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 ANAMNESIS APPLICATION FOR STRUCTURED PATIENT SYMPTOM RECORDING AND ANALYSIS

DAVID BRAU QUERALT
ENERO 2024

TRABAJO DE FIN DE GRADO

Título:	Diseño y Desarrollo de una Aplicación de Anamnesis Conversacional para el Registro y Análisis Estructurado de Síntomas en Pacientes
Título (inglés):	Design and Development of a Conversational Anamnesis Application for Structured Patient Symptom Recording and Analysis
Autor:	David Brau Queralt
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 ANAMNESIS APPLICATION FOR STRUCTURED PATIENT SYMPTOM RECORDING AND ANALYSIS

David Brau Queralt

Enero 2024

Resumen

La anamnesis médica es un proceso crucial en el cual un profesional de la salud recopila información detallada sobre la situación clínica de un paciente y su evolución. A pesar de su importancia, varias circunstancias comunes empeoran significativamente la capacidad para obtener y transmitir con precisión esta información. En primer lugar, existe un tiempo de consulta limitado, a menudo vinculado al gasto per cápita en salud, que a su vez depende en gran medida de la riqueza de un país, lo que resulta en que la mitad de la población mundial tenga 5 minutos o menos por consulta de atención primaria. En segundo lugar, durante estas breves consultas, los pacientes pueden tener dificultades para transmitir su situación debido a barreras idiomáticas, diferencias culturales y sesgos de memoria, especialmente cuando las consultas son temporalmente distantes. Además, hay una discrepancia significativa entre la información que los pacientes transmiten y lo que los médicos documentan. En tercer lugar, a pesar del uso de registros médicos electrónicos y el consumo de tiempo que su uso puede implicar, el 80% de la información registrada es no estructurada, lo que dificulta los procesos de análisis automatizado.

Este proyecto tiene como objetivo abordar estos problemas mediante el desarrollo de un sistema web automatizado de anamnesis conversacional. Este sistema podría mitigar muchos de los desafíos descritos, al proporcionar acceso instantáneo a un tiempo de anamnesis ilimitado, garantizar la usabilidad multilingüe e intercultural a través de la configuración personalizada y utilizar un sistema de registro completamente estructurado.

Para alcanzar este objetivo, se ha diseñado y desarrollado un sistema de anamnesis modular y paramétrico, basado en la web, con registro de usuarios, capaz de manejar tanto la entrada como la salida por voz y texto para gestionar múltiples tipos de anamnesis. Se han modelado dos tipos diferentes de anamnesis, una general y otra psicológica, definiendo los síntomas más relevantes y sus características correspondientes, con el fin de integrarse con el sistema de anamnesis desarrollado y demostrar su versatilidad. Una vez que el sistema ha sido desarrollado, se ha implementado y probado para evaluar la funcionalidad y el rendimiento en la grabación y análisis de datos así como para la generación de informes.

La arquitectura tecnológica de este sistema está diseñada en torno a microservicios Docker e implementada como una aplicación web. Esta arquitectura incluye, en el frontend, una página web minimalista, multilingüe y fácil de usar, construida con HTML, CSS y JavaScript. En el backend, una aplicación web Flask gestiona todo el proceso de anamnesis y genera los análisis e informes correspondientes. Esta aplicación y sus utilidades asociadas están desarrolladas en Python e incorporan multiples bibliotecas. La aplicación web se conecta a una base de datos no relacional, MongoDB, y a un ChatBot conversacional, Google Dialogflow.

El proceso de anamnesis es multiagente, totalmente paramétrico y utiliza la terminología SNOMEDCT. La gestión y actualización de los agentes esta completamente automatizada a través de archivos de configuración, facilitando una adaptación rápida a diversos contextos. Además de una recogida estructurada de los síntomas, el sistema tiene la capacidad de realizar y reportar un análisis de los sentimientos en la descripción de estos.

Palabras clave: Anamnesis Médica Automatizada, Registro Estructurado de Síntomas, Informe Automatizado, Telemedicina, Chatbot Conversacional, Google Dialogflow, Sistema Multiagente, Microservicios Docker, Aplicación Web, Flask, MongoDB, ReportLab, SNOMED CT, Análisis de Sentimientos.

Abstract

Medical anamnesis is a crucial process in which a healthcare professional gathers detailed information about a patient's medical condition and its progression. Despite its importance, several common circumstances significantly worsen the ability to accurately collect and transmit this information. First, there is a limited consultation time, which is frequently associated with per capita health spending and, in turn, with the wealth of a country, resulting in half of the world's population having 5 minutes or less per primary care consultation. Second, during these brief consultations, patients may face difficulties conveying their condition due to language barriers, cultural differences, and memory biases, especially when consultations are temporarily distant. Furthermore, there is a significant discrepancy between the information that patients disclose and what physicians document. Third, despite the use of electronic medical records and the time consumption it may imply, 80% of the recorded information remains unstructured, making it challenging for automated analysis processes.

This project aims to address these issues by developing an automated conversational web anamnesis system. This system will potentially mitigate many of the challenges described by providing instant access to unlimited anamnesis time, ensuring cross-lingual and cross-cultural usability through customization, and using a full structured recording system.

To achieve this goal, a modular and parametric web-based anamnesis system, with user registry, capable of handling both voice and text input and output, has been thoroughly designed and developed to manage multiple anamnesis types. Two different types of anamnesis, a general and a psychological one, have been modeled, defining the most relevant symptoms and their corresponding features to align with the developed anamnesis system and demonstrate its versatility. Once the system was ready, it was deployed and tested to assess functionality and performance for data recording, information reporting, and analysis.

The technological architecture of this system is designed around Docker microservices, and is implemented as a web application. This architecture includes a minimalist, multilingual, user-friendly web page on the frontend, built using HTML, CSS and JavaScript. On the

backend, a Flask web application manages the entire anamnesis process and generates the corresponding analyses and reports. This application and its associated utilities are developed in Python and incorporates several different libraries. The web application link with a non-relational MongoDB database and a Google Dialogflow conversational bot.

The anamnesis process is multi-agent, fully parametric, and employs SNOMEDCT terminology. Agents setup is completely automated through configuration files, enabling quick adjustment to different contexts. In addition to a structured collection of symptoms, the system has the ability to perform and report a sentiment analysis in the description of symptoms.

Keywords: Automated Medical Anamnesis, Structured Symptom Record, Automated Report, Telemedicine, Conversational Chatbot, Google Dialogflow, Multi-Agent System, Docker Microservices, Web Application, Flask, MongoDB, ReportLab, SNOMEDCT, Sentiment Analysis

Agradecimientos

A mi familia, por su fe y apoyo inquebrantables. El amor de mi esposa, Elena, y mis hijos David y Sofía me ha dado la fortaleza necesaria para superar cada desafío.

A mis padres, por su respaldo incondicional y por enseñarme el valor del esfuerzo y la perseverancia. Su dedicación y sacrifico son un ejemplo de vida.

A mi tutor, Óscar Araque, por su confianza, guía y orientación. Aprecio enormemente su generosa disponibilidad.

Contents

Resumen **I**

Abstract **III**

Agradecimientos **V**

Contents **VII**

List of Figures **XI**

1 Introduction **1**

1.1 Context 1

1.2 Project Goals 3

1.3 Structure of This Document 4

2 Enabling Technologies **5**

2.1 Microservices Technologies 5

2.1.1 Microservices Architecture 5

2.1.2 Docker and Docker Compose 6

2.2 Web Application 6

2.2.1 MVC Architecture 6

2.2.2 Flask Server and Server Sessions 7

2.3 User Interface and User Experience 7

2.3.1 Frontend Technologies. HTML, CSS and JavaScript 8

2.3.2 SMTP for Automated Email Sending 8
2.3.3 Accesibility. Multilingual and Audio Interaction 9
2.4 Data Management 10
2.4.1 Data Storage. MongoDB 11
2.4.2 Data Presentation. ReportLab 11
2.5 Chatbot Technologies 11
2.5.1 Chatbot Main Concepts 12
2.5.2 Dialogflow 12
2.6 Interoperability. Standardized Terminology. SNOMEDCT 14

3 Architecture **15**

3.1 Introduction 15
3.2 General Overview 15
3.3 Web Application 17
3.4 Database. MongoDB 18
3.5 Dialogflow. Anamnesis Agents 22
3.6 User Interface 26
3.7 System Onset: Configuration and Deployment 29
3.8 Anamnesis Process 31
3.9 Anamnesis Report 33

4 Case study **35**

4.1 Introduction 35
4.2 Motivation 35
4.3 Use Cases 36
4.4 Aplication Workflow 37
4.5 Use Case Example 40

5 Conclusions and Future Work **45**

5.1 Conclusions 45

5.2 Achieved Goals 46

5.3 Future Work and Challenges 47

Appendix A Impact of This Project **i**

A.1 Social Impact i

A.2 Economic Impact ii

A.3 Ethical Impact ii

Appendix B Economic Budget **iii**

B.1 Hardware Resources iii

B.2 Software Resources iv

B.3 Human Resources iv

B.4 Resources Summary iv

Appendix C Source Code **v**

Bibliography **vii**

List of Figures

1.1 Consultation Length vs. Per Capita Health Spending [34] 2

2.1 Dialogflow Basic Workflow [1] 13

2.2 Dialogflow API Interactions [1] 14

3.1 General System Architecture 16

3.2 Web Applications Architecture 18

3.3 Database Design 21

3.4 Anamnesis Agent “Anam”. Part 1 23

3.5 Anamnesis Agent “Anam”. Part 2 24

3.6 Anamnesis Agent Configuration File Structure 25

3.7 Main User Interface Areas 26

3.8 Language Selection Interface 27

3.9 Anamnesis View Design 27

3.10 Registry Field Validation 28

3.11 Agent Files Generation 29

3.12 System Onset. Configuration and Deployment 30

3.13 Anamnesis Design 32

3.14 Anamnesis Report 34

4.1 Use Cases 37

4.2 User Registration Email 37

4.3 Sequence Diagram Illustrating the General Workflow 38

4.4 First Part of the Anamnesis Workflow Implementation 39
4.5 Second Part of the Anamnesis Workflow Implementation 40
4.6 User Registration 41
4.7 Login 41
4.8 Creation of General Anamnesis 42
4.9 Completion of the Anamnesis 42
4.10 Anamnesis Report Example. Page 1 of 3 43
4.11 Anamnesis Report Example. Page 2 of 3 44
4.12 Anamnesis Report Example. Page 3 of 3 44

CHAPTER 1

Introduction

1.1 Context

When taking care for a patient, all information gathered by any means can play a crucial role in guiding the patient's diagnosis and treatment. In this context, the clinical history, as a set of data that forms the medical biographical narrative of the patient, seems to be a fundamental part. Due to this essential role and the sensitivity of the data it contains, its content, preservation, and access are extensively regulated [6].

The clinical history is typically organized into episodes and includes the anamnesis or patient interview, physical examination, supplementary data and tests, course or progression, and an epicrisis or medical review, which summarizes the clinical assessment of the case. The medical anamnesis (or medical history), is a crucial process in which a health care professional gathers detailed information about a patient's medical history. Some parts of the medical anamnesis cannot be retrieved from the patient's health record and therefore must be obtained in each episode, theses are the details concerning to a new episode, specifically, the history of the current illness and the corresponding systematic anamnesis by organ systems [43].

In recent decades, electronic health records (EHRs) have achieved widespread adoption

due to their significant potential to improve healthcare quality, although occasional barriers like interoperability and functionality issues, especially related to time consumption may arise [32]. The majority of studies have found a positive association between EHRs adoption, both from clinical and economic perspectives [35, 41].Despite its incredible potential to improve anamnesis collection, there are several obstacles towards optimally gathering this crucial information.

Firstly, there is typically limited consultation time, with significant variations between countries worldwide directly correlated with per capita health spending, which in turn is associated with the wealth of the country, although there are some exceptions. Approximately half of the world's population has 5 minutes or less per consultation, which is likely to generate a negative impact on the quality of health care and particularly by reducing the time available for patients to explain their symptoms [34].

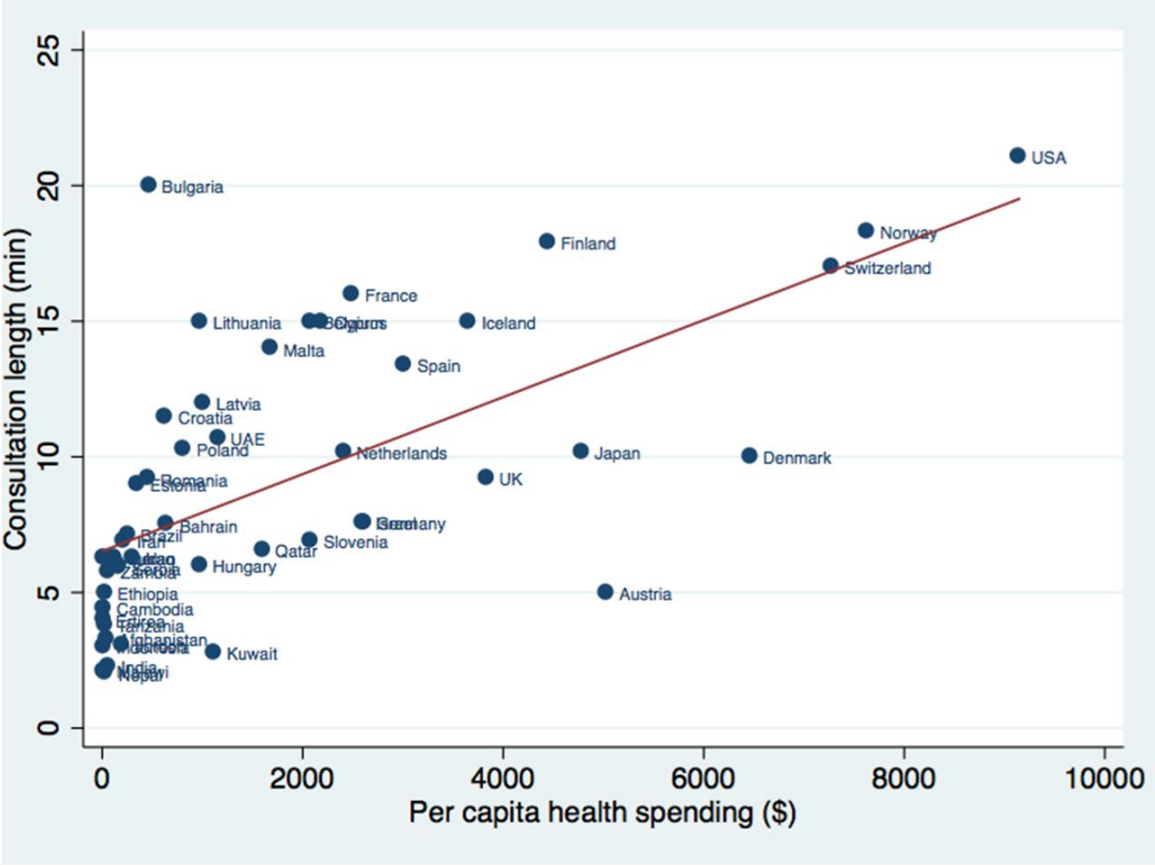


Figure 1.1: Consultation Length vs. Per Capita Health Spending [34]

Additionally, within a medical practice, both the perceived importance of the topic and the perception of limited time influences physicians behavior during visits. Even though in 15-minute consultations, non-main topics may not even receive a full minute of atten-

tion [50]. Moreover, time pressure as the workday progresses and fatigue appears have been observed to decrease patient-physician communication [40].

Secondly, although the results can be improved using certain techniques, there is a significant discrepancy between what patients disclose during consultations and what physicians document in their notes, particularly with regard to psychosocial and lifestyle information, which suggests differences in how patients and physicians define relevance in the context of medical information [52]. In addition, there is a decrease in memory accuracy over time, with a tendency to remember more recent events. This can particularly decrement the quality of the anamnesis in cases where consultations are temporarily distant from specific episodes or in certain patients such as the elderly [46]. But even when episode memory exists, patients recall of medical information is often poor and imprecise [38]. Communication gets even worse in the increasingly globalized world we live in due to language and cultural reasons that require additional training and communication skills from physicians, and that lead to performance and attention patterns that may be inconsistent and, in some cases, suboptimal [54].

Thirdly, despite the use of electronic medical records, 80% of the information remains unstructured data, making it challenging to apply automated analysis processes, particularly Artificial Intelligence techniques [39].

Finally, the use of computer media has been reported to consume between 20% and 50% of the consultation time [29, 48]. With an approximate consumption of 25% allocated to register information and create medical notes in the EHR [44].

Due to all the reasons mentioned above, the development of an efficient and accessible automated anamnesis system has a tremendous potential to improve healthcare attention, especially for the poorest half of the world. Therefore, this project aims to develop an automated open, multilingual and customizable conversational web anamnesis system. This system will be designed to systematically record patient symptoms and perform some analysis on the collected data.

1.2 Project Goals

The objectives pursued by this project are the following:

1. Design the architecture of a web-based anamnesis system capable of handling both voice and text input and output, generating summary reports, and providing basic analytics. Define the required specifications and the technologies to be used by each

of the system's components.

2. Model the basic structure of the anamnesis to use in the chosen conversational agent environment. Define the most relevant symptoms and their corresponding features that should be gathered, ensuring the proper documentation of the medical history.
3. Develop and deploy each of the system components according to their specifications.
4. Test the system to evaluate its functionality.

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*** establishes the context in which this project is framed, outlines the main challenges to be solved and sets out the objectives to be achieved.
- ***Chapter 2*** provides information on the technologies used in the development of this project, detailing their characteristics, purpose, operation and how they are integrated into the project, as well as the functionality they provide.
- ***Chapter 3*** describes the software architecture of the various components of the system, covering both the frontend and backend, along with their integration and functionality. In addition, the flow of information over time through the various components is detailed.
- ***Chapter 4*** showcases different use cases and details the complete patient experience, from the initial interaction with the user interface to obtaining a report, thus achieving a satisfactory experience. This provides a comprehensive view of the functionality and benefits of the system.
- ***Chapter 5*** addresses the key findings of the project and assesses the achievement of the stated objectives. In addition, challenges and possible areas for future work are explored.

CHAPTER 2

Enabling Technologies

2.1 Microservices Technologies

Microservices technologies are a modern software architecture approach that decomposes an application into independent services, and represent a significant evolution over traditional monolithic structures.

2.1.1 Microservices Architecture

Microservices Architecture is comprised of different decoupled and reusable components or services called microservices. They are designed for easy integration into cloud environments and independent deployment. With a specific task or responsibility attached to each component, the whole works together to achieve the proper functioning of the application. The ability to use custom-designed tools and technology components for each individual service, rather than relying on generic components for various functions, along with the inherent scalability of the system, boosts efficiency and flexibility of microservices architecture. However, we cannot overlook the associated challenges, such as the complexity of managing multiple service environments. [21, 25].

2.1.2 Docker and Docker Compose

In the context of microservices deployment and management, Docker becomes a fundamental tool, as it facilitates the packaging of applications along with their dependencies, tools and execution environments into isolated units known as containers. These containers provide isolation from the host runtime environment and allow multiple instances of applications to run concurrently on a single host.

The use of containers ensures that software always runs using the same configuration, avoiding compatibility issues and ensuring consistency. This gives Docker its greatest strengths: reliability and scalability [2]. In addition, Docker Compose complements this functionality by providing a tool that facilitates the construction and deployment of applications composed of multiple containers, defining the configuration of the entire application, including the interconnection between containers, thus simplifying the management of complex applications.

This project makes use of two Docker containers, one for the web application and one for the database, whose deployment and interconnection is addressed using docker compose.

2.2 Web Application

A web application is a software application accessible through a web browser that allows its users to interact with it, performing various functions and accessing different resources available through it.

2.2.1 MVC Architecture

Model-View-Controller (MVC) is a software design pattern that consists of dividing the application into 3 logical levels: Model, View and Controller. The model is responsible for managing the data logic and the integration of the application with the database. The view, on the other hand, manages the user interface. Finally, the controller contains the application logic and manages the requests and responses generated. This division facilitates the development and maintainability of the application by providing modularity and structural clarity, making it easier for developers to control the application architecture. Despite these advantages, the application may lose readability by requiring multiple files to be examined to understand the application processing flow [9]. The application developed in this project extensively applies these design principles.

2.2.2 Flask Server and Server Sessions

Flask is a Python microframework designed for the simple development of web applications under the MVC model. It uses WSGI (Web Server Gateway Interface) standards for communication with web servers and the Jinja2 template engine, used to manage and render HTML templates dynamically. The main advantages of Flask are its simplicity, flexibility and scalability [19]. In addition, it has a complete and practical documentation that greatly facilitates its use [20].

This project makes use of both a Flask web application and the development web server it incorporates to offer the anamnesis web service.

Sessions are a software tool that allows to temporarily store user-specific information throughout several interactions with a web application. This is essential in order to perform complex interactions, implement security measures for accessing resources or guarantee the persistence of specific data. The need for their implementation comes from the fact that the standard web communication protocol between the web client and the web server is HTTP, which is by default a stateless protocol.

In Flask there are two types of sessions depending on where the information is stored. On the one hand we have client-side sessions, which are stored in browser cookies, and on the other hand we have server-side sessions. In the latter case, the client browser only stores a session identifier with cookies, while it is the server that stores the specific data. This last type is implemented in Flask through the extension called "Flask-Sessions" and, although it is a bit more complex to implement, it offers several advantages such as greater data security, greater control of active sessions and more storage space for session data [17].

The system developed in this project implements such sessions on the server side so that each system user only has access to their anamnesis.

2.3 User Interface and User Experience

Both User Interface (UI) and User Experience (UX) are closely related elements and are key to successful software design. UI refers to the visible part of the software, with which users interact, while UX focuses on how users perceive and experience the whole interaction.

Since the UI is the user's first point of contact in a web applitation, it plays a crucial role in defining the final UX.

2.3.1 Frontend Technologies. HTML, CSS and JavaScript

The user interface in software development is commonly referred to as the frontend. This component covers the design, content and interactivity of the elements present in the user interface. In web development, the frontend is primarily built using a combination of three fundamental technologies: HTML, CSS and JavaScript.

The dynamic part is made up of HTML files, or HyperText Markup Language, which are used to represent the structure and content of the web. This section is considered dynamic because of its ability to change its content, through the web application, in response to user interactions. This implies that we can vary the information displayed and the structure of the web according to the user's interaction. As for the static part of the web, we find both CSS and JavaScript. CSS is used to configure the layout and presentation of the various elements defined in the HTML files, ensuring a consistent and attractive appearance in the user interface. JavaScript, on the other hand, is the programming language that provides functionality to the web, allowing decisions to be made and actions to be performed on the client side. Thanks to JavaScript, the user interface can respond dynamically to user actions, significantly improving the user experience [51]. In addition, there are other static files such as images and documents that are also part of the static web content.

The developed web anamnesis system implements those three technologies to achieve a minimalist and coherent frontend across the different interfaces. This coherence is achieved by making use of homogeneous styles and functionalities throughout the website, which aims to facilitate an intuitive use of the system.

2.3.2 SMTP for Automated Email Sending

The Simple Mail Transfer Protocol (SMTP) is a standard used for sending e-mails over the Internet. In the context of Gmail, one of the most popular email providers, it uses this protocol to send emails through its smtp.gmail.com server. Gmail, however, for security reasons, requires the creation of an application password for the use of this service [4]. This protocol can be used in the context of a Python Web Application and with Gmail as mail provider, for the automated sending of emails, by using the python "smtplib" library. This technology is used by the present anamnesis system for the automated delivery of personalized e-mails to users.

2.3.3 Accesibility. Multilingual and Audio Interaction

Accessibility has a key role to play in software development, especially in the medical field, where users often have a disability or are not in a good state of health. There are multiple aspects that can be implemented at the software level to improve accessibility. It can be improved through a simple and minimalist design, through flexible configuration of functionality, through multilingual availability or by enabling a simultaneous multichannel interface, allowing to interact with the system both by text and audio.

Implementing multilingual capability in a web server can be achieved by following a few steps. First, the server must be designed to accept page requests in different languages, and ideally, maintain the user's selected language as a session variable. In addition to this point, the server can inspect the "Accept-Language" variable in the header of the first request from a client. Secondly, the server must perform an opportune processing of the requests in order to render the html templates correctly, so that they reflect the correct language. To achieve this, it is essential to have a backend that contains all the texts that will be visible on the web in all available languages. There are different approaches to achieve this goal, but one of the most common strategies involves the use of json files to store the texts of the pages in each of the languages. Finally, it is desirable to incorporate a language-specific attribute in the html files and it is crucial to have an on-screen language selector menu [42].

While textual chat-like interactions can be handled between client and server through simple HTTP requests and responses, with a POST method from the client to the server, in which it sends the textual interaction and receives the response, audio interactions involve additional complexity as the audio must be recorded and encoded/decoded to an appropriate format. In order to allow users to interact with the application by voice, we can use the MediaRecorder API [8] in our JavaScript to record sound with the browser. At this point, depending on the web browser, we can get the audio files in one format or another. Then we can send the audio file to the server and this to the web application, Flask in our case, to first identify the format, using for example the Python library "Magic", and then perform the corresponding audio manipulation, if necessary, with the Python library "pydub" [31].

In this sense it is important to know the main characteristics of digital audio files, which are:

- **The format**: It indicates how the data is stored. The most relevant working formats in the context of this anamnesis system are "WebM" for Chrome, "Ogg" for Firefox and "WAV" for the Dialogflow chatbot.
- **Channels**: This is the number of channels in the audio track and affects the spatial

perception of the sound. This can be mono (1 channel), stereo (2 channels) or more in surround audio configurations.

- **Sampling frequency**: Indicates how many samples are obtained per time unit.
- **Bit resolution**: Indicates the number of discrete values that each sample is encoded with.
- **Bitrate**: Indicates the amount of bits used to encode the audio per second of audio. This is inversely related to audio compression. The bitrate is calculated as the sampling frequency times the bit resolution and the number of channels.

Each of these mentioned parameters must be adjusted to meet the requirements of the differing systems involved.

In the present project these technologies are used to capture and manage the audio files of the user interactions, in order to process them and obtain an appropriate response. The response, also in audio, is sent to the user's browser to be played if the user has playback enabled.

2.4 Data Management

The data generated during the anamnesis process must be properly stored for further processing, analysis and reporting to generate valuable information. To guarantee the persistence of the data there are two classic strategies, on the one hand we have the file management systems and on the other the database management systems. The latter offer many advantages, providing high performance, integrity, consistency and redundancy of data, concurrent access, and many other advantages, especially in terms of security, that make us select this option.

Among the different types of databases, there are two main models. On the one hand we have the relational model and on the other hand the non-relational model. Given the characteristics of the anamnesis system under development, where the nature of the data can dynamically change, with potentially high data volume and scalability needs, and where low latency is a priority, a non-relational system has been chosen. Specifically, MongoDB has been selected as the database management system.

In relation to data processing, the Python environment includes the Numpy [10] and Pandas [11] libraries for efficient data manipulation and analysis, as well as Matplotlib [7]

for the graphical visualization of information. Finally, for the generation of PDF reports with the analyzed and structured information, we have the Python library ReportLab [14].

2.4.1 Data Storage. MongoDB

MongoDB is a document-oriented non-relational database management system. In contrast to the tables of relational models, it uses collections of documents to structure the information. The documents are composed of key-value pairs and are stored in BSON format, a format similar to JSON but binary. Its most outstanding features are its flexibility in data structure, as well as its performance and scalability [5]. For its access and management in Python environments, the official driver called "pymongo" can be used. With this driver all database operations can be managed, including connection, CRUD operations and aggregation operations among others [13]. The present software makes an extensive use of this library in the modules dedicated to the model part of the anamnesis software, so that all the data generated is stored in this database.

2.4.2 Data Presentation. ReportLab

Reportlab is a widespread Python library that allows the dynamic generation of PDF documents in a parameterized way, including the insertion of graphs and tables. This allows to automate the generation of reports. This library has been used to present in a clear and structured way the information of the medical anamnesis once it has been completed.

2.5 Chatbot Technologies

A chatbot is a software designed to interact with humans through text or audio conversations. These systems make use of Artificial Intelligence (AI) technologies as well as Natural Language Processing (NLP) and Natural Language Understanding (NLU) techniques to understand and respond appropriately to user interactions, replicating a human conversation. There is currently a growing use of AI-based chatbots. These technologies allow minimizing human intervention in the use of technologies, saving operations and improving usability. Among their uses, their ability to efficiently gather information is one of the most important [30].

2.5.1 Chatbot Main Concepts

Some of the key concepts related to Chatbots are:

- **Intents**: Represent the correspondence between the input provided by the user and the action to be taken by the chatbot. The different intents are triggered by certain user inputs and contain parameters with specific information about the user.
- **Parameters**: Parts of the specific information that defines an intent.
- **Entities**: Tools for the extraction of specific values defined by the parameters.
- **Context**: References to the subject of a conversation and the previous conversation information.

2.5.2 Dialogflow

Dialogflow is a platform for the development of Chatbots owned by Google. Dialogflow in its ES version offers a free trial version that has been used for the development of this system.

Dialogflow ES is organized around **agents**, which are virtual assistants that handle user conversations in an independent way, so that different agents can manage conversations in completely different ways. The same google account can have multiple agents and each agent must be linked to a google cloud project. Making use of this feature, the present anamnesis system has different agents and thus different types of Anamnesis.

Each of the agents has its own intents. For the assignment of intents to each user input, Dialogflow ES makes use of **contexts**, which allows to prioritize some intents over others depending on it. Every intent consist of 4 basic elements.

- **Training sentences**: These are the sentences with which the AI model underlying the chatbot is trained.
- **Action**: Response triggered by the intent that can be used to activate some functions.
- **Parameters**: The specific parts that define each intent and that must be collected in order to have a complete intent, if specified. Each parameter is associated with an entity type that dictates what type of data each parameter can contain.
- **Response**: The message that the chatbot sends after completing the intent. An intent is understood to be complete when all its mandatory parameters are known.

The typical operation of Dialogflow ES is illustrated in Figure 2.1. After a user input, the agent to which the request is sent assigns that input to the closest intent, or to the fallback intent in case it does not resemble any sufficiently. After the assignment, it seeks to complete all the mandatory parameters in case it does not have them, making use of the questions that have been specified to collect each one of the parameters. While asking for the parameters of an intent, the active context is that of the intent itself, which largely prevents the conversation from leaving the present intent and diverting to another, maintaining consistency. However, there is the possibility of changing intent if required.
Once all parameters are completed, it triggers the set action if it exists and returns the response [1]. User interactions with Dialoglow ES can be done either with preconfigured

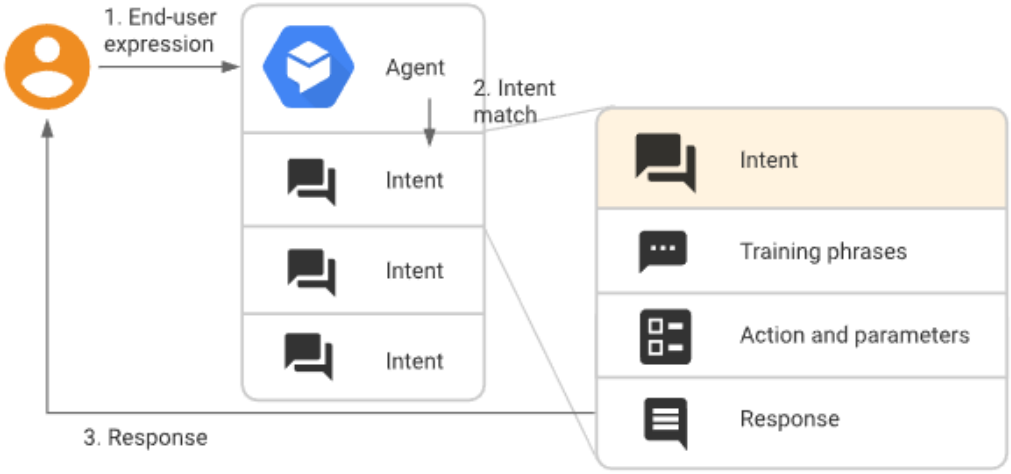


Figure 2.1: Dialogflow Basic Workflow [1]

integrations provided by the ecosystem itself or in a customized way to interact directly with the DialogFlow API for each user interaction as shown in Figure 2.2. In this custom configuration, the web application is the intermediary between the user and Dialogflow, which brings benefits in terms of increased capabilities, being able to modify messages or send messages to Dialogflow that have not been explicitly expressed by the user. In addition, it allows to directly obtain and store all the details of the Dialogflow response, such as the assigned intent, the text interpreted in an audio interaction, whether the intent is complete, the different parameters and even a sentiment analysis of the interaction [1].

This system communicates with the Dialogflow ES API using the Python library provided by Google for this purpose.

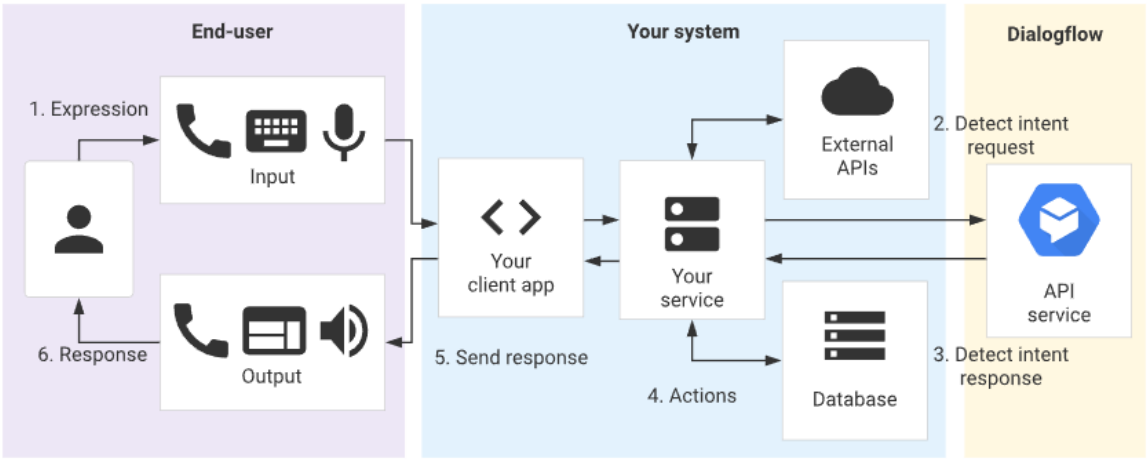


Figure 2.2: Dialogflow API Interactions [1]

2.6 Interoperability. Standardized Terminology. SNOMEDCT

The design of interoperable systems and the use of standardized terminology are crucial in healthcare to ensure effective exchange and communication of clinical information between different systems. In this context, SNOMED CT, as the most complete and widespread medical terminology in the world, plays a key role in enabling healthcare professionals to encode information in a precise and unified way using a unique identifier [24].

SNOMED CT presents integral and scientifically validated content, characterized by its comprehensiveness, scalability and flexibility. This terminology is multilingual and mapped to other international standards. SNOMED CT content is represented by three types of components:

- **Clinical concepts**, which are ordered by hierarchies and represented by numeric identifiers. SNOMED CT, in this sense, has a multi-heritance architecture.
- **Descriptions**, which link human-understandable terms to the concepts.
- **Relationships**, which serve to link related concepts.

In addition, SNOMED CT can be adapted to different functionality requirements by customizing it with the use of Reference Sets (Refsets).

The present system has coded and reports all symptoms and their corresponding coding based on this standard terminology.

CHAPTER 3

Architecture

3.1 Introduction

In this chapter, the design phase of this project is addressed, along with the implementation details related to its architecture. An overview of the project, divided into several modules, is initially provided. This is aimed at showing the reader a broad perspective of the project's architecture. Subsequently, each module is presented separately and in greater detail.

3.2 General Overview

The web system developed for anamnesis has been named "Anamon", and from now on, it will be referred to by this name.

The integral architecture of the application consists of Anamon as a web server, which connects, on the one hand, to the client through its web application or browser, and on the other hand, to the Google Cloud server with Dialogflow services. Figure 3.1 depicts the general architecture of the developed anamnesis web system.

Within the Anamon server, we find the Flask web application, whose architecture and

implementation is detailed in Sect. 3.3. This application contains all the logic of the anamnesis service allowing the entire process to be managed, as explained in Sect. 3.8, and generates the corresponding reports as detailed in the Sect. 3.9.
When starting the web application and, to allow its correct operation with the rest of the system components, it must go through a start-up and configuration process, which is detailed in the Sect. 3.7.

The web application is the central part of the architecture and is connected firstly with a non-relational database MongoDB, which is responsible for both storing and providing all relevant user, agent and anamnesis data as required. The detailed structure of the database is developed in the Sect. 3.4.

In addition, the web application also connects to the Dialogflow server on Google Cloud, which is the chatbot containing the various conversational agents that process the questions and return the appropriate answers to perform the anamnesis. Its structure and configuration is detailed in Sect. 3.5.

Finally, the web application establishes a connection with the user's browser in order to attend the user's requests and serve the web content as needed. The structure and the most relevant details concerning the user interface are described in Sect. 3.6.

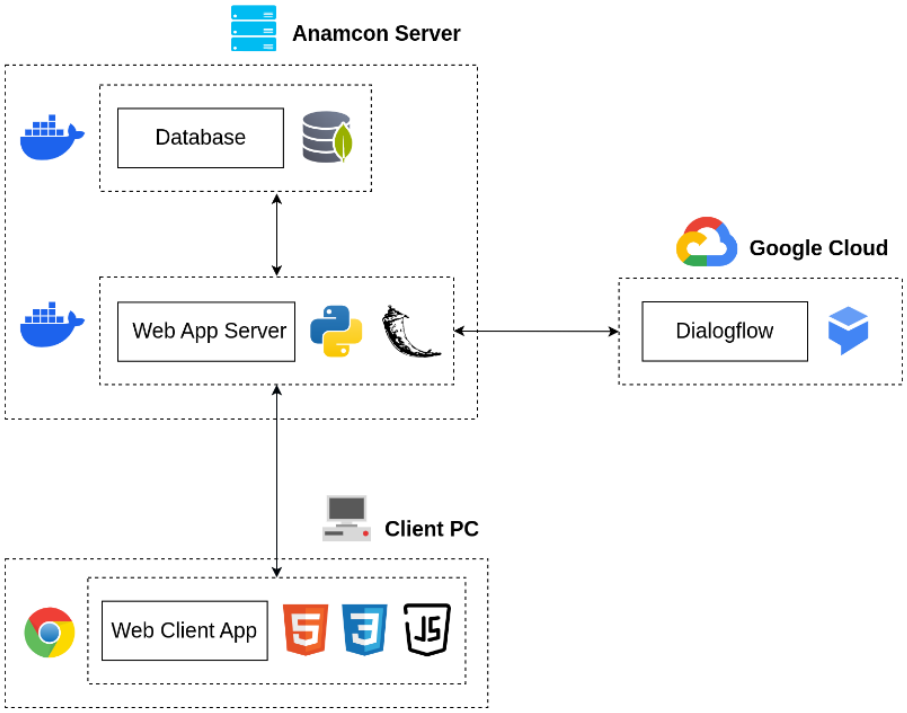


Figure 3.1: General System Architecture

3.3 Web Application

The Flask web application follows the MVC architecture and is deployed as a Docker container. It is divided into different packages that organize the related functionalities in such a way that the interaction during the application execution always occurs among files of the view-controller or controller-model pairs, as ilustrated in Figure 3.2.

In the application files, we first find the main files responsible for starting and configuring the Flask web application. Next to these, there are the folders corresponding to the main packages. The main elements of these packages according to the component they belong to are:

- **Models**: The components related to the models are in the "db_models" package. This contains all the files in charge of operating with the database, subdivided according to the type of operations carried out.
- **Views**: The components related to the views are in the "views" package. Here are all the elements in charge of handling the web requests made by the users, as well as the files, both dynamic and static, that are served to the users. This component is subdivided according to whether the routes served are related to registration and authentication, or to the main activity of the web, the anamnesis.
- **Controllers**: The components related to application control are divided into multiple packages according to the function performed:

  - The "auth" package contains the logic that manages user registration and validation.

  - The "net" package contains the logic related to the sending of e-mails.

  - The "report" package is responsible for generating and saving anamnesis reports.

  - The "anamnesis" and "dialog" packages integrate the logic for managing the anamnesis and the corresponding dialogs.

The main functionality of the controller is related to the sending, receiving and recording of interactions between the user and the agent, which is performed by the "anamnesis" and "dialog" components. Regarding the management of interactions by the application, although the handling of textual interactions is quite simple and only requires storing and transmitting the data properly, for the handling of audio some additional procedures implemented by these packages are required. The audio, after being recorded with the "MediaRecorder" API by the user's browser, sent to the web application and having its format recognized through the Python magic library, must be transformed according to Dialogflow's

needs. This is achieved by converting the audio, using the Python library "pydub" to "wav" mono format with a sampling frequency of 24000 Hz and a depth of 16 bits, which gives us an audio with a bitrate of 384 kbps or 48KBps. This audio can then be sent to Dialogflow and get a response similar to that obtained by textual interactions.

In addition, the web application also has folders for storing sessions and their associated data, as well as for storing the credentials necessary for communication with Google services.

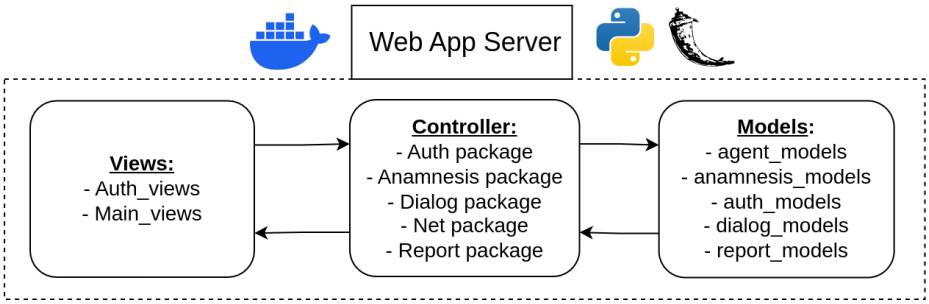


Figure 3.2: Web Applications Architecture

3.4 Database. MongoDB

The database architecture is based on a Docker container with the database management system (DBMS) MongoDB. It comprises a total of three collections, outlined as follows:

- **Collection "users"**: Stores all the data of the system users and is used in their registration and authentication process.
- **Collection "agents"**: Stores the information of the anamnesis agents available in DialogFlow. This allows the monitoring and management of the anamnesis process by the Server. Knowing at all times the details associated with each interaction performed, the server can take the appropriate actions. This collection contains a direct replica of the json in which the agents have been designed.
- **Collection "anamnesis"**: Stores all the anamnesis of the system together with its characteristics as well as the different interactions that have taken place in them.

All the documents contained in the 3 collections have an "_id" assigned by default by the system in an automated way, although these identifiers, for security reasons, never leave the server. The only data that are transferred to each user's interface are their username, which is unique, and their detailed anamnesis. The latter are identified to the frontend by the username and creation timestamp, which can be considered unique for each user.

As for the users collection, the fields stored for each registered user of the system correspond mostly to the personal data provided in the registration form and are shown in the Table 3.1. The last parameter (user_role) is designed to accommodate users with varying privileges in the system although, during the development, only patients have been included.

Table 3.1: User Collection Fields

"username"	"password"	"id_type"	"name"
"surname"	"birth_date"	"sex"	"gender"
"email"	"telephone_number"	"address"	"zip"
"country"	"health_provider"	"health_number"	"creation_datetime"
"language"	"user_role"		

Regarding the anamnesis collection, the most complex collection of the three, it contains as documents the different anamnesis of the users, with different fields explained in Table 3.2.

Table 3.2: User Collection Fields

Field	**Description**
"username" and "user_id"	The username and identifier associated with the anamnesis.
"title"	The title the user has assigned to the anamnesis.
"anamnesis_mode"	Type of anamnesis: General or psychological.
"symptoms"	The symptoms already completed in the anamnesis together with the associated anatomical region.
"creation_datetime"	The date and time of the creation of the anamnesis.
"last_interaction_datetime"	The date and time of the last interaction.
"duration"	Time elapsed between the creation and the last interaction
"status"	State of the anamnesis: created, recording or completed.
"dialog"	Contains each of the interactions of the anamnesis.

Each of the interactions that includes the dialog of an anamnesis includes different fields as shown in Table 3.3

Table 3.3: Dialog interaction Fields

Field	**Description**
"interaction_mode"	How the interaction has occurred, active if initiated by the user, passive if initiated by the server. Passive interactions start with "$__".
"interaction_type"	Whether the interaction was by text or audio.
"interaction_meaning"	Context or phase of the anamnesis the interaction belongs to, such as symptom description, systems "review, or a general interaction like describing an error.
"interaction_datetime"	The date and time of the interaction.
"intent"	The name of the assigned intent.
"complete_intent"	Whether the present interaction has completed the intent to which it belongs.
"query_text"	The text received by Dialogflow.
"interpreted_text"	The text interpreted by Dialogflow, different in audio interactions due to audio-text conversion.
"query_sentiment_type"	The type of sentiment interpreted by Dialogflow, ranging from -1 (very negative) to 1 (very positive).
"query_sentiment_intensity"	A positive number indicating the intensity of the symptom interpreted from the input.
"intent_parameters_and_values"	All parameters and their associated values.

During the execution of the server, only the "agents" collection remains unchanged. The rest change dynamically as users register and perform anamnesis, in order to record all relevant data. An outline of the database architecture can be seen in Figure 3.3.

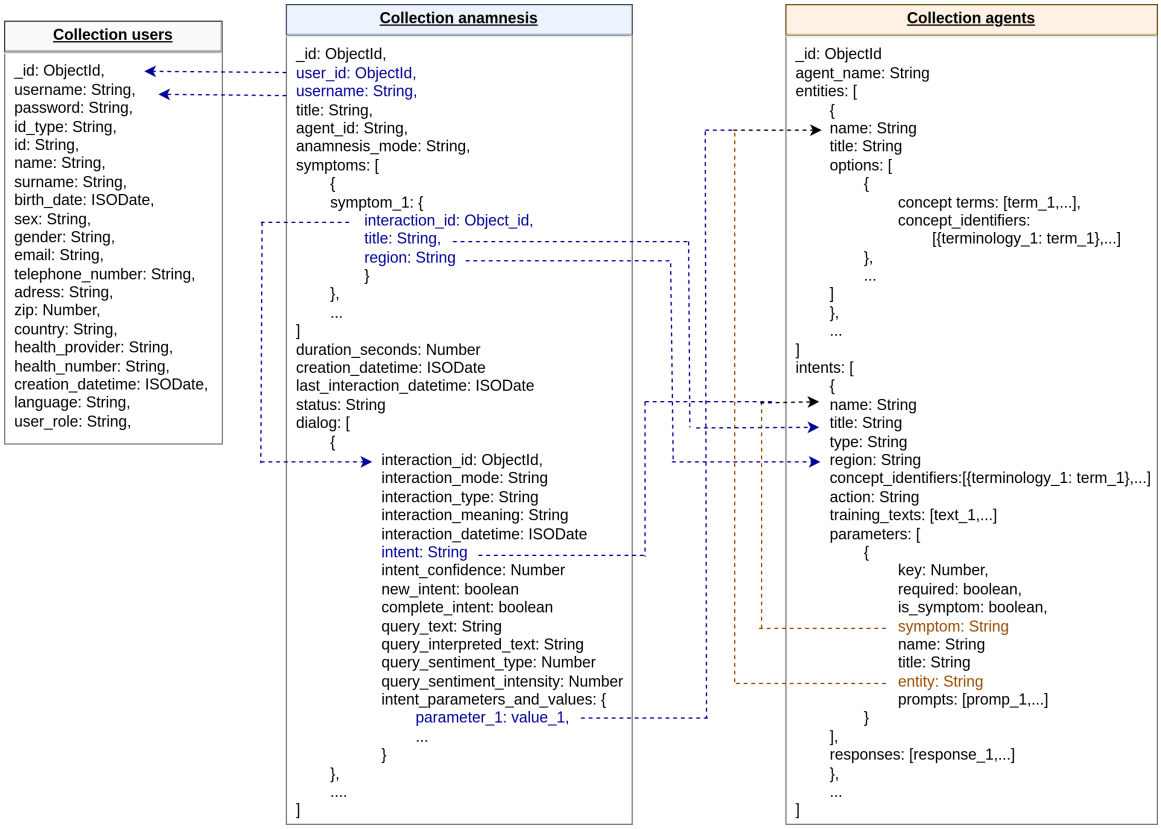


Figure 3.3: Database Design

3.5 Dialogflow. Anamnesis Agents

This anamnesis system makes use of Dialogflow as an integral Chatbot tool, allowing the creation and management of different anamnesis agents and multiple simultaneous anamnesis in an efficient way. These agents act as specialized conversational assistants, capable of understanding and processing questions and answers related to patients' symptoms.
Dialogflow, after configuring the associated Google Cloud account, allows to dynamically adapt, through the web application, the anamnesis agents according to the specific needs of each user or context.

The developed system has the ability to have several anamnesis agents, in order to cover different needs, so that when a new anamnesis is created, it is created based on a specific type, which is linked to the corresponding agent. This feature provides an enormous capacity of configuration and adjustment of the system to different environments and needs.
Each of the agents is parameterized in a corresponding json file in which a set of intents and entities are defined, each of which has a name, which is the value by which it is identified at the chatbot level and a title, that is the value that is externally presented to the user.

The entities define a set of values to which each of the parameters containing an intent and its possible synonyms can be attached.

The intents have multiple configuration parameters that are used internally by the web application to guide the anamnesis. In this way, the chatbot is simply in charge of fitting the user interactions into an intent and completing its parameters, while the web application, through the injection of hidden texts to the agent, can guide the conversation and even react to certain events without the user performing an explicit interaction. This enables a more dynamic conversation in which, for example, the agent, in appearance, sends two consecutive responses, as if in a conversation a person takes two consecutive turns to speak. In this context, some of the training sentences, both intents and entities, start with a code "$__" so that, if used, not only the desired action is produced, but it is also reflected that they are queries initiated by the web application in an autonomous way, so they will not be considered as part of the explicit dialog.

Each intent has an associated type. This can be either:

- **General**: These are related to the usual dynamics and flow of a conversation, such as a greeting, identification of the agent, goodbye, introduction to the system, change of section in the anamnesis or the end of the conversation.
- **Symptom**: Indicates that it is an intent referring to a symptom. These in turn indi-

cate, with a non-null value, the anatomical region to which the symptom is ascribed.

- **Review**: Indicates that this is the intent corresponding to the review, since given the anamnesis flow implemented, it has some particularities that are discussed in the corresponding section.

In this anamnesis system, two agents have been configured. On the one hand we have the agent "Anam" which performs a general anamnesis, based on a medical book of signs and symptoms [53], with a total of 61 symptoms and "Siam", which performs a psychological anamnesis and has only 3 symptoms. This is due to the fact that with the first agent the capabilities of the system can be sufficiently assessed. The second agent is only intended to show the versatility of the system, but configuring it as completely as the first agent would not add more features to the system, although it would have disproportionately increased the development cost of this part.

Figures 3.4 and 3.5 show the detailed content of the "Anam" agent in terms of entities and intents. For each entity, the number of options contemplated is specified. It is important to note that other Dialogflow entities such as dates and times as well as numbers, although not included in the description, are available for all agents and are widely used in our use case. In the case of intents, if they contain parameters as in the case of symptoms, the number of parameters required to complete it is specified.

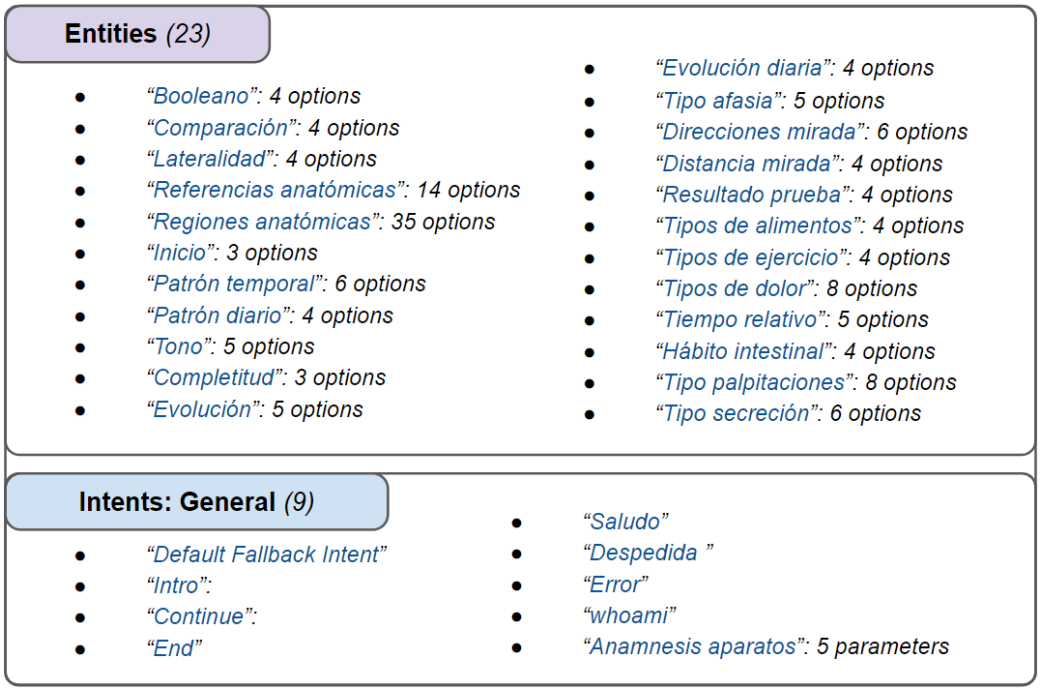


Figure 3.4: Anamnesis Agent "Anam". Part 1

Figure 3.5: Anamnesis Agent "Anam". Part 2

The general characteristics of the agents, such as voice, language, speaking rate or ML classification threshold, are configured when the agent is first created in Dialogflow and linked to a Google Cloud project.

One of the most significant features that directly affects the agent's performance is the ML classification threshold, which establishes the minimum similarity of a user interaction with one of the contemplated options, in order to accept it as correct and proceed consequently. In our case, the default parameter was left at 0.3, as shown in Figure 3.4, since it offered a very good sensitivity while keeping misclassifications very low.

All agents share a similar configuration file architecture, as depicted in Figure 3.6. Note that the files in which the agents have been configured do not allow their functionality, and that these must be processed to be uploaded to Dialogflow in order to properly configure the Chatbot and make it work correctly. This process has been automated and its development is discussed in detail in Sect. 3.7

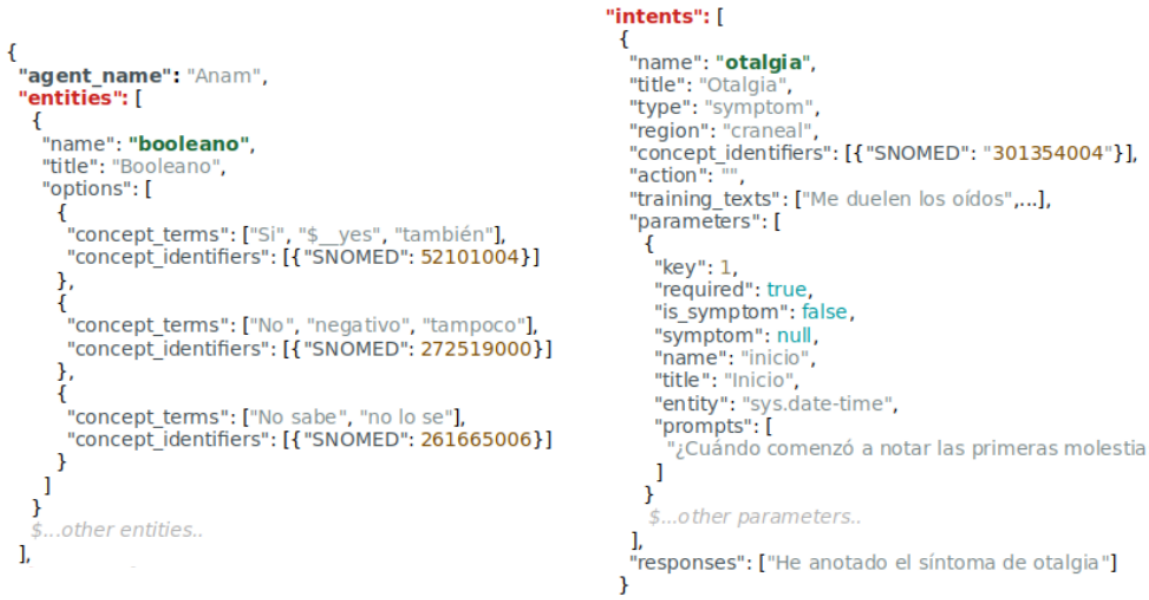


Figure 3.6: Anamnesis Agent Configuration File Structure

3.6 User Interface

The user interface design is based on a minimalist and continuous frontend through the different screens, with 3 differentiated areas, a header, a central body and a footer, as shown in Figure 3.7.

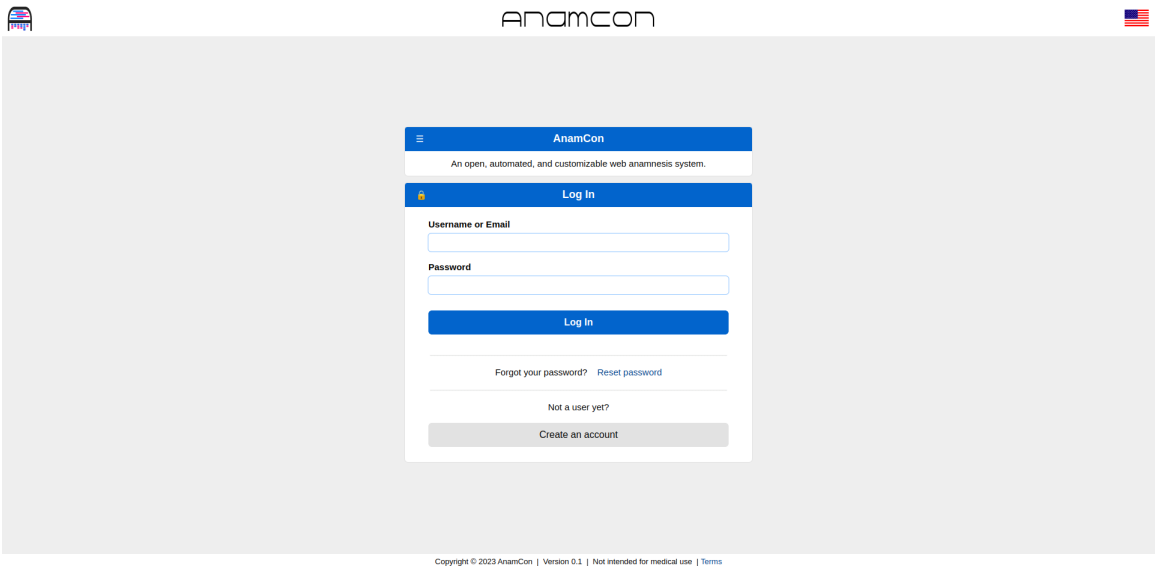


Figure 3.7: Main User Interface Areas

The header has the anamcom logo as a home button on the left side and the language button on the right side, which allows in all available views, to change the language of the content in a dynamic way as shown in Figure 3.8. In the central part is the name of "Anamcon" in the login and registration pages, while the menu appears, with the options available, when accessing the anamnesis screen as shown in Figure 3.9. The footer has a brief reference to the version and rights of the software, as well as a warning that the software is not intended for medical use. Finally, the footer has a link to the terms and conditions accepted in the registration.

The central and main part of the frontend varies significantly depending on the screen we are in. However, it always maintains the same architecture based on window-like boxes with a header indicating the title of each section. While the registration and authentication sections present a simple body design that basically consists of the corresponding form with dynamic field validation as shown in Figure **??**, the anamnesis interface is more detailed. In the main view of the anamnesis interface, on the one hand, in the left zone we have an anamnesis browser, where we can create, select and delete anamnesis. In this area the different anamnesis are represented as elements with their title, creation date and icons

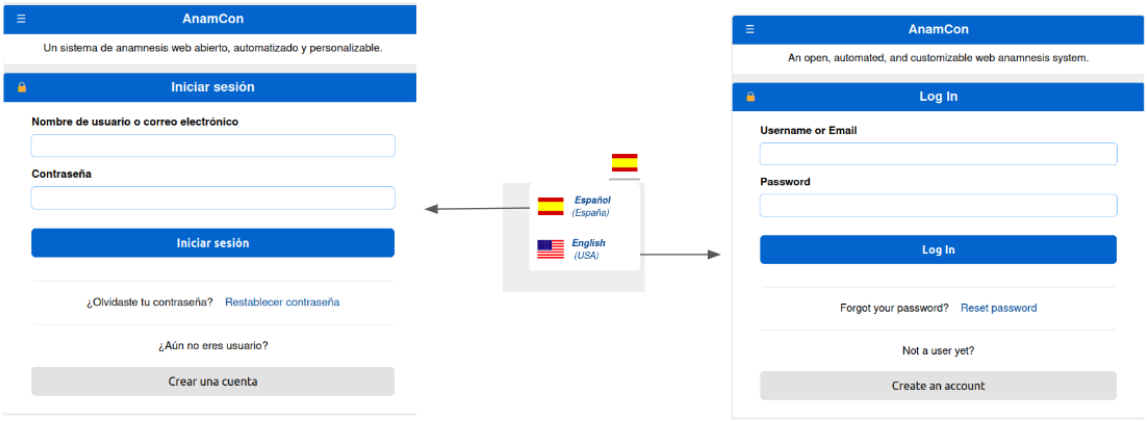


Figure 3.8: Language Selection Interface

indicating the type and status of the anamnesis. On the other hand, in the central area we have from top to bottom: A text box to enter the title of the selected anamnesis together with a button to enable or disable audio playback of the agent's interactions, the area where we can view the entire conversation and finally the area to make the entry by text or audio as desired. Finally, on the right side we have the summary area of the selected anamnesis, where the anatomical regions where the described symptoms are found are drawn on a human silhouette. In addition, it contains the main symptoms described and also a summary of the main data related to the current anamnesis updated in real time. Also in this area, there is a button to generate and download the anamnesis once it has been completed. The Anamnesis view design with its main areas is shown on Figure 3.9.

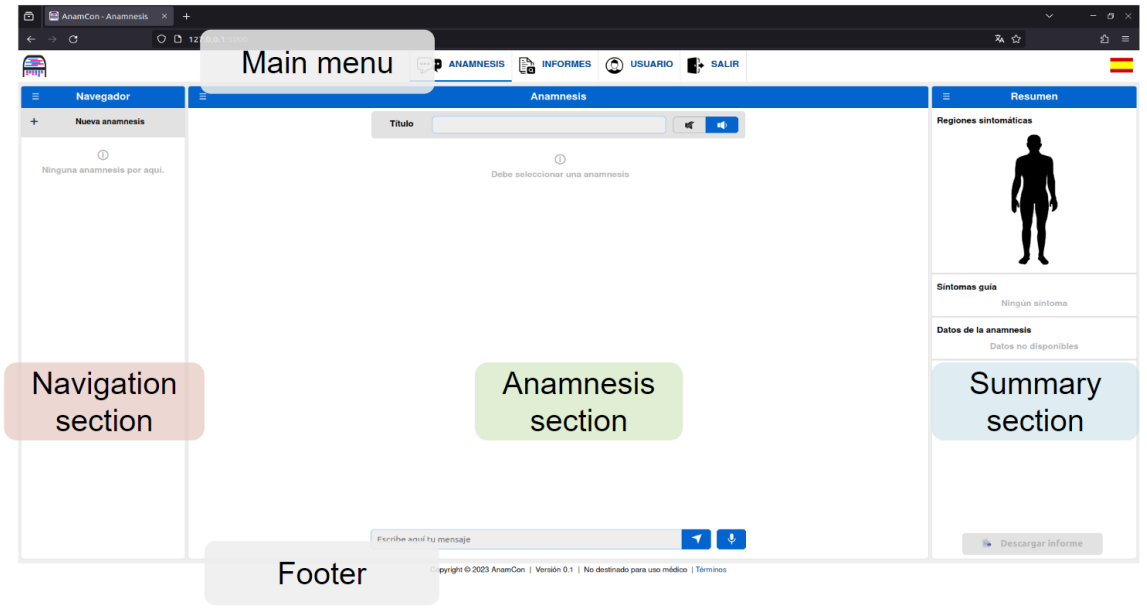


Figure 3.9: Anamnesis View Design

In addition, many of the interactive design elements described above have been developed to provide ease of use and assistance to the user. It is for this reason that some elements are disabled or hidden when it is not possible to use them and others, such as the registration form fields, indicate when erroneous values have been entered and how the problem can be solved, as shown in Figure 3.10.

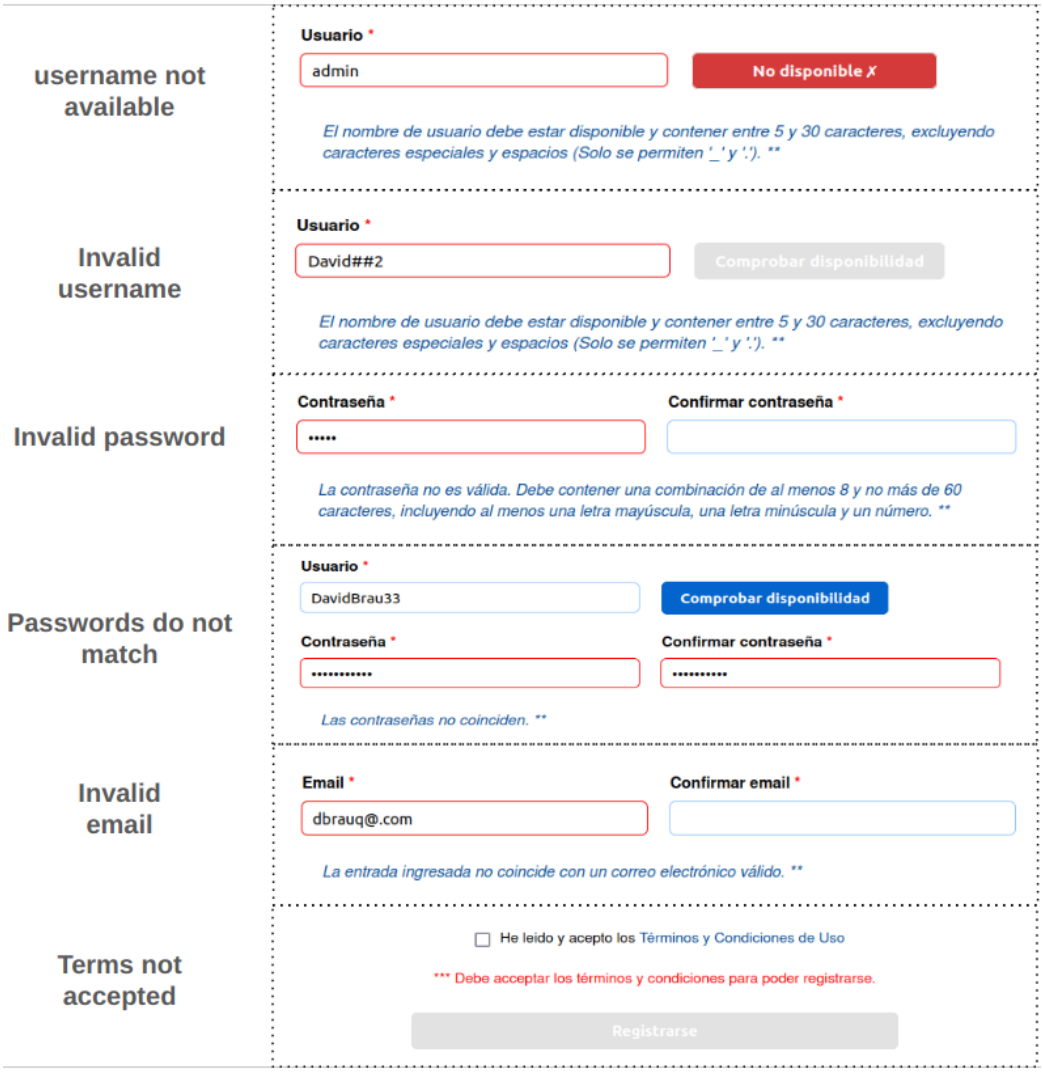


Figure 3.10: Registry Field Validation

3.7 System Onset: Configuration and Deployment

The initialization of the system involves a sequential process that seeks to configure the system according to the requested specifications as well as to create and start the docker containers corresponding to both the web application and the database.

When executing the main program file, "anamcon.py", the software first performs a configuration of the web application, then a configuration of the anamnesis system, and in the last step, executes the Docker Compose to generate the docker images of the web application and the database.

Regarding the configuration of the web application, there are, on the one hand, the main configuration files, existing a file with the default values and a file with the values to overwrite. The underlying idea is that the default file should be kept unchanged, so that in the absence of a specific configuration in the overwrite file, the default configuration is taken. Once the relevant configuration values have been selected, they are loaded into separate files in the different sections of the software as variables that will later be used by those modules. On the other hand, there are files with the web texts in different languages, which are also loaded in the corresponding sections of the web application.

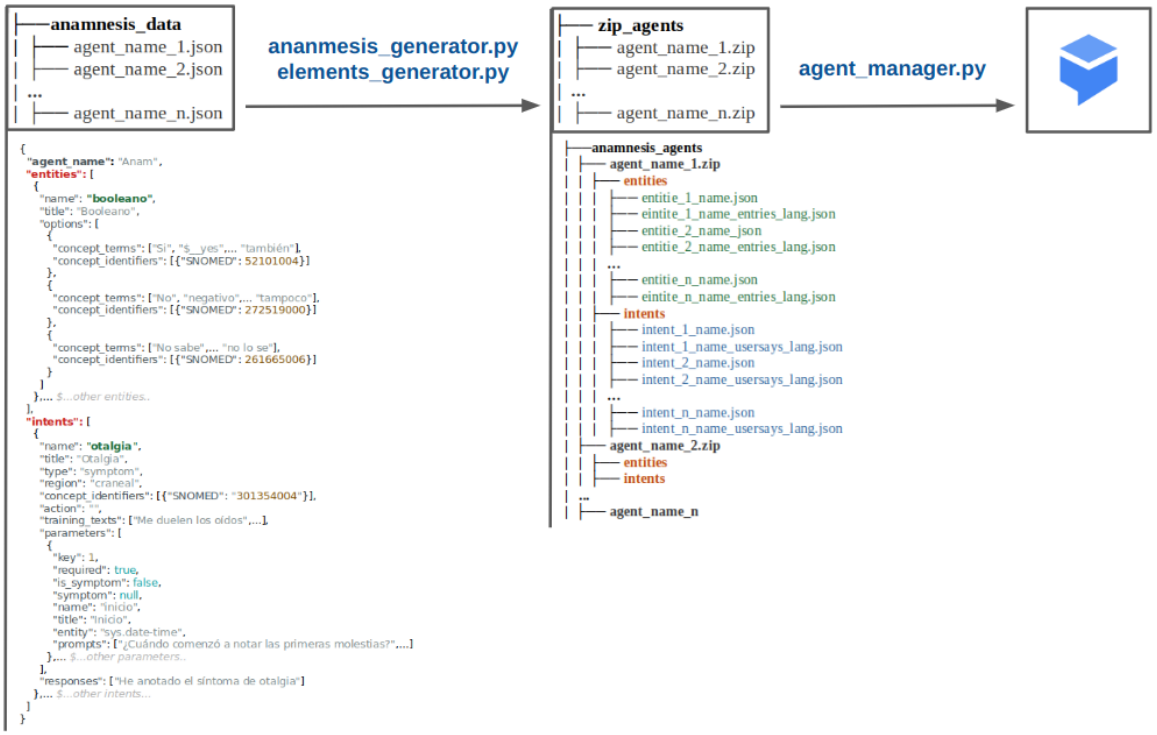


Figure 3.11: Agent Files Generation

For the configuration of the anamnesis system, this is only performed if the corresponding parameter has been enabled in the main configuration file, otherwise the previous configuration is maintained and this step is not performed. In case of being activated first and according to the corresponding configuration files of the agents, the files for each one of the agents referring to their intents and entities are generated in an automated way. After this, the files of each agent are compressed in zip files containing all the information of each agent. Finally, using the dialogflow API, these files are uploaded to the Dialogflow server to configure the different anamnesis agents. Figure 3.11 depicts the agents configuration process.

Once the configuration is complete, the Docker compose is run to initialize both services, the web application and the database. Additionally, the agent configuration files are loaded into the database created, as explained before, to enable the query of certain parameters of the anamnesis during the execution of the system. Figure 3.12 depicts the entire system start-up and configuration process.

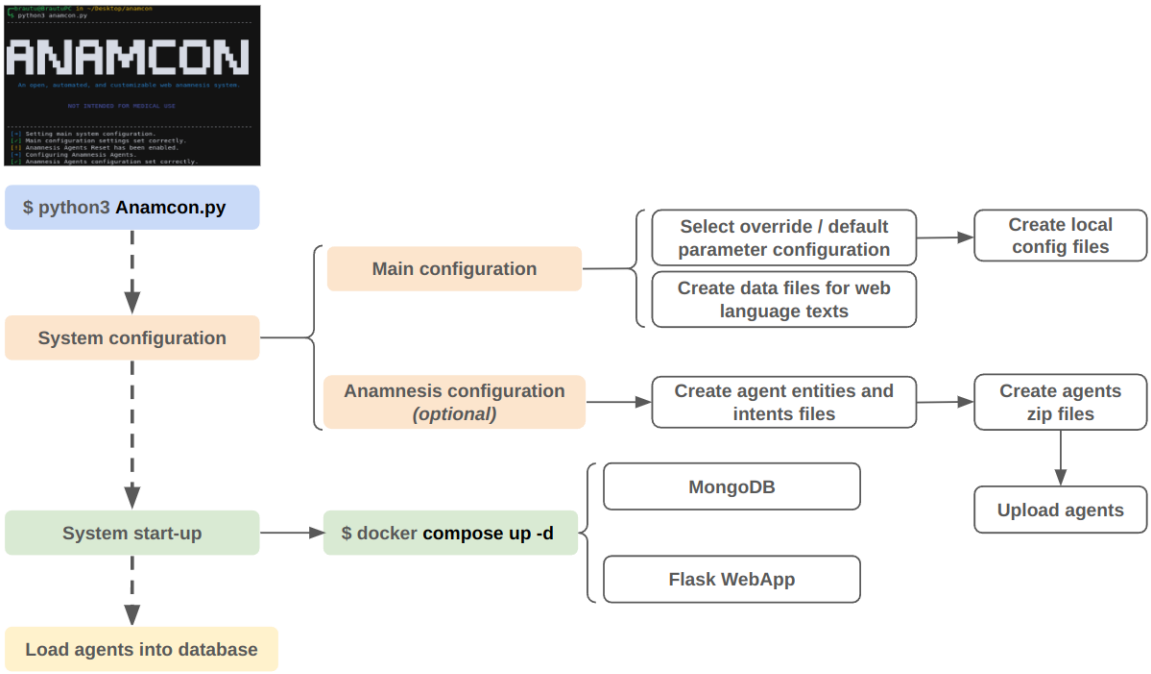


Figure 3.12: System Onset. Configuration and Deployment

3.8 Anamnesis Process

This section describes, from the beginning to the end, the anamnesis process programmed in the system, describing in detail the capabilities and possibilities of the system.

The anamnesis process starts when a user, after having created a particular type of anamnesis, in our case "general" or "psychological", and having it selected in the anamnesis bar, presses the start anamnesis button. At that moment the web application sends Dialogflow a "passive" interaction, to the corresponding agent, requesting the start of the anamnesis. Dialogflow in turn responds to the web application by welcoming the user, explaining basic instructions and asking for the main user's symptomatology. Since it is a passive interaction, the conversation shown to the user will only contain the server's response. This way of control by the application is repeated at different points of the anamnesis and allows a personalized management of the times and phases of the anamnesis, showing in appearance a more natural conversation that is not exclusively question-answer as is usually the case with chatbots.

After the above, the user can describe a symptom, generating an interaction that can be assigned to the intent corresponding to the symptom described or prompts the user to restate the symptom if it doesn't match any predefined symptoms. If correctly assigned, the system asks the user questions to gather details, starting with the symptom onset date for chronological organization in the summary report. If, at any time, the user reports an error, the system cancels the symptom collection, asks for symptomatology again, and allows the user to continue without restarting if the previous symptom collection is incomplete. At the end of the collection of a symptom, the system asks the user for a new symptom, repeating the process of collecting parameters until, when the user has described all his/her symptoms, he/she confirms so.

Finally, the system informs and starts the review by systems, asking the user for other predefined symptoms to be asked in all the anamnesis. If a symptom has already been previously completely described by the patient, the web application avoids asking the patient for that symptom, sticking to those not previously manifested. At the end of this review, the anamnesis is completed, allowing the user to generate and download the corresponding structured anamnesis report.

Figure 3.13 shows the complete flow of the anamnesis process. The lines with arrows indicate the normal flow of the anamnesis, while the dashed lines show other possible paths that may occur.

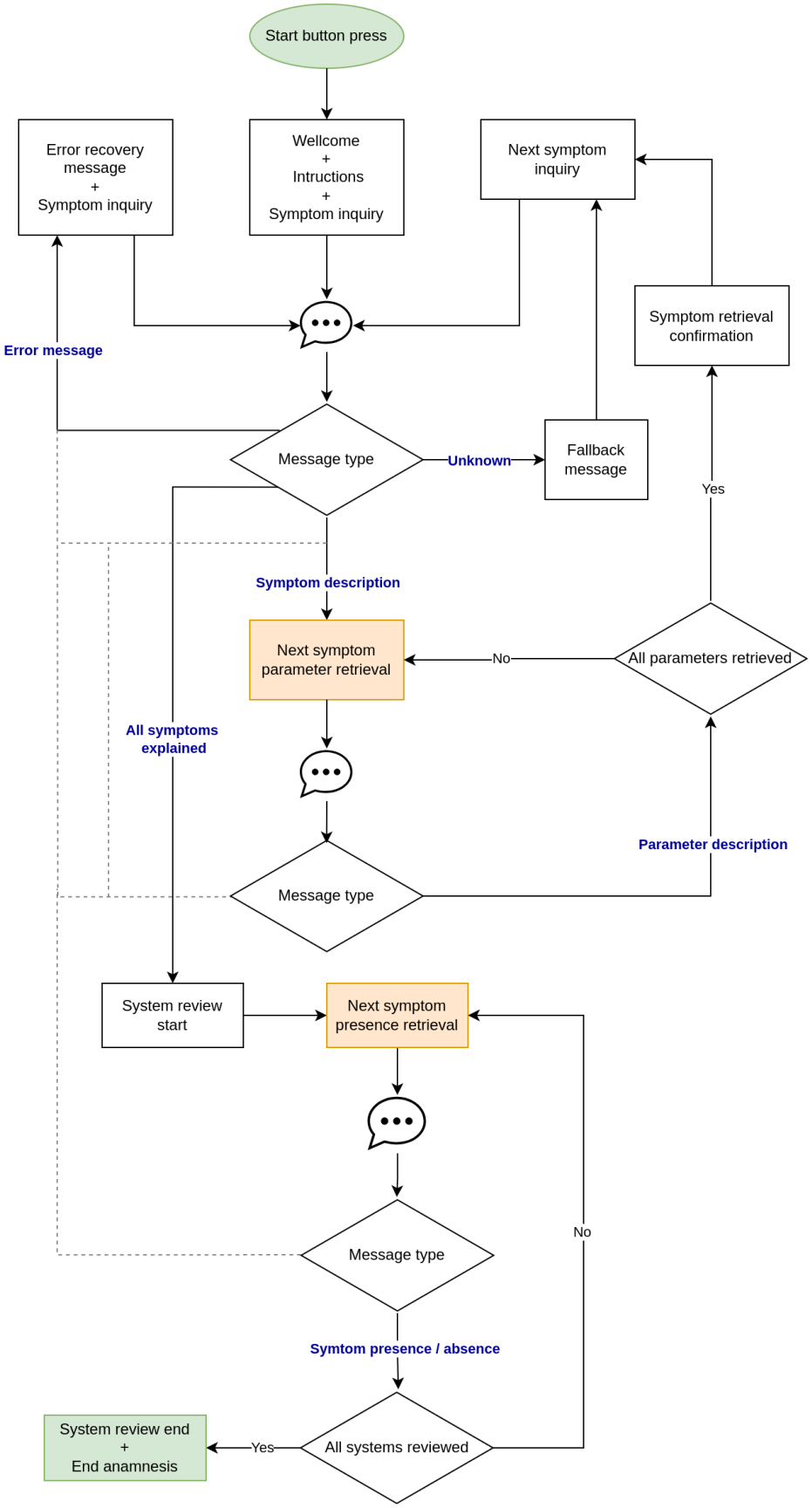


Figure 3.13: Anamnesis Design

3.9 Anamnesis Report

After the user presses the button for downloading the report, enabled after completing the anamnesis, the corresponding PDF report created by the web application using the Python Reportlab library is generated and opened in the user's browser. The report is structured from beginning to end in 6 sections, as detailed below:

- **Header**: Includes the name of the anamnesis system, "Anamcon", together with the title of the document, "Anamnesis Report", and the date on which the anamnesis was performed.
- **Patient information**: Contains the patient's personal data obtained in the registration process with the exception of the patient's access password.
- **Summary of the anamnesis**: Contains the most relevant data of the anamnesis. This includes the type of anamnesis, the date and time of onset, the duration, the average sentiment analysis and the guiding symptoms or main symptoms described by the patient (excluding other secondary symptoms that could be derived from the characteristics of a particular symptom). Regarding the analysis of sentiment, both in this section where the averages are presented, and in the section corresponding to the individual symptoms, the symptom type and intensity are included, as well as a sentiment index that is generated as the product of the two previous ones, trying to aggregate the sentiment analysis into a single value. It should be noted that the sentiment analysis for each symptom is extracted from the interaction in which the patient initiates the corresponding intent and therefore in which the patient describes the symptom. This is because if the interactions related to the collection of parameters were included, there could be several confounding effects that would invalidate the measure, either because of regression to the mean generated by the Boolean responses, or because of the conditionality of the response to direct and concrete questions.
- **Guiding symptoms over time according to the sentiment index**: Here the guiding symptoms are presented in chronological order in a bidimiensional graph representing on the X-axis time and on the Y-axis the sentiment index.
- **Individual symptoms** together with their parameters and sentiment analysis. This section presents each of the symptoms described by the patient together with their corresponding SNOMED CT standard terminology as well as their parameterized characteristics and corresponding sentiment analysis.

- **Review by systems**: Includes the additional symptoms for which the patient has been interrogated and whether they are present or not.

Figure 3.14 depicts the general structure of the automated reports generated by the application after completion of the anamnesis process.

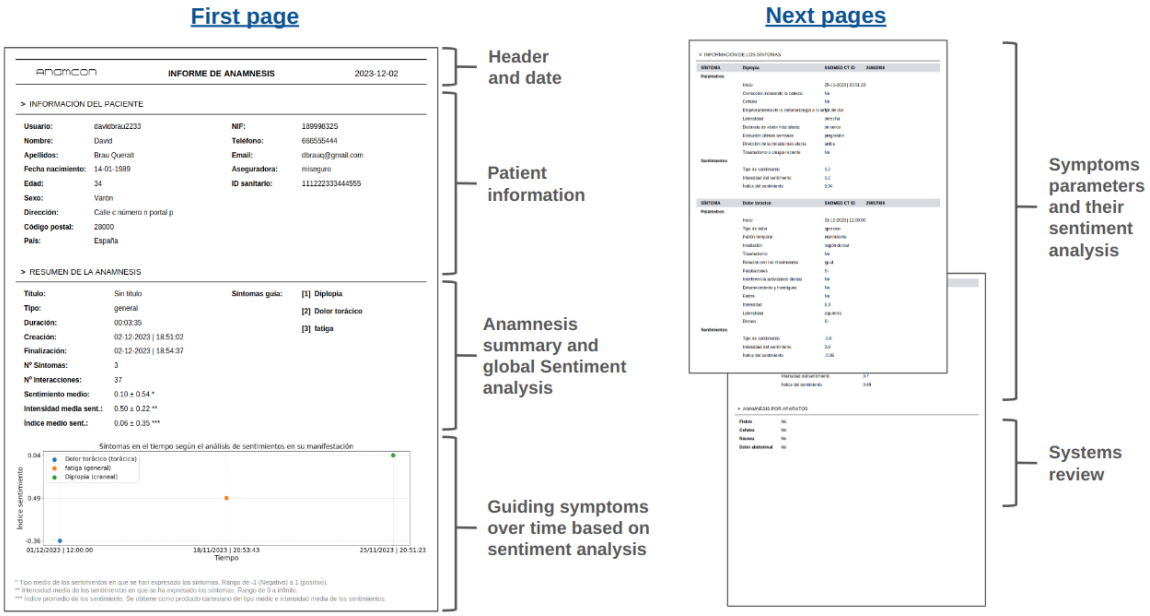


Figure 3.14: Anamnesis Report

CHAPTER 4

Case study

4.1 Introduction

This chapter describes the use cases based on an adapted UML modeling, without making a strict or formal use of it. This chapter will cover the main Anamcon features, and its main purpose is to completely understand the functionalities of Anamcon, and how to use it.

4.2 Motivation

The medical anamnesis is the key process by which medical professionals gather details about the patient's medical situation. This process is extremely important, as it is the basis on which various clinical decisions will be made, starting with the characteristics of the physical examination performed on the patient, the complementary tests that will be requested or even the initial diagnosis or empirical treatment that will be provided to the patient. Anamcon, as described in detail in previous chapters, is an open automated medical anamnesis system, with multilingual and multiagent approach and with customization capabilities, able to provide to users with unlimited time to perform parameterized anamnesis

processes that allow obtaining the patient's symptoms in a structured way that facilitates the availability and access to the data. In order to show all the functionality developed in the system, first, the main use cases of the application will be shown, and then, through a sequence diagram, the usual flow of use of the application will be illustrated. After that, the basic steps to correctly initiate and perform an anamnesis will be presented and finally, an example of use will be shown by means of a detailed development of an anamnesis process from the beginning to the obtaining of the report.

4.3 Use Cases

The anamnesis system has 5 fundamental use cases, which are are listed below:

- **Registration on the platform**: The user registers on the platform by providing some personal data, which will be present in all the anamnesis reports made by the patient. In order to use the system, the user must accept the terms of use.
- **Creation and validation of anamnesis schemes**: Anamcon's development team, in collaboration with healthcare professionals, is responsible for generating, maintaining and scientifically validating the anamnesis to meet the needs of users, maintaining scientific rigor at all times.
- **Anamnesis management**: The user, once inside the platform, can create new anamnesis corresponding to one of the available types or delete any of the previously created anamnesis. In addition, and as the main functionality, the user can complete a new anamnesis by talking with the agent, either by text, audio or a combination of the above.
- **Report visualization**: After successfully completing an anamnesis, users can request the automated generation of a summary report of the anamnesis process from the data generated. The objective is that these reports can be available to the healthcare professionals who will be attending the patients at a later time.

In each of these use cases, the Anamcon team provides technical support to users to ensure the correct use of the platform and deal with any incidents that may arise.

Figure 4.1 illustrates the UML based use case diagram of the application, which provides a high-level visual representation of the main interactions between the actors and the various potential use scenarios of the system, highlighting the main functionalities offered.

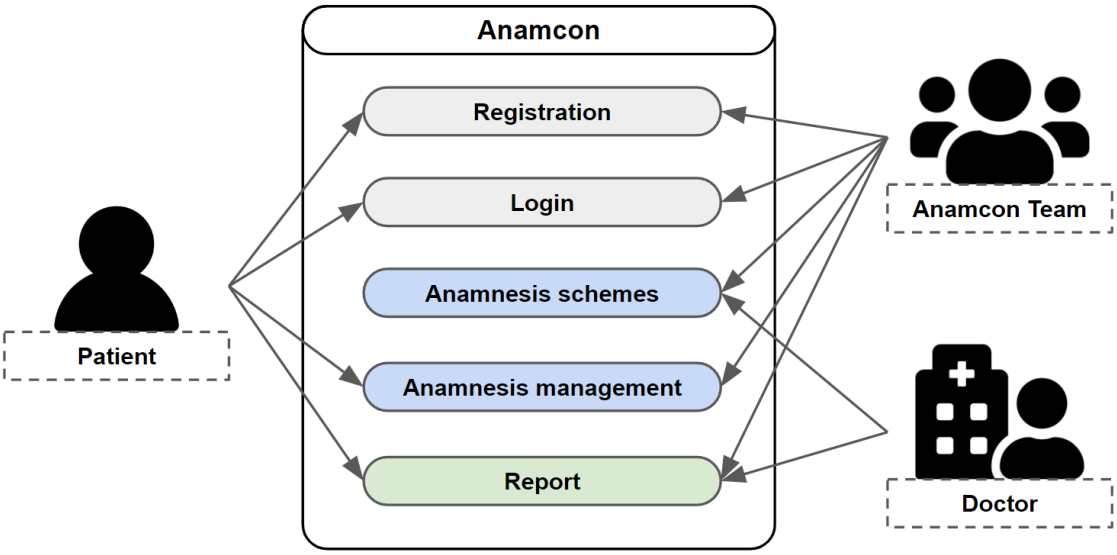


Figure 4.1: Use Cases

4.4 Aplication Workflow

The usual workflow of the application involves the different systems described in the previous chapters and is usually carried out in a sequential manner, as described below:

Firstly, the patient accesses the application through user browser or web client to register, and after completing the registration and the system has stored the new user, a confirmation email is sent as shown in Figure 4.2.

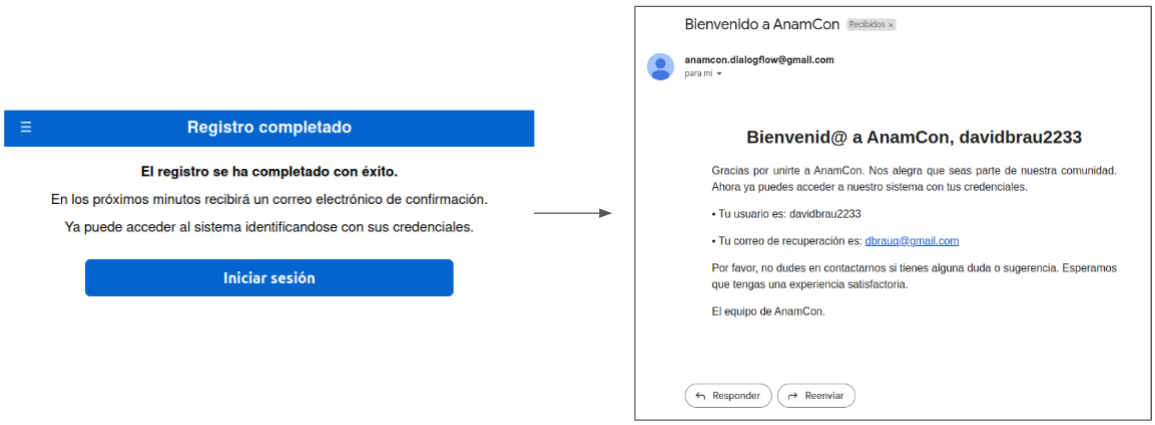


Figure 4.2: User Registration Email

Then, the user proceed to authenticate to access the main view where the anamnesis process can be performed. After authenticating and creating an anamnesis of a certain type, the patient can start the dialog with the anamnesis agent, which generates the appro-

priate interactions with the user while the web application not only manages the anamnesis process, but also stores all the interactions data in the database.

Once the anamnesis is finished, the user can request that the application, after analyzing the data stored on the anamnesis, generates the corresponding report. This report is not only intended to be received by the user, but it could also be sent in the future in an automated way to the health center to which the patient is assigned, although this last functionality has not yet been developed.

Figure 4.3. shows the functional modeling of the system by means of a sequence diagram, allowing the visualization of the system's operation over time through the different systems.

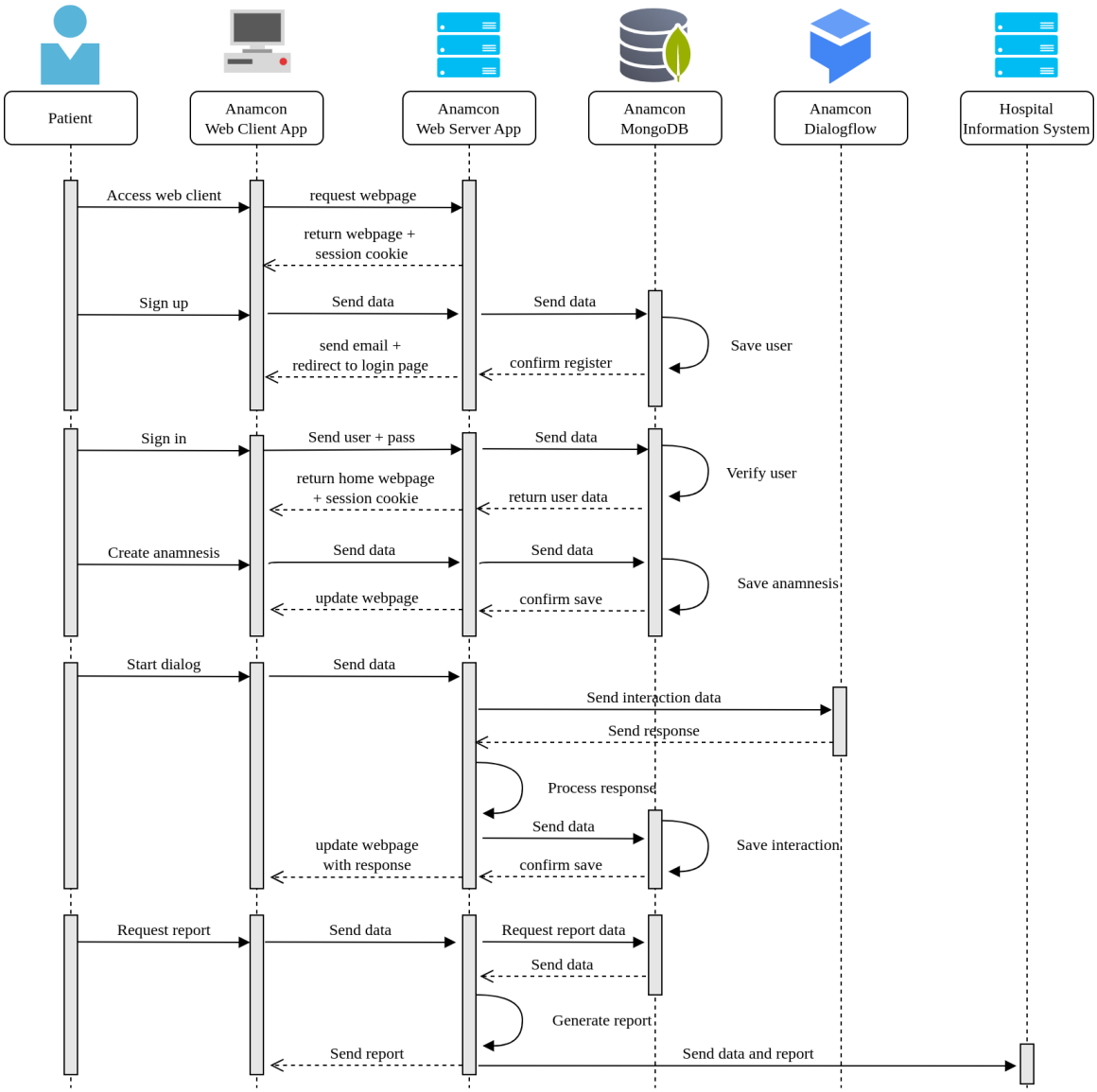


Figure 4.3: Sequence Diagram Illustrating the General Workflow

Regarding the detailed steps for the use and complete development of an anamnesis in the main view of the application, these can be listed in the following steps.

1. The user creates a new anamnesis of the desired type.
2. The user selects the desired anamnesis. By default, after the creation of an anamnesis, it is automatically selected.
3. After reading the anamnesis warning, the user presses the start button.
4. The user uses text or audio input as needed to interact with the anamnesis agent and develop the anamnesis.
5. The patient visualizes at all times the dialogue with the anamnesis agent and continues with the anamnesis until it is finished.
6. After enabling the download button of the report, once the anamnesis has been completed, the user proceeds to download the report with the parsed summary of the anamnesis.

The steps described and the sections in which they are developed are illustrated in Figure 14.4 (steps 1 to 5) and Figure 4.5 (steps 6 and 7). In addition, these figures also describe the complementary areas with information related to the anamnesis process.

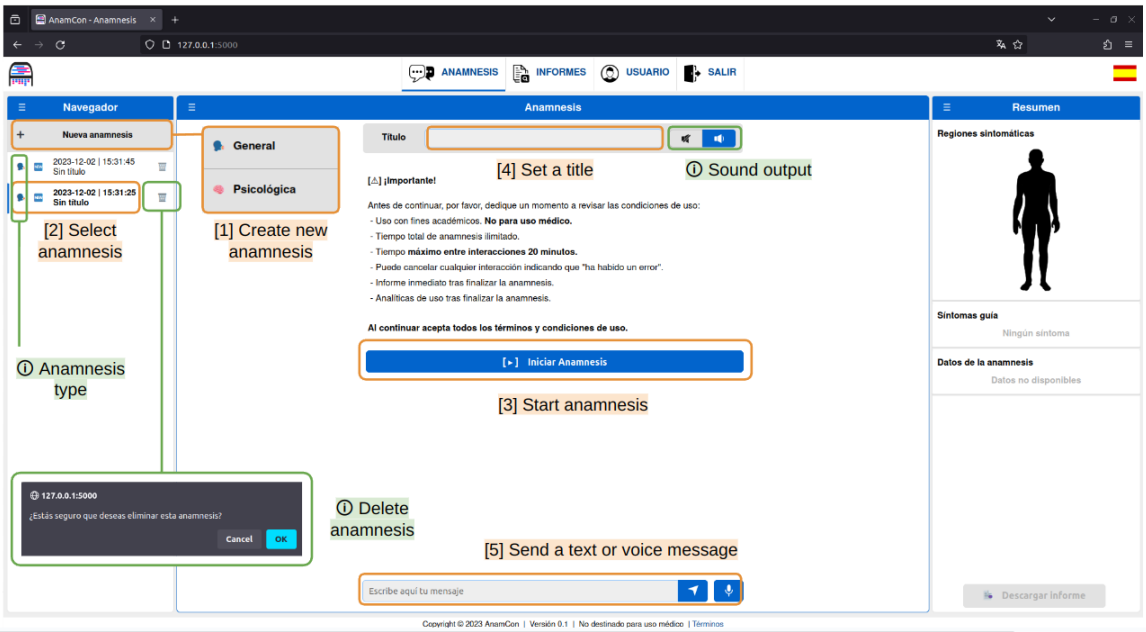


Figure 4.4: First Part of the Anamnesis Workflow Implementation

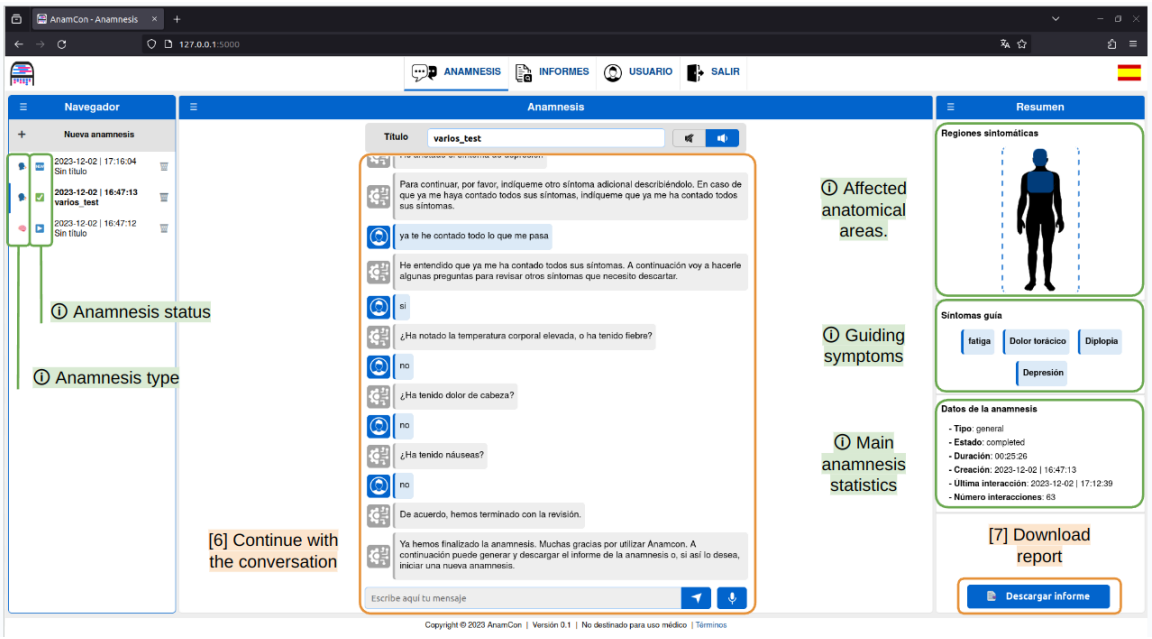


Figure 4.5: Second Part of the Anamnesis Workflow Implementation

4.5 Use Case Example

The following is an example of the whole functionality of the system. The case presented here is a mere example and does not correspond in any way to a real patient.

The invented case is about a patient named “Eric Garcia” with an incipient cardiomyopathy who has recently started to suffer from chest pain and edema in the lower limbs. This patient, before going to his doctor’s appointment, which is scheduled in a few days, registers and accesses the Anamcon system in order to perform a medical anamnesis. This allows the patient to calmly record all the symptoms in a structured way, which will facilitate subsequent access and analysis of the data generated and, in addition, will allow the doctor to save time by only needing to review the anamnesis. The steps followed and the corresponding illustrations are shown below:

1. User registration (Figure 4.6)
2. Login (Figure 4.7)
3. Creation of general anamnesis (Figure 4.8)
4. Completion of the anamnesis (Figure 4.9)
5. Downloading and obtaining the anamnesis report. The pages of the generated report are shown in Figures 4.10, 4.11, and 4.12.

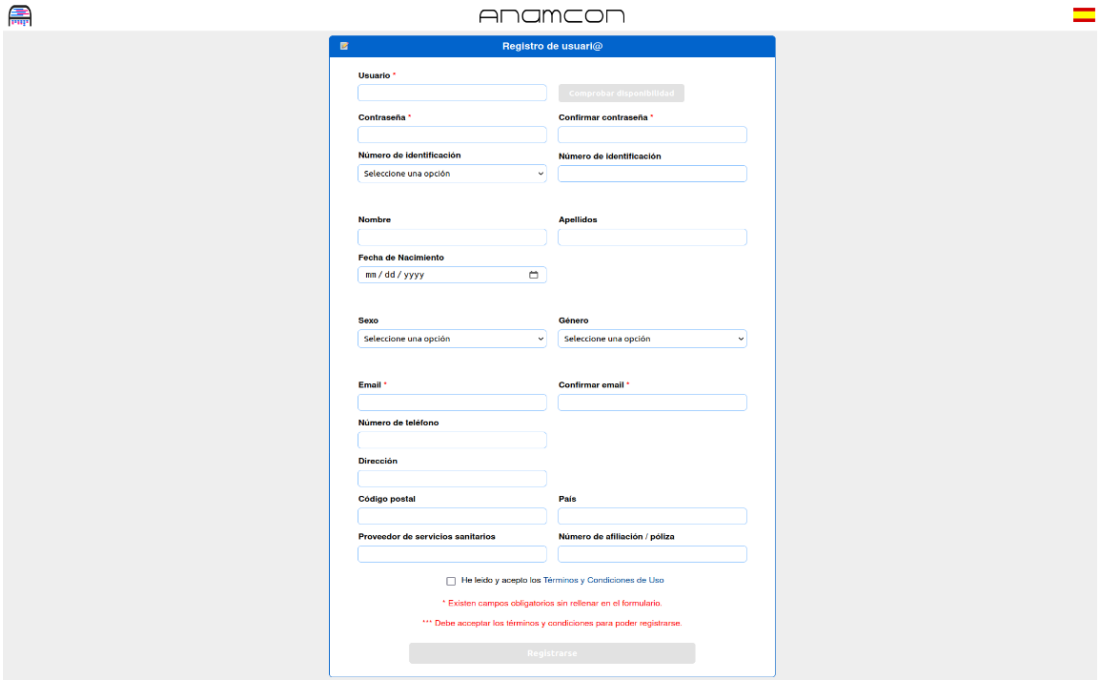


Figure 4.6: User Registration

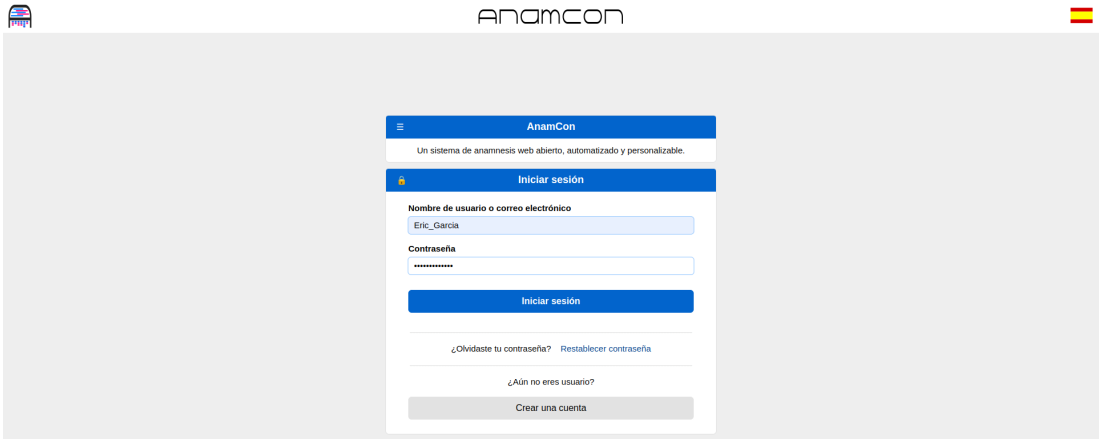


Figure 4.7: Login

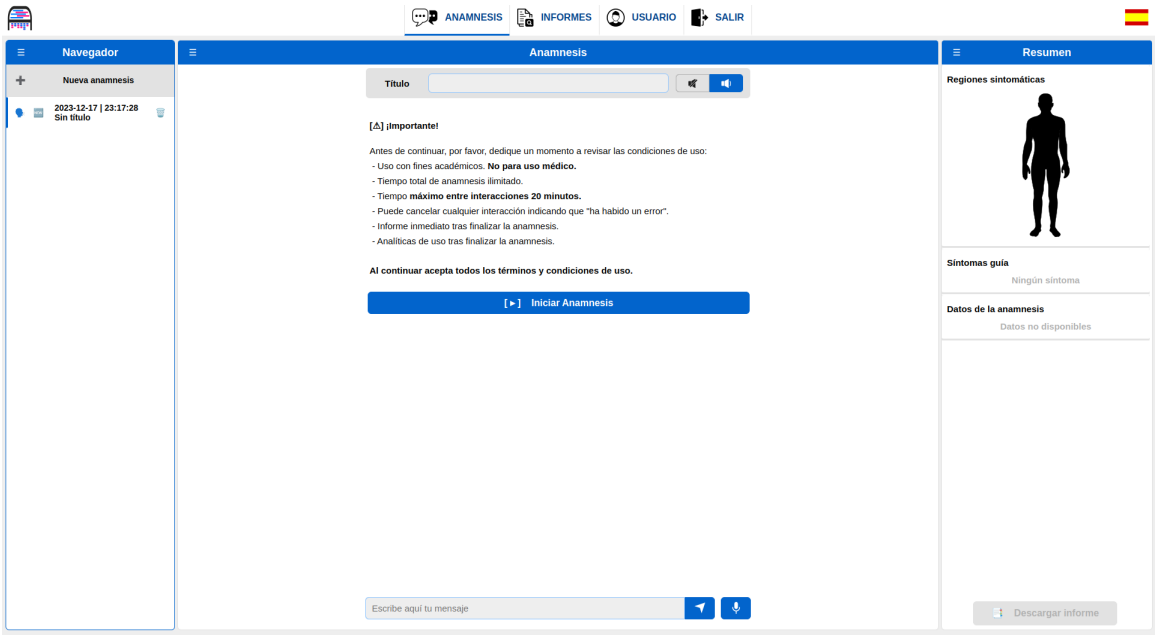


Figure 4.8: Creation of General Anamnesis

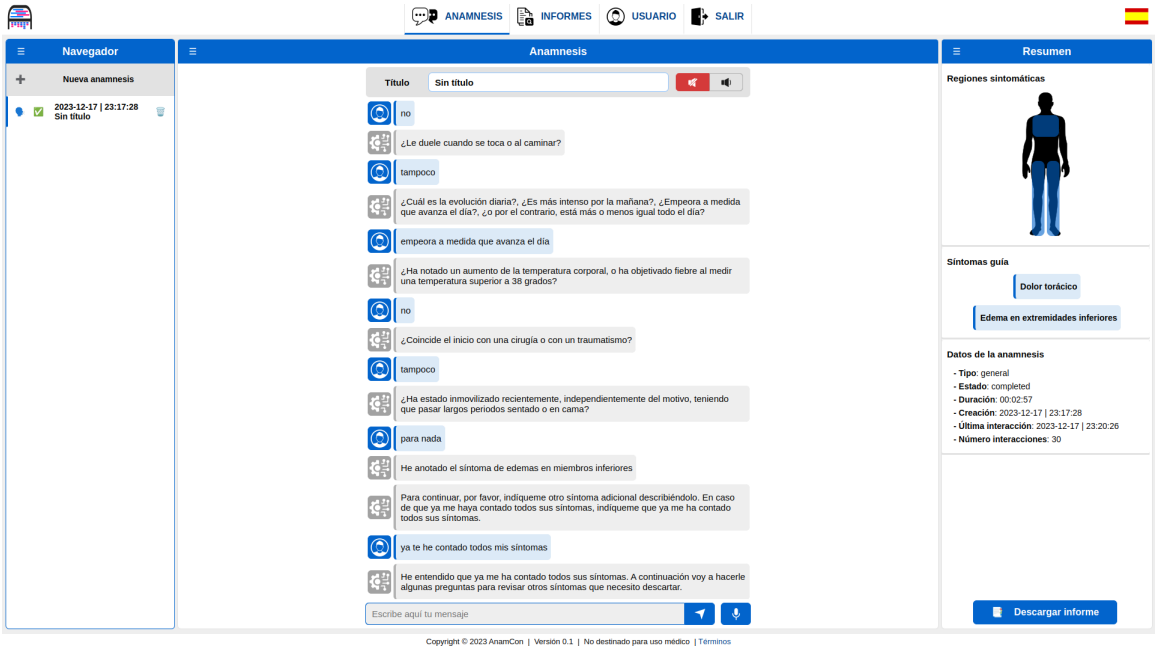


Figure 4.9: Completion of the Anamnesis

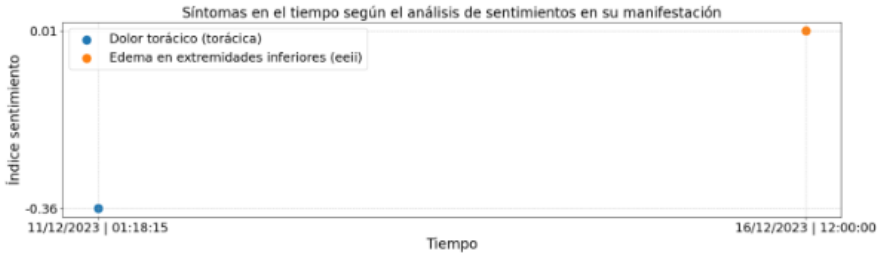
anamcon **INFORME DE ANAMNESIS** 2023-12-17

> INFORMACIÓN DEL PACIENTE

Usuario:	eric_garcia	**NIF:**	12345678Z
Nombre:	Eric	**Teléfono:**	666555444
Apellidos:	Garcia	**Email:**	eric_garcia@mail.com
Fecha nacimiento:	08-05-1989	**Aseguradora:**	miCompany
Edad:	34	**ID sanitario:**	111222333444555
Sexo:	Varón		
Dirección:	mydir		
Código postal:	28000		
País:	España		

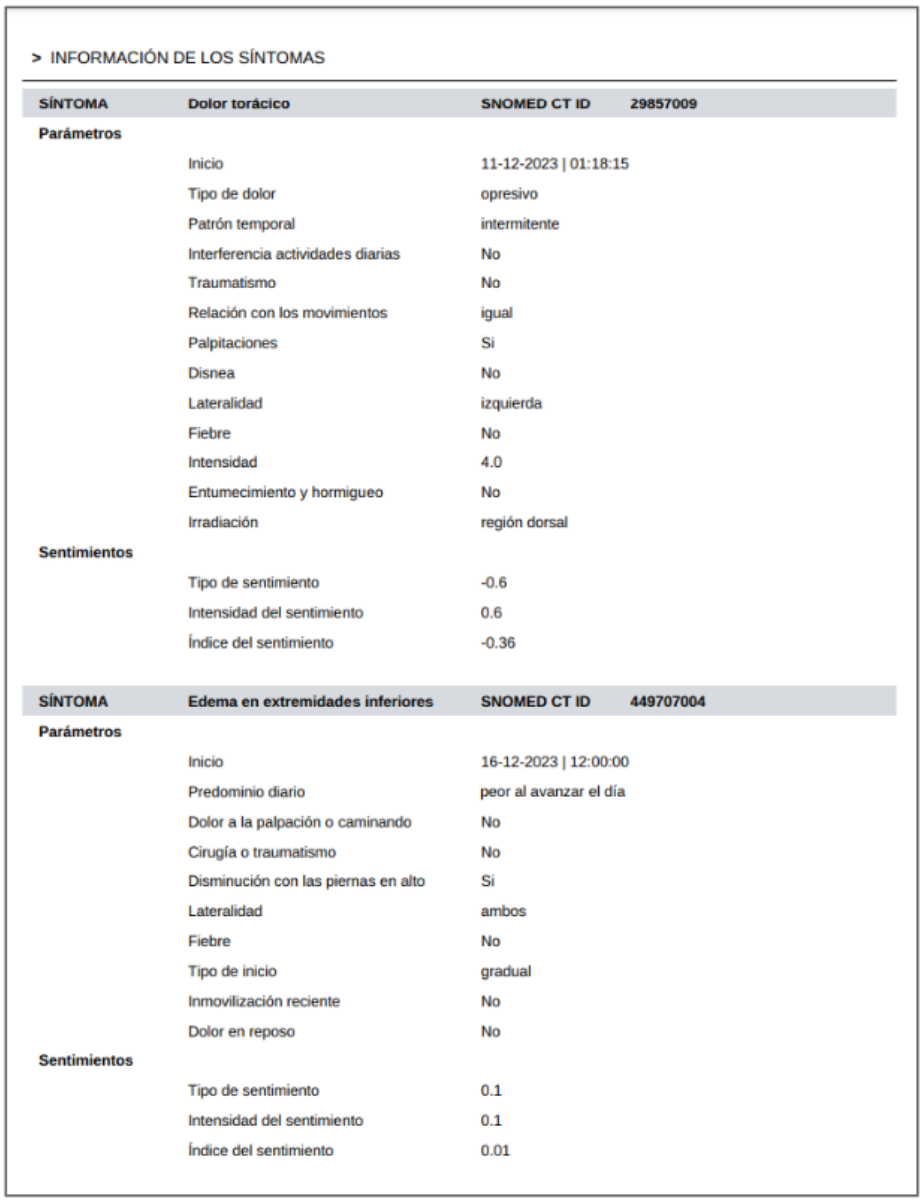
> RESUMEN DE LA ANAMNESIS

Título:	Sin título	**Síntomas guía:**	**[1] Dolor torácico**
Tipo:	general		**[2] Edema en extremidades inferiores**
Duración:	00:02:57		
Creación:	17-12-2023 | 23:17:28		
Finalización:	17-12-2023 | 23:20:26		
Nº Síntomas:	2		
Nº Interacciones:	30		
Sentimiento medio:	-0.25 ± 0.35 *		
Intensidad media sent.:	0.35 ± 0.25 **		
Índice medio sent.:	-0.17 ± 0.18 ***		



* Tipo medio de los sentimientos en que se han expresado los síntomas. Rango de -1 (Negativo) a 1 (positivo).
** Intensidad media de los sentimientos en que se ha expresado los síntomas. Rango de 0 a infinito.
*** Índice promedio de los sentimiento. Se obtiene como producto cartesiano del tipo medio e intensidad media de los sentimientos.

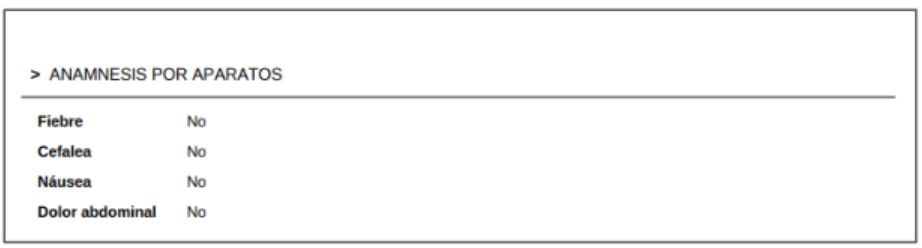
Figure 4.10: Anamnesis Report Example. Page 1 of 3



> INFORMACIÓN DE LOS SÍNTOMAS

SÍNTOMA	**Dolor torácico**	**SNOMED CT ID**	**29857009**
Parámetros			
	Inicio	11-12-2023 | 01:18:15	
	Tipo de dolor	opresivo	
	Patrón temporal	intermitente	
	Interferencia actividades diarias	No	
	Traumatismo	No	
	Relación con los movimientos	igual	
	Palpitaciones	Si	
	Disnea	No	
	Lateralidad	izquierda	
	Fiebre	No	
	Intensidad	4.0	
	Entumecimiento y hormigueo	No	
	Irradiación	región dorsal	
Sentimientos			
	Tipo de sentimiento	-0.6	
	Intensidad del sentimiento	0.6	
	Índice del sentimiento	-0.36	
SÍNTOMA	**Edema en extremidades inferiores**	**SNOMED CT ID**	**449707004**
Parámetros			
	Inicio	16-12-2023 | 12:00:00	
	Predominio diario	peor al avanzar el día	
	Dolor a la palpación o caminando	No	
	Cirugía o traumatismo	No	
	Disminución con las piernas en alto	Si	
	Lateralidad	ambos	
	Fiebre	No	
	Tipo de inicio	gradual	
	Inmovilización reciente	No	
	Dolor en reposo	No	
Sentimientos			
	Tipo de sentimiento	0.1	
	Intensidad del sentimiento	0.1	
	Índice del sentimiento	0.01	

Figure 4.11: Anamnesis Report Example. Page 2 of 3



> ANAMNESIS POR APARATOS

Fiebre	No
Cefalea	No
Náusea	No
Dolor abdominal	No

Figure 4.12: Anamnesis Report Example. Page 3 of 3

CHAPTER 5

Conclusions and Future Work

In this chapter, the conclusions extracted from this project, and the thoughts about future work, will be described.

5.1 Conclusions

The main conclusions arising from the design and development of this project are presented in this section.

The development of this project, Anamcon, constitutes an automated multilingual and highly configurable conversational anamnesis web system capable of providing unlimited and multimodal access as well as generating reliable and correctly structured data that allow the generation of complete and exhaustive reports on the patient's clinical situation. The system as a whole performs remarkably well, correctly capturing user interactions in most of the situations tested.

Anamcon, as the final result of the development, emerges as a simple and sober solution, as well as solvent and versatile. Its development has been achieved through the integration of multiple technologies, from the development of the medical anamnesis itself, through

the multilingual and minimalist frontend, the backend with the web application and the database as microservices and up to the Chatbot itself. This means that, despite its apparent simplicity of operation, it presents a significant complexity inherent to the integration of so many different technologies.

The Anamcon approach, as a technological project, stands as a system capable of becoming a tool with real possibilities to promote the democratization of access to structured and quality medical anamnesis processes. This has been achieved largely by focusing the technological development towards modularity and scalability. On the one hand, the implemented modular and microservices-based architecture allows functional independence and easy component exchange. On the other hand, the ability of the developed anamnesis system to offer multimodal operation, managing both text and audio input and output in a structured and parallel way, improves the usability and adaptability of the system to different clinical scenarios.

The system is not limited to efficiently capturing patient-reported symptoms, but incorporates a diversity of features that enhance the capabilities and scope of the system. First of all, it allows user registration and authentication, as well as session management and configuration. This is crucial to guarantee the privacy and persistence of the information, being able to establish and save the correlation between users and anamnesis. In addition, the system allows the analysis of feelings in the description of symptoms by the patient, that is, not only the symptoms are captured but also the way in which the patient has expressed their presence. Finally, the system can generate basic analytics and graphs of the anamnesis and present them in an organized way both in the frontend and in the report.

If we look at the possibilities of real application, the automated collection of basic anamnesis information, even if it is not complete and requires a medical professional to corroborate and complement the process, could represent a very significant cost saving, even more so if we take into account the enormous frequency of the medical anamnesis process.

5.2 Achieved Goals

This section highlights the main achievements of this project as well as the strengths of the developed system.

1. The present development performs an anamnesis in an automated way, extracting the main symptoms and their characteristics in a structured way, which allows the system

to generate complete and detailed reports.

2. The collected information is stored in a persistent and structured way in a database.

3. The rich configurability of the system offers numerous possibilities for expansion and customization such as the implemented multilingual functionality.

4. A procedure has been implemented for the automated configuration and deployment of the different anamnesis agents based on their start-up configuration.

5. Two anamnesis agents have been modeled, the first one: Anam, performs a general anamnesis and is very complete, with up to 61 symptoms with multiple parameters in each one of them. The second one: Siam, much more simple, performs psychological anamnesis.

5.3 Future Work and Challenges

This section briefly highlights the main challenges and future work that could result from this development. However, it should be emphasized that due to the enormous technical and legislative complexity of digital health systems, as is the case of a medical anamnesis system, as well as the large number of technologies involved, it is possible that the completion of most of the work proposed here could involve enormous costs, although the potential benefits could also be very significant, as discussed in the conclusions section. The proposals for improvement and future developments are presented below, categorized according to the main area of impact:

- In the area of **Chatbot technologies**, the adoption of more robust platforms capable of handling complex conversations and deeper contexts, such as Dialogflow CX [28] or advanced agents from other vendors could provide a huge improvement. In addition, the system would benefit from the sophisticated use of Natural Language Processing (NLP) techniques, leveraging Large Language Models (LLMs) with Prompt Engineering to personalize interactions and adjust the anamnesis flow in the backend according to technical requirements [49, 45]. This would not only allow handling more nuanced conversations, but also capture relevant contextual information, which together could achieve better results.

- In relation to the **user interface**, many potential improvements are identified, such as the incorporation of modules for user account management, as well as for system configuration, and the addition of a section for statistics and grouped anamnesis

reports. The implementation of specific sections for different user profiles could also be beneficial [47]. The development of a mobile application for Android and iOS with the aim of improving the usability and reach of the system should be a priority for future developments, as mobiles have a high penetration relative to PCs, especially in economically depressed areas [37].

- For the purpose of enhancing **interoperability** and to provide effective communication between Electronic Health Records (EHR), the more advanced implementation of standardized systems and vocabularies could be adopted, such as the deployment of a Snowstorm Docker service [27] to dynamically access SNOMED CT terminology. The incorporation of other terminologies, such as ICD-10 [3] for the capture of pathological history and Rx-Norm [15] for the identification of the patient's medication would also be positive. The use of OpenEHR-type archetyping [22] and clinical information transmission standards such as HL7 FHIR [23], in association with the previous standards could bring a major improvement.

- Regarding the **functionality** of the application, the implementation of advanced decision support systems based on Machine Learning algorithms, which could guide about recommended complementary tests and probable diagnoses would represent a significant functionality enhancement if the performance were adequate.

- In relation to **architecture of the anamnesis** itself, more types of anamnesis and more complex algorithms could be developed, in collaboration with clinical professionals, and validated through epidemiological studies to prove the effectiveness of the system [33].

- Regarding the **security and regulatory compliance**, the importance of assessing and adopting cybersecurity standards, as well as the compliance with regulatory standards on medical devices, personal data protection and the use of artificial intelligence systems in the field of healthcare should be also a priority for future developments if intended to become a commercial product in the future [36, 12].

- Finally, as a future perspective, the possibility of more advanced **multimodal integrations** could be considered, such as image or video capture, to improve the collection of information in contexts where mere verbal description may be insufficient.

Despite all the possible future work in relation to the system developed, and the enormous advantages that it could bring, even taking into account the considerable costs that might arise in its development, a major challenge for this and any similar systems developed

will always be to offer a service at a very low, if not zero cost for users, so that the service can be widely democratized and bring usefulness to the greatest possible number of users. This could be achieved through different strategies for monetizing the service, which could range from monetizing the use by healthcare service providers to monetizing the use of the data obtained for studies and research, all in compliance, with the legislation related with data protection.

APPENDIX A

Impact of This Project

This appendix reflects, quantitatively or qualitatively, on the possible social, economic and ethical impact of the project.

A.1 Social Impact

An automated conversational anamnesis system can have a significant social impact by effectively addressing access and time constraints of consultations, language and cultural barriers as well as medical documentation, which can be improved in quality and clarity through more structured recording.

All of this can contribute to improving access to medical care by obtaining detailed and accurate anamnesis in an automated way, which can have a positive impact on people's health, especially in those groups with more complicated access to health services.

A.2 Economic Impact

The implementation of an automated medical anamnesis system such as the one proposed can have important beneficial economic implications for several reasons:

First, it may allow optimizing the time of healthcare professionals, who would only have to review and complete the information collected by the system. This would allow them to focus on more complex and specialized tasks, which in turn could increase their efficiency.

Secondly, it can save the cost associated with interpreters or translators, especially in the case of environments with significant linguistic diversity.

Finally, it may allow for better planning of healthcare resources by providing managers with more detailed and structured information on patients' symptomatology in real time.

A.3 Ethical Impact

The introduction of automated anamnesis proposed by the developed system raises numerous ethical issues that must be adequately addressed.

Firstly, the system must comply with ethical and legal standards to protect the confidentiality of patient data, ensuring the security and privacy of the medical information obtained.

Secondly, the system must ensure equity of access, making the implementation of the system equally accessible to all communities and not discriminating against certain groups or individuals.

Finally, it is important to offer a transparent system in terms of decision making, providing access to the algorithms used to perform the anamnesis, so that both patients and health professionals can understand and trust the data generated and decisions made by the system. This also increases the chances of achieving early detection and correction of errors or biases in the operation of the system.

APPENDIX B

Economic Budget

This appendix details an adequate budget to bring about the project. All the resources required for the development of the current project will be presented, divided into the following sections: hardware resources, software resources and human resources. Finally, a summary of the total estimated monetary cost will be provided.

B.1 Hardware Resources

This section introduces the estimated budget required for the acquisition of the hardware components needed for this project.

The only hardware requirement has been a laptop with the following characteristics:

- **Operating System**: Ubuntu 22.04.3 LTS
- **CPU**: AMD Ryzen 5 4600H @ 3.0 GHz (Turbo Boost up to 4.0 GHz)
- **RAM**: 16 GB DDR4 @ 3200 MHz
- **Storage**: SSD 500 GB
- **GPU**: NVIDIA GeForce GTX 1650 Mobile

The cost of acquiring this computer, and therefore the total cost of hardware components for this project, has been 640€.

B.2 Software Resources

This section covers the economic costs related with the acquisition or licensing of software componets.

All the software technologies used in the development of this project are licensed under open-source licenses and therefore don't have any licensing cost related with their use. Consequently, the total cost of software resources is 0€.

B.3 Human Resources

This sections estimates the economical budget related to personal time.

For this estimation, it is assumed that the whole development has been carried out by a single individual with a professional category of a biomedical engineering student.

Since this project has a workload equivalent to 12 ECTS, and each of these credits corresponds to 30 hours, we obtain a total of 360 hours of work. If we estimate a cost of 10 euros per hour of work, in line with the minimum amounts approved by the External Academic Practices Committee of the UPM for the academic year 2023/24 [26], we obtain a personnel cost of 3600 euros.

B.4 Resources Summary

The final estimated economic cost required for the development of this project amounts to a total of 4240 €.

APPENDIX C

Source Code

In this appendix, the complete source code developed for the implementation of this project is provided. The code is available in an online open repository for ease of review and comprehension.

The repository can be found at **`https://github.com/dbrauq/anamcon`**

Note that the security files relating to Google Cloud project passwords have not been included, so the system will not be fully functional without them. In order to use the anamnesis agents, Google Cloud projects must be created and properly configured to use DialogFlow, obtaining the corresponding private credentials. In addition, the Gmail email account must also be configured to obtain the private credentials needed for the automated sending of emails. Once the different credentials have been obtained, they must be entered in the /src/web_app/secure folder and the configuration files must be adjusted accordingly.

Bibliography

[1] "Conceptos básicos de Dialogflow ES". Dialogflow ES. Google Cloud. Accessed: Aug. 11, 2023. [Online]. Available: https://cloud.google.com/dialogflow/es/docs/basics?hl=es-419.

[2] "Docker overview"'. Docker Documentation. Accessed: Dec. 1, 2023. [Online]. Available: https://docs.docker.com/get-started/overview/.

[3] "Edición electrónica de la CIE-10-ES Diagnósticos". Ministerio de Sanidad. Accessed: Dec. 23, 2023. [Online]. Available: https://eciemaps.mscbs.gob.es/ecieMaps/browser/index_10_mc.html.

[4] "Enviar correo electrónico vía Gmail y SMTP". Recursos Python. Accessed: Sep. 9, 2023. [Online]. Available: https://recursospython.com/guias-y-manuales/enviar-correo-electronico-via-gmail-y-smtp/.

[5] "Gillis, A. S., Botelho, B. (2023, 7 de marzo). What is MongoDB? Features and how it works". TechTarget Definition. Data Management. Accessed: Aug. 19, 2023. [Online]. Available: https://www.techtarget.com/searchdatamanagement/definition/MongoDB.

[6] "Ley 41/2002, de 14 de noviembre, básica reguladora de la autonomía del paciente y de derechos y obligaciones en materia de información y documentación clínica, BOE-A-2002-22188". BOE.es - Agencia Estatal Boletín Oficial del Estado. Accessed: Sep. 9, 2023.[Online]. Available: https://www.boe.es/buscar/act.php?id=BOE-A-2002-22188.

[7] "Matplotlib: Visualization with Python". Matplotlib. Accessed: Dec. 15, 2023. [Online]. Available: https://matplotlib.org/.

[8] "MediaRecorder - Web APIs". Recursos Python. Accessed: Sep. 23, 2023. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/MediaRecorder.

[9] "MVC Framework Introduction". GeeksforGeeks. Accessed on Aug. 11, 2023. [Online]. Available: https://www.geeksforgeeks.org/mvc-framework-introduction/.

[10] "Numpy"'. Numpy. Accessed: Dec. 15, 2023. [Online]. Available: https://numpy.org/.

[11] "Pandas"'. Pandas. Accessed: Dec. 15, 2023. [Online]. Available: https://pandas.pydata.org/.

[12] "Propuesta de marco normativo sobre inteligencia artificial". European Commission. Accessed: Dec. 27, 2023. [Online]. Available: https://digital-strategy.ec.europa.eu/es/policies/regulatory-framework-ai.

[13] "PyMongo 4.6.1 documentation". PyMongo 4.6.1 documentation. Accessed: Aug. 19, 2023. [Online]. Available: https://pymongo.readthedocs.io/en/stable/.

[14] "ReportLab PDF Library. User Guide". ReportLab Version 4.0.4. Accessed: Dec. 2, 2023. [Online]. Available: https://www.reportlab.com/docs/reportlab-userguide.pdf.

[15] "RxNorm". National Library of Medicine. Accessed: Dec. 26, 2023. [Online]. Available: https://www.nlm.nih.gov/research/umls/rxnorm/index.html.

[16] "Sentiment Analysis Result'. Dialogflow ES. Google Cloud. Accessed: Aug. 11, 2023. [Online]. Available: https://cloud.google.com/dialogflow/es/docs/reference/rest/v2/SentimentAnalysisResult.

[17] "Sessions in Flask'. Test-Driven Development, Microservices, Web Development Courses. Accessed: Sep. 8, 2023. [Online]. Available: https://testdriven.io/blog/flask-sessions/.

[18] "SNOMED CT Basics". SNOMEDCT. Accessed: Dec. 16, 2023. [Online]. Available: https://confluence.ihtsdotools.org/display/docstart/4.+snomed+ct+basics.

[19] "The Flask Web Framework: A Beginner's Guide.". CareerFoundry. Accessed: Sep. 8, 2023. [Online]. Available: https://careerfoundry.com/en/blog/web-development/what-is-flask/.

[20] "Wellcome to Flask". Flask Documentation (3.0.x). Accessed: Aug. 12, 2023. [Online]. Available: https://flask.palletsprojects.com/en/3.0.x/.

[21] "What are Microservices?". microservices.io. Accessed: Dec. 1, 2023. [Online]. Available: https://microservices.io/#:~:text=Microservices%20%2D%20also%20known%20as%20the,Organized%20around%20business%20capabilities.

[22] "What is openEHR?". openEHR. Accessed: Dec. 26, 2023. [Online]. Available: https://openehr.org/about/what_is_openehr.

[23] "What is HL7 FHIR?". TIBCO. Accessed: Dec. 26, 2023. [Online]. Available: https://www.tibco.com/reference-center/what-is-hl7-fhir.

[24] "¿Qué es SNOMED CT?". Ministerio de Sanidad. Accessed: Dec. 6, 2023. [Online]. Available: https://sanidad.gob.es/areas/saludDigital/interoperabilidadSemantica/factoriaRecursos/snomedCT/home.htm.

[25] "¿Qué son los Microservicios?". IBM. Accessed: Dec 1, 2023. [Online]. Available: https://www.ibm.com/es-es/topics/microservices.

[26] "Prácticas académicas curriculares". 2023. ETSI de Telecomunicación. Accessed: Dec. 7, 2023. [Online]. Available: https://www.etsit.upm.es/estudios/practicas-en-empresas/para-alumnos/practicas-academicas-curriculares.html.

[27] "SNOWSTORM". 2023. SNOMED International. Accessed: Dec. 23, 2023. [Online]. Available: https://github.com/IHTSDO/snowstorm.

[28] "Dialogflow. Documentación. CX.". 2023. Accessed: Dec 22, 2023. [Online]. Available: https://cloud.google.com/dialogflow/cx/docs/basics?hl=es-419.

[29] A. Ziv A. Shachak, M. Hadas-Dayagi and S. Reis. Primary Care Physicians' Use of an Electronic Medical Record System: A Cognitive Task Analysis. *J. General Internal Medicine*, 24(3):341–348, Jan. 2009. [Online]. Available: https://doi.org/10.1007/s11606-008-0892-6.

[30] E. Adamopoulou and L. Moussiades. " An overview of Chatbot technology". *Artificial Intelligence Applications and Innovations*, 584:373–383, May 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7256567/.

[31] D. Bourke. "Introduction to PyDub". Spoken Language processing in python. Accessed: Sep. 23, 2023. [Online]. Available: https://projector-video-pdf-converter.datacamp.com/17718/chapter3.pdf.

[32] B. Jones E. Mitchell C. S. Kruse, C. Kristof and A. Martinez. "Barriers to Electrrronic Health Record Adoption: a Systematic Literature Review". *J. Med. Syst.*, 40(12), Oct. 2016. [Online]. Available: https://doi.org/10.1007/s10916-016-0628-9.

[33] A. Choi et al. Development of a machine learning-based clinical decision support system to predict clinical deterioration in patients visiting the emergency department. *Scientific Rep.*, 13(1), May 2023. [Online]. Available: https://doi.org/10.1038/s41598-023-35617-3.

[34] G. Irving et al. "International variations in primary care physician consultation time: a systematic review of 67 countries.". *BMJ Open*, 7(10):e017902, Oct. 2017. [Online]. Available: https://doi.org/10.1136/bmjopen-2017-017902.

[35] R. S. Evans. "Electronic Health Records: Then, Now, and in the Future". *Yearbook Med. Inform.*, 25(S 01):S48–S61, Aug. 2016. [Online]. Available: https://doi.org/10.15265/iys-2016-s006.

[36] B. Hirt. "Medical device software (MDSW) under EU MDR and IVDR". 2023. Accessed: Dec. 27, 2023. [Online]. Available: https://decomplix.com/medical-software-mdr/.

[37] J. Howarth. "Internet Traffic from Mobile Devices (Dec 2023)". 2023. Accessed: Dec. 23, 2023. [Online]. Available: https://explodingtopics.com/blog/mobile-internet-traffic.

[38] R. P. C. Kessels. "Patient's Memory for Medical Information". *JRSM*, 96(5):219–222, May 2003. [Online]. Available: https://doi.org/10.1258/jrsm.96.5.219.

[39] H.-J. Kong. Managing Unstructured Big Data in Healthcare System. *Healthcare Inform. Res.*, 25(1):1, 2019. [Online]. Available: https://doi.org/10.4258/hir.2019.25.1.1.

[40] J. Chen L. Kolla and R. B. Parikh. "Time of Clinic Appointment and Serious Illness Communication in Oncology". *Cancer Control*, 30:07327482311704, Apr. 2023. [Online]. Available: https://doi.org/10.1177/10732748231170488.

[41] S. Modi and S. S. Feldman. "The Value of Electronic Health Records Since the Health Information Technology for Economic and Clinical Health Act: Systematic Review". *JMIR Med. Inform.*, 10(9):e37283, Sep. 2022. [Online]. Available: https://doi.org/10.2196/37283.

[42] Noha Nabil. "Building a Multilingual Static Website: A Step-by-Step Guide". Medium. Accessed: Sep. 2, 2023. [Online]. Available: https://www.freecodecamp.org/news/html-css-and-javascript-explained-for-beginners/.

[43] Noguer-Balcells. *Exploración Clínica Práctica*. Elsevier Masson, 2021.

[44] J. M. Overhage and D. McCallie. Physician Time Spent Using the Electronic Health Record During Outpatient Encounters. *Ann. Internal Medicine*, 172(3):169, Jan. 2020. [Online]. Available: https://doi.org/10.7326/m18-3684.

[45] S. Ramlochan. "Improving Large Language Models with Retrieval Augmented Generation: Redefining AI Conversations". 2023. Accessed: Dec. 22, 2023. [Online]. Available: https://promptengineering.org/improving-large-language-models-with-retrieval-augmented-generation/.

[46] J. E. Scofield and J. D. Johnson. "The diminishing precision of memory for time". *Psychonomic Bull. & Rev.*, 29:212–219, Aug. 2021. [Online]. Available: https://doi.org/10.3758/s13423-021-01984-z.

[47] S. Sen. "Case study: Designing roles and permissions". 2021. Accessed: Dec. 22, 2023. [Online]. Available: https://bootcamp.uxdesign.cc/designing-roles-and-permissions-ux-case-study-b1940f5a9aa.

[48] S. Álvarez-Montero L. Cabello-Ballesteros T. Pérez-Santonja, L. Gómez-Paredes and M. T. Mombiela-Muruzabal. Historia clínica electrónica: evolución de la relación médico-paciente en la consulta de Atención Primaria. *SEMERGEN - Medicina Fam.*, 43(3):175–181, Apr. 2017. [Online]. Available: https://doi.org/10.1016/j.semerg.2016.03.022.

[49] R. L. Logan IV-E. Wallace T. Shin, Y. Razeghi and S. Singh. "AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts". 2023. Accessed: Dec. 22, 2023. [Online]. Available: https://arxiv.org/abs/2010.15980.

[50] M. Tai-Seale and T. McGuire. "Time is up: increasing shadow price of time in primary-care office visits. Health Economics". *Health Econ.*, 21(4):457–476, Mar. 2011. [Online]. Available: https://doi.org/10.1002/hec.1726.

[51] K. Ubah. "Learn Web Development Basics – HTML, CSS, and JavaScript Explained for Beginners". Free Code Camp. Accessed: Aug. 26, 2023. [Online]. Available: https://www.freecodecamp.org/news/html-css-and-javascript-explained-for-beginners/.

[52] M. Nübling W. A. Langewitz, Y. Loeb and S. Hunziker. "From patient talk to physician notes—Comparing the content of medical interviews with medical records in a sample of outpatients in Internal Medicine". *Patient Educ. Counseling*, 76(3):336–340, Sep. 2009. [Online]. Available: https://doi.org/10.1016/j.pec.2009.05.008.

[53] Lippincott Williams & Wilkins. *"Manual básico de Signos y síntomas."*. Ovid Technologies, 2016.

[54] C. Zegers and M. Auron. Addressing the Challenges of Cross-Cultural Communication. *Med. Clinics North America*, May 2022. [Online]. Available: https://doi.org/10.1016/j.mcna.2022.02.006.