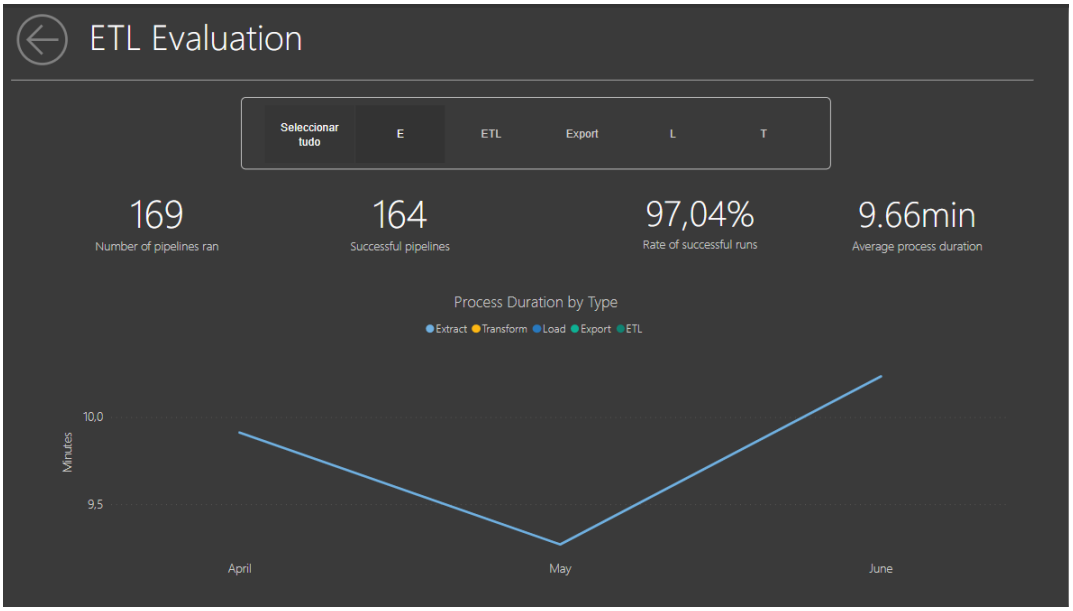

Figure 5.5: Extraction Process Evaluation Dashboard

Chapter 6

Conclusions

The purpose of this chapter is to provide a conclusive approach to the project developed, in an enlightening way regarding the results of the solution developed, as well as with regard to the summary of the work developed, limitations and future work.

6.1 Summary of work developed

The first chapter presents the project and its motivations. With this, the following agenda has been developed and subsequently clearly defined. This chapter also mentions the volatility of the stock market and how a BI architecture can have an impact on a solution that aims to complement the tools of analysis of an issue dominated by an increasing variety and volume of information.
With that said, the market survey was carried out in Chapter 2. Here were presented the essential concepts to fully understand the project, analyzed and compared the technologies present in the market, both for *cloud computing* and for *data reporting and visualization*. In the technological field, the concept of *web scraping* was also presented and deepened, demonstrating the two main ways of implementing it. The stock market concept is also introduced, supported by examples from today's world where the above-mentioned technologies are involved with it.
In the chapter 3, the solution proposed in the first chapter is explained in more detail from a theoretical perspective. A macro view of the whole project is finally given with a certain level of detail. The various phases of the project are described. These contribute to the outline of the data model which is also explained in detail in this same chapter and allows the solution to gain the necessary structure to start implementation. At the end of this chapter is also estimated a budget that includes all the necessary resources to keep the proposed solution running for a period of six months.
After all the theoretical and study work was done, the solution was implemented. This implementation, as shown in Chapter 4, approached a more detailed phased structure than that mentioned in the previous chapter, for the sake of properly detailing all the steps taken. The methodology that allows us to follow the performance of ETL processes is also presented, since these processes represent the backbone of the proposed solution.

Concluding the development of the project, comparative analyses with the most traditional solutions in the world of *Business Intelligence* were made and presented in Chapter 5. Also, the developed application that allows visual analysis both of the data related to the stock market and the data related to the ETL processes was presented.

6.2 Limitations

With regard to limitations in the implementation of this work, it was noted that the acronyms of companies and markets did not coincide between extraction sources at all, which made it very difficult to attribute the relationship between these acronyms and the descriptive name, which is more enlightening. This limitation is contradicted with the scrip regarding the load of dimensions, which annuls the entry of a new company that has not yet been integrated in the data model. Nevertheless, it is a limitation with little impact since almost all the passive firms of interest for analysis are contained in the current model.
Regarding limitations in the analysis, the comparison of extraction sources is only possible for the stock price indicator, the remaining indicators belong to only one source and, although it is a viable analysis, it does not allow comparing its values for the two sources. In order to overcome this limitation, it would be necessary to integrate more data extraction sources into the project.

6.3 Future Work

As for future work, there are some passive improvements to be implemented.
First, the selection of more sources of structured information. Here, the integration of a *API* that imported information relative to the stock market could be valorizing, since, in spite of leaving a bit the project scope (regarding the web scraping), it contributed with organized information and easy treatment. This would be fruitful in building more intrusive and detailed visual analysis.
Another improvement to be implemented is the development of a simple interface that allows the user to select the dates for the information request *on-demand*. Currently, the user needs to insert these parameters in the function link (the trigger), however, an interface would introduce the use of this project feature in an easier and more open way to any user.
On the other hand, as far as innovative progressions are concerned, it would be interesting to implement predictive models on the historical data of the current solution. This would require the use of *Machine Learning* (ML) and it would be intended to forecast, for example, the price of a company's shares, so as to predict perfect timings for investing or for executing a certain action on the investment.

References

[1] Seref SAGIROGLU and Duygu SINANC. Big data: A review. *International Conference on Collaboration Technologies and Systems*, 2013.

[2] Gray P Negash, S. Business intelligence. *Handbook of Decision Support 2*, 2008.

[3] W3Schools. *Cloud Computing Architecture*. https://www.w3schools.in/cloud-computing/cloud-computing-architecture///.

[4] Dan C. Marinescu. *Cloud Delivery Model*, 2013. https://www.sciencedirect.com/topics/computer-science/cloud-delivery-model//.

[5] *Meaning of data in English*, 2020. https://dictionary.cambridge.org/dictionary/english/data//.

[6] Oracle. *What Is a Data Warehouse?*, 2020. https://www.oracle.com/database/what-is-a-data-warehouse/.

[7] Peter Mell and Timothy Grance. The nist definition of cloud computing. Technical report, National Institute of Standards and Technology, September 2011.

[8] Anna-Louise Jackson and Arielle O'Shea. *What Is the Stock Market and How Does It Work?*, January 2020. https://www.nerdwallet.com/blog/investing/what-is-the-stock-market//.

[9] Geoff Boeing and Paul Waddell. New insights into rental housing markets across the united states: Web scraping and analyzing craigslist rental listings. *Journal of Planning Education and Research*, page 3, 2016.

[10] Kedar G. Pathare Anand V. Saurkar and Shweta A. Gode. An overview on web scraping techniques and tools. *International Journal on Future Revolution in Computer Science Communication Engineering*, page 2, April 2018.

[11] Hadi Pouransari and Hamidreza Chalabi. Event-based stock market prediction. pages 1–5, 2008.

[12] Carol Anne Hargreaves and Chandrika Kadirvel Mani. The selection of winning stocks using principal component analysis. *American Journal of Marketing Research*, August 2015.

[13] Carol Anne Hargreaves and Chandrika Kadirvel Mani. Stock trading using analytics. *American Journal of Marketing Research*, March 2016.

[14] Margaret Rouse. Data visualization, July 2017. https://searchbusinessanalytics.techtarget.com/definition/data-visualization.