UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA BIOMÉDICA

TRABAJO FIN DE GRADO

DESIGN AND DEVELOPMENT OF A CONVERSATIONAL-DRIVEN SYSTEM FOR UNMASKING HEALTH FAKE NEWS USING DATA MINING TECHINQUES

INÉS EL KHLOUFI LARRÁN

JUNIO 2022

TRABAJO DE FIN DE GRADO

Título:	Diseño y Desarrollo de un Sistema Conversacional para Desenmascarar Noticias Sanitarias Falsas usando Técnicas de Minería de Datos
Título (inglés):	Design and Development of a Conversational-driven System for Unmasking Health Fake News using Data Mining Techinques
Autor:	Inés El Khloufi Larrán
Tutor:	Óscar Araque Iborra
Departamento:	Departamento de Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL CALIFICADOR

Presidente:	—
Vocal:	—
Secretario:	—
Suplente:	—

FECHA DE LECTURA:

CALIFICACIÓN:

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE
INGENIEROS DE TELECOMUNICACIÓN

Departamento de Ingeniería de Sistemas Telemáticos
Grupo de Sistemas Inteligentes

TRABAJO FIN DE GRADO

DESIGN AND DEVELOPMENT OF A CONVERSATIONAL-DRIVEN SYSTEM FOR UNMASKING HEALTH FAKE NEWS USING DATA MINING TECHINQUES

Inés El Khloufi Larrán

Junio 2022

Resumen

Hoy en día, avalanchas de desinformación han obstruido y amenazado el flujo de operaciones tanto en los países como en las empresas. Un ejemplo claro es la pandemia de COVID-19, que lleva varios años arrojando toneladas de información poco fiable; desde curas y remedios caseros falsos, o que cuestionan la eficacia de las vacunas, hasta teorías conspirativas sobre el origen de la infección. Todo esto, unido a la actual situación de guerra en Europa del Este, crea un caldo de cultivo especialmente fértil para quienes utilizan la cultura del miedo y la desinformación en su beneficio político y económico. El creciente número de medios oficiales y no oficiales de transmisión de noticias hace que la circulación de este tipo de información esté al alcance de cualquiera. Además, el gran volumen de información dificulta la detección de bulos, noticias falsas, propaganda política contaminada y desinformación en todos los ámbitos. El objetivo de este proyecto es ofrecer un entorno fiable en el que los usuarios puedan encontrar un amplio abanico de fuentes para contrastar las noticias de actualidad y que les sea útil en la toma de decisiones basadas en la actualidad.

Para lograr este objetivo, se recopilará periódicamente una gran cantidad de noticias procedentes de diferentes fuentes, clasificadas, entre otras cosas, por su temática y origen. Además, también se obtendrán "Claims" de la API de Google Fact Checking, es decir, noticias previamente estudiadas por especialistas y clasificadas según su veracidad o falta de ella. El objetivo principal de utilizar estas dos fuentes de datos es dar al usuario más confianza a la hora de interpretar las noticias, en lugar de limitarse a ofrecer información de diferentes fuentes oficiales. Los datos obtenidos serán preprocesados y clasificados de forma que la navegación y el filtrado sean lo más intuitivos posible para el lector y todo ello se combinará con un bot conversacional que facilitará la interacción entre el usuario y el conjunto de datos para obtener información útil.

La arquitectura de este proyecto se basa en microservicios y software orientado a datos, como ElasticSearch. Esta arquitectura incluye un servicio de mesajería que hace las veces de interfaz de usuario, un motor de búsqueda que contiene los datos extraídos a través del web scraping y un servicio web transparente para el usuario, que actúa de intercomunicador entre los dos anteriores, además tambien se proporciona un dashboard de Kibana para la visualización de datos estadísticos a cerca de las tendencias encontradas. Para construir

el scraper se utilizan diferentes librerías de Python que facilitan la tarea de obtener el conjunto de datos. Por último, Google Dialogflow, y HTML como tecnologías utilizadas para el despliegue del bot.

Palabras clave: Fake News, Hoax, ChatBot, Telegram, Google Dialogflow, Webhook, Fulfillmet, Elasticsearch, Kibana, News API, Google Fact Cheking API

Abstract

Nowadays, floods of misinformation have clogged and threatened the flow of operations in both countries and companies. A clear example is the COVID-19 pandemic, which for several years has been spewing out tons of unreliable information, from bogus cures and home remedies, to questioning the efficacy of vaccines, to conspiracy theories about the origin of the infection. All this, coupled with the current war situation in Eastern Europe, creates a particularly fertile breeding ground for those who use the culture of fear and misinformation for their political and economic gain. The growing number of official and unofficial media outlets makes the circulation of this type of information accessible to anyone. Moreover, the sheer volume of information makes it difficult to detect hoaxes, fake news, contaminated political propaganda and disinformation in all areas. The aim of this project is to offer a reliable environment in which users can find a wide range of sources to contrast current news and which is useful for them to make decisions based on current events.

To achieve this goal, a large number of news items from different sources will be collected on a regular basis, classified, among other things, by topic and origin. In addition, "Claims" will also be obtained from the Google Fact Checking API, i.e. news items previously studied by specialists and classified according to their veracity or lack thereof. The main objective of using these two data sources is to give the user more confidence when interpreting the news, instead of simply offering information from different official sources. The data obtained will be preprocessed and classified in a way that makes navigation and filtering as intuitive as possible for the reader, and all this will be combined with a conversational bot that will facilitate interaction between the user and the dataset to obtain useful information.

The architecture of this project is based on microservices and data-driven software, such as ElasticSearch. It includes a messaging service that acts as a user interface, a search engine that contains the data extracted through web scraping and a transparent web service for the user, which acts as an intercommunicator between the two, and which also provides a Kibana dashboard for the visualisation of statistical data on the trends found. To build the scraper, different Python libraries are used to facilitate the task of obtaining the dataset. Finally, Google Dialogflow, and HTML as technologies used to deploy the bot.

Keywords: Fake News, Hoax, ChatBot, Telegram, Google Dialogflow, Webhook, Fulfillmet, Elasticsearch, Kibana, News API, Google Fact Cheking API

Agradecimientos

En primer lugar y por encima de todo, quiero agradecer a mis padres el enorme apoyo y paciencia que han demostrado en estos años, nunca han dejado de creer en mí y de animarme a ser siempre mejor. Si hoy estoy escribiendo estas líneas es gracias a ellos.

A Elías, que aunque él no lo sepa me ha enseñado a tener una actitud mucho más positiva, a creer en mí y en que *las cosas siempre van a salir bien* si de verdad quieres que así sea. Gracias, hermano.

A Natalia, Lucía, Laura y Ana, por ser las mejores amigas que podía encontrar aquí y acompañarme en este maravilloso tiempo, siempre vais a estar conmigo.

A María José Melcón, que sin su mano amiga en el momento más necesario, tampoco estaría aquí. Gracias por escucharme y dedicarme tiempo.

Por suspuesto, a Óscar Araque, por estar siempre disponible y haber hecho de este proyecto no solo un Trabajo de Fin de Grado, sino una experiencia enriquecedora. Gracias por tus consejos y apoyo.

A mis compañeros de laboratorio, que más de dos y de tres dramas han escuchado, por ayudarme en todo lo que han tenido en su mano y sobre todo, por no hacer el tfg desde casa ;).

Por último, agradecer al GSI por contar conmigo para colaborar con ellos, por haberme acogido tan bien estos meses y poner a mi disposición toda la ayuda disponible.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Context

About 6,000 years ago, the first human civilizations flourished between the Tifris and Euphrates rivers, and with them the first conceptions of medicine. From then until today, medical practices have progressed and moved towards scientific empiricism, although other conceptions of medicine have continued to exist.

Over the years, there have been moments in which medicine has been strengthened and have made great advances over the knowledge that existed previously; the discovery of the microscope or penicillin are examples of this. There is also no doubt that the arrival of new technologies, and the intersection of these with biology, have brought great advances in modern medicine. The clearest example is that, thanks to technological progress —specially in computer engineering —, in 2003 the *Genome Project*, considered the greatest advance in the history of medicine, was completed.

This progress has also left a trail of numerous explanatory and in some cases almost conspiratorial theories about the origin or cure of diseases. With the birth of medicine, what is now known as *pseudoscience* or *alternative medicine* was also born, and whether due to the ignorance of times past or to current illicit interests, the medical malpractice

that takes place and the intrusion of false information into the universe available on the internet — an indisputable source of medical information that precedes and sometimes replaces professional consultations — are a reality that is difficult to combat.

The problem of medical misinformation mentioned above has always existed, but it is in recent years, with the COVID-19 health crisis, that it has worsened and factors such as economic or business interests have come together to launch false or partially distorted news: from *influencers* advising drugs, usually antibiotics, to combat acne or other disorders, to staunch *denialists* and their various derivations, which have re-emerged with the pandemic.

The aim of this project is to provide a safe environment for its users, where they can easily find reliable health news on different topics and provide fake news that has been verified by professionals, a system that automatically captures, stores, analyses and compares a large number of news items on different topics. It is configurable, so for this use case it focuses on the health environment, but it is easily adaptable to any environment.

1.2 Project goals

The objectives of this project are the following:

1. Identify particularly controversial use cases that are misrepresented or misreported in the media and collect data from different specialised and verified news sources, both true and false.

2. Pre-process data for its indexation according to future requirements.

3. Implementation and deployment of the conversational bot that will play the role of the main user interface.

4. Provide a display of statistics and individualised access to each new.

1.3 Structure of this document

In this section we provide a brief overview of the chapters included in this document. The structure is as follows:

Chapter 1 puts the work in context for the reader and presents the objectives of it. It gives a brief overview of how the information will be handled in the project.

Chapter 2 provides information on the different technologies that have been used to carry out this project: there is information on what they are used for, where they come from and, briefly, how they are used.

Chapter 3 deals with how the data was obtained for this work, which filters they had to pass through, and how it is divided. It also shows the distribution of these data in space and time.

Chapter 4 It details the use cases for the application and shows the interface with which the end user will interact. Provides a complete view of the UX/UI.

Chapter 5 Discusses the conclusions drawn from this project, the goals achieved, the problems faced and their respective solutions, and suggestions for future work.

CHAPTER 2

Enabling Technologies

This chapter discusses the different technologies selected to carry out each part of the project, as well as the possibilities considered before decisions were made.

2.1 Data collection

The main line of data collection is web-scraping, modification and subsequent indexing in *Elasticsearch*. For this purpose, different reliable sources of information have been selected, which provide verified news that have been classified as false or true.

2.1.1 APIs Newsfeeds

Two different APIs will be used, depending on the source of the data:

- **NewsAPI**: This is an HTTP REST API for searching and retrieving live articles from across the web [22]. For a given query, it returns an HTTP response in JSON format containing all available information about the articles in which that keyword appears. Various parameters such as date, language and more can be selected to filter the results.

The selected option for this case, was to use a Python library provided by the *NewsAPI* developers that facilitated the task of collecting data without having to make HTTP requests directly.

The open source version of this API is limited to 100 articles per request and 50 requests per day. The latter is not a problem for the normal operation of the system, but the fact that it only returns 100 articles is. In contrast to this problem is the possibility offered by this source to download them sorted by relevance.

- **Google Fact Check**: Tool designed for content verification. It is a search engine for verified and mainly, disproven content [15]. In addition to having their own platform to perform searches, they also provide two APIs, one for professionals to add their checks, and one for clients to make use of their dataset through queries: [16]:

  1. The Google FactCheck ClaimReview Read/Write API
  2. The Google FactCheck Claim Search API

  The second one is used in this project in order to collect all verified fake news all over the web. To access Google's data, HTTP requests will be done [17].

2.1.2 Data Storage

Once all the articles from both sources have been collected, it is time to modify their structure so that it is unique and meets the requirements we will need for it for indexation. The technology used to store the data obtained will be Elasticsearch and the format of the documents will be JSON.

- **Elasticsearch**: Lucene-based search server through which we can aggregate and subsequently search and analyse large amounts of data in real time in a distributed manner. Configuration and data access can be done through its RESTful API. It also has several client libraries (Java, Ruby, PHP, .NET, Python, etc.)[10].

  For what this project requires, we will need to do text searches and this technology allows us to do Full-Text searches that will return relevant results and not just exact searches.

  In addition to the above, another reason for choosing Elastic is the existence of another of their products - Kibana - for interactive visualisation [11].

2.1.3 Data Access

An API created to handle the bot's requests to the database. To do this, an application will be created using Fask technology that will analyse the information provided by the user and convert it into a query to the database containing the collected news.

- **Flask:** It is a "micro" Framework written in Python and developed to simplify and make easier the creation of Web Applications under the MVC pattern, this is a way of working that allows to differentiate and separate what is the view (HTML page), the data model (the data that the App will have), and the controller (where the requests of the web app are managed) [13].

2.2 Data Visualisation

One of the functionalities offered by the proposed system is the visualisation of statistics on the data collected. For this purpose, and based on the choice of Elasticsearch as the search engine and information storage base, Kibana has been chosen as the interface for the visualisation of this information.

Kibana is a frontend application on top of Elastic Stack that allows visual analysis and data search of one or several Elasticsearch indexes [12].

We can have dashboards with interactive visualisations of our data bringing together graphs, maps and filters to show the full picture of the data and even allowing users to go directly from a Kibana dashboarcd to any URL or web application.

In addition, and as a further point, it offers the option to share the dashboard via link, PDF, PNG or CSV, which makes it a very useful option for later integration into the messaging service offered by our system [12].

2.3 User Interface

The user interface (UI) is the means by which the user can communicate with a machine, computer or device, and comprises all points of contact between the user and the equipment. It is intended to be usable, as well as friendly and intuitive to the user's eyes, and also to ensure that any user, regardless of their level of adherence to the technology, is able to interact successfully with it, obtaining the expected results.

2.3.1 Chatbot technologies

For this project, it is thought that the best way to create a UI that fulfils the required functionalities and also guarantees an easy adaptation to other areas is a conversational bot -also called chatbot-. In the world of chatbots we can find different alternatives; they can be programmed from scratch, but there are also multiple ways to develop advanced ones with a very low complexity.

The options studied for this purpose are set out below:

- ***Azure Bot Service*** [20]:
  - Available for Windows, Mac and Linux.
  - AI Technology: Natural Language Processing, voice and high-end vision thanks to Azure Cognitive Services.
  - Compatible - making some code changings - with telephony and Microsoft Teams.
  - Connection to voice response (IVR), web and mobile applications.
- ***Amazon Lex*** [4]:
  - Uses machine learing algorithms to learn how users express their intentions.
  - Able to create complete linguistic models from a few sentences.
  - Allows voice or text interactions
  - Available in several languages
  - Compatible with Facebook Messenger, Slack, Twilo and SMS.
  - Integration with databases or visualisation tools is limited to AWS.
- ***IBM Watson Assistant*** [19]:
  - Runneable in any Cloud.
  - Use of machine learning algorithms to learn how to solve the questions it could not solve in the first instance.
  - Available in 13 languages
  - Creates just Andoid bots
  - Runnable in IBM cloud or in any others like Amazon, Google, Microsoft and local hosts.
  - Deployable on any website, mobile applications and customer service tools.

- ***Google Dialogflow*** [1]:
  - Bot creation through a very simple graphic interface.
  - Bot development option using Node.js language.
  - Supports Google's machine learning technologies.
  - Integrable with: Google Assistant, Facebook messenger, Slack, Line, Kik, Skype, Cisco, Telegram, Cisco tropo, Twilop, Twitter and Viber.
  - Allows requests to be sent to "*Webhooks*" so that the response is much more natural when retrieving information. This option enables easy integration with external databases.

2.3.2 Used technologies

After a detailed study of the functionalities offered by each technology, it was concluded that Google Dialogflow and Telegram meet the needs of the project.

- **Dialogflow**

The technology used for the chatbot is Google Dialogflow, a natural language understanding platform used to design and integrate a conversational user interface into mobile apps, web applications, devices, bots, interactive voice response systems and related uses [14]. Components explained below [6]:

1. **Intents:** Defined to specify the user's intentions. To create a useful bot, we need to set as many intents as possible situations. To detect the user's intentions, there are actions and parameters, which extract useful information from the message received and relate it to the responses predefined by the developer.

2. **Entities:** Each intent has an entity type, which determines exactly how data is extracted from a user expression, they are made to identify groups of words to make easier the way to understand the intents. An entity is configured with a meaning, this is called *entity type* and synonyms are added that will allow it to be detected, these are called *entity entry*.

   Dialogflow contains multiple system entities inside its structure. A few examples are locations, dates, times, etc. These entities are already provided and users has to do nothing with them, they are automatically detected.

3. **Context:** As defined, a context is a set of circumstances that, without which there is no way to understand a situation or a conversation properly. Thanks to them, it is possible to understand the flow of the conversation and in Dialogflow, are defined for each intent and it is essential to define the input and output contexts.

   The Fig. 2.1 shows an example using the context of a banking agent.

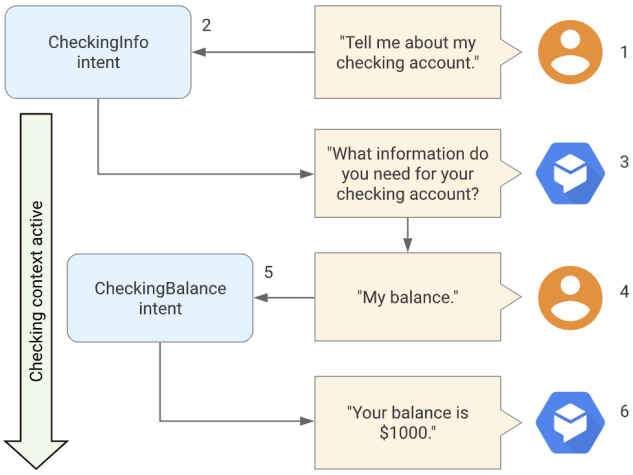

Figure 2.1: Dialogflow Contexts Flow (Banking Agent Example)

4. **Fulfillment:** Fulfillment is code that is deployed as a webhook request that allows your Dialogflow agent to call business logic on an intent-by-intent basis. During a conversation, Fulfillment allows you to use the information extracted by Dialogflow's natural language processing to generate dynamic responses or trigger actions on your back-end [8].

This platform makes it possible to design a bot of medium to high complexity in a fast and agile way. The programming code remains invisible to the user who uses it and provides results that can be easily integrated with different messaging systems or with your own application. In addition, it also provides access to databases through the webhook service [18], which allows us to address the whole purpose of the project.

In the case of this system, a bot has been designed with conversational capabilities oriented to different use cases and the platform chosen for its integration is Telegram.

- **Telegram**

Telegram is a messaging platform focused on instant messaging , sending multiple files and mass communication. The service offers functionalities focused on chats between users

with the option of attaching files of all kinds, whether documents, multimedia or graphic animations, with an upload limit of 2 GB each.

For the automation of massive tasks, bots are developed, which carry out extra activities and services such as payments, games, group moderation or the assignment of other tasks under artificial intelligence. Bots are also used in the business and social sphere [26].

The reason for choosing this rather than another messaging platform as the user interface for the service is as follows:

1. It is the messaging platform that offers the greatest ease of use when it comes to establishing a conversation with a bot — It has a pre-designed bot called "Bot Father" [25] that guides users in the task of creating a new interface —.

2. Dialogflow automatically configures the bot in telegram.

3. In addition to the aforementioned integrations, Telegram is a platform that is widely used by many users and can be deployed on both mobile devices and computers, making it an adaptable option for different end-user needs and situations.

2.4 Docker

Docker is a software platform that allows you to quickly build, test and deploy applications. Docker packages software into standardised units called containers that include everything needed for the software to run, including libraries, system tools, code and runtime. With Docker, you can rapidly deploy and scale applications in any environment with the certainty of knowing that your code will run [5].

Similar to how a virtual machine virtualises (eliminates the need to directly manage) server hardware, containers virtualise a server's operating system. Docker is installed on each server and provides simple commands you can use to create, start or stop containers.

The main reasons for using Docker containers in this project are the following:

- Containerised applications make it easier to deploy, identify problems and return to an earlier stage to fix them.

- Dockerised applications can be seamlessly transferred from local development teams to production deployments.

- Docker containers make it easier to run more code on each server, improving usability and saving resources.

CHAPTER 3

Architecture

In this chapter, we cover the design phase of this project, as well as implementation details involving its architecture. Firstly, we present an overview of the project, divided into several modules. This is intended to offer the reader a general view of this project architecture. After that, we present each module separately and in much more depth.

3.1 General Overview

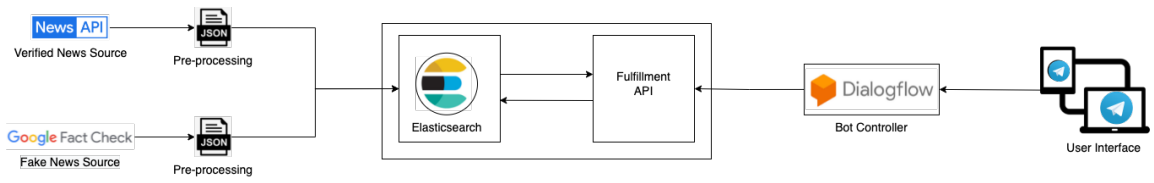

Figure 3.1: General Architecture

The general architecture of the system is shown inFig. 3.1, It is composed of five main modules described in detail below.

Web Scraping and Data Pre-processing: Module that obtains the data from the sources detailed in 2.1.1, then modifies them to unify its schema.

Data Storage: Module that ingests data extracted from the web and stores them in a single index.

Data Access API Service: Webhook service application that receives calls from Dialogflow with the necessary information to perform queries to the database. It is responsible for performing these queries and returning the result.

Chatbot: Structure of the bot created for the project and its workflow specifications. How it integrates with Telegram.

Data Visualisation: Architecture and configuration of the dashboards created for the visualisation of statistics on the information collected.

3.2 Web Scraper and Pre-processor

This module is responsible for collecting articles from all over the web by making web requests to the aforementioned APIs in 2.1.1. Let's take a closer look at the process:

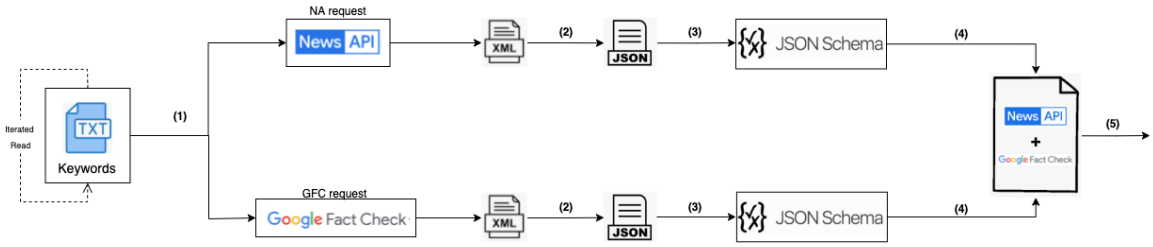

Figure 3.2: Web Scraper and Pre-processor

The points in the Fig. 3.2 are detailed below:

1. A web request is made to *News API* and another to the *Google Fact Check API* for each of the keywords collected in a text file on which we must iterate.

2. For each keyword a response object is returned whose format must be transformed to JSON.

3. From the answers obtained, another JSON document is created, modifying the fields, eliminating those that are not useful and adding those that will be necessary. At this point it is important to take into account the nature of both documents according to their origin: the *Fact Checking Articles* will have a valuation field that the others do

not have, and those from reliable sources will have a relevance factor. These differences will be managed so that the schema is the same and responds to the needs of both.

4. Once we have the articles from both sources in the same format for all keywords, they are merged into a single document. A field has been created to indicate the API they come from.

5. Having all the information contained in a single document, it is sent to the data storage module.

We have used *Crontab* [3] to schedule the daily data collection, it is a simple text file that stores a list of commands to be executed at a time specified by the user. It will check the date and time at which the script or command should be executed, the execution permissions and it will do it in the background. In this case, it has been specified to run at 8.00 a.m. every day.

3.3 Data Storage

As mentioned many times, the storage will take place in Elasticsearch. An index has been created, called *"all_claims"*, in which all the news items that are collected periodically are stored.

The schema we discussed in 3.1 is as follows:

- Author
- Title
- Header
- Keyword
- URL
- ID: *hashed URL*
- Publication Date
- Download Date
- Source API

- Textual Rating: *Verified or Rated as false*
- Factor of Relevance: *Useful only for articles from News API*


```
"all_claims": [
 {
     "author": "The New York Times",
     "title": "Boston Marathon Live Updates: Race Returns to Patriots
         \u2019 Day",
     "description": "The marathon is finally back on the third Monday
          in April. The coronavirus forced organizers to cancel the
         race in 2020 and hold it in October last year.",
     "url": "https://www.nytimes.com/live/2022/04/18/sports/boston-
         marathon",
     "publishedAt": "2022-04-18T11:59:45Z",
     "content": "Runners from Russia and Belarus will not be allowed
         to compete in the Boston Marathon on Monday, another example
          of the countries deepening isolation over the invasion of
         Ukraine.Like so many around \u2026 [+2035 chars]",
     "keyword": "coronavirus",
     "API": "NA",
     "name": "New York Times",
     "textualRating": "True",
     "language": "en",
     "id": "07092824c5a86d177c2700b0049e3960",
     "relevanceFactor": 0,
     "downloadDate": "2022-04-20"
 }
```


Figure 3.3: Elasticsearch Indexed Document

Every time the scraper code is run, i.e. once a day, the index is updated, so that we have a record of the most relevant news, as well as the most striking fake posts over time.

The Elasticsearch created to store the collected posts is hosted in a docker container. *Elasticsearch:7.12.1* has been used as the image, thanks to which a single-node container has been built, listening on ports 9200 and 9300.

In the structure of the document it can be seen that there is a field called *Relevance-Factor*, it is important to highlight that it has been generated especially for articles coming from *News API*, as this platform offers its documents ordered by relevance, if so indicated when making the request, but there was no predefined field to show it. Thus, the lower the relevance factor, the higher the relevance of the news item.

3.4 Data Access API Service

This section deals with the Elasticsearch data access module, i.e. how Dialogflow makes requests to the database to obtain the information the user has demanded.

As shown in Fig. 3.4, when the Dialogflow logic detects that the user wants to receive some news, it makes an HTTP POST request to the API that has been configured for it. Its role is to act as an intermediary between the conversational agent service and Elasticsearch.

Once the HTTP POST request is received, the fulfilment API collects the necessary data to formulate a query that will be sent to Elasticsearch via an HTTP GET request. The response from this is formatted in the API and will be sent as HTML text that Dialogflow will display directly in the end-user's chat.

A Flask-based application supports the API, whose URL is provided to Google's Fulfillment module (Fig. 3.5).

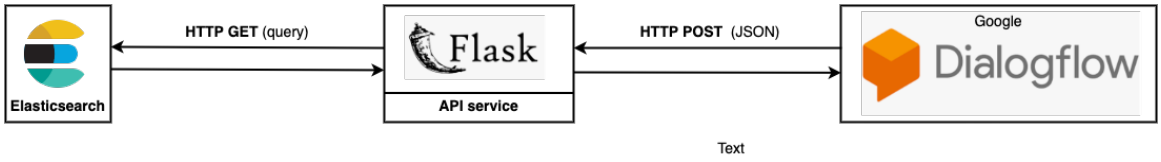

Figure 3.4: Webhook Structure API service

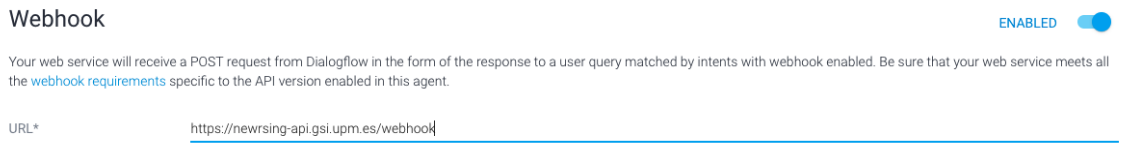

Figure 3.5: Fulfillment-Webhook Dialogflow's Module

This application is also built in a docker container. In this case port 5000 has been enabled and, due to Dialogflow's requirement to only allow access to an https service, the address `https://newrsing.gsi.upm.es/webhook` has been enabled for it.

3.5 Chatbot

Detailed here the structure created for the bot, as well as its flow. For all of them, button-type responses have been established, as for the purpose of the bot it is considered more usable and less error-prone.

3.5.1 Intents Structure

All the intents created, as well as their functionality and responses, are detailed below. Fig. 3.6 shows an overview of the connection and flow between them.

- **Welcome Intent:** This intent is triggered every time the bot receives a greeting message, e.g. *Hello*, *Hi*, *Hey*, *What's up*, etc. On detecting any of these or other terms, it will send this introductory message:

  " Welcome to Newrsing:

  You are in contact with the Newrsing Health News Assistant. Two options are offered:

  - *If you want to see trends in fake news, indicate that you want to see statistics.*
  - *If you want to see specific news on a specific topic, check the list of available themes and ask for the one you are most interested in.*

  We hope you find it useful,

  Best regards."

  On receiving this message, two alternative responses are offered:

  - Statistics
  - Available Themes

- **Available Themes:** If *Available Themes* selected in the previous step, then a list of generalist topics available would be displayed. Then the user is aimed to select one of those.

  - COVID-19 Pandemic
  - Mental Health

- **Statistics Theme Selector:** In the same way as in the previous section, the user must choose the subject on which statistics are to be displayed.

  - COVID-19 Pandemic
  - Mental Health

- **Mental Health and COVID-19 Topics:** Once the user has selected the topic they want to know more about, a list of keywords contained in the chosen category will be displayed. A button selector offered again, this one for the last time.

  1. Coronavirus

2. Mask
3. Vaccine
4. Infection

- **News Dispenser:** This intent recognises the keyword about which the user has requested to know more and thanks to the compliance technology detailed in section 3.4, a query is made to the database and the news item is returned classified as true or false.
- **Statistics Dispenser:** This intent recognises the topic on which the user has requested to view statistics and returns a link to the Kibana dashboard where the statistics are located.

3.5.2 Entities Structure

Three entities have been created for the automatic recognition of the variables that are handled for each of the use cases, so we distinguish the following three:

- **News Use Cases:** For each use case, i.e. COVID-19 Pandemic and Mental Health, two entity types have been created.
- **Keywords:** Type of entity created to detect each keyword and store it as an input object for the webhook request.
- **Statistics Use Cases:** Same as the News Use Cases, for each use case, i.e. COVID-19 Pandemic and Mental Health, two entity types have been created.

3.5.3 Telegram

Telegram has been selected as the instant messaging system that will support the bot. Dialoglow offers a great ease of integration with telegram, which is detailed below:

1. First, download the Telegram mobile or desktop application and then search for the *Bot Father* chat.
2. When interacting with it, a message with instructions for use is sent as shown in Fig. 3.7.
3. We indicate that a new bot is created with the command */newbot*

4. After indicating the name the bot will receive, using the */token* command, an HTTP API access token will be provided.

5. The token provided will be inserted into the Telegram integration section of Dialogflow and the bot will be up, running and accessible to any user.

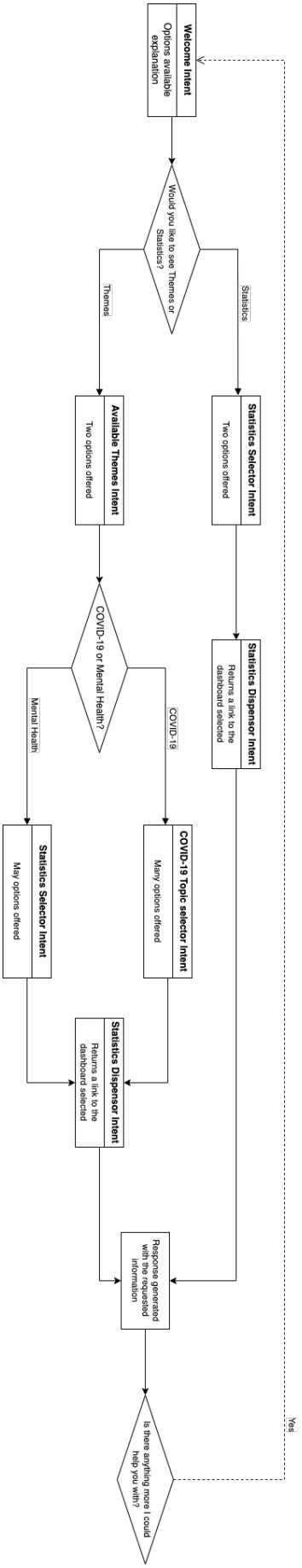

Figure 3.6: Chatbot Intents Structure

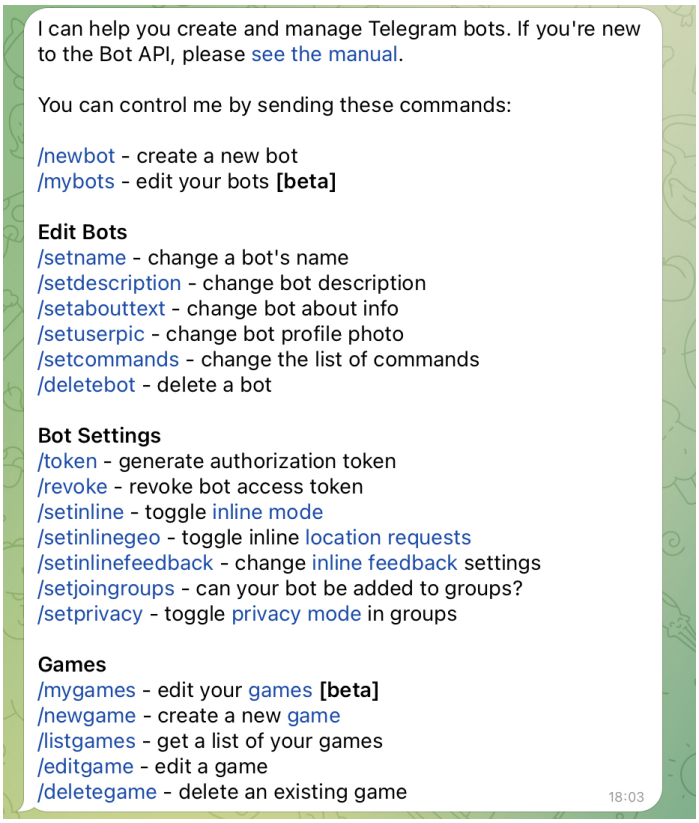

Figure 3.7: Bot Father Instructions Message

3.6 Data Visualisation

In order to provide the user with an overview of trends among the news collected, two dashboards have been created, one for each use case — which we will detail in section 4 —. The Elastic Stack tool, Kibana, has been used for this purpose.

For each dashboard, a separate dual view is provided:

- Left side for the review of fake news, showing a summary of the keywords that have appeared the most and the sources that publish the most fake news.
- Right side follows the same structure, but on verified news.

All the visualisations can be filtered, so if we want to see the most reliable sources but do not want to see those that come from CNN News, for example, we can remove them from the visualisation. In addition, we can also access the individualised news published by each of these sources. The same will happen with the keywords.

As with Elasticsearch, Kibana, being part of the same feature set, must also be dockerised. For this, port 5601 has been enabled and the *Kibana 7.12.1* image has been used to match the version of Elasticsearch used above.

CHAPTER 4

Case study

4.1 Introduction

In this chapter we will describe the basic use of the chatbot, to cover the main features, understand its functionalities and how to use it. This information will be completed with annotations, for each module, on the use cases that are currently configured.

4.2 Motivation

NewRsing, as we have already mentioned, is an easily scalable and configurable system, as well as being accessible by any user, from any condition, with the only requirement of having Telegram downloaded on the device they wish to use.

For this project, NewRsing is focusing on the area of health and, within this, on the following sub-themes:

- **Mental Health:** Mental health refers to cognitive, behavioral, and emotional well-being. It is all about how people think, feel, and behave. People sometimes use the term "mental health" to mean the absence of a mental disorder [2].

The Cambridge dictionary defines stigma as *"a strong feeling of disapproval that most people in a society have about something, especially when this social judgement is unfair"* [9]. Nowadays, mental health problems, from the patient's point of view, are lived in silence, generally provoked by fear, shame or incomprehension. This is the social stigma that surrounds them and has been latent since time immemorial.

The media is one of the main sources of information for the population; in this sense, its potential to promote mental health by improving the levels of education, awareness and knowledge of the population is evident [24]. Therefore, it was felt that the application of the principles of this project to the topic of mental health would be valuable, as a good publication on this topic can be very positive, while malicious ones can cause particular harm.

- **COVID-19 Pandemic:** COVID-19 is the disease caused by the new coronavirus known as SARS-CoV-2. World Health Organisation (WHO) first became aware of this new virus on 31 December 2019, when it was informed of a cluster of "viral pneumonia" cases reported in Wuhan, Republic of China.

  On 11 March 2020, the disease was declared pandemic by the WHO, and on 9 November 2020, the US pharmaceutical company Pfizer announced that the vaccine candidate it was preparing in collaboration with the German biotech company BioNTech was 90% effective. On 19 May 2022, 60% of the world's population has at least the first dose of the vaccine [23], and finally, as of today, the WHO has not yet declared the end of the COVID-19 pandemic.

  Since the beginning of the pandemic, there have been many and varied sources of disinformation, as well as fake news, hoaxes and unscientific news that have permeated global society, leading them to believe in miracle cures or sowing panic. The aim of including COVID-19 Pandemic among the use cases of this application is to help combat the infodemics generated around this topic.

4.3 General Overview

Before explaining how it works, Fig. 4.1 shows the general behaviour of the bot, previously seen in chapter 3.5.1, but this time from a more instructive point of view for the end-user.

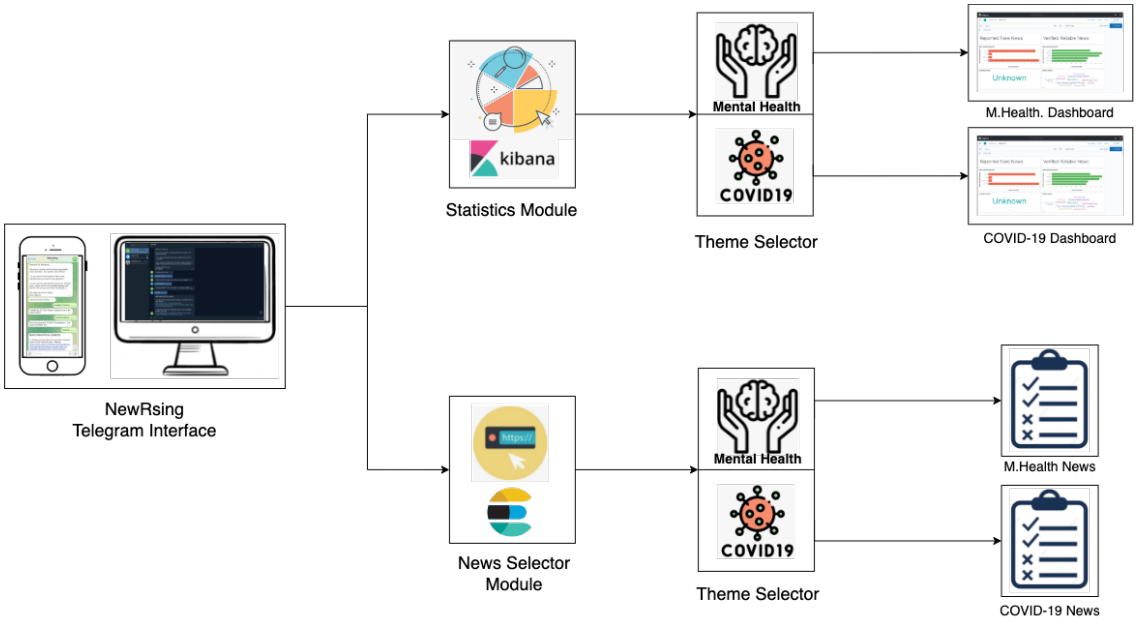

Figure 4.1: NewRsing Utilities

4.4 ChatBot usage

Having seen the general scheme of use, let's explain in more detail how NewRsing works.

4.4.1 Welcome Message and Module Selector

In Fig. 4.2 we can see the welcome message, which indicates to the user the available possibilities and a button selector to facilitate the interaction between the user and the bot.

4.4.2 Theme Selector Module

The users will select this module in case they want to display a list of true news and news that have turned out to be false.

1. First, there is a choice between mental health or the COVID-19 pandemic (Fig. 4.3).
2. Then, once the preferred option has been selected, a list of keywords on which news has been collected will appear(Fig. 4.4).
3. Finally, a list of news items is sent as shown in, separating true and false ones (Fig. 4.5).

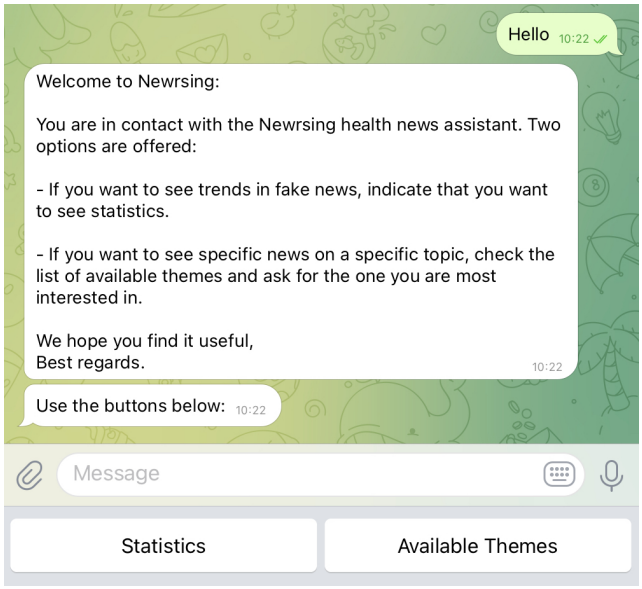

Figure 4.2: NewRsing Welcome Message, Module Selector

4.4.3 Statistics Module

Users will select this module in case they want to see statistics on the veracity or dishonesty of sources, news or terms.

1. Once the statistics module has been selected, a choice is given between the two cases already discussed: mental health and the Covid-19 pandemic (Fig. 4.6).
2. And after choosing one of the two, a link is provided to the Kibana dashboard where the graphs generated from the data obtained can be found (Fig. 4.7).
3. Finally, by following the link provided, you will be able to view a Kibana dashboard like the one in the Fig. 4.8.

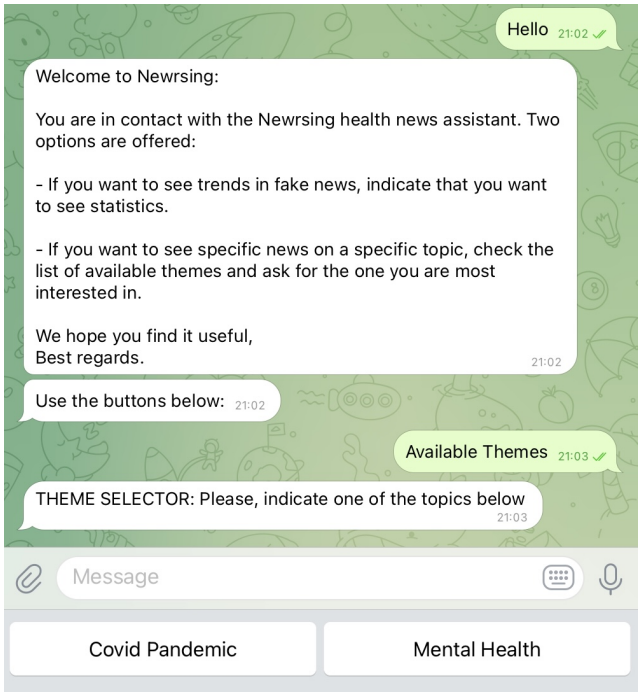

Figure 4.3: News Theme Selector

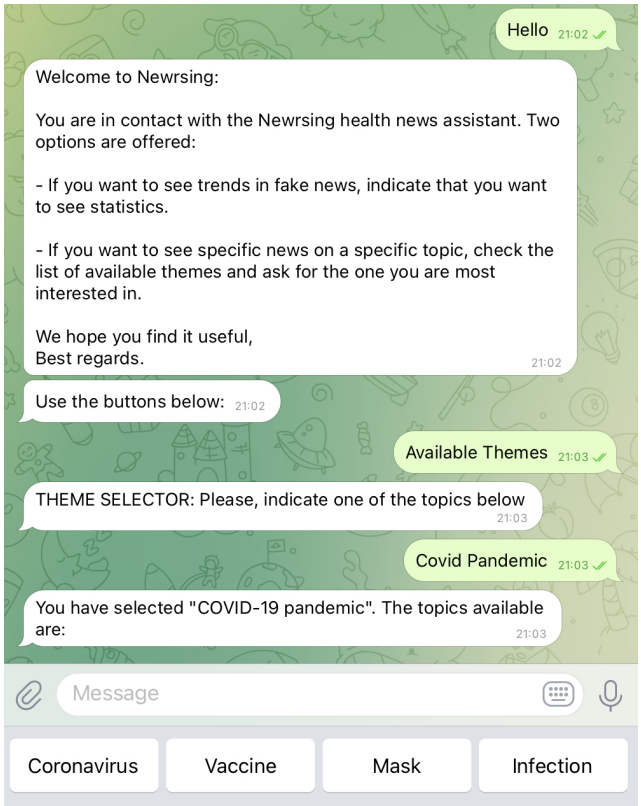

Figure 4.4: COVID-19 Pandemic Keyword Selector

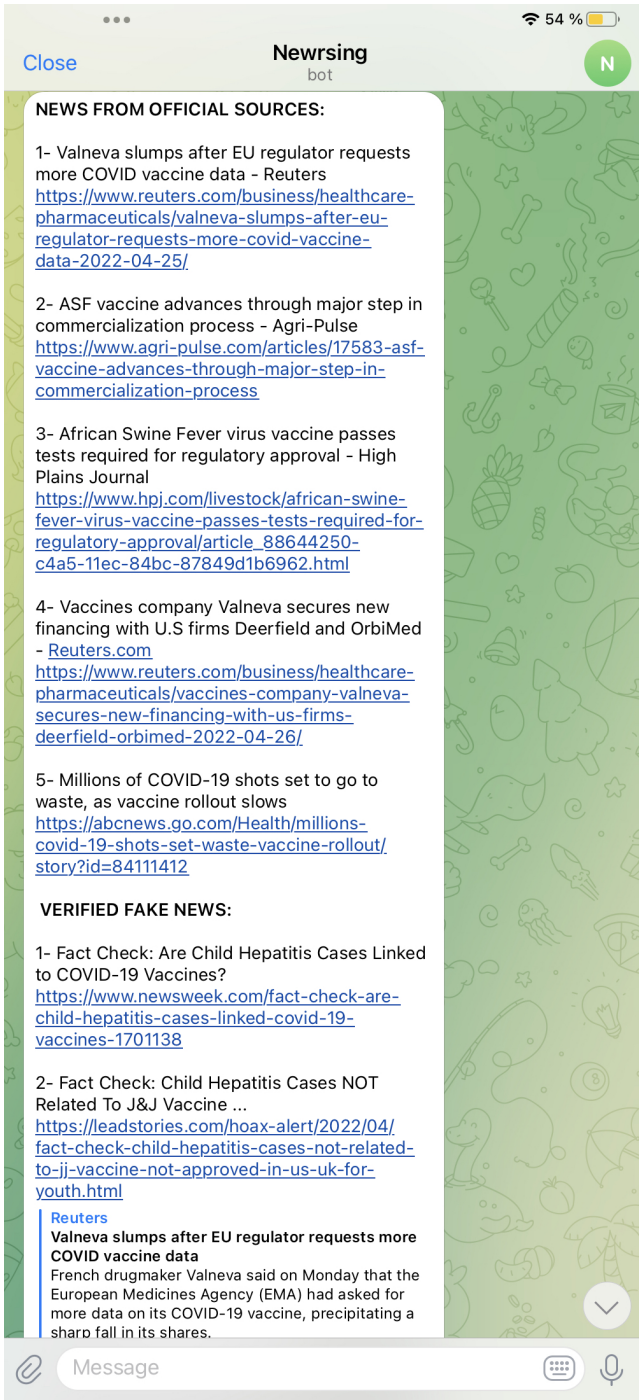

Figure 4.5: COVID-19 Pandemic News List

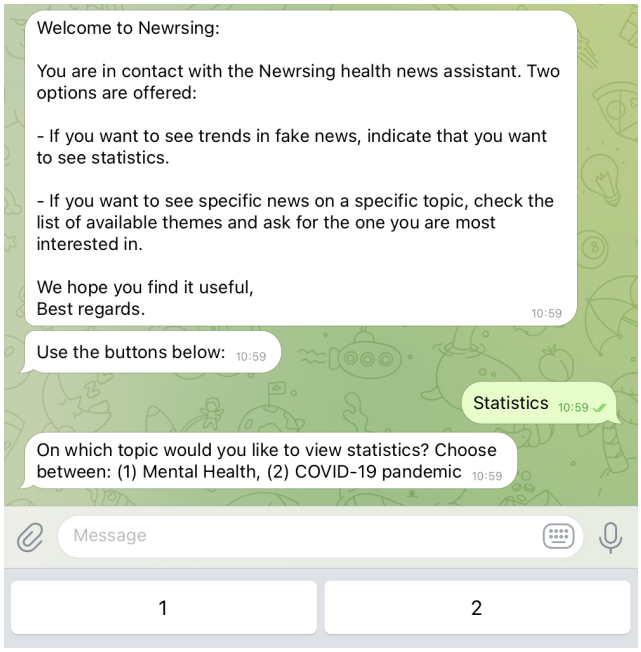

Figure 4.6: Statistics Theme Selector

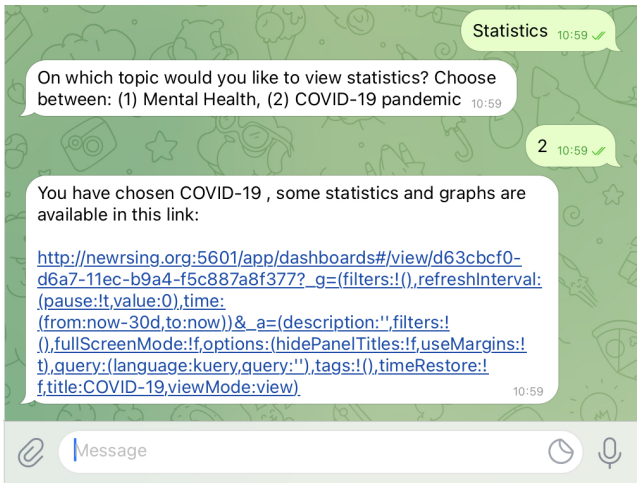

Figure 4.7: Link Provided to Kibana

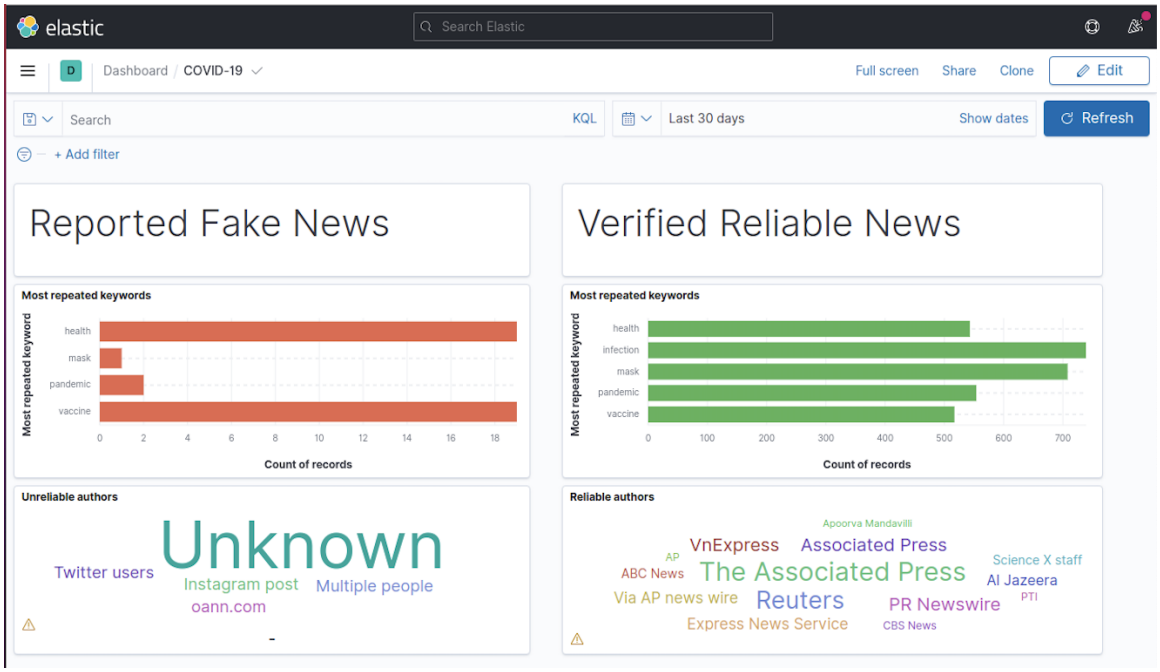

Figure 4.8: COVID-19 Pandemic Kibana Dashboard

CHAPTER 5

Conclusions and future work

This chapter describes the conclusions drawn from this project and reflections on future work. It also discusses the difficulties encountered and the strengths of the project.

5.1 Conclusions

The result of this project is *NewRsing*, a ChatBot for Telegram useful for all types of users in the detection of fake news or hoaxes that may be harmful to health. But *NewRsing* does not stop there, it also offers filtered news from reliable sources and an additional platform to visualise different statistics on the information obtained over time.

In retrospect, the main objective was to provide the user with a reliable environment to use whenever they want, but especially when they need information on particularly controversial topics.

The motivation for focusing this work on the health field was the recent wave of misinformation generated by the Coronavirus pandemic that started in 2020; numerous cases were seen of people putting their health at risk by following false home remedies or even being scammed due to the lack of control this generated. Numerous social networks such as

Facebook and Instagram have reached agreements with the WHO to commit to facilitating access to official and authorised information, and have already seen detectors of this type of comments, both related to COVID and mental health — blocking those that could be harmful such as those inciting suicide [21] or non-vaccination, among others —.

As can be seen with a quick glance at the web, this type of filtering that is practised in some social networks and information sources, although useful, is not enough, since despite its existence, numerous hoaxes continue to be leaked - even if they are considered reliable enough to be published in official media - so we can say that, although means are put in place to combat disinformation, they are insufficient.

Taking all of the above into account, NewRsing has been created, a platform that aims to facilitate the task of searching for information, offering the possibility not only of being able to check which have been the latest publications classified as false, but also giving greater credibility to the true ones and offering an environment of visualisation of trends over time, which helps to acquire greater and better criteria for the selection of sources and/or authors.

5.2 Achieved goals

In this section we will summarise the strengths of the developed system.

It is a system capable of obtaining, ordering and standardising information, as well as filtering in order to offer it to the user in a simple and easily accessible way.

Data capture and processing is automated, with a robust infrastructure that ensures daily updates, giving the user greater confidence to consider *NewRsing* as a source of information.

The modular architecture of the system makes it easily configurable for new use cases without any thematic limitation and in a few steps.

The user is provided with an extra functionality that allows to display trends on topics and sources that publish fake news, useful for learning how to make decisions based on these factors.

The only requirement for users to use NewRsing is to install Telegram on their mobile device, computer or tablet.

It is easy to use, guided, free and accessible to all. No prior knowledge of the subject or technology are required.

At this point, it is time to look back and check whether the objectives set at the beginning of the project have been achieved:

1. *"Identify particularly controversial use cases that are misrepresented or misreported in the media and collect data from different specialised and verified news sources, both true and false."*

   Two use cases have been selected that are particularly contentious at present, as we have seen in the previous sections, namely the COVID-19 pandemic and mental health.

2. *"Pre-process data for its indexation according to future requirements."*

   The documents offered by the different APIs have been formatted to fit the needs of the project, i.e. relevance factor, download date and standardisation to follow the same scheme.

3. *"Implementation and deployment of the conversational bot that will play the role of the main user interface."*

   A Chatbot has been created to act as a user interface that is supported by Telegram and accessible by any user with an electronic device.

4. *"Provide a display of statistics and individualised access to each new."*

   A Kibana dashboard is provided as a means of visualising the trends and statistics obtained from the data collected.

5.3 Problems Faced

There are some points that have complicated the development of the project. They are detailed in this section, as well as their solutions:

- The first problem found is that, when it comes to compiling news, there are many news items that can be generated in the same day and not all of them have the same importance or the same relevance for the user who accesses them. It is a complicated problem to solve in order to be able to serve a selection of news items to the user that are relevant in a personalised way.

The solution that was found was that of one of the news API utilities, which offers a relevance sorting parameter when making the request.

- One of the news sources used, News API, had a limit of 200 news items per request and 50 requests per day in its free version. It might seem that an easy solution would be to make several requests that collected all the news, but they are listed by relevance, so it was impossible to predict the order of the news from one request to another. This inconvenience slowed down the development phase of the scraping and preprocessing module, because once this limit was reached, you had to wait until the next day.

  The solution to this problem is to handle this exception in the application code. In case the limit exception appears, we will be left with the 200 news items, but in case the licence has been paid, no modification will be necessary and all available news items will be collected.

- For Google Fact Checking API posts, it was found that all of them included a field with a verdict on their veracity or lack thereof. This form of classification was not standardised, i.e. in some of them a simple *false* appeared and in others it could be true, false or a whole range of intermediate connotations, which could tip the balance more to one side than the other when read by a person, but which makes the task of automatic classification very difficult.

  Looking at the dataset obtained from this source, it became apparent that in general, the vast majority of news items were false, and those that were not, were partially false or with some exceptional connotation, but false nonetheless. That is why it was decided to consider that all the news coming from this source - remember that it is a platform whose task is to disprove false news per se - was false.

- The last problem we faced was how to deliver the service. Initially, the idea was to create a website with a noticeboard and a bot to help the user filter the news. We thought about using Kibana to create this noticeboard, but finally the idea was discarded because this technology is only useful for data aggregation and in this case the data needed to be segregated. In addition, we were not aware of previous integrations between Kibana and a Dialogflow chatbot.

  The solution to this problem is clear in the development of the system. The main interface is finally Telegram and Kibana has been used as the statistics distributor.

5.4 Future work

In this section a brief outline of possible improvements to the project will be given.

- The most feasible improvement at this stage would be to broaden the range of possible health topics covered by NewRsing. Some of the options proposed are the most searched diseases on Google:
  - STD (*Sexually Transmitted Diseases*), such as HIV (*Human Immunodeficiency Virus*) or HPV(*Human Papiloma Virus*), which have always been a taboo subject and about which people often turn to the Internet for information instead of going to the doctor.
  - Diabetes, which is one of the most widespread chronic diseases, with 463 million people affected by the disease in 2019.
  - The Common Cold is an illness caused by numerous viruses that affects the vast majority of the population every year. It is widespread and does not usually cause serious symptoms, so information on the Internet about home remedies to cure it is a potential danger.
- And extend the applications of the service beyond the health sector to other sectors that are also relevant and of interest. Some of the proposals are:
  - The stock market, as a leak of hoaxes on this issue can be a risk factor for many companies.
  - War in Ukraine, as it is a topical issue and it is known that a lot of fake news is circulating, endangering the relationship between countries, the economy and the mental health and peace of mind of citizens.
- It would be interesting to add an Alarms section, which would be activated every time the system detects a hoax or fake news that goes viral, or a potential danger to the user.
- Developing a system to detect the percentage of truthfulness of a news item based, among other things, on
  - Its source
  - Its author
  - Its format
  - Its content

that is trained with data collected from sources that are known to be reliable and that are engaged in fact-checking.

- In line with what was mentioned in the previous section on the classification of fake news, a possible improvement to offer greater reliability to the user would be the creation of a classification standard. In this way, we would no longer have categorically false articles, but would open the door to intermediate connotations.

  One possible way of doing this would be the creation of a dictionary of the terminology seen in the ratings collected, which would analyse the text string in this field and assign new ratings such as:

  - True
  - Partially true
  - Misleading
  - False
  - Under review

- Finally, and taking a general look at the system architecture, it can be seen that there is a bottleneck at the level of the flask server (fulfilment API). At any given time, the volume of requests may be higher than the server can handle. For this reason, and as a possible improvement of the service, it would be to move it to a cloud server that can provide the resources it needs.

APPENDIX A

Impact of this project

This appendix reflects, quantitatively or qualitatively, on the possible social and economic impact, as well as the ethical implications.

A.1 Social impact

This section will discuss the social impact that this project could have.

This project aims to give people resources to fight a problem that has always existed, but is now the order of the day.

Misinformation and misinformation have caused headaches for ordinary citizens, entrepreneurs, banks, the health system and virtually all sectors. And it is in recent years that it has increased more and more with the establishment of social networks and the possibility of making any information from any source viral.

NewRsing is an agent that helps the user to detect and acquire more knowledge in this field. It is distributed as a conversational agent on Telegram and is available to anyone who needs to make use of its services.

A.2 Economic impact

This section deals with the possible economic impacts that may arise from this work. The presence and standardisation of the use of this platform could have a significant economic impact on the national health system, which could reduce the investment in debunking hoaxes. In addition, it would also have a significant impact on companies and commercial sectors, which would see markets more protected from the influence of this type of news.

A.3 Ethical implications

In this section we will evaluate the ethical problems that our project could involve.

The biggest impact NewRsing could have is an ethical one, for what is less ethical than spreading false information for one's own benefit? And even more so if this practice could end up putting the health of the population at risk, since, as we well know and as has been demonstrated in the COVID-19 cisis, numerous alternative medicine remedies have been proposed and carried out by people who, after having taken them, have finally been hospitalised.

APPENDIX B

Economic budget

This appendix details an adequate economic budget to bring about the project. We will expose the physical and human resources, the licenses, and the taxes.

B.1 Physical resources

This section details the estimated budget that has been needed for the project in relation to hardware.

The project was carried out with a computer with the following characteristics:

- **CPU:** Intel Core i5 CPU 3.20GHz
- **RAM:** 8 GB
- **DISK:** 500GB
- **Operative System:** Ubuntu 20.04.3 LTS

The approximate cost of the equipment is 500,00€

B.2 Human resources

This section will be similar to the previous one, this time dealing with budgets for people.

To estimate this budget, we assume that everything is carried out by a single person. The time spent must be the equivalent of 12 ECTs, so multiplying it by the estimated hours to which one of them is equivalent, 30 hours, we obtain a total of 360 hours.

We consider a part-time shift of 4 hours and that in a month there are an average of 21 working days. We also estimate a salary of about 450€/month for a scholarship in the GSI group. Therefore, we calculate a total salary of about 2000,00€.

B.3 Licenses

All the costs for licenses used for this project will be covered here.

All the technologies used to develop this project are Open Source, so we conclude that the licensing costs for this project amount to a total of 0,00€.

B.4 Conclusion

The final economic budget for the project amounts to 2500,00€ and the duration is 360 hours.

Bibliography

[1] "Dialogflow Features". https://digitalherramienta.com/dialogflow/, 2021. Accessed: 1 nov 2021.

[2] Adam Felmann, Timothy J. Legg. "What is Mental Health?". https://www.medicalnewstoday.com/articles/154543, 2020. Accessed: 21 May 2022.

[3] Alejandro (a.k.a KZKGGaara). "Cron and Crontab, explicados". https://blog.desdelinux.net/cron-crontab-explicados/?utm_source=destacado-inside, 2013. Accessed: 23 may 2022.

[4] AWS. "Amazon Lex Features".https://docs.aws.amazon.com/lexv2/latest/dg/what-is.html, 2021. Accessed: 1 nov 2021.

[5] AWS. "What is Docker?". https://aws.amazon.com/es/docker/, 2022. Accessed: 20 May 2022.

[6] L. Caballero. Development of a Movile Nutritional Assistant for Supermarkets with the Technology Dialogflow. Master's thesis, ETSIT-UPM, 2019.

[7] Codelabs Developers. "How Webhook Service Conects With Dialogflow". https://cloud.google.com/dialogflow/es/docs/fulfillment-overview?hl=es-419, 2022. Accessed: 16 Apr 2022.

[8] Codelabs Developers. "Understand fulfillment by integrating Dialogflow". https://codelabs.developers.google.com/codelabs/chatbots-dialogflow-fulfillment#1, 2022. Accessed: 3 dic 2021.

[9] Paulette Delgado. "Rompiendo el Estigma de la Salud Mental". https://observatorio.tec.mx/edu-news/rompiendo-el-estigma-de-la-salud-mental, 2021. Accessed: 21 May 2022.

[10] Elastic NV. Elasticsearch definition. https://www.elastic.co/es/elasticsearch/, 2022. Accessed: 14 Apr 2022.

[11] Elastic NV. "Elasticsearch Documentation". https://www.davincigroup.es/introduccion-a-elasticsearch-que-es-casos-de-uso-instalar/, 2022. Accessed: 15 may 2022.

[12] Elastic NV. "Kibana Documentation". https://www.elastic.co/es/kibana/, 2022. Accessed: 15 may 2022.

[13] Epitech. "What is Flsak and How does it works". https://www.epitech-it.es/flask-python/, 2022. Accessed: 19 May 2022.

[14] Google Inc. "dialogflow definition". https://en.wikipedia.org/wiki/Dialogflow, 2022. Accessed: 14 Apr 2022.

[15] Google Inc. "Google Fact Check Documentation", https://newsinitiative.withgoogle.com/es-es/resources/journalism/verification/lessons/google-fact-check-tools/#:~:text=La%20API%20Google's%20Fast%20Check,ClaimReview%20en%20su%20propio%20CMS, 2022. Accessed: 14 Apr 2022.

[16] Google Inc. "Google FactCheck ClaimReview Read-Write-Search API". https://developers.google.com/fact-check/tools/api#the-google-factcheck-claim-search-api, 2022. Accessed: 14 Apr 2022.

[17] Google Inc. "Google FactCheck REST" https://developers.google.com/fact-check/tools/api/reference/rest/v1alpha1/claims, 2022. Accessed: 14 Apr 2022.

[18] Google Inc. "webhook service, google dialogflow". https://cloud.google.com/dialogflow/es/docs/fulfillment-webhook?hl=es-419, 2022. Accessed: 14 Apr 2022.

[19] IBM Inc. " IBM Watson Assistant Features". https://cloud.ibm.com/docs/solution-tutorials?topic=solution-tutorials-android-watson-chatbot&locale=es, 2021. Accessed: 1 nov 2021.

[20] Microsoft Azure Inc. "Azure Bot Services Features". https://azure.microsoft.com/en-us/services/bot-services, 2021. Accessed: 1 nov 2021.

[21] Adam Mosseri. "Un paso importante para proteger a nuestra comunidad en Europa". https://about.instagram.com/es-la/blog/announcements/an-important-step-towards-better-protecting-our-community-in-europe, 2022. Accessed: 21 May 2022.

[22] News API. "News API Documentation". https://newsapi.org/docs, 2022. Accessed: 14 Apr 2022.

[23] Datos RTVE. "La vacuna contra la COVID-19 en el mundo: El 66% de la población tiene al menos una dosis". https://www.rtve.es/noticias/20220519/vacuna-coronavirus-mundo/2073422.shtml, 2022. Accessed: 21 May 2022.

[24] Saludemia. "Salud mental y Estigma y medios de comunicación". https://www.saludemia.com/-/vida-saludable-salud-mental-y-medios-comunicacion, 2022. Accessed: 21 May 2022.

[25] Telegram FZ-LLC. "bot father telegram bot documentation". https://telegram.me/botfather, 2022. Accessed: 14 Apr 2022.

[26] Telegram FZ-LLC. "telegram definition". https://es.wikipedia.org/wiki/Telegram, 2022. Accessed: 14 Apr 2022.