

# **Cryptocurrency Technical Analysis**

Tools & BOTs Development



Bachelor thesis

Degree Programme in Computer Applications  
Hämeenlinna University Centre  
fall, 2021

Alessandro Zanni

---

Author	Alessandro Zanni	Year 2021
Subject	Cryptocurrency Technical Analysis: Tools & BOTs Development	
Supervisors	Esa Huiskonen	

---

## ABSTRACT

The purpose of this thesis is to create a web application hosted in AWS and a BOT which automatically trades cryptocurrencies based on technical analysis indicators. Investing in cryptocurrencies is very risky and many times sentimental investing makes the investor buy and sell crypto at the wrong moment. Thanks to the implementation of these tools investors can utilize technical analysis and thus mathematics to understand when it is a good moment to buy or sell a specific coin. The web app enables the user to study an investing strategy and the BOT uses the strategy found to trade cryptocurrencies automatically.

In order to achieve the development of these tools a good knowledge of Python and Nodejs and React is needed as well as a good understanding of React. The creation of the web app and the BOT is possible thanks to the creation of an algorithm that uses specific libraries dedicated to technical indicator calculations. This thesis is mostly practical, since the amount of code needed to develop all the services is rather massive. However, everything that is going to be developed is firstly introduced in the theoretical part, especially everything regarding the main subjects, which are cryptocurrencies and technical analysis.

The conclusion of the paper is that a developer can create a web app hosted in AWS without having high costs. Moreover, the creation of an investing BOT without having a deep knowledge of the subject thanks to the use of libraries is achievable.

Keywords development, technical analysis, BOT, cryptocurrency

Pages 52

## Glossary

HTML	HyperText Markup Language for web pages
DNS	Domain Name System
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
CLI	Command Line Interface
API	Application Programming Interface
POS	Prof of Stake
POW	Prof of Work
SCSI	Small Computer System Interface
CIFS	Common Internet File System
NFS	Network File System
SSL	Secure Socket Layer

## Contents

1	Introduction.....	6
2	Cryptocurrencies and technical analysis a general overview .....	7
2.1	Introduction to the crypto world .....	8
2.2	Crypto exchange platforms .....	9
2.3	Technical analysis.....	10
2.3.1	Technical indicators.....	11
2.3.2	MACD & RSI .....	11
2.3.3	BOT & technical analysis .....	13
3	AWS .....	15
3.1	AWS storage and application delivery .....	15
3.2	AWS routing .....	15
3.3	AWS cloud computing.....	16
4	Web app .....	17
4.1	Architecture .....	17
4.1.1	S3.....	18
4.1.2	Route 53 .....	20
4.1.3	Cloudfront implementation.....	21
4.1.4	EC2 and algorithm deployment.....	22
4.1.5	API Gateway setup .....	23
4.2	Web app frontend.....	25
4.2.1	The home page.....	27
4.2.2	Tools implementation .....	28
4.2.3	Additional pages.....	33
4.2.4	Google Analytics .....	34
5	Automated BOT.....	36
5.1	BOT infrastructure.....	36
5.1.1	Fetching historic data .....	36
5.1.2	Algorithm and signals generator .....	38
5.1.3	Exchanging in Binance .....	38
5.1.4	Telegram notifications.....	39
5.2	Running the BOT .....	41
6	Results .....	43
6.1	Products analysis.....	43

6.2	Tools results and profit .....	44
6.3	Comparing BOT and holding strategy .....	45
6.4	Possible negative impact.....	46
6.5	A better future impact for cryptocurrencies.....	47
7	Summary .....	48

## Figures, program codes, commands and tables

Figure 1	- MACD plot taken at 22:12 on the 1st of October 2021 .....	12
Figure 2	-RSI plot taken at 22:43 on the 1st of October 2021 .....	13
Figure 3	– Web app plot generated at 23:00 on the 27th of September 2021 .....	14
Figure 4	– Web app architecture .....	17
Figure 5	– S3 Buckets view .....	18
Figure 6	– S3 Buckets access.....	19
Code 1	– S3 Buckets policy .....	19
Figure 7	– S3 Bucket hosting settings .....	20
Figure 8	– Route 53 console view .....	20
Figure 9	– Cloudfront console view .....	21
Figure 10	– SSL certificate.....	22
Figure 11	– EC2 instance view .....	22
Figure 12	– Logged into EC2 instance .....	23
Figure 13	– API Gateway methods and resources .....	24
Figure 14	– API Gateway CORS settings.....	25
Code 2	- React Router implementation.....	26
Code 3	- Helmet example .....	26
Code 4	– Home page.....	27
Code 5	- Styled Components containers .....	28
Code 6	- Styled Components titles .....	28
Code 7	– React Hooks .....	29
Code 8	- Setting a variable.....	29
Code 9	- Plot Generator implementation .....	29
Code 10	- Plot Generator price fetcher .....	31
Code 11	- Plot Generator showing the image .....	31

Code 12 – Plot Generator Youtube video implementation .....	32
Code 13 - Plot Generator API request towards API Gateway.....	32
Code 14 - Profit Calculator implementation.....	33
Code 15 – Privacy Policy .....	34
Code 16 - Google Analytics implementation .....	34
Figure 15 – Google Analytics .....	35
Figure 16 – BOT architecture.....	36
Code 17 - Fetching historic data .....	37
Code 18 - Cron job.....	37
Code 19 - Binance buy method .....	38
Code 20 - Binance sell method .....	39
Code 21 – Telegram notification .....	40
Figure 17 - Telegram message.....	41
Figure 18 - Running BOT .....	42

## **Annexes**

Annex 1      Material management plan

# 1 Introduction

2020 was the year of the beginning of cryptocurrency mass adoption. Since Decentralized finance awareness has been spreading widely around the globe, big companies, such as Tesla, have acquired Bitcoins and one of the richest men on the planet, Elon Musk, has publicly supported the cryptocurrencies world. One of the biggest American stock exchange Nasdaq and many others are comparing the rising of crypto to the rising of the Internet in the 90s, stating that Bitcoin has already outpaced the Internet growth. This opened enormous opportunities for investors and entrepreneurs. The latter category of people needs a good idea in order to be successful and this thesis presents my personal idea and every detail about its realization. (Nalawade, 2021)

This paper explains the creation of a never seen before web application, which enables users to develop crypto trading strategies by using technical analysis indicators. The web app includes a profit calculator that gives to the user the possibility to test their strategies and a plot generator, which enables the user to visually analyze the trend of their decisions.

Moreover, the tools and algorithms created for the web app are utilized to create a BOT, which automatically manages a Binance Account (crypto exchange platform) Portfolio. The BOT exchanges the assets between USDT (stable cryptocurrency) and ADA (the coin of the Cardano blockchain) automatically basing its decision on buy and sell signals calculated using an algorithm coded in Python.

This thesis reports the path of realizing these tools from scratch, the thinking behind the idea and technologies selection, the software architecture, cost management, implementation of Google analytics, utilization of source control through GitHub, cloud development, and, of course, a lot of code.

The research questions which this thesis answers are as follows:

- How to create a working algorithm that gives trustable outcomes?
- How to deploy a fully working web-app in the cloud?
- How does BOT automation work and what technologies to use for development?

## 2 Cryptocurrencies and technical analysis a general overview

This thesis mainly focuses on the development of the tools and BOT. Most of the code used for the development is going to be shared, with the exception of the Python algorithm. There are three main subjects that are going to be discussed:

- The basics of cryptocurrencies
- The basics of technical analysis
- Software development methods (including cloud, frontend, backend, software architecture)

The basics of cryptocurrencies and technical analysis are an overall introduction to the topics, but they are not studied deeply, since the scope of this document is the development process and the creation of the web app and BOT. However, to fully understand the tools, it is essential to have a general understanding of the subjects.

Up until the period during which this document has been written, no similar web apps are publicly available on the net. Instead, there is a different situation for the BOT. Currently there are many BOTs circulating, but the one developed in this case is going to be different, since the users are going to be able to decide the technical indicators to utilize after checking the profitability of the web app. This is because the users do not have the power of deciding the technical indicators to utilize.

If this is not making much sense yet, it is related to what was reported at the beginning of the chapter. Without a general understanding of the subject it is challenging to create any application or even follow a paper.



## 2.1 Introduction to the crypto world

Cryptocurrency is a digital currency, which is secured by cryptography. This means that a cryptocurrency does not exist as a material asset, it is not possible to have a material Bitcoin in the wallet. Most of the cryptocurrencies are decentralized and they are based on blockchain technology, which is a network of computers which work as a distributed ledger. The main characteristic of cryptocurrencies is that they are not controlled by a central authority, such as banks and thus they are not controllable by governments. (Brown, 2020, pp. 32-35)

There are different types of blockchains. The Bitcoin blockchain, for example, is different from the Ethereum and from the Cardano blockchains for example. One of the main differences for example between the Bitcoin Blockchain and Cardano is that the first has a proof of work model, where the second one has a proof of stake one. These terms are part of a higher concept, the 'consensus mechanism' which is a mechanism that is utilized in computers and blockchain to have an agreement on a single piece of data or state of the network between the distributed process or multi-agent ones, exactly like cryptocurrencies. The main information hidden in the above definition is that this mechanism is very useful for record-keeping. In simple words the consensus mechanism in blockchain system is how the crypto transaction is ensured, since there is no central entity doing it as for example banks do in centralized finance. (Frankenfield, 2021a)

Before going to the definitions and differences between the two different models, there is an additional concept that requires to be introduced: mining. The work of the miners is to confirm transactions by solving cryptographic mathematical problems, also called hashes. Once a hash is solved by a miner, which is done in a "block" from which the name blockchain, the following block is going to have a new cryptographic signature and it is going to be visible by everyone. In fact, in the decentralized finance it is possible to see every single transaction that happened, but it is not possible to know the parties involved in the transaction. (Brown, 2020, pp. 27)

In different consensus mechanisms, the miner work changes. However, the final goal of the miner is always the same. The difference is the "how to" process, and this identifies the main differences between blockchains. The Proof of Work (PoW) requires nodes on a network to provide evidence that they have expended computational power (i.e. work) in order to achieve consensus in a decentralized manner and to prevent bad actors from overtaking the network. An example of a

blockchain that uses PoW is the Bitcoin Blockchain or the Ethereum one. Cardano blockchain uses a PoS model instead. (Frankenfield, 2021b)

The Proof of Stake (PoS) concept states that a person can mine or validate block transactions according to how many coins they hold. This means that the more coins owned by a miner, the more mining power they have. (Frankenfield, 2021c)

## **2.2 Crypto exchange platforms**

As cryptocurrencies are decentralized and totally detached from the FIAT currencies (Fiat money is government-issued currency that is not backed by a physical commodity), there is the need of crypto exchange platforms to play the role of exchanges between the two.

An exchange platform is where people can exchange cryptocurrencies with FIAT and viceversa. Following a list of the most famous platforms, based on a research done by CoinMarketCap (CoinMarketCap, 2021)

- Binance (<https://www.binance.com/en>)
- Coinbase (<https://www.coinbase.com/>)
- Huobi Global (<https://www.huobi.com/en-us/>)
- FTX (<https://ftx.com/>)
- KuCoin (<https://www.kucoin.com/>)
- Kraken (<https://www.kraken.com/>)
- Binance.US (<https://www.binance.us/en/home>)
- Bitfinex (<https://www.bitfinex.com/>)
- Bithumb (<https://en.bithumb.com/>)

- Gate.io (<https://www.gate.io/>)

The BOT created utilizes the Binance Exchange platform and the APIs directly provided by them. I personally selected Binance as my main choice for two main reasons. The first one is that I already have an account and I am familiar with the interface of the application as well as the functionalities. Moreover, the APIs provided by Binance are clearly documented and thus easy to use.

Binance is one of the most trusted exchange available in the market. The company was launched in July 2017 by Changpeng Zhao (CZ). The platform not only gives the opportunity to exchange FIAT currencies with Crypto, but it also provides tools for analysis as well as an NFTs (Non-fungible tokens) marketplace. In this paper we are not going to cover NFTs.

The BOT as well as the web-app utilize the APIs to retrieve information about the markets and thus the crypto prices in real time. Moreover, the BOT uses the APIs to make real asset movement in the Binance account. APIs are the best way for the tools developed to automate the BOT and to get information about the market in real time.

### **2.3 Technical analysis**

The term “technical analysis” has been used many times in this thesis. Therefore, it is time to give a definition to this term. Technical analysis is the process of identifying trend changes, which indicates a good moment to buy or sell a certain asset. Technical analysis methods can be used for example for stocks and cryptocurrencies and this latter one is the topic that interest us the most. In other words, technical analysis uses technical indicators, for example RSI or MACD to determine if it is a good moment to buy, sell or do nothing. (Pring, 2014, pp. 3)

One of the main aspects of technical analysis is the subjectivity. When an analyst analyze the data there are three main activities that can be observed: the identification of the price and indicator patterns, the data interpretation and the potential future price behavior. Analyzing this kind of data is subjective because different behavioral traits and filters are unique to each analyst or observer. As a result, every analyst can have a different perception of the market. (Lim, 2016, pp. 16)

### **2.3.1 Technical indicators**

Technical indicators are used to illustrate the characteristics of a market. They represent different mathematical models and studies the goal of which is to describe what the current status of the market is. However, it is the task of the analyst to gather the information given by these indicators in order to understand what the indicators are telling in a specific moment in time. (Pring, 2014, pp. xii)

More indicators shall be used together in order to have an extensive view and understanding of the situation. This does not mean that any indicator can be associated to any other. In fact, there are specific indicators, which identify specific characteristics of a market trend. The BOT developed during the creation of this thesis uses mainly two indicators, of which two are consider oscillators, the moving-average convergence divergence (MACD) and the relative strength indicator (RSI). (Pring, 2014, pp. 244)

### **2.3.2 MACD & RSI**

My profession is software developer and not investor. Therefore, even if I developed a certain curiosity in the investing field, I would never address myself as a knowledgeable person in the field of investing money and using technical analysis at its best. However, developers can create a well-working-application thanks to libraries created by experts in the field. This enables people, such as me, to materialize ideas and products without the need of having a deep knowledge of the specific subject. The realization of the algorithm I developed and overall, the products that work around it, are an example of how much open-source libraries can elevate a developer in reaching a goal. The algorithm created is slightly analyzed later in this paper. However, it is important to focus on the utilization of the technical indicators and how they can perform. Before introducing the indicators, it is essential to learn the concept of candlestick chart. A single candle represents the high, low, open and close prices for a specific period of time for, in our case, a crypto asset. (Hayes, 2020)

For instance, MACD and RSI are momentum indicators and even if they are part of the same category, they differ from each other. The MACD is mostly used to measure the strength of the price movement. This is done by gauging the divergence between two exponential moving

averages (EMAs). Usually, the period selected when calculating the MACD is the 12 and 26 period EMAs. This indicator is very simple to understand when it is plotted as Figure 1 demonstrates:

Figure 1 - MACD plot taken at 22:12 on the 1st of October 2021



The upper part of the picture represents the price trend of the Bitcoin up to the 1st of October 2021. Each candlestick represent one day. In the second half of the pictures, it is represented the MACD value for each single candle (ergo day). The indicator is composed by three main elements. The first one is the 12 months period EMA represented by the purple line. The second one is the 26 months EMA, which is represented with the blue line. Finally, the signal length, which looks like a candle, but it shows the indicator sentiment in that specific moment in time. Comparing the candlesticks of the Bitcoin price and the indicator line strength it is possible to notice that the two are correlated. When the MACD value changes from negative to positive, thus whenever the color of the MACD value changes from red to green in the graph, the indicator tells that it is a good moment to buy Bitcoin. Vice versa, whenever the value changes to negative, thus the color of the line strength of the MACD changes from green to red color and it is translated as a good moment for selling.

The green circle is the moment in which the indicator suggests buying and with red when it suggests to sell. It is noticeable that not always the indicator would have suggested to buy low and sell high. Therefore, more indicators shall be used when using technical analysis, since only using the MACD for example would not give excellent results.

The other indicator utilized is the RSI, which tells if a market is overbought or oversold. The scale of the measurements range from 0 to 100. When the RSI is over 70, it means that the market is overbought and thus the price supposedly will go down, thus it is time to sell. If the RSI index is below 30, it means that the market is oversold and thus it indicates that the price might start to go up, in other words it is a good moment to buy.

Figure 2 -RSI plot taken at 22:43 on the 1st of October 2021



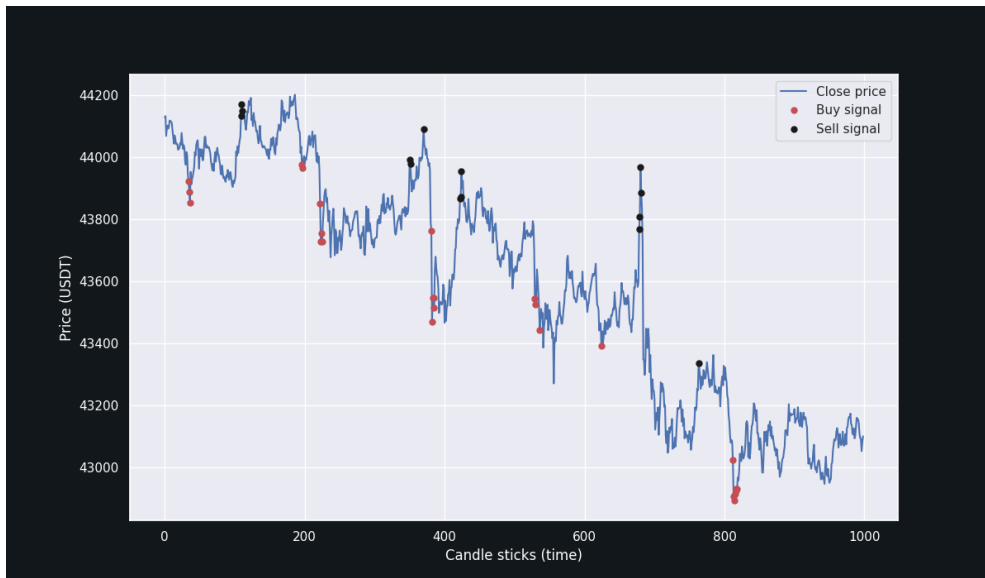
The Figure 2 it is reported the graph about the Bitcoin price in the same way as in the MACD picture. However, in this picture the RSI value is represented. The blue line on the bottom part of the picture shows the RSI which at that specific moment had around a value of 68, which means that the price of the Bitcoin was just about to reach the value of 70 and be considered overbought. As for the MACD the RSI indicator taken alone would have not given us the best buy and sell indicators. In fact, the buy and sell signals given in that period, represented respectively by the green and red circles, would have not provided a profit (Maverick, 2021).

### 2.3.3 BOT & technical analysis

The decision of using these two indicators when running the BOT is based on their simplicity. Testing the Python algorithm and the signals generated using this instrument showed that paring these two indicators gave rather good results. The following plot has been generated using as buy signal an  $RSI < 30$  and as sell signal an  $RSI > 70$  &  $MACD > 0$  (positive). The crypto selected in this case is Bitcoin and the price history data is based on 1-minute candlesticks, but this is not yet the

moment to focus on the details. The takeaway from Figure 3 is that by paring those two indicators, acceptable buy and sell signals were received:

Figure 3 – Web app plot generated at 23:00 on the 27th of September 2021



As the legenda says the red dots are the buy signals and the black dots are the sell indicators.

What matters the most now is that the sell signals are higher than the buy signals. This means that there is space for profit. If the BOT I developed would have followed these indicators during this range of time, it surely would have ended up with a decent profit.

This is to demonstrate that even if I am not an analyst it was enough to learn about few indicators in order to understand how the algorithm was supposed to be built and what outcome I should expect once the coding was done.

### **3 AWS**

Back In 2006, Amazon Web Services (AWS) started to offer IT infrastructure services to businesses in the form of web services. Nowadays, these services are called cloud computing. The advantages of cloud computing is the opportunity to save infrastructure expenses, in fact thanks to the scaling possibilities that the cloud offers, customers can pay only for what they need and use. (Baron, 2017, pp.)

The infrastructure used in the web app is going to use only a small part of the services provided by AWS. In the following chapter, these services are analyzed to have a clear understanding of why they were my main choice for the deployment of the web application.

#### **3.1 AWS storage and application delivery**

Amazon S3 is cloud considered an object storage. Amazon S3 storage is accessed over the Internet, but it is not associated to a server. In fact, S3 data is managed through API and not using the most typical protocols SCSI, CIFS, or NFS.

The most important feature of s3 that is in contrast with typical file systems is that when we GET an object or PUT an object, it operates on the whole object at once. One of the main goals of s3 is to be highly scalable and durable. (Kuppusamy & Vyas, 2014 pp. 36)

Another AWS service used when delivering the web app is Cloudfront, which is a fast content delivery network. It is perfect for delivering data, videos applications and APIs to users all over the world. Another great feature are the security capabilities and since I want to deliver the application securely through https, Cloudfront is exactly what I need to achieve this. (Amazon Web Services inc., 2021a, pp. 10)

#### **3.2 AWS routing**

Amazon Route 53 is a highly available and scalable cloud DNS web service designed with the purpose of giving to developers a trustable and cheap way to route end users to Internet applications.



Amazon Route 53 functions can be summarized in domain registration, since it enables the user to register domains. Also, it provides DNS services, thus it translates IP addresses to domain names. Finally Route 53 can be used for health checking. In fact, it can send automatic requests to application over the Internet to make sure they are up and running. (Amazon Web Services inc., 2021b, pp. 10)

### **3.3 AWS cloud computing**

Amazon API Gateway is utilized to manage, monitor and secure APIs. It is used as the entrance for application to access the data provided by the backend services. There are two different types of APIs that can be created, Websocket APIs and REST APIs. REST APIs are the ones that are used by our web application and BOT, thus it is necessary to give a short introduction to them.

API stands for Application Programming Interface and it is a set of definitions and protocols for building and integrating applications. A REST API is an application programming interface that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services. REST stands for representational state transfer. (Amazon Web Services inc., 2021c, pp. 8)

Amazon Elastic Compute Cloud (EC2) provides a resizable compute capacity, and it utilizes it comes with an easy-to-use interface. It allows complete control over the computing resources. The main focus of EC2 is to be able to scale easily, reliable, highly secure and easy to use. This service is very important in the matter of the web application since it contains the algorithm with which the frontend communicates. (Kuppusamy & Vyas, 2014 pp. 50)

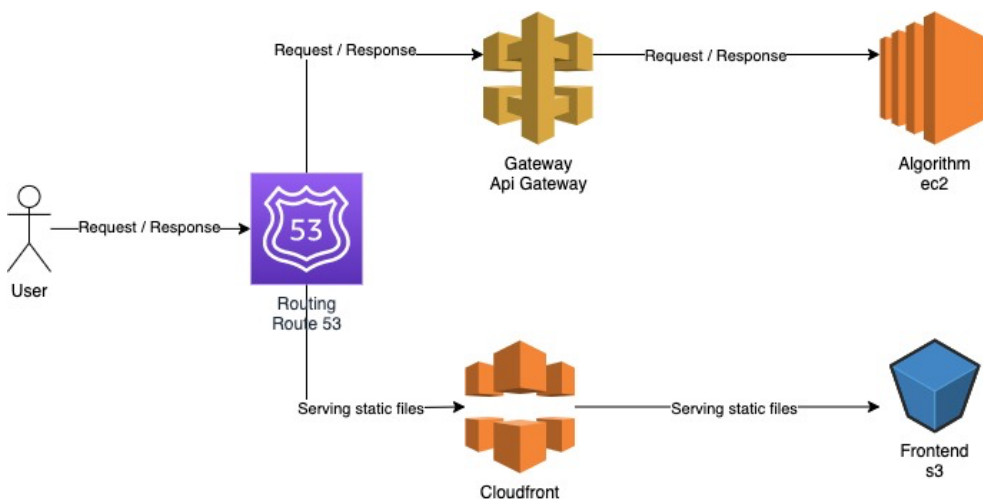
## 4 Web app

The goal of the web app is to use the Python algorithm to enable the user to see what signals technical indicators when certain values are provided. For example it was previously stated that the RSI would theoretically give a good sell signal once it reaches the value of  $> 70$ . However, someone could argue this statement and it could also be argued that RSI sell value should vary based on which other indicators we want to query the results. Another goal expected by the web app is to have a profit calculator, which would compare how trading using technical analysis over a certain period of time would result in a comparison between just buying and holding the coin for the same amount of time.

### 4.1 Architecture

I decided to opt for the cloud solution for the hosting and development of the web app. The cloud selected is Amazon Web Services since I have personal knowledge about the service and because it provides, if used correctly, one year of free hosting (with the exception of the domain costs). The final architecture of my solution is represented in Figure 4:

Figure 4 – Web app architecture



The static files of the web applications are going to be served through an s3 bucket, the delivery is done through Cloudfront which enforces the https usage as well. This is important to ensure secure connections to the users. Concerning the actual usage of the services provided by the web app, API Gateway is used as a gateway of the API requests made by the user when generating the

plot or calculating the profit. These requests are routed towards the EC2 instance in which the algorithm is running. If the request is successful, the response runs through the same services and it will result in the user viewing either the plot requested, or the profit calculated.

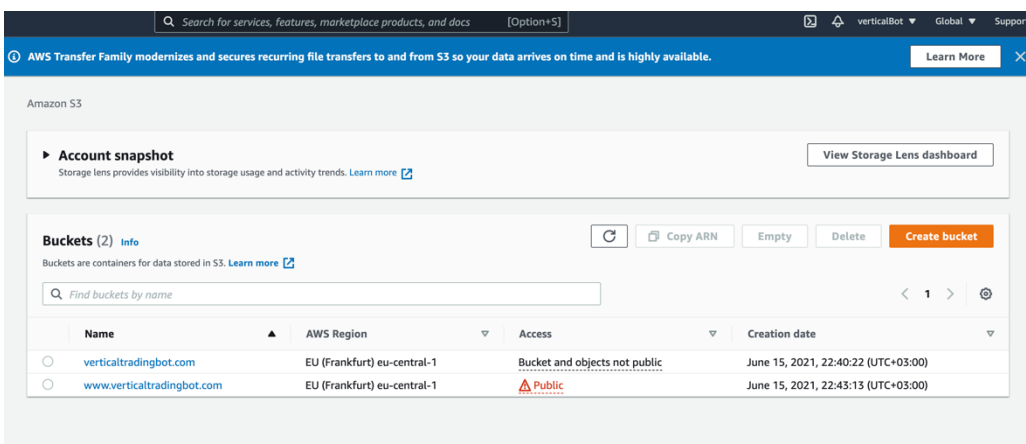
#### 4.1.1 S3

After creating a root account and logging as a user afterwards in AWS, I started to focus on the web app hosting in s3. However, it is necessary to have something to host at first.

The frontend of the application is going to be coded in React, thus before moving anything into the cloud, I started by creating a simple React application in my local machine by running the command `npx create-react-app NAME_OF_THE_APP`. After double checking that everything was working properly by moving to the created folder using `cd NAME_OF_THE_APP` and running `npm start`, which enabled me to see the application running in localhost, I was ready to deploy the dummy application to the bucket.

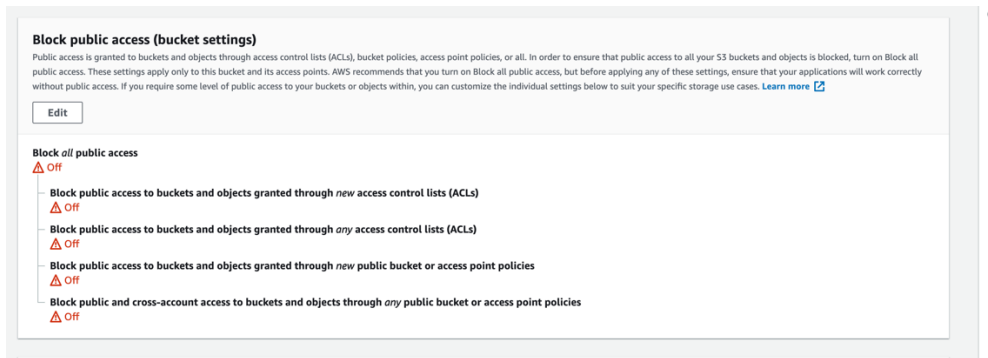
In s3 two different buckets are created. One is the public bucket, which is named using the complete URL of the web app and thus using the www naming convention. The other one is private and it contains the exact same files, but the name of it omits the www part of the URL. The domain name used for the web app is “verticaltradingbot”.

Figure 5 – S3 Buckets view



Once the bucket was created, I uploaded the static files into it and started to set the www bucket to be accessible by everyone. This process is obtained by turning off all the public access blocks, which are enabled as default for safety reasons by AWS.

Figure 6 – S3 Buckets access



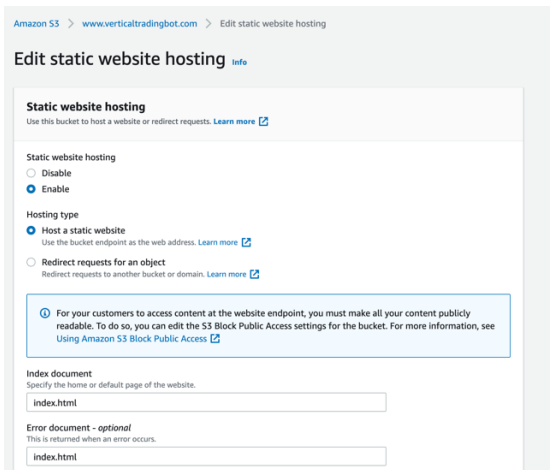
Next configuration step is to create a bucket policy, which allows public access to the public. The policy code is shown in Code 1:

Code 1 – S3 Buckets policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowPublicReadAccess",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::www.verticaltradingbot.com/*"
    }
  ]
}
```

The last step for the moment is to move to the properties tab and edit the “static website hosting” options. The important steps are to enable the static website hosting, set it as static hosting and finally indicate the index document, which in this case is “index.html”.

Figure 7 – S3 Bucket hosting settings

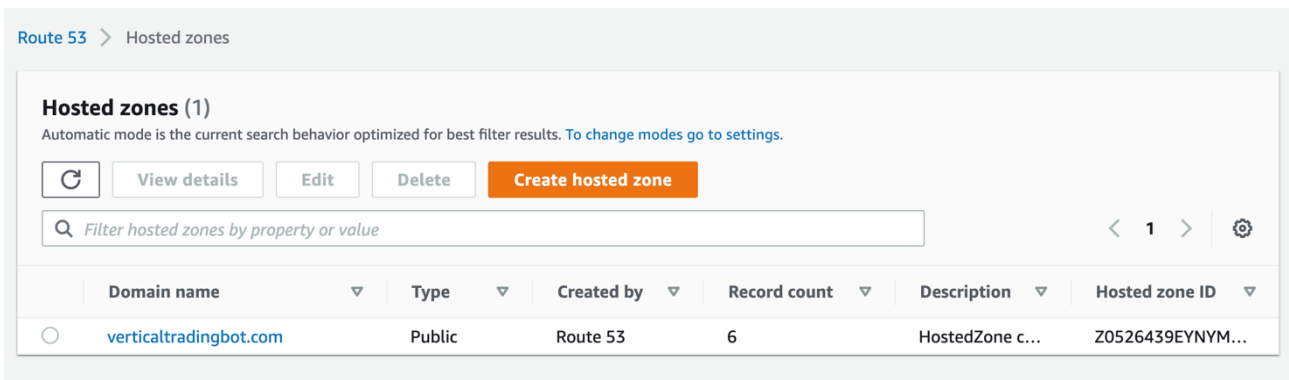


#### 4.1.2 Route 53

There are two main steps to be taken in Route 53, the domain registration and the DNS management. Luckily the domain verticaltradingbot.com was not taken by anyone, thus I was able to purchase it until June 2022, but thanks to the auto-renewal option, no one will be able to use it unless I decide to give up on it.

After purchasing the domain, I started setting up the routing. For security reasons the exact routing process is not shown, but it is important to say that the setting up of the routing is done at the same time while deploying other services, such as the EC2 instance and especially Cloudfront, which will be introduced in the next chapter. The result of the work is a working routing for our web application.

Figure 8 – Route 53 console view

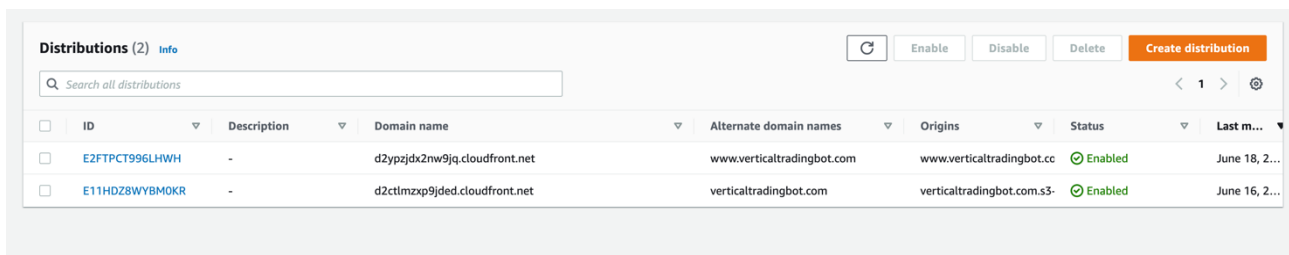


### 4.1.3 Cloudfront implementation

There are two main things to accomplish in Cloudfront, one is to create a distribution where we are able to create a default Cloudfront domain, which is routed to the purchased domain verticaltradingbot.com. The other accomplishment is to secure the connection using an SSL certificate. Thanks to this the web app is going to be accessible through HTTPS and thus be considered secured by more famous browsers such as Chrome.

More precisely, there are two different distributions that are going to be created, since the goal is to have people redirected to the web application in both of the cases, when a user tries to access the website through www.verticaltradingbot.com and when they insert in the URL search bar only verticaltradingbot.com.

Figure 9 – Cloudfront console view



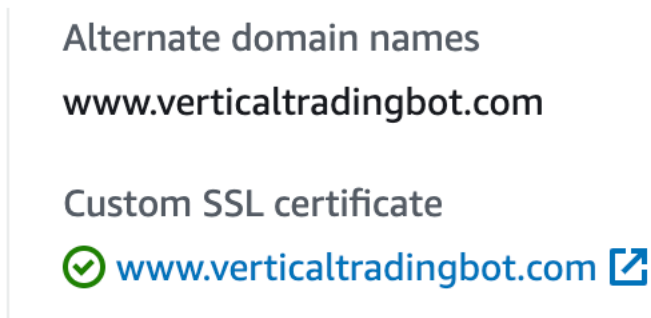
The screenshot shows the AWS CloudFront console interface. At the top, there's a header 'Distributions (2) info' with a search bar and buttons for 'Enable', 'Disable', 'Delete', and 'Create distribution'. Below the header is a table with the following columns: ID, Description, Domain name, Alternate domain names, Origins, Status, and Last m... (Last modified). There are two rows of data:

ID	Description	Domain name	Alternate domain names	Origins	Status	Last m...
E2FTPCT996LHWH	-	d2ypzidx2nw9jq.cloudfront.net	www.verticaltradingbot.com	www.verticaltradingbot.cc	Enabled	June 18, 2...
E11HDZ8WYBMOKR	-	d2ctlmzxp9jded.cloudfront.net	verticaltradingbot.com	verticaltradingbot.com.s3-	Enabled	June 16, 2...

It is noticeable that the Domain name provided by Cloudfront is not professional for a website therefore purchasing a domain name is needed. However, having the new domain being the main one, it does not mean that the Cloudfront domain is disabled. In fact, if we try to access <https://d2ypzidx2nw9jq.cloudfront.net/>, we are still able to open the web application without any issue.

The second step to be done is to create a certificate in order to have a secure HTTPS connection. AWS is very helpful in this sense, since it guides the user to create the certificate and it stores it in the certificate manager.

Figure 10 – SSL certificate

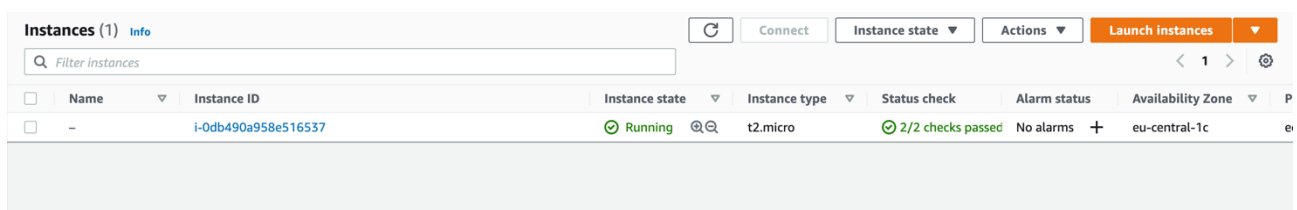


After setting up domain and certificate in Cloudfront console web app is served to public internet. It is possible to access the web app by using a secure HTTPS connection and the custom domain name <https://www.verticaltradingbot.com/>.

#### 4.1.4 EC2 and algorithm deployment

The Python algorithm is deployed in a micro EC2 instance, since I am not expecting high traffic for the web application, for the moment micro was the best and cheaper option for the deployment. The first thing to do after heading to the EC2 instance page is to press on launch instances and go through the options that AWS console offers. There are plenty of different options available. However, for our case there is no need to add any additional service. Therefore, starting the instance is a matter of a minute. When creating the instance, it takes a moment before it is actually possible to log into it.

Figure 11 – EC2 instance view



Once the instance has been turned up, it is enough to press connect and a new CLI interface open. The next steps are setting up the instance with the algorithm and open the correct ports in order to make the instance accessible from outside. For security reasons this process is not shown. Next step is to upload the algorithm to the instance. The easiest way is to have the code in a Github

repository, so that it is enough to clone it directly into the instance with the command `git clone NAME_OF_REPO`. The final result is that the instance is running the algorithm using Gunicorn which is a Python WSGI HTTP Server for UNIX.

Figure 12 – Logged into EC2 instance

```
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-1038-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

System information as of Wed Oct  6 19:20:07 UTC 2021

System load:  0.0          Processes:    154
Usage of /:   55.1% of 7.69GB   Users logged in:  0
Memory usage: 60%          IPv4 address for eth0: 172.31.1.10
Swap usage:  0%

* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.

  https://ubuntu.com/aws/pro

125 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

*** System restart required ***
Last login: Fri Jun 25 09:04:44 2021 from 34.202.136.122
ubuntu@ip-172-31-1-10:~$ ls
verticalTrading
ubuntu@ip-172-31-1-10:~$ █
```

The algorithm running in the instance is now able to communicate with the outer world by accepting incoming requests and sending responses. The main work of the algorithm is to receive API calls from the frontend including historic data of a specific crypto, elaborate the data based on technical analysis libraries and send the response back to the frontend.

There is one last service that needs to be configured before moving to coding the frontend of our project. API gateway, which works between the EC2 instance and the frontend acting as the middleman between the two.

#### 4.1.5 API Gateway setup

Thanks to API Gateway we are able to make the frontend communicate with the EC2 instance in a simple and more secure way. The goal is to set up two different endpoints that are going to be the core of the plot generator and profit calculator of the web app. In fact, these two tools are able to work thanks to the communication with the backend code running in the EC2 instance.

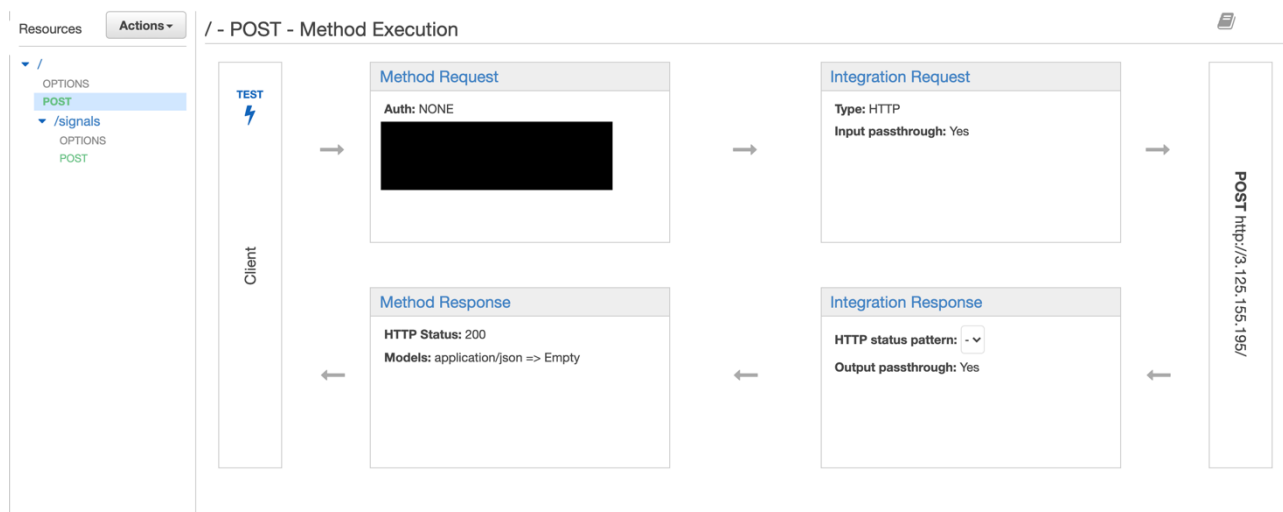


The creation of the service and the two endpoints is very straightforward and since we do not want to expose the IP address of the EC2 instance, I changed the gateway re-routed the calls received from the domain to the IP address. This way the IP address of the EC2 instance is behind the gateway itself and is not publicly visible. The endpoints created are the two following POST requests

- <https://k67g3ukds3.execute-api.eu-central-1.amazonaws.com/test/> , which is used for the plot generator
- <https://k67g3ukds3.execute-api.eu-central-1.amazonaws.com/test/signals> , which is used for the profit calculator

Now that the Method and resources have been set up, the AWS console for this specific service is reported in Figure 13:

Figure 13 – API Gateway methods and resources



There is still another important matter to solve. CORS, Cross-Origin Resource Sharing can be very troublesome, but thanks to the interface of API Gateway enabling the CORS is a matter of a click. This is done by adding the Access-Control-Allow-Headers, Access-Control-Allow-Methods and Access-Control-Allow-Origin values directly into the method integration response.

Figure 14 – API Gateway CORS settings

← Method Execution / - OPTIONS - Integration Response

First, declare response types using [Method Response](#). Then, map the possible responses from the backend to this method's response types.

HTTP status regex	Method response status	Output model	Default mapping
-	200		Yes

Map the output from your HTTP endpoint to the headers and output model of the 200 method response.

HTTP status regex:

Content handling:

Cancel Save

▼ Header Mappings

Response header	Mapping value
Access-Control-Allow-Headers	'Content-Type,X-Amz-Date,Authorization,X-Api-Key,X-Amz-Security-Token'
Access-Control-Allow-Methods	'OPTIONS,POST'
Access-Control-Allow-Origin	'*'

▶ Mapping Templates

+ Add integration response

Everything is now set up in the cloud. It was rather easy to create the hosting environment for the web application. AWS provides simple and cheap ways to host any kind of website in the net. Only few of the, almost 200 services have been utilized to host the web app. The only thing missing for completing the web application is the frontend development, which is going to be explained in the next chapter.

## 4.2 Web app frontend

The frontend of the web app is going to be coded using React. At the beginning of the AWS implementation a new application has been created in order to start uploading something to S3 while setting up the cloud environment.

However, it is now time to write some code. The main idea is to have a main page where the tools available are introduced as well as the available cryptocurrencies put available for the technical analysis calculations. At the top of this page there is going to be a menu with links to the tools and an about section.

Some information about tools and methods that are not going to be covered in the following chapters. The routing of the application has been achieved using React Router Dom, this enabled us to route to different URLs when rendering a new page. A small part of the code used for the routing is reported in Code 2. Another important tool used in the development has been Helmet which enables to enhance SEO related matters, such as page descriptions and tags. Helmet implementation is shown in Code 3.

### Code 2 - React Router implementation

```

<Router>
  <LogoContainer>
    <Link style={{ textDecoration: 'none', color: '#fff' }} to="/">
      <VerticalSVG />
    </Link>
  </LogoContainer>
  <BodyContainer>
    <CenterTabs>
      <AppBarStyled>
        <LinkMarginDivFirst>
          <Link style={{ textDecoration: 'none', color: '#fff' }}
to="/">
            HOME
          </Link>
        </LinkMarginDivFirst>
        <LinkMarginDiv>
          <Link style={{ textDecoration: 'none', color: '#fff' }}
to="/plot">
            PLOT GENERATOR
          </Link>
        </LinkMarginDiv>
        .....
      <Switch>
        <Route path="/privacy-policy">
          <PrivacyPolicy />
        </Route>
        <Route path="/plot">
          <Plots />
        </Route>
        .....
      </Switch>
    </CenterTabs>
  </BodyContainer>
</Router>

```

### Code 3 - Helmet example

```

<Helmet>
  <title>Plot generator</title>
  <meta
    name="description"
    content="Use technical analysis indicators and generate a plot
that shows your crypto strategy results. Vertical Trading enables the user
to visually understand if the investing strategy chosen can bring profits.

```

```
There are many crypto available as well as tens of technical analysis
indicators."
  />
</Helmet>
```

The last important thing to mention is that all of the code written below has been uploaded to the S3 bucket, which has been set up in chapter 4.1.1. In order to do so it is enough to drag and drop the public static files of the code in the bucket. AWS S3 will then recognize that the files in the bucket have changed and thus will start serving the new ones. The files and thus the web app is now accessible from the domain: <https://www.verticaltradingbot.com/>.

#### 4.2.1 The home page

The design of the home page is going to be fine and simple. When a user opens the web application a 100vh view opens showing only the menu and the name of the website. This is to give a strong first impression. The code for this page is straight forward since most of the content is written text or logos.

#### Code 4 – Home page

```
<Helmet>
  <title>Home</title>
  <meta
    name="description"
    content="Invest in crypto using technical analysis and avoid any
sentimental investment mistakes"
  />
</Helmet>
<TitleContainer>
  <TitleContainerSubtitle>
    <H1Title>Vertical Trading</H1Title>
    <SubtitleMainP>Your Crypto Trading Strategy</SubtitleMainP>
    <CryptoTradeGraphSVG />
  </TitleContainerSubtitle>
</TitleContainer>
<BoxContainer>
```

It is noticeable that the components used in Code 4 are not default components. This is because I am using React styled component library. This is a great library to style React application and create styles for components that can be utilized multiple times. After installing it using: `npm install --save styled-components` I started styling the Home page components. For example the Code 5 is the code used for styling the BoxContainer. Another simple example is the

code written to style the H2 elements as shown in Code 6. This is the Styled H2 which is applied to all the H2 in the entire project.

#### Code 5 - Styled Components containers

```
export const BoxContainer = styled.div`
  display: flex;
  justify-content: space-around;
  flex-wrap: wrap;
  margin-top: 7rem;
`
```

#### Code 6 - Styled Components titles

```
export const H2Titles = styled(H2)`
  place-self: center;
  font-size: 1.25rem;
  font-weight: 100;
  letter-spacing: 0.5vw;
  background-color: transparent;
  border: 2px solid #0371b9;

  @media (max-width: 768px) {
    font-size: 1.25rem;
    letter-spacing: 0.5vw;
  }
`
```

In Code 6 I have included also the media, which is used to make the website responsive and thus readable on any device. As previously mentioned, the home page did not require much logic since most of the content is explanatory text. The result is a good looking, responsive and clear introductory page.

#### 4.2.2 Tools implementation

The heart of the application is the implementation of the plot generator and the profit calculator. The first challenge is to make these tools clear for the user, since the concept of technical analysis is not known by many people.

The implementation of the plot generator consists in fetching data from Binance APIs about the historic prices of the selected crypto, enriching this information with the wanted technical analysis indicator and values and send them to the algorithm through a REST API. The first step to implement is the form in which the user will insert the crypto of interest, the candlestick time period and the technical indicators and values. The form includes a lot of selectable values, but the

just part of the code can give the main idea of how the form is created. I utilized React Hooks, which are considered a newer way to develop in React. The main concept is to have variables at the beginning of the file and set their value in the code and since React is used to develop single page application, we use to re-render the page with the updated information every time that the values of these variables is updated.

In Code 7 it is shown how React hooks are declared. These information are going to be updated once their wanted value is ready, in this case when the plot has been sent from the algorithm. In Code 8 it is shown how React hooks values are set.

### Code 7 – React Hooks

```
const [isImageLoading, setImageLoading] = useState(false)
const [pic, setPic] = useState()
```

### Code 8 - Setting a variable

```
.then((data) => {
  setImageLoading(false)
  setPic(data.data)})
```

Now it is the moment to create the form with which the user will provide the wanted technical analysis indicators and values. In Code 9 it is reported a big part of the code utilized for the implementation. Moreover, I left useful comments in the code in order to explain more in details what is happening in specific lines of the implementation. The comments can be identified, since they have an "#" at the beginning of the line.

### Code 9 - Plot Generator implementation

```
<MainContainer>
  <FormContainer>
    <form onSubmit={handleSubmit(onSubmit)}>
      <InitialForm>
        <FormTitles>Crypto</FormTitles>

#This is the code for the crypto coin selectors

        <Selector {...register('coinSelected', { required: true })}>
          <OptionStyle>BTC</OptionStyle>
          .....
        </Selector>
      <FormTitles>Candlestick time range</FormTitles>

#Here there is the creation of the candlestick time period input value
```

```

        <SelectTime {...register('selectedTime', { required: true
    }}}>
        <option>1m</option>
        .....
    </SelectTime>
</InitialForm>
<FormDivided>
    <SingleFormColumn>
        <FormTitles>Buy indicators</FormTitles>
        <InputContainerSelector>
            <Label>Indicator</Label>

```

#At this point there is the creation of the selector of the technical indicator

```

        <SelectIndicator
    {...register('selectedIndicatorFirstBuy', { required: true })}>
        <option disabled>MOMENTUM INDICATOR FUNCTIONS</option>
        <option value="buyRSI">RSI</option>
        .....
    </SelectIndicator>

```

#The following code enables the user to select the value and the "greater" or "less than" information

```

        <Label>Greater / Lower</Label>
        <SelectIndicatorGreater
    {...register('selectedSymbolSecondBuy', { required: true })}>
        <option value="<">{'<'}</option>
        <option value=">">{'>'}</option>
        </SelectIndicatorGreater>
        </SelectIndicatorGreater>
        <Label>input value</Label>
        <InputStyle id="thirdValueSell"
    {...register('thirdValueSell', { required: false })} />

```

#The following lines of code are for error handling

```

        <ErrorMessage
            errors={errors}
            name="thirdValueSell"
            render={({ message }) =>
<RequiredStyle>{message}</RequiredStyle>
        />
        </InputContainerSelector>
    </SingleFormColumn>
</FormDivided>

```

#Here there is the creation of the submit button

```

        <SubmitButton type="submit" onClick={() =>
setImageLoading(true)}>
            Create Plot
        </SubmitButton>
    </form>
</FormContainer>
<div>

```

#This last part of the code is where we set the image that we received back after submitting the request

```

        {pic && !isLoading && <DynamicPlot src={pic} />}
        {isLoading && (
          <LoadingSpinnerContainer>
            <LoadingSpinnerAnimation>Generating
Plot...</LoadingSpinnerAnimation>
          </LoadingSpinnerContainer>
        )}
      </div>
</MainContainer>

```

There are other important parts for the plot generator logic. For instance, once the form is submitted the following logic is followed. Firstly, based on the candlestick time period and the crypto selected by the user, the Binance API is invoked:

#### Code 10 - Plot Generator price fetcher

```

const singlePrices = async (coin) => {
  return await axios.get(
    `https://api.binance.com/api/v3/klines?limit=10000&interval=${coin.selected
Time}&symbol=${coin.name}USDT`
  )
}

```

In order to do this I utilized Axios, which is a great tool for making API requests. All I needed to do in order to use it was to first `npm install axios` and then `import axios` from `'axios'`.

The second step is to transform the data and make a REST API request towards the API Gateway, which will then forward the request to the algorithm running in the EC2 instance. The content sent towards the algorithm contains the Binance API response (after some transformation) and the values of the technical indicators that the user provided when submitting the form. It is important to say that the technical indicators values provided are queried using an “AND”, this means that if one indicator would give a buy signal in a specific moment in time and another indicator would not give the signal, the result would be that the plot will not have the signal at all.

The generation of the plot is done directly in the backend. This means, that the plot is received back from the backend and the only thing needed in the frontend, once the successful response is received is to display the plot.

#### Code 11 - Plot Generator showing the image

```

<div>
  {pic && !isLoading && <DynamicPlot src={pic} />}
  {isLoading && (
    <LoadingSpinnerContainer>

```



```

        <LoadingSpinnerAnimation>Generating
Plot...</LoadingSpinnerAnimation>
    </LoadingSpinnerContainer>
  )}
</div>

```

Now that the plot generator tool is completed, I can implement the introductory video in the page, where I show how to utilize the tool. This is simply done by installing ReactPlayer with `npm install react-player`. Since I want to style the video, I import the library into the styled component file and I give to the video container some details:

#### Code 12 – Plot Generator Youtube video implementation

```

export const VideoYoutube = styled(ReactPlayer) `
  margin: auto;
  @media (max-width: 768px) {
    width: 100%;
  }
`

```

Code 12 is then utilized in the tool file:

```

<VideoYoutube width="80vw" height="40vw"
url="https://www.youtube.com/watch?v=TPt6VTeUV4A" />

```

Finally, I added some text to introduce the tool is added at the beginning of the file and the plot generator is now working properly. The functionality implemented can be found at:

<https://www.verticaltradingbot.com/plot>

Concerning the other tool, there are not many differences coding wise. The concept is the same, the user provides the technical analysis information and values, after submission the Binance API is called in the same way as the plot generator, finally the data are enriched and the other endpoint available in API gateway is invoked as follow:

#### Code 13 - Plot Generator API request towards API Gateway

```

return await axios.post('https://k67g3ukds3.execute-api.eu-central-1.amazonaws.com/test/signals', payload)

```

The main difference with the plot generator is the way the data is enriched and the type of response given by the endpoint. The interesting point is how the profit is calculated, this is done in the frontend. Basically, based on the buy and sell signals that the algorithm provided after the user submission the Code 14 is invoked:

## Code 14 - Profit Calculator implementation

```
data.data.forEach((element) => {
  if (element.buy === 1 && action === 'sell') {
    money = money / element.close
    action = 'buy'
    if (actionCount === 0 && !hodlBuy) {
      hodlBuy = money
    }
    actionCount++
  } else if (element.sell === 1 && action === 'buy') {
    money = money * element.close
    action = 'sell'
    actionCount++
  }
  actualValue = element.close
})
setHodlProfit(hodlBuy * actualValue)
if (action === 'sell') {
  setCurrencySelected('USDT')
} else if (action === 'buy') {
  setCurrencyCurrentValueConverted(money * currentValue)
}
setCalculationReady(false)
setProfit(money)
```

The idea is that by starting at the beginning of the time period given by the user with 1000 USDT and the buy and sell signal received during that time length, the function calculates if the user would have had a profit or loss money. USDT is selected because it is a stable coin which floats around the price of the dollar. Moreover, it compares the results with people that would have just bought at the first buy signal and then never sold during the same time frame.

This tool can be a bit difficult to understand as well, thus I created an instructor Youtube video for this as for the plot generator. The tool can be found at:

<https://www.verticaltradingbot.com/calculator>

### 4.2.3 Additional pages

The core of the application has already been covered. However, there are still two main pages to create. One is the About us page, for which I do not report the code for this specific page, since it is only informative text about the website. However, the other page is very interesting, the privacy policy. European legislation is quite strict with the personal data and it is dictated by law that when personal data are collected the user must be informed and most importantly accept to give

this information. This is done through the acceptance of cookies. A very useful React library enables the developer to easily create a pop-up to handle cookies, it is named React Cookie Consent. After installing it with the usual npm command. I was able to implement the code in a very short period of time thanks to the CookieConsent component available in the library:

#### Code 15 – Privacy Policy

```
<CookieConsent
  style={{ background: '#11181c', borderTop: '1px solid #fff' }}
  buttonStyle={{ background: '#0371b9', fontSize: '13px', color: '#fff'
}}
  debug={false}
  buttonText="Accept"
  declineButtonText="Reject"
  enableDeclineButton
>
```

Finally, I created a page with the information requested by the legislation and thus everything that is collected when accessing the website. The privacy policy can be visited at

<https://www.verticaltradingbot.com/privacy-policy>

#### 4.2.4 Google Analytics

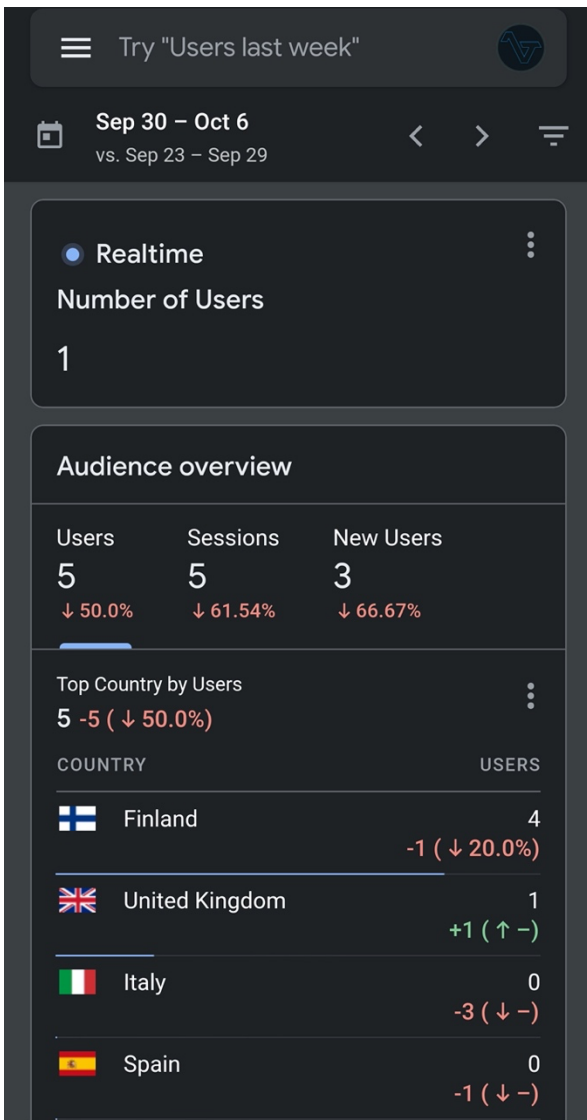
The final step of the web application is to enable Google analytics and monitor the statistics. After creating an account, it has been enough to insert in my index.html, which can be found in the React public folder Code 16:

#### Code 16 - Google Analytics implementation

```
<script
  data-ad-client="KEY"
  async
  src="PROVIDED_URL"
></script>
```

Now it is possible to monitor the web application traffic by using the Google analytics website or application:

Figure 15 – Google Analytics



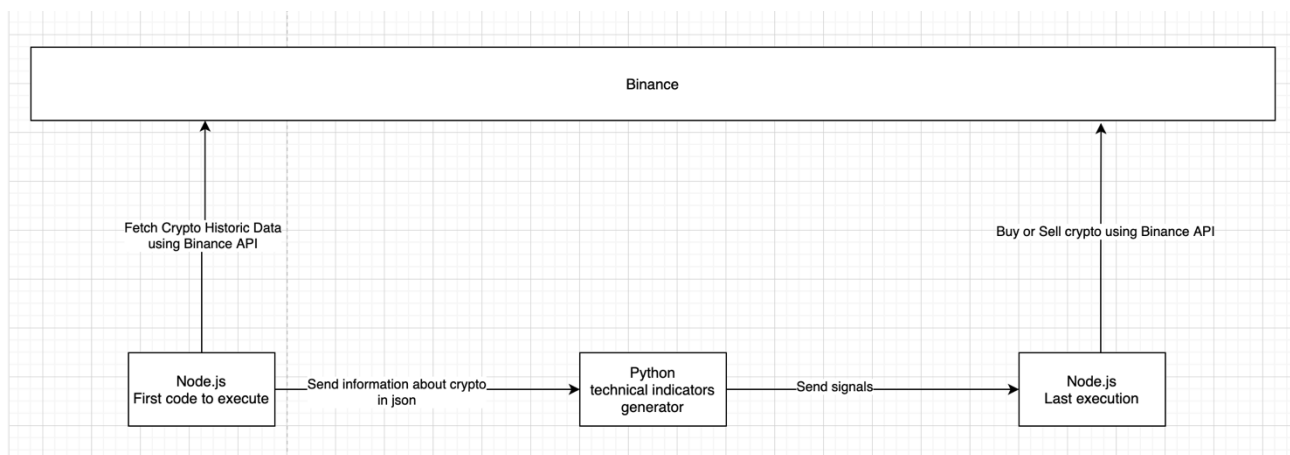
## 5 Automated BOT

The creation of the BOT is a great opportunity to safely invest in Crypto using technical analysis and avoid investment based on feelings. The BOT should automatically exchange money using technical analysis indicators. This can happen thanks to the Binance APIs, which enable the user to move cryptocurrencies using REST API calls.

### 5.1 BOT infrastructure

The creation of the BOT can be separated in three main parts. The first part is the code which fetches the historic price of the specified crypto and it creates the payload to be sent to the algorithm, which is the second element of the BOT. The algorithm receives the historic data of the desired crypto, it elaborates the results and it sends the technical analysis results to the third main part of the BOT. The last part of the BOT receives the technical analysis indicator and based on this information it either buy, sell or do nothing.

Figure 16 – BOT architecture



#### 5.1.1 Fetching historic data

The first part of the implementation consists in fetching the data of the specified cryptocurrency. The tool that executes this work is developed in Node.js and the fetching of the historic data is done using the Binance API endpoint:

<https://api.binance.com/api/v3/klines?limit=10000&interval=15m&symbol=ADAUSDT>

In order to utilize this REST API call an API key is not needed, because these information are available to everyone, since they are not related to my personal portfolio. The specific request reported above is requesting the historic data based on candlesticks of 15 minutes of the Cardano blockchain currency called ADA. If you click on the link, it will open your browser and show you the result in a web page. Code 17 is the one utilized to fetch the data using Axios:

### Code 17 - Fetching historic data

```
const config = {
  method: "get",
  url:
`https://api.binance.com/api/v3/klines?limit=10000&interval=15m&symbol=ADAUSDT`,
  headers: {
    "Content-Type": "application/json",
    "Cache-Control": "no-cache",
    timestamp: date.toTimeString(),
    Pragma: "no-cache",
    Expires: "0",
  },
},
};
//Fetching the data about BTC
const singlePrices = () => {
  const dataPromise = axios(config).then((response) => response.data);
  return dataPromise;
};
```

Once the response is successful the data is cleaned and sent using a json file to the algorithm. However, the BOT needs to do this automatically without the need of any human pressing a button every 15 minutes. Therefore, I decided to utilize Cron, which can be seen as an execution timer. This means that the application is going to be stored in a Cron batch job that triggers the run command every 15 minutes. The Cron implementation is represented in Code 18, where in the second part of the code the algorithm is triggered and potential error messages are reported.

### Code 18 - Cron job

```
cron.schedule("*/15 * * * *", async () => {
  await sleep(10000);
  //Writing to the input folder of the price-analysis folder
  mappedPrices().then((mappedPricesObject) => {
    mappedPricesObject.pop();
    console.log("wrote file to input folder in price-analysis");
    fs.writeFile(
      "../price-analysis/input/BTCValues.json",
      JSON.stringify(mappedPricesObject),
```

```

function (err) {
  if (err) return console.log(err);
}
);
exec("sh runPython.sh", (error, stdout, stderr) => {
  if (error) {
    console.log(`error: ${error.message}`);
    return;
  }
  if (stderr) {
    console.log(`stderr: ${stderr}`);
    return;
  }
  console.log(`stdout: ${stdout}`);
});
});
});

```

### 5.1.2 Algorithm and signals generator

As mentioned at the beginning of this thesis the code of the algorithm is not going to be shown. However, the functionality is what matter the most in this case. The data are now stored in a json file and they can be read and analyzed by the algorithm. Once the data is analyzed the algorithm returns three possible outcomes, a buy signal, which means that it is time to acquire the ADA, a sell signal which means that it is the moment to sell ADA and finally a “do nothing” signal which means either to keep ADA or to not buy ADA. In a perfectly working BOT the buy signals are given when the price is going to spike, a sell signal is going to be given is the price of ADA is about to drop and a “do nothing” signal is sent when the price is stable.

### 5.1.3 Exchanging in Binance

Now that the algorithm generated the signal, the final part of the program needs to act based on the type of signal received. This can be done by using the Binance APIs which require an API key in order to authenticate the user. In case the signal is a buy then Code 19 is triggered:

#### Code 19 - Binance buy method

```

binance
  .marketBuy(
    "ADAUSDT",
    Math.floor(
      balances.USDT.available /

```

```

        response.data[response.data.length - 1][4]
    )
)

```

Since the BOT is programmed to buy and sell the full amount every time a signal comes, it is important to select the full amount of available FIAT money when buying ADA. This is done inside the `Math.floor` method. In case the sell signal is received the following code is triggered:

#### Code 20 - Binance sell method

```

binance
    .marketSell(
        "ADAUSDT",
        Math.floor((balances.ADA.available * 1000000) / 1000000)
    )
    .then((res) => {
        console.log(res);
        previousAction = "sell";
        priceBought = 0;
        telegramText(
            `BREAKING ${sellMessage.text} \n ***** \n\n
date: ${getDate} \n\n close price: ${fetchResult.close} \n\n
***** \n\n real-price: ${realPrice} \n\n quantity:
${balances.ADA.available}`
        );
    })

```

The principle is the same of the buy, but there is an exception. In order to prevent errors and big loss of money I have added a “break” function, which can be seen as a security exit. In case the price of ADA drops more than 7% then all the ADA are going to be sold. However, this does not stop the BOT, after selling everything if a buy signal is received the BOT will still buy.

In order to instruct the BOT to know what the previous move was and thus understand if now in the portfolio we have either ADA or FIAT a global variable is used, which can be seen in the previous code snippets as: `previousAction`. If the `previousAction` variable is `sell` then it means that only buy signals are going to be listened by the BOT and vice versa.

#### 5.1.4 Telegram notifications

Since the BOT works in automatic on a local machine, I needed a system to know what the status of the BOT is and what are the action taken by it. Therefore, I decided to implement a Telegram



notification channel, every 15 minutes the BOT run and takes a decision, once the decision is taken, other than moving the assets in the portfolio it also send the signal received and a graphical visualization of the coin value.

The visualization is going to be the exact same plot that can be generated using the web application. Indeed, the plot sent through notification uses the same code that the web app utilizes, in fact the plot is generated and stored in a folder by the algorithm itself. In Code 21 it is visible that the plot itself is read from the `"../price-analysis/output/signals_plot.png"`.

### Code 21 – Telegram notification

```
const TelegramBot = require("node-telegram-bot-api");
const { telegramToken } = require("../secrets");
// replace the value below with the Telegram token you receive from @BotFather
const token = telegramToken;

// Create a bot that uses 'polling' to fetch new updates
const bot = new TelegramBot(token, { polling: true });

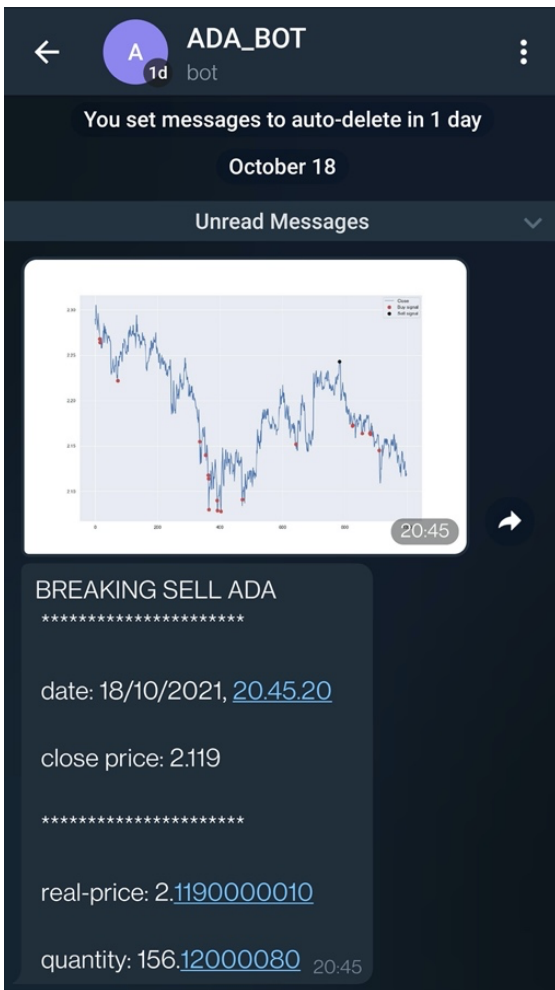
const telegramText = (actionMessage) =>
  bot.sendMessage(HIDDEN, actionMessage);

const telegramPic = () =>
  bot.sendPhoto(HIDDEN, "../price-analysis/output/signals_plot.png");

module.exports = { telegramText, telegramPic };
```

Code 21 is the method utilized in the main file to send the notification every 15 minutes to the Telegram channel. In Figure 17 it is represented a notification when the price of ADA was dropping more than 7% and thus the safety selling happened. The Plot in the message is the one generated by the algorithm with the buy and sell signals. It is noticeable that in this specific case the plot did not have a sell signals. In fact, no black dots are present. However, since the price was starting to fall sharply the BOT identified the drop and decided to sell. This is an additional feature added to the algorithm, which prevents big losses of money. In the second part of the message a resume of the current situation of the coin is reported, showing the real-price and the quantity of coins exchanged in Binance.

Figure 17 - Telegram message



As you may notice the notification includes the plot and the information about the quantity of sold coins as well as the price for single coin. This is very useful since in this case I was able to login in my Binance Portfolio and check the overall crypto market situation

## 5.2 Running the BOT

The implementation is now finished, the only thing left is to run the code and let the BOT do its job. Every 15 minutes an action is taken and a notification is sent to Telegram. The technical indicators and value are manually set into the algorithm. However, the web application comes very handy since it can be used to decide what indicators to use for a specific cryptocurrency.

Figure 18 - Running BOT

```
aleza@DESKTOP-A4QUPJV MINGW64 ~/Desktop/CryptoBOT-by-mn/price-fetcher (main)
$ node index.js
wrote file to input folder in price-analysis
stdout: Processing inputs!
Sleeping...

npm
node-telegram-bot-api deprecated In the future, content-type of files you send will default to "application/octet-stream". See https://github.com/yagop/node-telegram-bot-api/blob/master/doc/usage.md#sending-files for more information on how sending files has been improved and on how to disable this deprecation message altogether. telegramSender.js:13:7
sell indicator: null
buy indicator: null
[Object: null prototype] {
  symbol: 'ADAUSD',
  orderId: 2495489781,
  orderListId: -1,
  clientOrderId: 'Rqpo47paBNNs225mqRj3mR',
  transactTime: 1634579123473,
  price: '0.00000000',
  origQty: '156.00000000',
  executedQty: '156.00000000',
  cumulativeQuoteQty: '330.40800000',
  status: 'FILLED',
  timeInForce: 'GTC',
  type: 'MARKET',
  side: 'SELL',
  fills: [
    [Object: null prototype] {
      price: '2.11800000',
```

In Figure 18, it is shown phase one and phase three running, since the algorithm is activated automatically by the Node code in phase one. As shown in the CLI above, phase one processed the inputs and sent the outcome to the algorithm which sends the signal to phase three. In phase three a selling signal is received, so the code uses the Binance API to sell ADA and buy FIAT currency. At this point both phases enter in sleeping mode for 15 minutes after which the same process starts all over again.

## **6 Results**

After few months both the BOT and the web app have been properly working. Thanks to Google Analytics I was able to see that people from all around the world have visited the web application. The marketing of the web app can be improved, but it was not in the scope of this paper. The algorithm has been working properly and has given correct buy and sell signals to both the web application and the BOT.

The BOT itself was able to produce a profit, but a real result must be measured in a longer period of time. The technical indicators should be regulated time to time, but this does not take much effort, since the web app can be used to calibrate new values for the indicators.

Personally, I consider the result of this thesis a success, since I was able to accomplish the development of something new, innovative and everything was self-made, from the idea to the implementation of the tools.

### **6.1 Products analysis**

The web application is unique, the market did not have anything like this existing. We could consider it a blue ocean without competitors. The challenge is the complexity of the tools. Cryptocurrency is already a quite difficult theme and technical analysis is not any easier. This is going to influence the usage of the tool. Many people are not interest in investing in crypto and many prefer other type of market analysis than technical analysis. Therefore, it is going to be challenging to create an environment around the website. Cryptocurrency even if expanding, is still at its early stages. This means that there is a huge pool of people that could flow in the market and get interested in the subject. The future opportunities are great and being first in the market is rather a great advantage.

In case many people would start using this tool, sentimental investing will be erased. Therefore, less people will fall for the trap of buying high and selling low. Thanks to this investor could make their investment more profitable and thus increase the positive perception of the market. In case this would happen, it could create a chain and help the market to grow even wider. At the moment, Google analytics reported that the web app is used visited in average 7 times per week,

which is a very low result, but it needs to be considered that the marketing of the website is minimal and browser such as Google tend to not promote financial websites, thus the easiest way to find the web app is to actually find a link to it. Now, the only marketing tool is Twitter. There is much to improve in the advertisement of the website, but every success has its starting points.

Obviously, this would happen just once or maximum two times at the macro level of years. Because people would realize that the technical indicators they were using are not working anymore and consequently they would stop using them. However, it does not mean that this “avalanche effect” would not happen again. After, few years an alignment of usage of the same indicators could happen again and cause the same market crash.

Concerning the BOT, it has been running during the worse period for cryptocurrencies in 2021, thus during the period between April and July. The results were great, by the end of July the value of the asset invested was even to the one with which the BOT was started. This means that the profit was 0%, but this is not at all a bad result, since Cardano had a drop of over 50% during May and it did not recover until the end of August. Therefore, the BOT and technical indicators used have outperformed the holding strategy.

## **6.2 Tools results and profit**

After launching the web application and starting to use the BOT, the most important thing was to register the profit and identify what are the best strategies that can be used when developing. Personally, I noticed that in average every week any strategy needs to be re-calibrated. Meaning that the price behavior changes and thus the technical indicators and values that worked for a while will eventually stop working.

Assuming that the starting investment is 1000 USDT in the period between May and October 2021 and that the coin bought was ADA. A simple strategy such as the one described in this paper, which utilizes MACD and RSI would give approximately a profit of around 5% for the first days. However, in around a week the profit per day would decrease to an average of around 2%. After two weeks the profit average decreases between 0 and 1 percent, which indicates that a good moment to calibrate again the tool is every time after one week. It is important to clarify that these percentages are averages of many months of testing. It can happen that a strategy that is

working well while creating it, it might stop working the day after. Constant supervision of the BOT results is a must in order to protect the user assets.

The result of the above strategy would bring a profit of around 19% monthly. Therefore, in just one month the profit in average would be of 215 USDT with a final amount of 1215 USDT (when starting with 1000 USDT). However, the outcome of this strategy depends on the time and the selected coin. This means that in one week everything could change and the strategy used in our case would not work anymore. The data used to reach these results comes from the pattern of my personal profits during a period of around 6 months. These data are store in my personal Binance account and the 1000 USDT example does not mirror my profit, this amount is just an example, but the percentages are based on real results.

The above results are not to be considered as given. Every strategy can give different outcomes as well as the same strategy in different moments in time. The volatility of the market is highly difficult to predict, also more and more laws aimed to regulate the market are incoming and thus the instability of the market can be high. Many technical indicators are based on averages and unpredictable movement are not considered in these calculations, thus if there is a sudden drop of the market, surely the profit would be far away from +5%.

A solution to this problem should be to have a deep understanding of many other indicators so that it would be possible to query average based indicators with others, for example based on volume. However, as anticipated in this thesis, I personally do not have much experience in the investing side. Therefore, I could not test more complicated type of strategies.

### **6.3 Comparing BOT and holding strategy**

A holding strategy consists in simply buy one coin and then keep it without selling it. Many investors consider this to be the best strategy. Therefore, I considered important to compare the profit that the BOT made along the 6 months period with the ones that a holder would have made.

At the beginning of May 2021 ADA price was 1.35 USDT and the closing price for the Month of October 2021 was 1.96 USDT. This means that an investor that started with 1000 USDT investment

would have earned 450 USDT and thus a total of +45%. The final amount in USDT would be 1450 USDT.

Starting with the same amount of money, the result of the BOT would be a net profit of 1839 USDT, which means a +184% with a final amount of 2839 USDT. This is an amazing result that shows how simple technical analysis applied in a constant and repetitive manner can outperform the hold strategy by a lot.

It is important to clarify that in this case the market sentiment was very positive within the 6 months of testing. It will be interesting to see the performance of the BOT during a worse period of the market, where all the prices drop and they stay in the same range for a longer period of time.

#### **6.4 Possible negative impact**

An important concept is the mass adoption of the tools. Technical analysis works because it is able to predict the decision of people in the specific market. Therefore, in case more and more people would start utilizing technical analysis they all would receive similar outcomes and thus they would start buy and selling at the same time. This means that the first people that would start using the web app tools could make a very big profit. However, this would not apply to the newcomers. If everyone buys at the same moment this means that the prices will increase, which will result in receiving sell signals. When people will start receiving sell signals, they all will start selling, which will cause the price to decrease drastically. Drastically is the correct word, because when people see their gains disappearing, they start to fear that the market would crash even more, which bring them to sell more and cause a sort of “avalanche effect”.

The issues identified for the web applications are reflected to the BOT as well. Since the BOT mimic human behavior, buy and sell signals would be align and cause the same type of issues reported above.

This does not mean that the mass adoption of these tools would surely cause such problems. The technical indicators provided are plenty and with the possibility to query these indicators together,

the outcome are millions. Therefore, I personally consider unlikely that the mass adaption would bring people to develop the same strategies.

## **6.5 A better future impact for cryptocurrencies**

My point of view sees the mass adoption of these tools to bring more awareness in the investing field for everyone that approaches the cryptocurrency world. More and more people will trust the indicators and not be afraid of a sudden drop. Nevertheless, other tools will be created and more websites will be available on the market, where people could create any sort of investing strategy. This would differentiate so much the investing mentality that the “avalanche effect” would be impossible.

In the future, I see the cryptocurrency investing getting closer and closer to the stock market. Where people invest based on the expectation of the company itself and its outcome, more than simply watching the micro-trend of the price. If I am bullish towards Tesla stocks in the next years, it is based on the performance that such a company is having. If I expect the stock to increase, it is because I like the way that Tesla is acting in the market and I believe it would continue to follow this trend. The same will apply to cryptocurrencies, more people will start looking more into the behavior of the company behind the project and the coin itself. Discovering developers spending hours in trying to create something new and revolutionary. Something that will change the world and thus something that they would be willing to invest into.



## 7 Summary

The result of this paper is the creation of two tools, which can be very useful for someone that wants to invest safely in cryptocurrencies. The algorithm created gives trustable outcome, showing that a developer with not too deep knowledge in the technical analysis field is still able to create a very well working algorithm that gives trustable outcomes.

Moreover, the web application deployment went smoothly and the web app has been running for quite a long time now without having any downtime. The costs of AWS have been minimal and since the application is not yet used by many people it did not need any scaling up bringing the monthly cost to around a dollar.

Another achievement reached was the deployment of a working BOT on my local machine, which successfully exchanges cryptocurrencies in my Binance account.

This paper also demonstrates the power that we, developers, have. It does not matter if we do not have a deep knowledge of the subject, frameworks and libraries solve the knowledge gap for us. All we need is consistency and perseverance in creating something new, innovative and useful.

A final word goes to the cryptocurrency world, which I personally believe is still in an early stage compared to what it will be in the future. More blockchain applications will be created and new unique ideas will be invented, the blockchain is here to stay and year by year more people will start understanding the potential of such an incredible technology.

## References

Amazon Web Services Inc. (2021a). Amazon Cloudfront Developer Guide. Amazon Web Services Inc., from [https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AmazonCloudFront\\_DeveloperGuide.pdf#Introduction](https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AmazonCloudFront_DeveloperGuide.pdf#Introduction)

Amazon Web Services Inc. (2021b). Amazon Route 53 Developer Guide. Amazon Web Services Inc., from <https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/route53-dg.pdf#Welcome>

Amazon Web Services Inc. (2021c). Amazon API Gateway Developer Guide. Amazon Web Services Inc., from <https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-developer-guide.pdf#welcome>

Brown G., & Whittle, R. (2020). *Algorithms, Blockchain & Cryptocurrency*. Emerald Publishing.

CoinMarketCap, (2021). *Top Cryptocurrency Spot Exchanges*. Retrieved September 24, 2021, from <https://coinmarketcap.com/rankings/exchanges/>

Frankenfield, J. (2021a). *Consensus Mechanism (Cryptocurrency)*. Retrieved September 24, 2021, from <https://www.investopedia.com/terms/c/consensus-mechanism-cryptocurrency.asp>

Frankenfield, J. (2021b). *Proof of Work (PoW)*. Retrieved September 24, 2021, from <https://www.investopedia.com/terms/p/proof-work.asp>

Frankenfield, J. (2021c). *Proof of Stake (PoS)*. Retrieved September 24, 2021, from <https://www.investopedia.com/terms/p/proof-stake-pos.asp>

Hayes, A. (2020). *Candlestick Definition*. Retrieved October 2, 2021, from <https://www.investopedia.com/terms/c/candlestick.asp>

Prabhakaran K. & Uchit V. (2014). *AWS Development Essentials*. Packt Publishing.

Maverick, J. B. (2021). *How do the MACD and RSI indicators differ?*. Retrieved September 26, 2021, from <https://www.investopedia.com/ask/answers/122214/what-are-main-differences-between-moving-average-convergence-divergence-macd-relative-strength-index.asp>

Lim, M. A. (2016). *The Handbook of Technical Analysis*. Wiley.

Nalawade, A. (2021). *Bitcoin Has Already Won, Soon The Price Will Reflect That*. Retrieved September 24, 2021, from <https://www.nasdaq.com/articles/bitcoin-has-already-won-soon-the-price-will-reflect-that-2021-09-20>

Pring, M. (2014). *Technical Analysis Explained*. Mc Graw Hill Education.

**Annex 1: Material management plan**

During the development of this paper many backups of this same document have been created in my private OneDrive account. Every time a major change or addition to the thesis happened a new version has been backed up in both format, PDF and docx. In case a minor change was introduced the version of the thesis did not change and the previous version was simply overwritten.

**Development project:**

Concerning the development Git has been used for source control. The code has been stored in Github in a private repository. Whenever a big feature needed implementation, I did not code directly in master, a separate Git branch was created and only when the full feature implementation was done I finally merged it to master.

After the completion of this thesis all the code will remain in Github and in the Cloud. Concerning the thesis, all the older versions are going to be destroyed.

**Research work:**

All the research work used in the thesis has been done either by using books or open-source documentation. All the references have been reported. There is not documentation to be destroyed at exception of the Binance API key which was for a short period of time stored locally in my machine. The secret has already been destroyed and transferred to Secret Manager in AWS.