



**UNL**

Universidad  
Nacional  
de Loja

## Universidad Nacional de Loja

### Facultad de la Energía, las Industrias y los Recursos Naturales No Renovables

#### Carrera de Ingeniería en Sistemas

**Sistema de información sobre Covid-19 mediante llamadas  
telefónicas con la ayuda de un agente inteligente**

Trabajo de Titulación previo a  
la obtención del título de  
Ingeniero en Sistemas

**AUTOR:**

Sandro Michael Córdova Carrión

**DIRECTOR:**

Ing. José Oswaldo Guamán Quinche, Mg.Sc.

Loja - Ecuador

2023

*Educamos para Transformar*

## **Certificación**

Ing. José Oswaldo Guamán Quinche, Mg. Sc.

### **DIRECTOR DEL TRABAJO DE TITULACIÓN**

#### **Certifico:**

Que he revisado y orientado todo el proceso de elaboración del Trabajo de Titulación denominado: "**Sistema de información sobre covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente**", previo a la obtención del título de **Ingeniero en Sistemas** de autoría del estudiante **Sandro Michael Córdova Carrión** con **cédula de identidad: 1150261905**, una vez que el trabajo cumple con todos los requisitos exigidos por la Universidad Nacional de Loja, para el efecto, autorizo la presentación del mismo para su respectiva sustentación y defensa.

Loja, 13 de febrero del 2023.

---

Ing. José Oswaldo Guamán Quinche, Mg. Sc.

### **DIRECTOR DEL TRABAJO DE TITULACIÓN**

### Autoría

Yo, **Sandro Michael Córdova Carrión**, declaro ser autor del presente Trabajo de Titulación y eximo expresamente a la Universidad Nacional de Loja y a sus representantes jurídicos, de posibles reclamos o acciones legales que se puedan ocasionar por el contenido del mismo. Adicionalmente acepto y autorizo a la Universidad Nacional de Loja, la publicación de mi Trabajo de Titulación en el Repositorio Digital Institucional – Biblioteca Virtual.



**Firma:**

**Cédula de identidad:** 1150261905

**Fecha:** 16/03/2023

**Correo electrónico:** [sandro.cordova@unl.edu.ec](mailto:sandro.cordova@unl.edu.ec)

**Teléfono:** +593 996 484 221

**Carta de autorización por parte del autor, para consulta, producción parcial o total, y/o publicación electrónica del texto completo, del Trabajo de Titulación.**

Yo **Sandro Michael Córdova Carrión**, declaró ser el autor del Trabajo de Titulación denominado: **Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente**, de la autoría del estudiante **Sandro Michael Córdova Carrión**, con **cédula de identidad Nro.1150261905**, como requisito para optar por el título de **Ingeniero en Sistemas**, autorizo al sistema Bibliotecario de la Universidad Nacional de Loja para que, con fines académicos, muestre la producción intelectual de la Universidad, a través de la visibilidad de su contenido en el Repositorio Institucional.

Los usuarios pueden consultar el contenido de este trabajo en el Repositorio Institucional, en las redes de información del país y del exterior con las cuales tenga convenio la Universidad.

La Universidad Nacional de Loja, no se responsabiliza por el plagio o copia del Trabajo de Titulación que realice un tercero.

Para constancia de esta autorización, en la ciudad de Loja, a los dieciséis días del mes de marzo del dos mil veintitrés.



**Firma:**

**Autor:** Sandro Michael Córdova Carrión

**Cédula:** 1150261905

**Correo Electrónico:** [sandro.cordova@unl.edu.ec](mailto:sandro.cordova@unl.edu.ec)

**Teléfono:** +593 996 484 221

**DATOS COMPLEMENTARIOS:**

**Director del Trabajo de Titulación:** Ing. José Oswaldo Guamán Quinche, Mg.Sc.

## **Dedicatoria**

A mi padre, por brindarme su soporte desde el primer día de mi vida , su apoyo moral, consejos y educación han sido de los mejores. A mis hermanos por haberme apoyado a lo largo de toda mi carrera universitaria. A todas las personas especiales que me acompañaron en esta etapa, aportando a mi formación tanto personal como profesional.

***Sandro Michael Córdova Carrión***

## **Agradecimiento**

A mis padres por ser el pilar fundamental y haberme apoyado incondicionalmente, pese a las adversidades e inconvenientes presentadas. Agradezco a mi director del Trabajo de Titulación, quien con su experiencia y conocimiento me orientó en la investigación. Agradezco a los todos docentes que, con su sabiduría, conocimiento y apoyo, motivaron a desarrollarme como persona y profesional en la Universidad Nacional de Loja.

***Sandro Michael Córdova Carrión***

## Índice de contenidos

<b>Portada</b> .....	i
<b>Certificación</b> .....	ii
<b>Autoría</b> .....	iii
<b>Carta de autorización</b> .....	iv
<b>Dedicatoria</b> .....	v
<b>Agradecimiento</b> .....	vi
<b>Índice de contenidos</b> .....	vii
Índice de tablas .....	xi
Índice de figuras.....	xiii
Índice de Anexos.....	xvi
<b>1. Título</b> .....	1
<b>2. Resumen</b> .....	2
2.1. Abstract.....	3
<b>3. Introducción</b> .....	4
<b>4. Marco teórico</b> .....	5
4.1. Inteligencia artificial .....	5
4.2. Aprendizaje automático.....	5
4.3. Sistema experto .....	5
4.4. Sistema experto y agente inteligente.....	6
4.5. Agente inteligente .....	6
4.5.1. Características de los agentes .....	6
4.5.2. Clases de agentes inteligentes.....	6
4.5.3. Tipos de arquitecturas de agentes .....	7
4.6. Sistemas multiagente.....	7
4.7. Java.....	8
4.8. JADE .....	8
4.9. SPEECH.....	8
4.10. MongoDB.....	9
4.11. TestNG .....	9
4.12. Metodologías de desarrollo de software .....	9
4.13. Selección de la metodología de desarrollo de software .....	10
4.14. Selección de la metodología de desarrollo ágil .....	10
4.15. Modelo de desarrollo SCRUM .....	11
4.16. Metodología de desarrollo Kendall y Kendall.....	12

4.17. Muestreo no probabilístico por conveniencia .....	13
4.18. Identificación de pacientes con Covid-19 .....	14
4.19. Medidas de bioseguridad .....	14
4.20. Trabajos relacionados.....	15
<b>5. Metodología .....</b>	<b>16</b>
5.1. Metodología de desarrollo de software.....	16
5.2. Proceso.....	16
5.2.1. Objetivo 1: Diseñar el módulo de software del agente inteligente para el sistema de llamadas telefónicas utilizando el lenguaje de programación Java y la metodología de desarrollo Kendall y Kendall. ....	16
5.2.2. Objetivo 2: Implementar el módulo de software del agente inteligente al sistema de llamadas telefónicas. ....	17
5.2.3. Objetivo 3: Evaluar la aceptación y correcto funcionamiento del módulo de software en el sistema de llamadas telefónicas.....	17
5.3. Métodos .....	17
5.3.1. Método Analítico .....	17
5.3.2. Método Científico .....	18
5.3.3. Método inductivo-deductivo.....	18
5.3.4. Estudio de Caso .....	18
5.4. Técnicas.....	18
5.4.1. Muestreo no probabilístico por conveniencia .....	18
5.4.2. Entrevistas .....	18
5.4.3. Encuestas .....	18
5.4.4. Prototipado.....	19
5.5. Estándares.....	19
5.5.1. IEEE 830.....	19
5.6. Materiales .....	19
5.6.1. Recursos Académicos .....	19
5.6.2. Recursos humanos .....	19
5.6.3. Recursos de software y hardware.....	20
<b>6. Resultados.....</b>	<b>21</b>
6.1. Objetivo I: Diseñar el módulo de software del agente inteligente para el sistema de llamadas telefónicas utilizando el lenguaje de programación Java y la metodología de desarrollo Kendall y Kendall. ....	21
6.1.1. Definir el grupo objetivo que utilizará el módulo de software .....	22

6.1.2. Establecer los actores .....	22
6.1.3. Selección de la muestra a entrevistar.....	22
6.1.4. Historias de usuario .....	23
6.1.5. Requisitos funcionales .....	24
6.1.6. Requisitos no funcionales .....	26
6.1.7. Diagrama de casos de uso.....	28
6.1.8. Actores del sistema.....	28
6.1.9. Especificación de casos de uso .....	29
6.1.10. Diagrama de actividades.....	31
6.1.11. Diagrama de general de la arquitectura de software .....	34
6.1.12. Representación de la arquitectura.....	35
Diagrama de clases .....	36
Diagrama de paquetes.....	36
Diagrama de despliegue .....	37
Persistencia de la base de datos .....	37
6.1.13. Agente inteligente: Representación de la arquitectura.....	37
6.1.14. Iteraciones (Sprints) .....	44
6.1.15. Control de versiones .....	44
6.1.16. Estructura del módulo de software .....	45
6.1.17. Codificación del módulo de software .....	46
6.2. Objetivo II. Implementar el módulo de software del agente inteligente al sistema de llamadas telefónicas .....	66
6.2.1. Costos de implementación de un centro de llamadas telefónicas.....	66
6.2.2. Elementos y librerías a implantar .....	67
6.2.3. Implementación del módulo de software .....	69
6.2.4. Pruebas unitarias .....	71
6.3. Objetivo III: Evaluar la aceptación y correcto funcionamiento del módulo de software en el sistema de llamadas telefónicas .....	79
6.3.1. Funcionamiento del módulo de software .....	80
6.3.2. Pruebas de aceptación a usuarios finales .....	83
6.3.3. Pruebas de aceptación a expertos en medicina .....	87
<b>7. Discusión.....</b>	<b>93</b>
<b>8. Conclusiones.....</b>	<b>98</b>
<b>9. Recomendaciones.....</b>	<b>99</b>
<b>10. Bibliografía .....</b>	<b>100</b>

<b>11. Anexos.....</b>	<b>103</b>
------------------------	------------

## Índice de tablas:

<b>Tabla 1.</b> Metodologías de desarrollo de software .....	10
<b>Tabla 2.</b> Metodologías de desarrollo de software ágiles.....	11
<b>Tabla 3.</b> Trabajos relacionados .....	15
<b>Tabla 4.</b> Recursos humanos .....	19
<b>Tabla 5.</b> Recursos de software.....	20
<b>Tabla 6.</b> Recursos de hardware .....	20
<b>Tabla 7.</b> Datos para el cálculo de la muestra .....	22
<b>Tabla 8.</b> Historias de usuario.....	24
<b>Tabla 9.</b> Requisitos funcionales .....	26
<b>Tabla 10.</b> Requisitos no funcionales.....	27
<b>Tabla 11.</b> Actor del sistema 01 .....	28
<b>Tabla 12.</b> Actor del sistema 02.....	29
<b>Tabla 13.</b> Especificación de casos de uso.....	30
<b>Tabla 14.</b> Arquitectura 4+1 .....	35
<b>Tabla 15.</b> Clasificación de pacientes .....	42
<b>Tabla 16.</b> Iteración 1 .....	44
<b>Tabla 17.</b> Iteración 2 .....	44
<b>Tabla 18.</b> Iteración 3 .....	44
<b>Tabla 19.</b> Iteración 4 .....	44
<b>Tabla 20.</b> Detalles de la estructura general .....	46
<b>Tabla 21.</b> Costos de implementación de un centro de llamadas telefónicas.....	67
<b>Tabla 22.</b> Requerimientos y características.....	68
<b>Tabla 23.</b> Librerías y dependencias .....	68
<b>Tabla 24.</b> Lista de módulos .....	74
<b>Tabla 25.</b> Lista de correcciones aplicadas.....	78
<b>Tabla 26.</b> Lista de preguntas aplicadas.....	83
<b>Tabla 27.</b> Resultados pregunta 1 .....	84
<b>Tabla 28.</b> Resultados pregunta 2 .....	84
<b>Tabla 29.</b> Resultados pregunta 3 .....	84
<b>Tabla 30.</b> Resultados pregunta 4 .....	85
<b>Tabla 31.</b> Resultados pregunta 5 .....	85
<b>Tabla 32.</b> Resultados pregunta 6 .....	85
<b>Tabla 33.</b> Resultados pregunta 7 .....	85

<b>Tabla 34.</b> Resultados pregunta 8 .....	85
<b>Tabla 35.</b> Resultados pregunta 9 .....	85
<b>Tabla 36.</b> Lista de encuestados .....	87
<b>Tabla 37.</b> Lista de preguntas aplicadas.....	88
<b>Tabla 38.</b> Resultados pregunta 1 .....	89
<b>Tabla 39.</b> Resultados pregunta 2 .....	89
<b>Tabla 40.</b> Resultados pregunta 3 .....	89
<b>Tabla 41.</b> Resultados pregunta 4 .....	90
<b>Tabla 42.</b> Resultados pregunta 5 .....	90
<b>Tabla 43.</b> Resultados pregunta 6 .....	90
<b>Tabla 44.</b> Resultados pregunta 7 .....	90
<b>Tabla 45.</b> Resultados pregunta 8 .....	90
<b>Tabla 46.</b> Resultados pregunta 9 .....	91
<b>Tabla 47.</b> Resultados pregunta 10 .....	91
<b>Tabla 48.</b> Resultados pregunta 11 .....	91
<b>Tabla 49.</b> Resultados pregunta 12 .....	91

## Índice de figuras:

<b>Figura 1.</b> Diagrama de casos de uso .....	28
<b>Figura 2.</b> Diagrama de actividades, transferir llamada. ....	32
<b>Figura 3.</b> Diagrama de actividades, listar síntomas.....	33
<b>Figura 4.</b> Diagrama general del módulo de software en un centro de llamadas telefónicas.	34
<b>Figura 5.</b> Diagrama general del módulo de software utilizado.....	35
<b>Figura 6.</b> Diagrama de clases.....	36
<b>Figura 7.</b> Diagrama de paquetes. ....	36
<b>Figura 8.</b> Diagrama de despliegue.....	37
<b>Figura 9.</b> Colecciones de la base de datos. ....	37
<b>Figura 10.</b> Arquitectura JADE  .....	38
<b>Figura 11.</b> Arquitectura JADE usada.....	38
<b>Figura 12.</b> Estrategia de procesamiento .....	39
<b>Figura 13.</b> Estrategia de procesamiento usada.....	39
<b>Figura 14.</b> Flujograma del comportamiento del agente .....	40
<b>Figura 15.</b> Flujo del Proceso de entrenamiento .....	42
<b>Figura 16.</b> Flujo del Proceso de entrenamiento .....	43
<b>Figura 17.</b> Control de versiones en GitHub. ....	45
<b>Figura 18.</b> Estructura general. ....	46
<b>Figura 19.</b> Selección de “Opción 1”.....	48
<b>Figura 20.</b> Selección del usuario “Opción 1”. ....	48
<b>Figura 21.</b> Método para transferir llamada. ....	49
<b>Figura 22.</b> Selección de “Opción 2”.....	49
<b>Figura 23.</b> Agente inteligente.....	50
<b>Figura 24.</b> GUI de JADE .....	50
<b>Figura 25.</b> One Setup .....	51
<b>Figura 26.</b> One Shot Behaviour .....	51
<b>Figura 27.</b> Solicitud de datos al paciente .....	52
<b>Figura 28.</b> Procesamiento de los datos del paciente.....	52
<b>Figura 29.</b> Almacenamiento de los datos del paciente .....	53
<b>Figura 30.</b> Solicitud de síntomas al paciente.....	53
<b>Figura 31.</b> Procesamiento de los datos del paciente.....	54
<b>Figura 32.</b> Extracción de los síntomas de referencia. ....	55
<b>Figura 33.</b> Estableciendo prioridad respecto a los síntomas de referencia.....	55

<b>Figura 34.</b> Clasificación del paciente .....	56
<b>Figura 35.</b> Medidas de bioseguridad.....	56
<b>Figura 36.</b> Método medidasBioseguridad() .....	57
<b>Figura 37.</b> Cita generada .....	57
<b>Figura 38.</b> Cita generada .....	58
<b>Figura 39.</b> Medicina en caso de emergencia .....	59
<b>Figura 40.</b> Método recetar medicina .....	59
<b>Figura 41.</b> Vista “Generar reporte de pacientes”.....	60
<b>Figura 42.</b> Método buscarPacientes() .....	60
<b>Figura 43.</b> Vista “Generar reporte de citas”.....	61
<b>Figura 44.</b> Método buscarCitas () .....	61
<b>Figura 45.</b> Vista “actualizar/eliminar cita” .....	62
<b>Figura 46.</b> Método actualizarCita().....	62
<b>Figura 47.</b> Método eliminarCita() .....	62
<b>Figura 48.</b> Entrenamiento del agente (numeros).....	63
<b>Figura 49.</b> Entrenamiento del agente (símbolos) .....	63
<b>Figura 50.</b> Almacenamiento de los números/símbolos.....	64
<b>Figura 51.</b> Entrenamiento del agente (dominios web) .....	64
<b>Figura 52.</b> Almacenamiento de los dominios .....	64
<b>Figura 53.</b> Llamada al método identificador de números.....	65
<b>Figura 54.</b> Llamada al método identificador de dominios .....	65
<b>Figura 55.</b> Método buscarNumeros() .....	65
<b>Figura 56.</b> Método buscarSimbolos() .....	65
<b>Figura 57.</b> Método identificador de dominios .....	66
<b>Figura 58.</b> Instalación de Java.....	69
<b>Figura 59.</b> Instalación de la base de datos Mongo DB.....	69
<b>Figura 60.</b> JDK y librerías. ....	70
<b>Figura 61.</b> Instalación del módulo de software.....	70
<b>Figura 62.</b> Pantalla principal del módulo de software.....	71
<b>Figura 63.</b> Prueba unitaria al método principal.....	75
<b>Figura 64.</b> Resultado de prueba unitaria al método principal. ....	75
<b>Figura 65.</b> Prueba unitaria al método hay_llamada.....	75
<b>Figura 66.</b> Resultado de prueba unitaria al método hay_llamada .....	76
<b>Figura 67.</b> Prueba unitaria al método transferir_llamada.....	76
<b>Figura 68.</b> Resultado de prueba unitaria al método transferir_llamada .....	76

<b>Figura 69.</b> Prueba unitaria al método leer .....	76
<b>Figura 70.</b> Resultado de prueba unitaria al método leer.....	77
<b>Figura 71.</b> Ingreso de datos personales del usuario. ....	80
<b>Figura 72.</b> Listar síntomas del usuario. ....	80
<b>Figura 73.</b> Cita generada. ....	81
<b>Figura 74.</b> Medidas de bioseguridad.....	81
<b>Figura 75.</b> Recetar en caso de emergencia. ....	82

## **Índice de Anexos:**

<b>Anexo 1.</b> Especificación de requisitos de software IEEE 830.....	103
<b>Anexo 2.</b> Informe de entrevistas a centros médicos.....	114
<b>Anexo 3.</b> Arquitectura de software.....	133
<b>Anexo 4.</b> Desarrollo y documentación .....	149
<b>Anexo 5.</b> Plan de pruebas Unitarias.....	202
<b>Anexo 6.</b> Plan de pruebas funcionales.....	207
<b>Anexo 7.</b> Manual de usuario .....	219
<b>Anexo 8.</b> Manual de despliegue para desarrolladores .....	228
<b>Anexo 9.</b> Informe de encuesta de síntomas.....	237
<b>Anexo 10.</b> Pruebas de aceptación a expertos.....	249
<b>Anexo 11.</b> Pruebas de aceptación a usuarios finales.....	266
<b>Anexo 12.</b> Solicitud de implementación del módulo de software .....	280
<b>Anexo 13.</b> Certificado de traducción del resumen.....	284

## **1. Título**

**Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda  
de un agente inteligente**

## 2. Resumen

El gran número de contagios por Covid-19 provocó la saturación de los medios brindados por los diferentes centros de atención médica a sus pacientes, exponiendo así la existencia de falencias en el proceso de atención de los mismos.

Partiendo de esta problemática el presente trabajo de titulación pretende dar respuesta a la pregunta de investigación: ¿El uso de agentes inteligentes ayuda en el proceso de atención médica mediante llamadas telefónicas en un centro médico, considerando la saturación de los centros de salud debido al estado de emergencia sanitaria por Covid-19? y tiene por objetivo desarrollar un prototipo funcional de un agente inteligente que simule la gestión de llamadas telefónicas en un centro médico, mediante un módulo de software encargado de identificar y clasificar a los pacientes.

Para ello se utilizó la metodología ágil Kendall y Kendall que dio como resultado tres secciones principales que corresponden a los objetivos específicos planteados. La primera sección abarca las cinco primeras fases de la metodología, que dio como resultado la identificación de los problemas, oportunidades y objetivos, la lista de los requerimientos, análisis de las necesidades del sistema, el diseño del sistema recomendado y el desarrollo del módulo de software. La segunda sección contiene la sexta fase de la metodología, aquí se obtuvo el manual de despliegue y los resultados de las pruebas unitarias. En la tercera sección se ejecuta la fase de implantación y evaluación del sistema, en donde se obtuvo el manual de usuario, resultado de las pruebas funcionales y el resultado de las pruebas de aceptación.

Para finalizar, el módulo de software desarrollado permite la clasificación e identificación de pacientes con Covid-19 de forma correcta, eficiente y confiable mejorando así la atención médica brindada al paciente.

**Palabras claves:** Aplicación informática, centro médico, Covid-19, agente inteligente, inteligencia artificial, sistema experto.

## 2.1. Abstract

The large number of Covid-19 infections led to the saturation of the means of care provided by the different health care centers to their patients, thus exposing the existence of shortcomings in the care process.

Based on this problem, this degree work aims to answer the research question: Does the use of intelligent agents help in the process of medical care through telephone calls in a medical center, considering the saturation of health centers due to the state of health emergency by Covid-19? and aims to develop a functional prototype of an intelligent agent that simulates the management of telephone calls in a medical center, through a software module responsible for identifying and classifying patients.

For this purpose, the Kendall and Kendall agile methodology was used, resulting in three main sections corresponding to the specific objectives set forth. The first section covers the first five phases of the methodology, which resulted in the identification of the problems, opportunities and objectives, the list of requirements, the analysis of the system needs, the recommended system design and the development of the software module. The second section contains the sixth phase of the methodology, where the deployment manual and the results of the unit tests were obtained. In the third section the implementation and evaluation phase of the system is executed, where the user manual, functional test results and the results of the acceptance tests were obtained.

Finally, the software module developed allows the correct, efficient and reliable classification and identification of patients with Covid-19, thus improving the medical care provided to the patient.

Keywords: Communication process, computer application, health, medical center, patient.

### **3. Introducción**

De acuerdo con los datos presentados en el “Sistema nacional de vigilancia en salud pública del Ecuador” en el documento “Consenso multidisciplinario informado en la evidencia sobre el tratamiento de Covid-19” [1], se expone que existe un gran número de contagios a nivel nacional, en donde las personas entre 20 a 49 años de edad son el grupo de mayor incidencia con el 60% del total de los casos de Covid-19. Además, según lo presentado en la “Gaceta epidemiológica semanal No. 34” proporcionada por el ministerio de salud pública de la República del Ecuador [2], de los presuntos casos de Covid-19 analizados, el 67% han sido confirmados, el 30% han sido descartados y el 3% han sido no concluyentes.

Por lo tanto, es importante atender el mayor número de casos posibles para que estos puedan ser descartados o confirmados de forma oportuna, permitiendo así controlar la taza de contagios y por ende reduciendo la saturación de los centros de atención médica.

El Ministerio de Salud Pública (MSP) del Ecuador es consciente de ello y presenta a los ciudadanos la línea telefónica “171” y cuatro medios telemáticos (WhatsApp, página web, Facebook Messenger y aplicativo móvil) para el agendamiento de citas médicas [3], esto para facilitar la atención médica a cada uno de los ciudadanos que lo necesiten.

Para aportar a la atención médica brindada por el MSP se plantea el desarrollo de un módulo de software que se encargue de identificar y clasificar pacientes con Covid-19, con la finalidad de ayudar a reducir la saturación de los centros de atención médica al permitir que los pacientes reciban información sobre las medidas de bioseguridad, generen una cita médica o recetarle medicina de emergencia si es que lo requiere.

El Trabajo de Titulación surge de la pregunta de investigación ¿El uso de agentes inteligentes ayuda en el proceso de atención médica mediante llamadas telefónicas en un centro médico, considerando la saturación de los centros de salud debido al estado de emergencia sanitaria por Covid-19?, tiene como objetivo general el “Desarrollar un prototipo funcional de un agente inteligente que simule la gestión de llamadas telefónicas en un centro médico”, y para cumplirlo se establecen tres objetivos específicos: 1. Diseñar el módulo de software del agente inteligente para el sistema de llamadas telefónicas utilizando el lenguaje de programación Java y la metodología de desarrollo Kendall y Kendall, 2. Implementar el módulo de software del agente inteligente al sistema de llamadas telefónicas y 3. Evaluar la aceptación y correcto funcionamiento del módulo de software en el sistema de llamadas telefónicas.

## **4. Marco teórico**

### **4.1. Inteligencia artificial**

La inteligencia artificial (IA) se refiere a sistemas que presentan comportamientos inteligentes, haciendo uso de algoritmos para analizar su entorno, aprender de los datos y utilizarlos en la toma de decisiones con cierto grado de autonomía como lo haría un ser humano. Los sistemas basados en la IA son utilizados para ayudar a las personas, beneficiarse de mejoras significativas y disfrutar de una mayor eficiencia en los diferentes subcampos que abarca en la actualidad [4].

### **4.2. Aprendizaje automático**

El aprendizaje automático o también conocido como machine learning, es uno de los principales enfoques que tiene la inteligencia artificial. Se trata de un aspecto de la informática, donde los ordenadores se caracterizan por la capacidad de aprender mediante algoritmos para la detección de patrones y uso del conocimiento adquirido en la toma decisiones [5].

Existen tres tipos de aprendizaje automático: aprendizaje supervisado, aprendizaje no supervisado y el aprendizaje de refuerzo.

**Aprendizaje supervisado:** Basado en tareas, se hace uso de datos que ya han sido etiquetados y organizados para indicar cómo será categorizada la nueva información. Aquí se hace uso de la intervención humana para poder recibir control y retroalimentación.

**Aprendizaje no supervisado:** Basado en datos, no se hace uso de datos etiquetados u organizados, aquí los algoritmos deben encontrar la manera de clasificar los datos y aprender de los resultados ya que no hay intervención humana.

**Aprendizaje de refuerzo:** Aprende a reaccionar en su entorno, aquí los algoritmos son recompensados por sus aciertos y aprenden de ellos cada vez que aciertan, aquí la intervención humana se da de forma parcial.

### **4.3. Sistema experto**

Los sistemas expertos son considerados como un subconjunto de la inteligencia artificial. Un sistema experto emplea el conocimiento humano para resolver problemas que normalmente requieren la intervención de expertos humanos. Los sistemas expertos imitan el proceso de razonamiento que los expertos humanos utilizan para resolver problemas específicos, a su vez estos sistemas pueden ser empleados por no expertos para mejorar sus habilidades en la resolución de problemas [6].

#### **4.4. Sistema experto y agente inteligente**

Para un sistema experto no es posible aprender de sus errores ni de errores ajenos. No son capaces de distinguir cuáles son los factores relevantes de un problema y separarlos de factores secundarios. Los sistemas expertos carecen de sentido común, no interactúan con su entorno ni cooperan entre si ya que estos son diseñados para tareas complejas en donde solo ellos tienen el control.

Por otro lado, un agente inteligente hace uso del aprendizaje automático para mejorar sus falencias, aquí el usuario interactúa directamente con el agente ya que este hace el papel de consultante y a su vez tiene la capacidad de comunicarse con otros agentes para poder cumplir sus tareas u objetivos.

#### **4.5. Agente inteligente**

Un agente inteligente se puede considerar una entidad capaz de percibir su entorno, tener autonomía y retroalimentar su conocimiento. Este se encarga de percibir el nivel de inteligencia y las dificultades que presenta el usuario, obteniendo un perfil de este acorde a lo que percibe, con la finalidad de poder brindar soluciones a las necesidades del individuo [7].

##### **4.5.1. Características de los agentes**

Los agentes inteligentes deben cumplir con determinadas características y estas tienen relevancia de acuerdo al ambiente en que se desarrolla. Las siguientes características de fueron obtenidas de [7].

- **Reactivo:** el agente debe ser capaz de responder a cambios en el entorno donde se encuentra.
- **Proactivo:** el agente siempre debe tratar de cumplir su propio plan u objetivo.
- **Social:** el agente debe poder comunicarse con otros agentes por medio de cualquier lenguaje de comunicación entre agentes.

##### **4.5.2. Clases de agentes inteligentes**

Los agentes se pueden clasificar de acuerdo a la función que ejecutan y a las propiedades que tienen [7].

- **Agentes de interfaz:** estos tienen la capacidad de adaptarse a la forma de aprendizaje del usuario y sirve de asistente en la comprensión de nuevos temas.

- **Agentes móviles:** procesos computacionales capaces de viajar por nodos de una red, por computadoras, navegar por internet y efectuar intercambios de información.
- **Agentes de información:** sirven de ayuda para mejorar la recopilación, manipulación y el control de la información para dar una respuesta al usuario.

#### **4.5.3. Tipos de arquitecturas de agentes**

##### **Arquitecturas deliberativas**

Su enfoque es tradicional y se basa en el lema “pensar mucho, luego actuar”, aquí las decisiones se obtienen mediante el razonamiento lógico. La información recolectada es fusionada y transformada en símbolos para tener una representación interna del estado del ambiente, las acciones del agente y sus objetivos.

##### **Arquitecturas reactivas**

Se basa en la filosofía “no pensar, actuar”, inspirada en el modelo estímulo-respuesta el comportamiento del agente emerge de la interacción entre el agente y su ambiente. El sistema es descompuesto en módulos de comportamientos orientados a tareas específicas acorde al problema en donde cada módulo es conectado a sus sensores y efectores.

##### **Arquitecturas hibridas**

Combina el tiempo de respuesta de la arquitectura reactiva con el razonamiento de la arquitectura deliberativa, su lema es “pensar y actuar independientemente, en paralelo”. Su capa reactiva es de rápida reacción y tiene comportamientos primitivos y su capa intermedia ejecuta los planes elaborados en la capa deliberativa.

#### **4.6. Sistemas multiagente**

El desarrollo de los agentes no siempre se da de forma individual, existen casos donde es necesario desarrollar diversos agentes con roles definidos para optimizar y facilitar su funcionamiento, esto se denomina como sistema multiagente y el funcionamiento no depende solamente de un agente, sino de varios que interactúan entre sí, con el fin de comunicarse, consultar información relevante entre ellos y establecer su rol dentro del sistema [8].

Los sistemas multiagente son diseñados teniendo en cuenta una estructura que permita una operación efectiva, además de que puedan interactuar de forma productiva entre ellos, dicha estructura debe contemplar todos los aspectos relacionados con los procesos de comunicación entre agentes, refiriéndose al lenguaje que emplean los agentes para el envío y recepción de los mensajes, además de permitir el intercambio de información a un alto nivel.

#### **4.7. Java**

Java es un lenguaje de programación en donde se ha desarrollado un sin número de aplicaciones, actualmente Java sigue siendo indispensable en dispositivos electrónicos debido a la cantidad de aplicaciones desarrolladas con este lenguaje. Java es muy valorado debido a que sus programas se pueden ejecutar en diversas plataformas con diversos sistemas operativos como Windows, Mac o Linux.

Los programas en Java se compilan a un nivel intermedio, este código es interpretado por la máquina virtual de Java (JVM) del entorno de ejecución (JRE) y se consigue la portabilidad ya que es independiente de la plataforma o del sistema operativo. La portabilidad que ofrece java es una de las principales razones por las cuales muchas empresas han preferido optar por este lenguaje para el desarrollo de aplicaciones [9].

#### **4.8. JADE**

El framework de desarrollo de agentes en Java (JADE) está desarrollado en el lenguaje de programación Java y es presentado a la comunidad por TILAB (Telcom Italia Lab), este se enfoca principalmente en la implementación de sistemas multiagente.

Está regido a los estándares de sistemas basados en agentes FIPA (Fundation for intelligent physical agents), esto con el fin de obtener interoperabilidad entre los sistemas basados en agentes.

Entre las principales ventajas de JADE se encuentra la portabilidad y movilidad que ofrece para los agentes, así como su comunicación la cual se lleva a cabo mediante mensajes asíncronos cuya estructura se basa en el lenguaje ACL (Agent communication language), lo que le permite la comunicación entre agentes de diferentes plataformas mediante su API (Application programming interface), siempre y cuando estos agentes cumplan con los estándares FIPA [10].

Además, JADE facilita la asignación de comportamientos preestablecidos en su entorno de desarrollo (behaviours), permite aplicar cambios en tiempo de ejecución y proporciona una interfaz gráfica de usuario (GUI) para el control y manipulación de agentes.

#### **4.9. SPEECH**

La API de Java Speech (JSAPI) extiende de la plataforma de Java, es un paquete de clases escritos en Java utilizado para el reconocimiento y síntesis de voz. La primera versión de Java Speech fue lanzada por Sun en 1998 y actualmente se trabaja con la versión 2.0.

El objetivo de speech es proveer soporte para el reconocimiento de voz, ser simple, compacto y fácil de utilizar. Las aplicaciones sobre las que se usa Speech se implementan generalmente sobre hardware dedicado capaz de soportar un gran número de conexiones simultáneas ya que los sintetizadores de voz requieren de único canal de salida de audio y los reconocedores de voz requieren de un único canal de entrada [11].

#### **4.10. MongoDB**

MongoDB es un sistema de administración de bases de datos NoSQL, el modelo de datos y las estrategias de persistencia planteadas por MongoDB son construidas para un alto rendimiento de lectura y escritura con la finalidad de escalar fácilmente.

MongoDB se basa en JSON(JavaScript Object Notation), un esquema popular para almacenar estructuras de datos arbitrarias en donde las estructuras JSON consisten en claves y valores que pueden anidarse arbitrariamente. Un modelo de datos basado en documentos JSON puede representar estructuras de datos ricas y jerárquicas.

MongoDB tiene el mismo concepto de cualquier base de datos: Dentro de una instancia de MongoDB puede tener cero o más bases de datos, cada una actuando a un alto nivel. Una base de datos puede tener cero o más colecciones y puede considerarse como una tabla tradicional. Las colecciones se componen de cero o más documentos y un documento puede considerarse como una fila. Un documento se compone de uno o más campos y se puede considerar como una columna [12].

#### **4.11. TestNG**

Considerado como framework para pruebas unitarias y basado en la librería JUnit de Java, tiene gran relevancia en el contexto de pruebas a código en Java, ya que es usado por muchas herramientas de soporte a pruebas unitarias debido a que TestNG agrega al menos tres funcionalidades adicionales que lo caracterizan: Permite pasar objetos como parámetros, permite definir grupos de pruebas y permite definir dependencia entre pruebas [13].

TestNG puede ser usado al mismo tiempo que JUnit, las dos pueden ser usadas en las pruebas aplicadas a un sistema, sin embargo, TestNG será la mejor opción cuando se requiera realizar pruebas concurrentes, test multihilo o condicionar un test con los resultados de otro.

#### **4.12. Metodologías de desarrollo de software**

Una metodología es un conjunto de técnicas y métodos que permite abordar de manera uniforme cada una de las actividades que conforman el ciclo de vida de un proyecto de

desarrollo de software. Es un proceso detallado y completo que describe los procedimientos, herramientas, técnicas y documentos, normados y comprendidos en un marco de trabajo. Lo cual sirven de soporte en la estructuración, planificación y controles requeridos para lograr la conversión de una o varias necesidades a un sistema de información de manera eficiente [14].

Las metodologías se basan en una combinación de modelos de proceso, definen las etapas, restricciones, tareas y distribución de recursos, así como también los roles, prácticas y técnicas utilizadas. La metodología para el desarrollo de software es una forma sistemática de realizar, gestionar y administrar un proyecto para llevarlo a cabo y garantizar el éxito del mismo.

Las metodologías de desarrollo de software establecen los procesos a seguir sistemáticamente para idear, desarrollar, implementar y mantener un producto software desde que surge la necesidad del mismo con el planteamiento del problema hasta que se cumple el objetivo por el cual fue planteado.

#### **4.13. Selección de la metodología de desarrollo de software**

La Tabla 1 contiene las principales características de las metodologías de desarrollo de software tradicional y ágil, la misma fue elaborada a partir de la información presentada en los documentos: [15] [16] [17].

**Tabla 1. Metodologías de desarrollo de software**

<b>Tradicionales</b>	<b>Ágiles</b>
Proceso riguroso y controlado	Proceso menos riguroso y controlado
Resistencia a los cambios.	Diseñada para cambios durante el proyecto.
Equipo de trabajo grande con roles específicos.	Equipo de trabajo pequeño con roles flexibles.
Gran cantidad de roles.	Pocos roles.
La arquitectura del software es esencial.	Menos énfasis en la arquitectura del software.
Orientada a proyectos de cualquier tamaño.	Orientada a proyectos pequeños.
Proyectos de media o alta duración.	Proyectos de corta duración.
Proyecto cerrado.	Proyecto abierto a cambios.
Documentación rigurosa.	Poca documentación.
Alto coste de prototipado.	Bajo coste de prototipado.
Existe una planificación a seguir.	Cambios sobre la marcha.
Pruebas tardías y pesadas.	Pruebas continuas.

#### **4.14. Selección de la metodología de desarrollo ágil**

En la Tabla 2 se evalúan las principales características de las metodologías de desarrollo de software ágiles, la misma que fue elaborada a partir de la información presentada en el documento: [18].

**Tabla 2. Metodologías de desarrollo de software ágiles**

Características	XP	Kendall y Kendall	Desarrollo basado en funciones (FDD)
<b>Enfoque</b>	- Ciclos de entrega rápidos.	- Resultados relevantes con cierta frecuencia.	- Iteraciones cortas. - Entregas tangibles.
<b>Tamaño adecuado del proyecto</b>	- Proyectos pequeños.	- Todo tipo de proyectos.	- Proyectos pequeños
<b>Involucramiento del usuario</b>	- Activamente	- Activamente	- Frecuentemente.
<b>Documentación</b>	- Básica.	- Alta.	- Básica.
<b>Principales prácticas</b>	- Estrecha interacción con el cliente. - Pruebas constantes.	- Historias de usuario. - Diagramado UML. - Marco de trabajo de SCRUM.	- Entrega de resultados cada 2-10 días. - Pocas reuniones en equipo.

#### **4.15. Modelo de desarrollo SCRUM**

Scrum es un modelo ágil y flexible que puede ser usada para la gestión del desarrollo de software, su enfoque son aquellos proyectos de entornos complejos, donde se necesita obtener resultados inmediatos, con requisitos cambiantes o definidos parcialmente [19].

Scrum es un modelo de desarrollo ágil caracterizado por:

- Adoptar una estrategia de desarrollo incremental evitando la planificación y ejecución completa del software o producto a desarrollar.
- Basar la calidad del resultado en el conocimiento de los participantes y en equipos auto organizados.
- No darle importancia a la calidad de procesos empleados.

Según [20], los componentes de la metodología SCRUM se pueden describir como:

1. **Acumulación de productos:** El cual abarca la lista de entregables a partir de las especificaciones de los productos.
2. **Acumulación de corrida o sprint:** Es una lista dinámica sobre las tareas que se van a completar en la siguiente corrida.
3. **Corrida:** Periodo de días o semanas en donde el equipo elabora parte del producto de software que se puede presentar.
4. **Scrum diario:** Reunión rápida donde la comunicación es muy importante. Los miembros del equipo explican lo que se hizo desde la última reunión, si se presentaron inconvenientes y lo que planean hacer hasta la siguiente scrum diaria.
5. **Demo:** Software funcional que se puede mostrar al cliente.

En Scrum se realizan entregas constantes y parciales del producto final, se centra en las actividades de control y no en prácticas de ingeniería, Scrum tiene como base que los procesos definidos funcionan bien sólo si las entradas están bien definidas y la ambigüedad o cambio es mínimo.

#### **4.16. Metodología de desarrollo Kendall y Kendall**

La metodología Kendall y Kendall fue seleccionada debido a la importancia de su documentación, dado que detalla cada una de las fases del ciclo de desarrollo de sistemas (SDLC) y brinda las herramientas necesarias para la ejecución de cada una de estas. Además, durante el análisis de la metodología se identificó que la mayoría de las herramientas brindadas serán utilizadas para el desarrollo del trabajo de titulación.

Según la metodología Kendall y Kendall, el ciclo de vida de un sistema consta de siete fases:

A continuación, se listan las siete etapas de la metodología Kendall y Kendall [20]:

1. **Identificación de problemas, oportunidades y objetivos:** Para esta etapa se identifica lo que la organización intenta realizar y luego se determina si el uso del sistema de información ayuda a alcanzar sus metas.
2. **Determinación de los requerimientos de información:** Esto se hace a partir de los usuarios involucrados, para determinar los requerimientos de información dentro de la organización. Se puede utilizar diferentes instrumentos los cuales incluyen: el muestreo, estudio de los datos, la entrevista, los cuestionarios o la observación de la conducta.

- 3. Análisis de las necesidades del sistema:** Se analizan las necesidades propias del sistema. Aquí se definen las herramientas de soporte para los requerimientos, se ilustra al sistema de una forma gráfica y estructurada. Se hace el uso de diagramas de flujo, diagramas de actividades y diagrama de secuencia. Se define el resultado que se va a obtener.
- 4. Diseño del sistema recomendado:** Se usa la información recolectada y se elabora el diseño lógico del sistema, esta etapa, además, incluye el diseño de la base de datos que almacenará los datos del sistema y también se da el diseño de la interfaz, los procedimientos y el diseño de la seguridad del mismo.
- 5. Desarrollo y documentación del software:** El/los analistas trabajan junto con los programadores para el desarrollo del software planteado. Dentro de las técnicas estructuradas para el diseño y documentación del software se tienen: los diagramas de flujo y el pseudocódigo. Además, se desarrollan los manuales de procedimientos.
- 6. Pruebas y mantenimiento del sistema:** El sistema debe probarse antes de ser utilizado, ya que el costo es menor si se detectan problemas previos a su despliegue. Gran parte de las pruebas se lleva a cabo por los programadores.
- 7. Implementación y evaluación del sistema:** La última etapa del desarrollo del sistema, incluye la capacitación que el usuario final. Aquí se plantea la conversión del sistema antiguo al nuevo. Se evalúa el sistema desplegado.

#### **4.17. Muestreo no probabilístico por conveniencia**

En una investigación, lo ideal es hacer uso de una muestra probabilística que represente a una población. Sin embargo, en ocasiones es muy difícil lograr abarcar toda la muestra debido a que la población a evaluar es demasiado grande o poco accesible.

El muestreo por conveniencia no es probabilístico y tampoco se considera aleatorio, aquí el investigador alige la muestra por su proximidad y no por obtener una muestra representativa de la población.

No existen criterios de aceptación que deban considerarse para formar parte de la muestra, cada elemento de la población puede formar parte de dicha muestra. Es utilizada de acuerdo a la facilidad de acceso y disponibilidad de la muestra, se caracteriza por su velocidad, sencillez, menor costo de inversión y resultados inmediatos [21].

#### **4.18. Identificación de pacientes con Covid-19**

Covid-19 pertenece a los coronavirus que son una familia de virus que causan enfermedades que pueden ir desde un resfriado común hasta enfermedades respiratorias más graves que pueden terminar con la vida de una persona [22].

Según los datos presentados por la Organización Panamericana de la Salud (OPS) [23], una persona se considera sospechosa de portar Covid-19 en tres circunstancias:

- 1) La persona cumple con los criterios clínicos y epidemiológicos.
  - a) Criterios clínicos:
    - Cumplir con tres o más síntomas de la siguiente lista: Fiebre, tos, debilidad general, fatiga, cefalea, mialgia, dolor de garganta, resfriado nasal, disnea, anorexia, náuseas, vómitos, diarrea, estado mental alterado.
  - b) Criterios epidemiológicos:
    - Haber residido o trabajado en un entorno de alto riesgo de transmisión del virus.
    - Haber residido en una zona en la que haya transmisión comunitaria.
    - Haber trabajado en un entorno de atención de salud.
- 2) Paciente con enfermedad respiratoria grave.
- 3) Individuo asintomático que ha dado positivo para prueba rápida de detección de Covid-19.

#### **4.19. Medidas de bioseguridad**

A continuación, se presenta una lista con las medidas de protección básicas contra el Covid-19, las mismas que han sido obtenidas del Protocolo de prevención de Covid-19 [24] presentado por parte del ministerio de salud pública de la república del Ecuador, y también de las entrevistas realizadas a los diferentes centros de salud (

Anexo 2. Informe de entrevista).

- Lavarse las manos con frecuencia con agua y jabón.
- Cubrirse al toser o estornudar.
- No escupir.
- Distanciamiento social.
- Evite tocarse los ojos.
- Uso de mascarilla.
- Uso de guantes.
- Al momento de acudir a un centro médico, hacerlo de preferencia solo o con un acompañante.

#### **4.20. Trabajos relacionados**

En la Tabla 3 se presenta un resumen de los trabajos relacionados y sus respectivos aportes al desarrollo del trabajo de titulación.

**Tabla 3. Trabajos relacionados**

<b>Titulo</b>	<b>Aporte al trabajo de titulación</b>	<b>Referencia</b>
Perspectiva de aplicación de la tecnología de inteligencia artificial basada en un centro de llamadas.	Expone la importancia de utilizar agentes inteligentes para la gestión de sistemas y la ventaja de utilizarlos para mejorar la experiencia conversacional con el usuario.	[25]
Inteligencia artificial conversacional: sistemas de diálogo, agentes conversacionales y chatbots.	Brinda una idea de cómo debe realizarse el flujo de interacción entre el agente inteligente y el paciente. Así como la importancia de desarrollar una interacción intuitiva.	[26]
Soporte para la vida diaria en casa a través de un socio de soporte virtual.	Plantea el flujo a seguir para el reconocimiento de voz y detalla cómo llevar la asistencia virtual en aspectos de salud.	[27]
Ampliación de una base de conocimientos de chatbot convencional a una fuente de conocimiento externa e introducción de sesiones basadas en el usuario para la educación sobre la diabetes.	Establece el flujo a seguir para el reconocimiento de palabras que no estén preestablecidas, ayudando a comprender los datos de entrenamiento y los tipos de aprendizaje.	[28]
Construcción de un sistema de diálogo completo, en lengua española, para la biblioteca braille del honorable consejo provincial de loja.	Brinda información del funcionamiento y el proceso de implementación de la librería Speech de Java.	[11]

Diseño de un sistema multi-agente para monitoreo de redes utilizando jade y jpcap	Detalla la arquitectura y funcionamiento de JADE. Expone el uso de behaviours y manejo de la interfaz gráfica de usuario (GUI) de JADE.	[29]
---	---	------

Gracias a la identificación y análisis de los trabajos relacionados se obtuvo una serie de tecnologías, librerías y arquitecturas que serán de ayuda para contestar la pregunta de investigación durante el desarrollo del trabajo de titulación. Entre los principales resultados obtenidos durante este análisis, se identificó la librería de agentes inteligentes para Java (JADE) junto con las ventajas que ofrece para la gestión de agentes; así mismo, se identificó la librería Speech y su flujo a seguir para el reconocimiento de voz en Java.

## 5. Metodología

El presente trabajo de titulación logró obtener como resultado un sistema de gestión de llamadas telefónicas controladas por un agente inteligente. El cual surgió en base al problema de investigación y pregunta: **¿El uso de agentes inteligentes ayuda en el proceso de atención médica mediante llamadas telefónicas en un centro médico, considerando la saturación de los centros de salud debido al estado de emergencia sanitaria por Covid-19?**, a partir de la cual se dio inicio al desarrollo del trabajo de titulación hasta su culminación.

Además, se definió, elaboró e implementó en conjunto la metodología, métodos, técnicas y estándares que establecieron la secuencia de pasos para el desarrollo del trabajo de titulación. Las cuales se describen a continuación:

### 5.1. Metodología de desarrollo de software

Para la elaboración del presente trabajo de titulación, se utilizó la metodología de desarrollo ágil Kendall y Kendall (K&K), la misma que se identificó como óptima de acuerdo a las características analizadas en el marco teórico.

### 5.2. Proceso

El proceso para alcanzar el objetivo general del trabajo de titulación se detalla a continuación, mencionando cada uno de los objetivos con sus respectivas actividades:

#### 5.2.1. Objetivo 1: Diseñar el módulo de software del agente inteligente para el sistema de llamadas telefónicas utilizando el lenguaje de programación Java y la metodología de desarrollo Kendall y Kendall.

- a) Se realizó entrevistas iniciales a los diferentes centros de atención médica en la ciudad de Loja para identificar los problemas, oportunidades y objetivos del módulo de software (Objetivo 1 en la sección Resultados).

- b) Se determinó los requerimientos del módulo de software mediante el documento de especificación de requisitos de la IEEE 830 (Objetivo 1 en la sección Resultados).
- c) Se identificó las necesidades del módulo de software con la ayuda de la entrevista como herramienta de recolección de datos y a los diagramas de flujo como técnica para representar la entrada de datos, los procesos y la salida de información (Objetivo 1 en la sección Resultados).
- d) Diseño del módulo de software, en donde se definió el diseño lógico del sistema haciendo uso de los diferentes diagramas UML que forman parte del documento de especificación de requisitos de la IEEE 830 (Objetivo 1 en la sección Resultados).
- e) Codificación del módulo de software basado en los requerimientos establecidos (Objetivo 1 en la sección Resultados).

#### **5.2.2. Objetivo 2: Implementar el módulo de software del agente inteligente al sistema de llamadas telefónicas.**

- a) Se desarrolló el manual de despliegue para el módulo de software (Objetivo 2 en la sección Resultados).
- b) Se implementó de forma local el módulo de software (Objetivo 2 en la sección Resultados).
- c) Se aplicó pruebas funcionales y unitarias al módulo de software (Objetivo 2 en la sección Resultados).

#### **5.2.3. Objetivo 3: Evaluar la aceptación y correcto funcionamiento del módulo de software en el sistema de llamadas telefónicas.**

- a) Se elaboró el manual de usuario del módulo de software (Objetivo 3 en la sección Resultados).
- b) Se aplicó pruebas de aceptación del módulo de software al usuario final y a expertos en medicina (Objetivo 3 en la sección Resultados).

### **5.3. Métodos**

#### **5.3.1. Método Analítico**

El método analítico se encarga de la descomposición de un todo en sus elementos básicos, de lo general a lo específico [32]. Se hizo uso del método analítico para dividir el trabajo de titulación en diferentes etapas y actividades a seguir durante la realización del mismo, ayudando así a la realización de los diferentes objetivos planteados.

### **5.3.2. Método Científico**

El método científico [33], fue utilizado para la definición y planteamiento del problema, formulación de la hipótesis la cual se encuentra dentro de la pregunta de investigación, recolección, procesamiento y análisis de datos en la sección de resultados, también se lo utilizó para dar contraste de los datos obtenidos con las hipótesis planteadas en la discusión y por último en la elaboración de las conclusiones.

### **5.3.3. Método inductivo-deductivo**

Empleado en la recolección de datos, análisis de la información e interpretación de los hechos y descubrimiento de nuevos procedimientos. El método de razonamiento inductivo se utilizó para poder obtener la secuencia lógica que debe tener el comportamiento del sistema desarrollado en base a las respuestas dadas por las personas entrevistadas. El método de razonamiento deductivo se utilizó para poder definir qué tareas dentro del desarrollo del sistema se deben considerar de importancia, así como también para poder obtener la lista de funcionalidades que satisfacen los requerimientos de los entrevistados.

### **5.3.4. Estudio de Caso**

Este método se utilizó para la revisión de trabajos relacionados al tema del trabajo planteado, ayudando así a aclarar el estado del arte, identificando trabajos similares y ayudando a mejorar el enfoque del trabajo de titulación.

## **5.4. Técnicas**

### **5.4.1. Muestreo no probabilístico por conveniencia**

Permite seleccionar la muestra de una población de acuerdo con su disponibilidad [21]. Utilizado para seleccionar las muestras a las cuales se les efectuó las diferentes entrevistas y encuestas que ayudaron en la fase de elicitation de requerimientos y en la fase de pruebas de aceptación.

### **5.4.2. Entrevistas**

Para recaudar la información que ayude a la obtención de requerimientos del módulo de software, se realizaron entrevistas en distintas instituciones de salud pública y privada como hospitales y clínicas de la ciudad de Loja. Utilizada también como técnica para evaluar la aceptación por parte del usuario final y la validación por parte de los expertos en medicina.

### **5.4.3. Encuestas**

Se utilizó dos tipos de encuestas dirigidas a profesionales en medicina. La primera con la finalidad de obtener una lista de síntomas presentados por personas con Covid-19, mismos

que fueron usados en la aplicación para tener una clasificación precisa de los pacientes y poder tomar las respectivas decisiones.

La segunda encuesta, fue utilizada para la validación de la aplicación y análisis de la funcionalidad de la misma.

#### **5.4.4. Prototipado**

Utilizado para modelar el software final, obteniendo así información objetiva del sistema que permitió efectuar las diferentes pruebas a sus atributos, permitiendo implementar mejoras sin que se despliegue el módulo de software final.

### **5.5. Estándares**

#### **5.5.1. IEEE 830**

Conjunto de recomendaciones para la especificación de los requerimientos o requisitos de software. Utilizado para dar cumplimiento a las actividades del primer objetivo específico: diseñar el módulo de software del agente inteligente para el sistema de llamadas telefónicas.

### **5.6. Materiales**

#### **5.6.1. Recursos Académicos**

- Guías de desarrollo.
- Foros.
- Artículos.
- Documentación gris.
- Bases de datos científicas.
- Revistas.
- Conferencias.

#### **5.6.2. Recursos humanos**

La Tabla 4 presenta la lista de los recursos humanos utilizados para el desarrollo del presente trabajo de titulación.

**Tabla 4. Recursos humanos**

<b>Nombre</b>	<b>Descripción</b>
Sandro Michael Córdova Carrión.	Estudiante a cargo de la ejecución del proyecto.
José Oswaldo Guamán Quinche.	Docente director del Proyecto de Trabajo de Titulación.
Liliana Elizabeth Sanmartín Cruz.	Médico a cargo de la validación del módulo de software desarrollado.
Johnny Minga Tapia.	Médico a cargo de la validación del módulo de software desarrollado.

Yakeline Cartuche.	Médico a cargo de brindar la clasificación de pacientes con Covid-19 acorde a su sintomatología.
Hugo Ramiro Córdova Córdova.	Enfermero a cargo de la validación del módulo de software desarrollado.
Bertha Beatriz Narváez Briceño.	Enfermera a cargo de la validación del módulo de software desarrollado.
Maricela Jesús Tenesaca Álvarez.	Enfermera a cargo de la validación del módulo de software desarrollado.
Andrés Rivas.	Enfermero a cargo de la validación del módulo de software desarrollado.
Danny Renato Mora Huiracocha.	Enfermero a cargo de brindar la clasificación de pacientes con Covid-19 acorde a su sintomatología.
Verónica Estefanía Gutiérrez Luzón.	Enfermera a cargo de brindar la clasificación de pacientes con Covid-19 acorde a su sintomatología.

### 5.6.3. Recursos de software y hardware

A continuación, la Tabla 5 presenta los recursos de software y la Tabla 6 los recursos hardware utilizados para el desarrollo del trabajo de titulación.

**Tabla 5. Recursos de software**

Recurso	Descripción
Java v17.0.2	Lenguaje de programación en donde se desarrolló la aplicación.
JDK v17.0.2	Paquete para el desarrollo de Java.
JADE v4.3.	Librería de Java para la creación y gestión de agentes.
Mongo DB v5.0.1	Base de datos no relacional.
MongoDB-connection v2.9.3	Librería para la conexión y gestión de Mongo DB.
TestNG v6.14.3	Librería de Java utilizada para realizar las pruebas a la aplicación.
Speech v2.0	Librería para convertir texto a voz y voz a texto.
Swing v1.0.3	Librería para interfaz gráfica.
IDE NetBeans v12.6	Plataforma sobre la que se desarrolló la aplicación.
Zoom	Herramienta utilizada para las reuniones con el director del trabajo de titulación.
GitHub	Utilizado como repositorio en la nube y para el control de versiones de la aplicación.
Windows 11	Sistema operativo sobre el que se desarrolló la aplicación.

**Tabla 6. Recursos de hardware**

Recurso	Descripción
Computador personal.	Equipo indispensable para el desarrollo del trabajo de titulación.
Micrófono	Dispositivo de entrada de voz.
Altavoz	Dispositivo de salida de voz.

## **6. Resultados**

En el siguiente apartado se describe el proceso que llevo a cabo para cumplir con los objetivos planteados en el presente TT.

### **6.1. Objetivo I: Diseñar el módulo de software del agente inteligente para el sistema de llamadas telefónicas utilizando el lenguaje de programación Java y la metodología de desarrollo Kendall y Kendall.**

Para identificar el proceso seguido y las situaciones a mejorar durante la interacción con los pacientes se aplicó entrevistas a diferentes centros médicos de la ciudad de Loja, con el fin de analizar el proceso seguido durante la comunicación por llamada telefónica, así como el soporte y ayuda que se brindan al paciente (

Anexo 2. Informe de entrevista).

Gracias a estas entrevistas se obtuvo los requisitos funcionales, requisitos no funcionales e historias de usuario destallados en las secciones siguientes.

#### **6.1.1. Definir el grupo objetivo que utilizará el módulo de software**

El grupo objetivo al que está dirigido el módulo de software según su incidencia por casos de Covid-19 (punto 3. Introducción), es a personas comunes entre los 20 a 49 años de edad, que no padeczan enfermedades crónicas o problemas de salud persistentes. También se requiere conocimiento básico de tecnologías similares.

#### **6.1.2. Establecer los actores**

El módulo de software desarrollado tiene los siguientes actores:

**Paciente:** Actor que realiza la llamada y accede al centro de información sobre Covid-19.

Se asume que la persona que se contacta con el centro de atención médica es el paciente.

**Administrador:** Encargado de actualizar, eliminar y generar reportes de las citas médicas agendadas, así como generar reportes de los pacientes que se comunican con el centro de atención médica.

#### **6.1.3. Selección de la muestra a entrevistar**

Según los datos presentados en el documento “Establecimientos de salud de primer nivel autorizados” por el ministerio de salud pública [34] y las instituciones registradas en el directorio web “Edina” [35], dentro del área urbana de la ciudad de Loja existen 36 centros de atención médica entre centros de salud, clínicas y hospitales.

Para obtener una muestra significativa de una población de 36 centros de atención médica con un nivel de confianza del 90% y un margen de error de 5% se debe aplicar la fórmula de muestreo con los datos especificados en la Tabla 4:

**Tabla 7. Datos para el cálculo de la muestra**

Variable	Valor	Descripción
z	1.65	El valor de z para el 90% de confianza.
N	36	Tamaño de la población.
e	5	Margen de error.
p	50	Probabilidad de que ocurra el evento.
q	50	Probabilidad de que no ocurra el evento.

$$Tamaño\ de\ muestra = \frac{\frac{z^2 * p(1 - q)}{e^2}}{1 + (\frac{z^2 * p(1 - q)}{e^2 * N})}$$

$$Tamaño\ de\ muestra = \frac{\frac{1,65^2 * 50(1 - 50)}{5^2}}{1 + (\frac{1,65^2 * 50(1 - 50)}{5^2 * 36})}$$

$$Tamaño\ de\ muestra = \frac{266,805}{4,411} = 31.72$$

$$= 32\ Centros\ médicos\ //$$

Al realizar la solicitud para la aplicación de las entrevistas, únicamente 6 de 32 centros médicos aceptaron proporcionar los datos solicitados, esto debido a factores como tiempo limitado de los funcionarios, información confidencial de la entidad o por protocolos de bioseguridad.

Dando como resultado un muestreo probabilístico insatisfactorio ya que no se logró cumplir con el tamaño de la muestra esperada, lo que conllevo a realizar un muestreo no probabilístico por conveniencia.

#### **6.1.4. Historias de usuario**

En la Tabla 5 se presentan las historias de usuario establecidas para el desarrollo del módulo de software.

**Tabla 8. Historias de usuario**

Código	Nombre de la historia	Descripción
01	Transferir llamada.	Cómo paciente quiero que mi llamada sea transferida al departamento de atención al cliente del centro médico para comunicarme directamente con una persona.
02	Clasificación del paciente	Cómo paciente quiero que el sistema me clasifique acorde a mis síntomas para poder recibir atención médica.
03	Medidas de bioseguridad	Cómo paciente quiero que se me brinden las medidas de bioseguridad para prevenir contagios.
04	Generar cita médica	Cómo paciente quiero generar una cita médica para atenderme en el centro médico.
05	Recetar en caso de emergencia	Cómo paciente quiero que se me receten medicamentos de emergencia como medida preventiva hasta poder ser atendido en el centro médico.
06	Visualizar pacientes	Cómo administrador quiero visualizar la lista de pacientes que se han comunicado con el centro médico para generar informes.
07	Visualizar citas	Cómo administrador quiero visualizar la lista de citas que se han agendado en el centro médico para generar informes.
08	Actualizar cita	Cómo administrador quiero actualizar la información de las citas agendadas en el centro médico para corregir errores durante el proceso de agendamiento.
09	Borrar cita	Cómo administrador quiero eliminar una cita de forma lógica para cancelar la cita agendada por el paciente.

#### **6.1.5. Requisitos funcionales**

En la Tabla 6 se presentan los requisitos funcionales establecidos para el desarrollo del módulo de software, resultado de las entrevistas realizadas a seis centros médicos de la ciudad de Loja (

Anexo 2. Informe de entrevistas a centros médicos).

**Tabla 9. Requisitos funcionales**

Código	Requisitos	Descripción
01	Transferir llamada.	El usuario puede solicitar al sistema comunicación directa con el departamento de atención al cliente del centro médico.
02	Listar síntomas.	El sistema permitirá al usuario listar los síntomas que este posee para poder determinar el estado del paciente.
03	Clasificación del estado de los pacientes.	El sistema compara y clasifica los síntomas dados por el paciente y toma decisiones dependiendo de la clasificación.
04	Informar sobre medidas de bioseguridad.	El sistema debe presentar al usuario las medidas de bioseguridad necesarias para poder acercarse al centro médico.
05	Agendar citas.	El usuario podrá agendar una cita médica. Los datos para agendar una cita son: Especialidad médica, doctor, número de cédula, nombres, apellidos, fecha y hora.
06	Recetar en caso de emergencia.	En caso de emergencia, el sistema debe recomendar al usuario el consumo exclusivo de Paracetamol, hasta que este pueda ser atendido en el centro médico.
07	Gestión de citas.	El administrador podrá visualizar, actualizar y cancelar las citas médicas agendadas.
08	Generar reportes de pacientes.	El administrador podrá visualizar la información consolidada de los pacientes que se han comunicado con el centro médico.
09	Generar reportes de citas.	El administrador podrá visualizar la información consolidada de las citas que se han generado en el centro médico.

#### 6.1.6. Requisitos no funcionales

En la Tabla 10 se presentan los requisitos no funcionales establecidos para el desarrollo del módulo de software.

**Tabla 10. Requisitos no funcionales**

Código	Requisitos	Descripción
01	Rendimiento.	El sistema debe garantizar que las consultas al sistema u otro proceso no afecte el desempeño de la base de datos, ni a la respuesta del sistema de forma general. Monitoreando el tiempo de respuesta de la base de datos y del sistema en general, antes y después de la implementación del módulo de software.
02	Seguridad.	El sistema garantizará la seguridad de la información y datos personales que el usuario proporcione al momento de generar una cita médica. Aplicando mecanismos de protección a los datos personales de tipo confidencial (encriptación).
03	Usabilidad.	El sistema deberá contar con una interfaz de uso intuitiva y sencilla, evaluada a través de pruebas de aceptación del usuario en el uso del sistema.
04	Disponibilidad.	La disponibilidad del sistema deberá ser continua con un nivel de servicio mínimo del 95% durante los 365 días del año 24/7, exceptuando horarios de mantenimiento y fallos no planificados. Al medir el nivel de satisfacción del usuario respecto a la disponibilidad del módulo de software.
05	Mantenibilidad.	El sistema debe disponer de documentación que permita realizar operaciones de mantenimiento.
06	Portabilidad.	El sistema deberá ser compatible con la mayoría de los sistemas operativos en los que se ejecute. A través del uso de herramientas de código abierto y en sus versiones estables para el desarrollo del módulo.
07	Voz aguda y agradable.	A través de pruebas de aceptación el sistema debe garantizar una voz aguda y agradable al oído del usuario durante la interacción.

Durante el análisis de las necesidades del sistema se obtuvo el diagrama de casos de uso y el diagrama de actividades correspondientes a vista de escenarios y a la vista de procesos respectivamente del modelo 4+1 en la arquitectura de software (

Anexo 3. Arquitectura de software).

#### 6.1.7. Diagrama de casos de uso

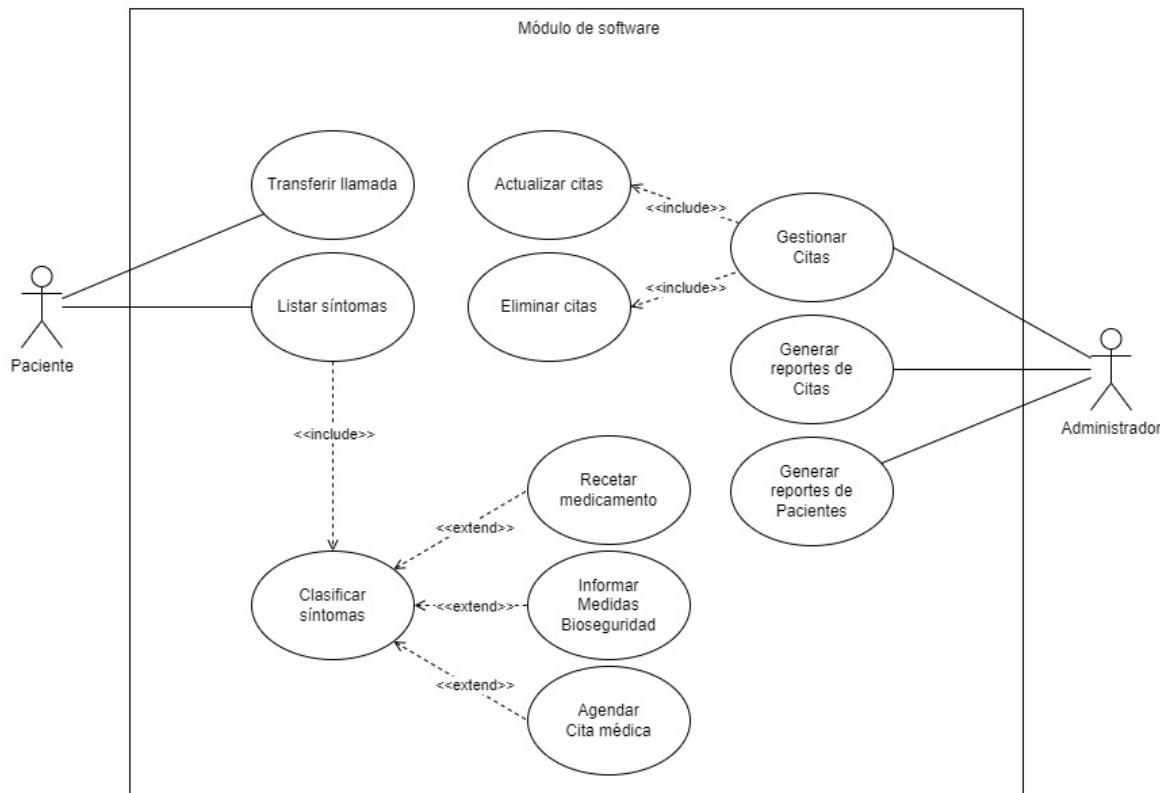


Figura 1. Diagrama de casos de uso.

#### 6.1.8. Actores del sistema

La Tabla 8 brinda detalles del actor “Paciente”. La Tabla 9 brinda detalles del actor “Administrador”.

**Tabla 11. Actor del sistema 01**

<b>Identificador.</b>	AS-01
<b>Nombre.</b>	Paciente.
<b>Descripción.</b>	Actor que realiza la llamada y accede al centro de información sobre Covid-19. Se asume que la persona que se contacta con el centro de atención médica es el paciente.

**Tabla 12. Actor del sistema 02**

<b>Identificador.</b>	AS-02
<b>Nombre.</b>	Administrador
<b>Descripción.</b>	Encargado de actualizar, eliminar y generar reportes de las citas médicas agendadas, así como generar reportes de los pacientes que se comunican con el centro de atención médica.

**6.1.9. Especificación de casos de uso**

La Tabla 13 contiene la especificación de cada uno de los casos de uso (

Anexo 3. Arquitectura de software).

**Tabla 13. Especificación de casos de uso**

Código	Nombre del caso de uso	Descripción
UC-01	Transferir llamada.	El paciente solicita que se transfiera su llamada a la línea designada para atención al cliente del centro médico.
UC-02	Listar síntomas	El paciente brinda una lista de los síntomas que posee.
UC-03	Clasificar síntomas	El agente inteligente compara y clasifica los síntomas dados por el paciente, esta clasificación puede ser de prioridad alta, prioridad media y prioridad baja.
UC-04	Informar Medidas de bioseguridad.	En caso de que los síntomas del paciente lo ameriten, el agente inteligente brinda al paciente las medidas de bioseguridad que debe cumplir.
UC-05	Agendar cita médica.	En caso de que los síntomas del paciente lo ameriten, el agente inteligente genera una cita médica con los datos personales del paciente.
UC-06	Recetar medicamento.	En caso de que los síntomas del paciente lo ameriten, el agente inteligente indica al paciente los medicamentos que puede tomar de manera preventiva.
UC-07	Gestionar citas	Permite al administrador actualizar la fecha de la cita médica agendada. Permite la eliminación lógica de la cita médica agendada.
UC-08	Generar reportes de citas.	Permite al administrador del sistema generar reportes de las citas médicas agendadas en el centro médico.
UC-09	Generar reportes de pacientes.	Permite al administrador del sistema generar reportes de los pacientes que se han comunicado con en el centro médico.

#### **6.1.10. Diagrama de actividades**

A continuación, se presenta un resumen de los diagramas de actividades realizados en la vista de procesos del documento de arquitectura de software (

Anexo 3. Arquitectura de software).

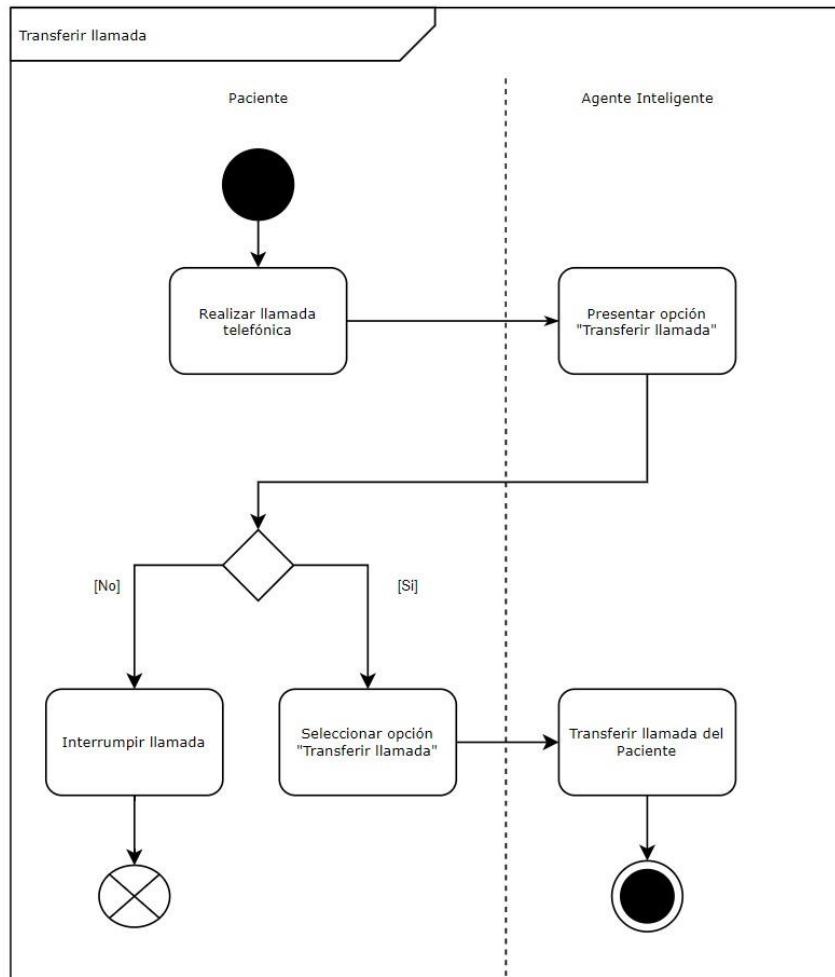


Figura 2. Diagrama de actividades, transferir llamada.

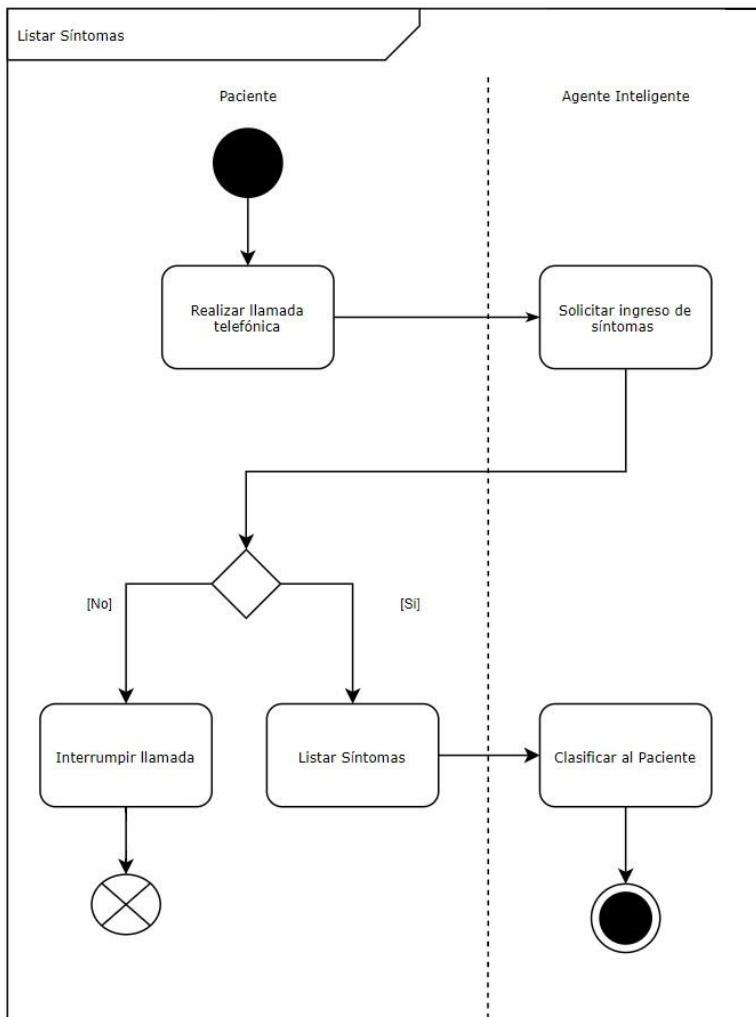


Figura 3. Diagrama de actividades, listar síntomas.

Para el diseño del sistema recomendado se obtuvo el diagrama general de la arquitectura de software y la representación de la arquitectura basada en el modelo 4+1, que abarca el diagrama de clases, diagrama de paquetes, diagrama de despliegue y la representación de la base de datos (

Anexo 3. Arquitectura de software).

#### 6.1.11. Diagrama de general de la arquitectura de software

Las siguientes figuras contienen el diagrama general de la arquitectura del módulo de software en donde se observa la interacción entre los diferentes componentes del sistema y las tecnologías que posee.

La Figura 4 muestra el diagrama general del módulo de software implementado en un centro de llamadas telefónicas.

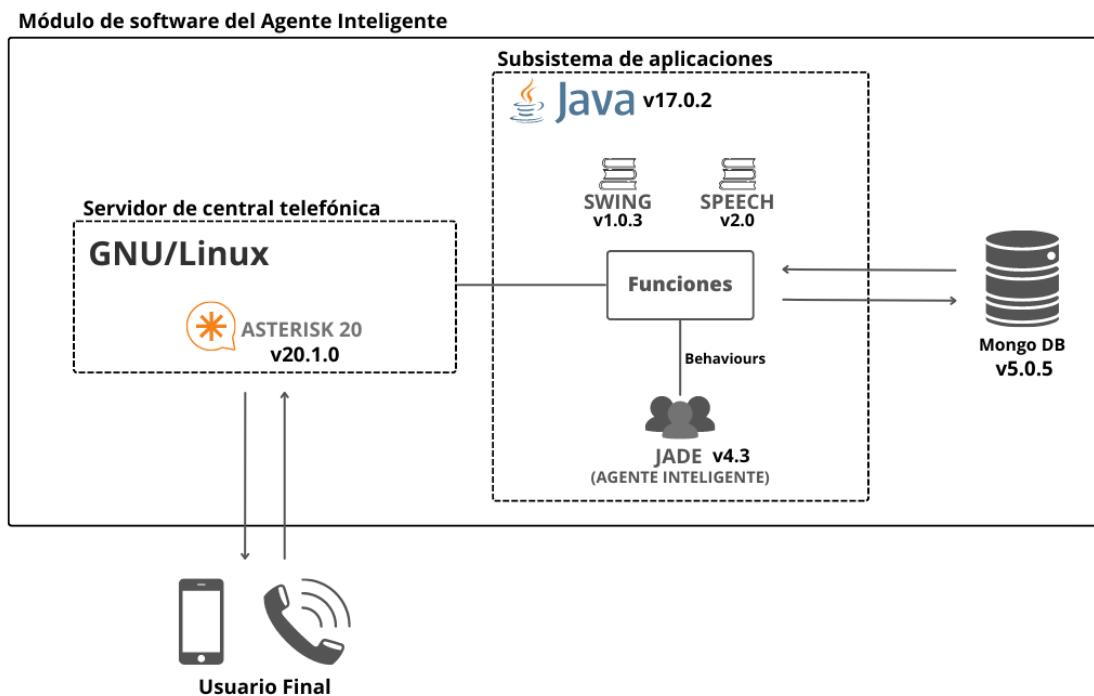


Figura 4. Diagrama general del módulo de software en un centro de llamadas telefónicas.

La Figura 5 muestra el diagrama general del módulo de software usado en un servidor local.

#### Módulo de software del Agente Inteligente

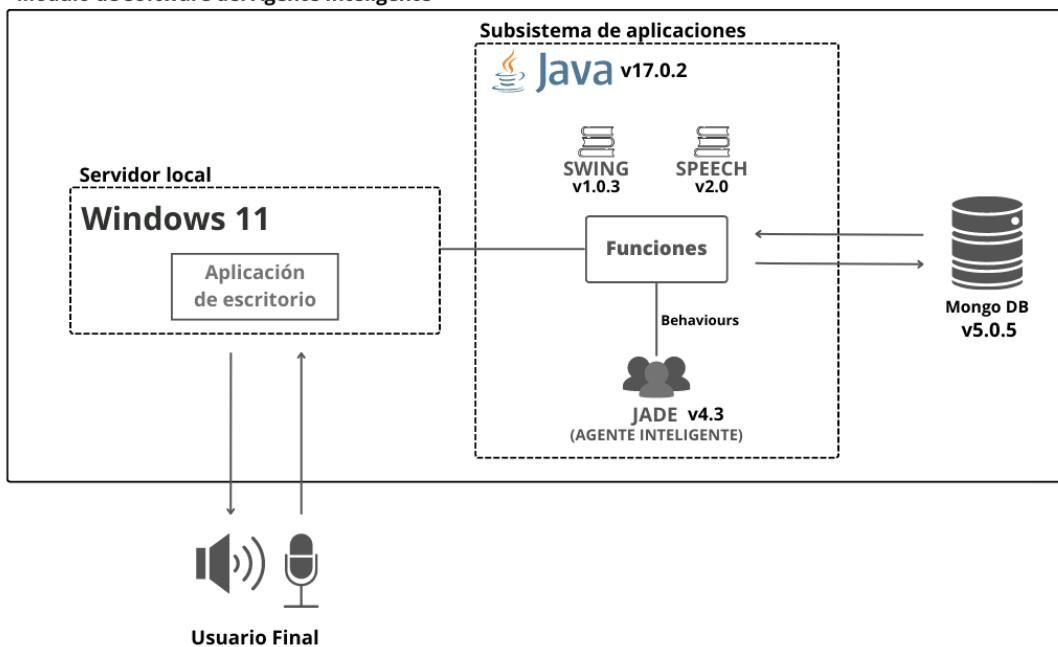


Figura 5. Diagrama general del módulo de software utilizado.

#### 6.1.12. Representación de la arquitectura

La solución informática planteada utiliza una arquitectura diseñada con la ayuda del modelo 4+1, el cual plantea el uso de cuatro vistas que describen diferentes aspectos de la arquitectura y que se relacionan entre sí con una vista extra, denominada “vista ‘+1’” o vista de escenarios. En la Tabla 14 se describe cada una de las vistas del modelo.

Tabla 14. Arquitectura 4+1

Vista	Perspectiva	Elemento	Descripción
Vista Lógica.	Usuario Final.	Diagrama de Clases y representación de base de datos.	Representa la funcionalidad que el sistema proporcionara a los usuarios finales.
Vista de Despliegue.	Administrador de Software.	Diagrama de Paquetes.	Se muestra como está dividido el software en componentes y las dependencias que hay entre los componentes.
Vista Física.	Administrador de Software.	Diagrama de Despliegue.	Se muestra todos los componentes físicos del sistema, así como todas las conexiones físicas entre los componentes que conforman la solución.
Vista de Procesos.	Programador.	Diagrama de Actividad.	Se muestra los procesos que hay en el sistema y el flujo que siguen.
Vista de Escenarios (+1).	General.	Diagrama de Casos de Uso.	Esta vista tiene como objetivo unir y relacionar las 4 vistas. Presentada en la Figura 1 de la Fase 3.

## Diagrama de clases

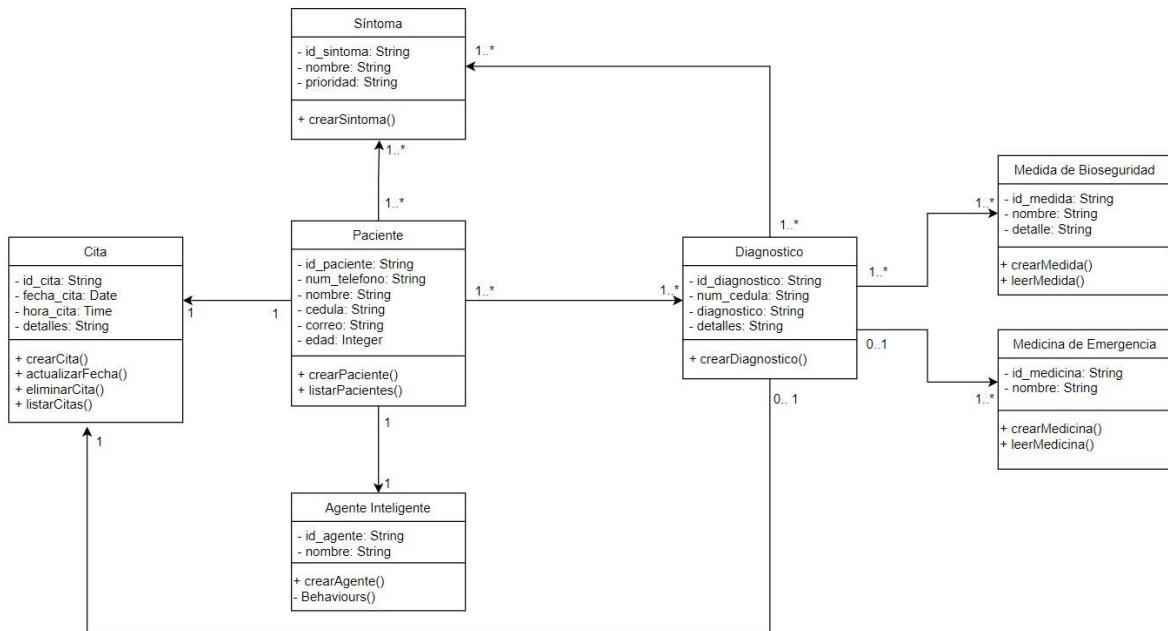


Figura 6. Diagrama de clases.

## Diagrama de paquetes

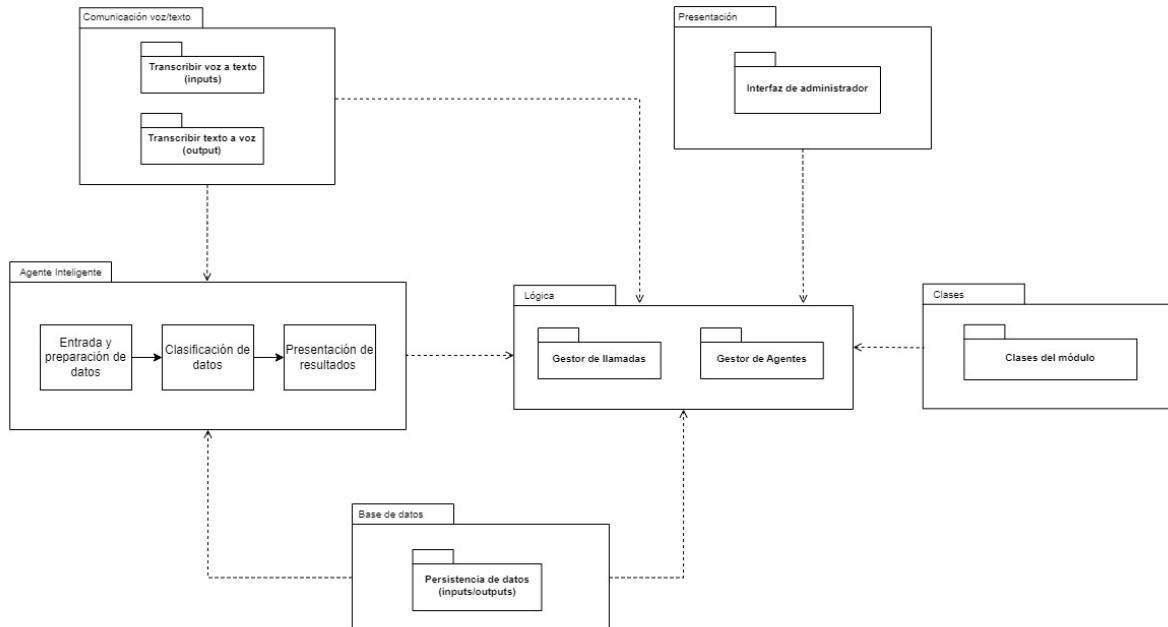


Figura 7. Diagrama de paquetes.

## Diagrama de despliegue

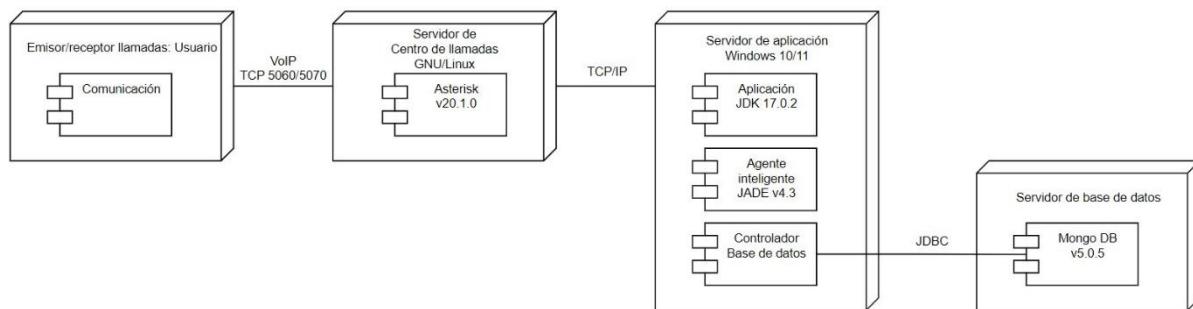


Figura 8. Diagrama de despliegue.

## Persistencia de la base de datos

La solución informática planteada utiliza MongoDB para la persistencia de datos por lo que a continuación se muestra un gráfico representativo de las diferentes fases definidas para este proceso y sus respectivas colecciones de datos.

Fase de Recolección	Fase de Clasificación	Fase de Resultados	Entrenamiento del agente inteligente
llamada	diagnostico	medida_bioseguridad	dominio
paciente	sintoma_paciente	medicamentos	extension
extensiones		cita	numeros simbolos

Figura 9. Colecciones de la base de datos.

### 6.1.13. Agente inteligente: Representación de la arquitectura

#### Arquitectura JADE

JADE se considera un entorno de ejecución en donde los agentes pueden existir, este entorno se denomina “contenedor” el contenedor puede almacenar un número indeterminado de agentes en donde el conjunto de contenedores se denomina “plataforma” la cual es una capa extra dentro/fuera del entorno en donde se ejecuta la aplicación.

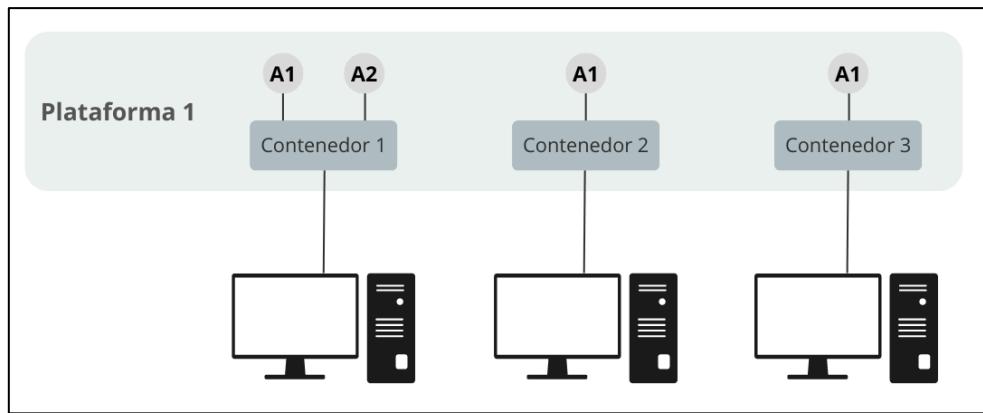


Figura 10. Arquitectura JADE

### Arquitectura JADE utilizada

En el módulo de software desarrollado, se utiliza un “contenedor” que alberga a los agentes inteligentes, en donde la “plataforma” es implementada dentro del entorno en donde se ejecuta la aplicación (módulo desarrollado).

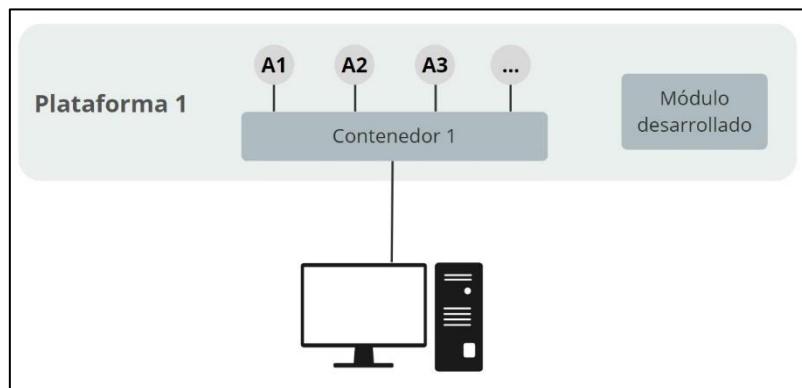


Figura 11. Arquitectura JADE usada

### Escalabilidad

Como se indicó previamente, un sistema que utiliza JADE puede tener varias plataformas y una plataforma puede abarcar a varios contenedores, que a su vez tienen un sin número de agentes. Depende de la capacidad de procesamiento y almacenamiento de la plataforma para tener uno o varios contenedores, si la demanda de un centro de llamadas es alta se puede incrementar el número de contenedores o de plataformas necesarias garantizando así la escalabilidad del módulo desarrollado.

## Estrategia de procesamiento

El agente inteligente debe tomar decisiones en base a la clasificación del paciente, este concepto le pertenece a la estrategia de procedimiento de agentes con reflejos simples o también llamados agentes reactivos puros. Esta estrategia de procesamiento se muestra en la Figura 12.

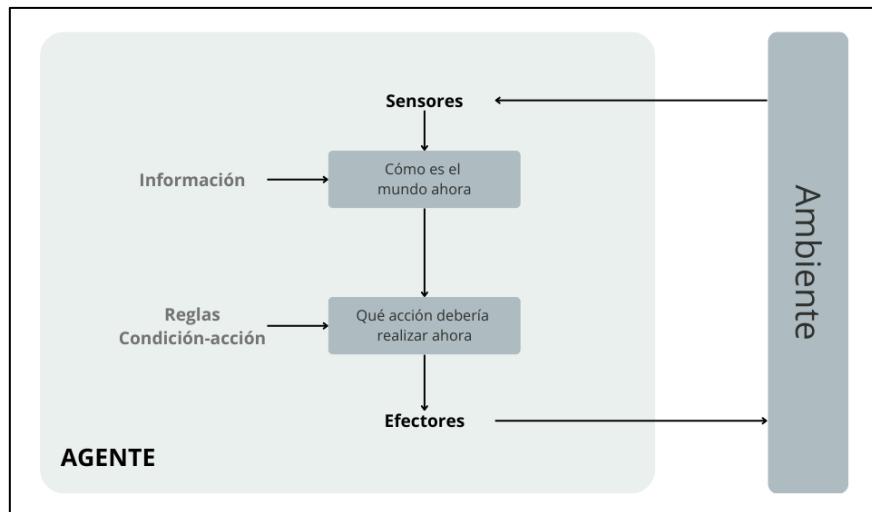


Figura 12. Estrategia de procesamiento

La Figura 13 muestra el procedimiento que el agente lleva a cabo dentro del módulo de software, aquí la información percibida por los sensores del agente son los síntomas del paciente, estos síntomas pasan a la etapa de procesamiento en donde se analiza con qué reglas de clasificación cumplen y pasar a los efectores que se dan acorde a la clasificación dada.

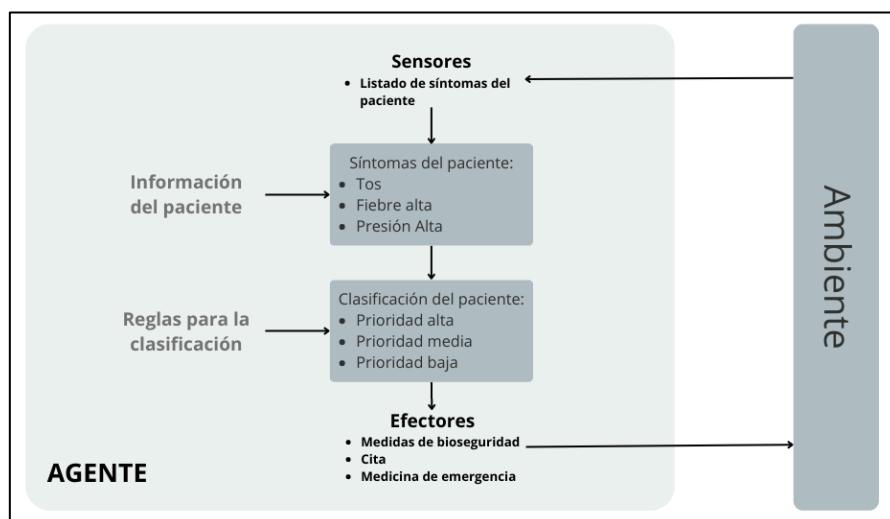


Figura 13. Estrategia de procesamiento usada

## Comportamiento del agente

Una vez definida la arquitectura y la estrategia de procesamiento, se plantea el flujograma del comportamiento del agente inteligente dentro del módulo de software Figura 14. El agente inteligente es llamado desde la clase raíz e inicializado, acto seguido el agente inteligente tiene como tarea inicial solicitar los datos personales del paciente para luego solicitar el ingreso de los síntomas, una vez ingresados los síntomas el agente les asigna un grado de prioridad y clasifica al paciente, de la clasificación asignada dependen los eventos siguientes en donde se puede: brindar medidas de bioseguridad, crear una cita médica o recetar medicamento en caso de emergencia.

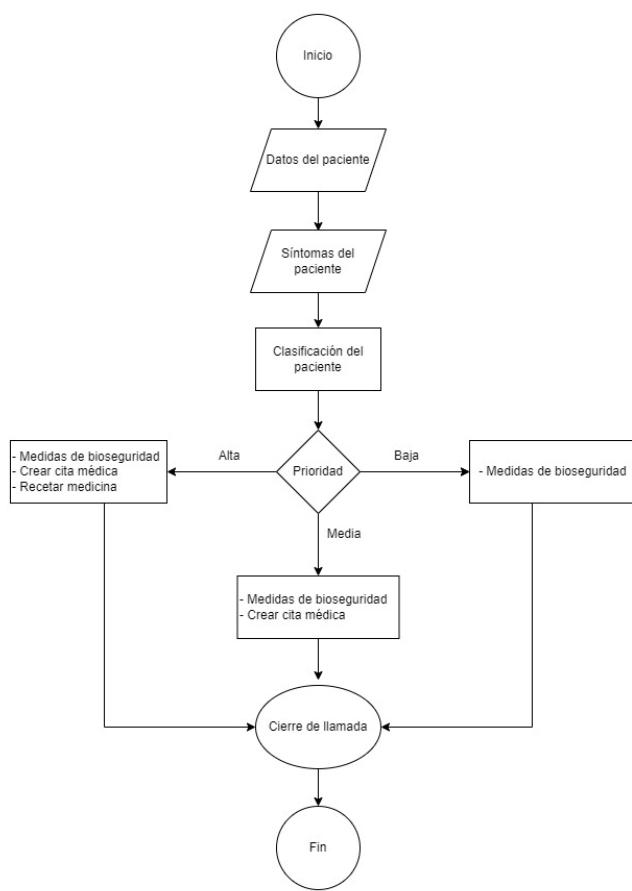


Figura 14. Flujograma del comportamiento del agente

## **Clasificación del paciente**

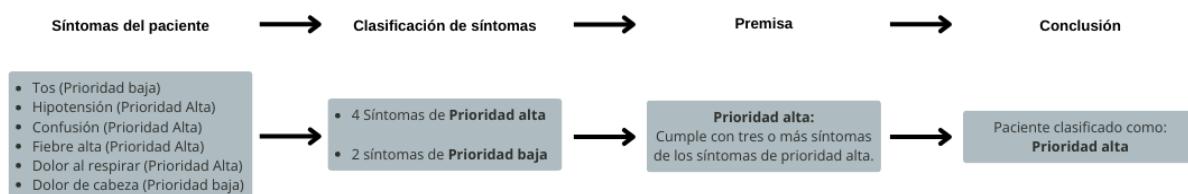
El grado de prioridad a los síntomas se asigna considerando el

Anexo 9. Informe de encuesta de síntomas y la clasificación del paciente se basa en las reglas definidas de la Tabla 15 la cual se generó a partir de los criterios establecidos en el punto 4.18 Identificación de pacientes con Covid-19.

**Tabla 15. Clasificación de pacientes**

Prioridad	Grado	Detalle
P03	Prioridad alta	Cumple con tres o más síntomas de los síntomas de prioridad alta.
P02	Prioridad media	Cumple con menos de tres síntomas de prioridad alta o con dos o más síntomas de prioridad media.
P01	Prioridad baja	No cumple con las condiciones de prioridad alta y media.

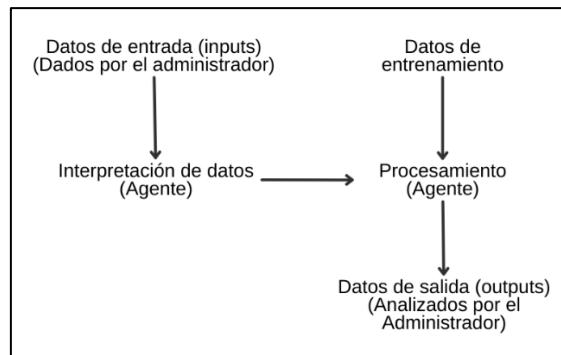
A continuación, la Figura 15 presenta un ejemplo de clasificación. Una vez ingresados los síntomas del paciente, estos son comparados con los síntomas registrados en la base de datos y se les asigna la clasificación correspondiente (síntoma de grado alto, medio, bajo), una vez clasificados se procede analizar a qué premisa corresponde (P03, P02 o P01) para poder asignarle un grado de prioridad al paciente en base a sus síntomas. En el ejemplo podemos observar que los síntomas del paciente corresponden a la premisa P03 por lo que el paciente es clasificado como prioridad alta.



*Figura 15. Flujo del Proceso de entrenamiento*

### Entrenamiento del agente inteligente

Para el entrenamiento del agente se hace uso del aprendizaje supervisado en donde el administrador del módulo de software es el encargado de entrenar al agente. Para su entrenamiento el proceso seguido se muestra en la Figura 16 aquí el administrador se encarga de ingresar los datos de prueba para posteriormente analizar los datos de salida interpretados por el agente, si la interpretación es correcta se continúa con más datos de prueba y si la interpretación es incorrecta se procede a ingresar datos de entrenamiento.



*Figura 16. Flujo del Proceso de entrenamiento*

El módulo desarrollado permite únicamente entrenamiento de símbolos, números y dominios. No se permite el entrenamiento de síntomas debido ya que estos datos fueron recolectados inicialmente y requieren de un proceso de selección para ingresar más síntomas de referencia.

Para el desarrollo y documentación del software se utilizó el modelo planteado por SCRUM el cual ayudó a cumplir con los entregables requeridos en los lapsos de tiempo planteados, para ello se hizo uso de las iteraciones, donde cada iteración tiene la estructura de: planificación, diseño, codificación, pruebas y resultados (

Anexo 4. Desarrollo y documentación).

#### 6.1.14. Iteraciones (Sprints)

Para la fase de desarrollo se establecieron cuatro iteraciones, cada iteración tendrá como resultado final los diferentes nodos que conforman al módulo de software desarrollado.

**Tabla 16. Iteración 1**

Actividades	Resumen	Duración
AC – 01	Instalación del framework NetBeans.	10 horas.
AC – 02	Instalación del JDK de Java.	
AC – 03	Creación del proyecto.	
AC – 04	Agregar la libreta JADE dentro del proyecto.	
AC – 05	Crear repositorio de GitHub para el proyecto.	
AC – 06	Instalación de la base de datos.	
AC – 07	Creación de la base de datos.	
AC – 08	Creación de las colecciones en la base de datos.	

**Tabla 17. Iteración 2**

Actividades	Resumen	Clave HU	Duración
AC – 01	Transferir llamada.	HU01	40 horas.
AC – 02	Listar síntomas	HU02	
AC – 03	Clasificación del paciente.	HU03	
AC – 04	Generar cita médica.	HU08	
AC – 05	Medidas de bioseguridad.	HU09	
AC – 06	Recetar en casos de emergencia.	HU10	

**Tabla 18. Iteración 3**

Actividades	Resumen	Clave HU	Duración
AC – 01	Visualizar cita.	HU07	30 horas.
AC – 02	Actualizar cita.	HU07	
AC – 03	Borrar cita.	HU07	
AC – 04	Generar reporte de pacientes	HU08	

**Tabla 19. Iteración 4**

Actividades	Resumen	Duración
AC – 01	Voz aguda y agradable.	10 horas.
AC – 02	Diseño de interfaces gráficas	

#### 6.1.15. Control de versiones

Para el control de versiones del módulo de software se hizo uso del repositorio de GitHub en la Figura 17 se observan las diferentes actualizaciones del módulo de software que corresponden a cada una de las iteraciones realizadas. Tanto el código como las diferentes versiones se pueden encontrar en el repositorio:

<https://github.com/sandrocordova/AgenteInteligenteJava.git>

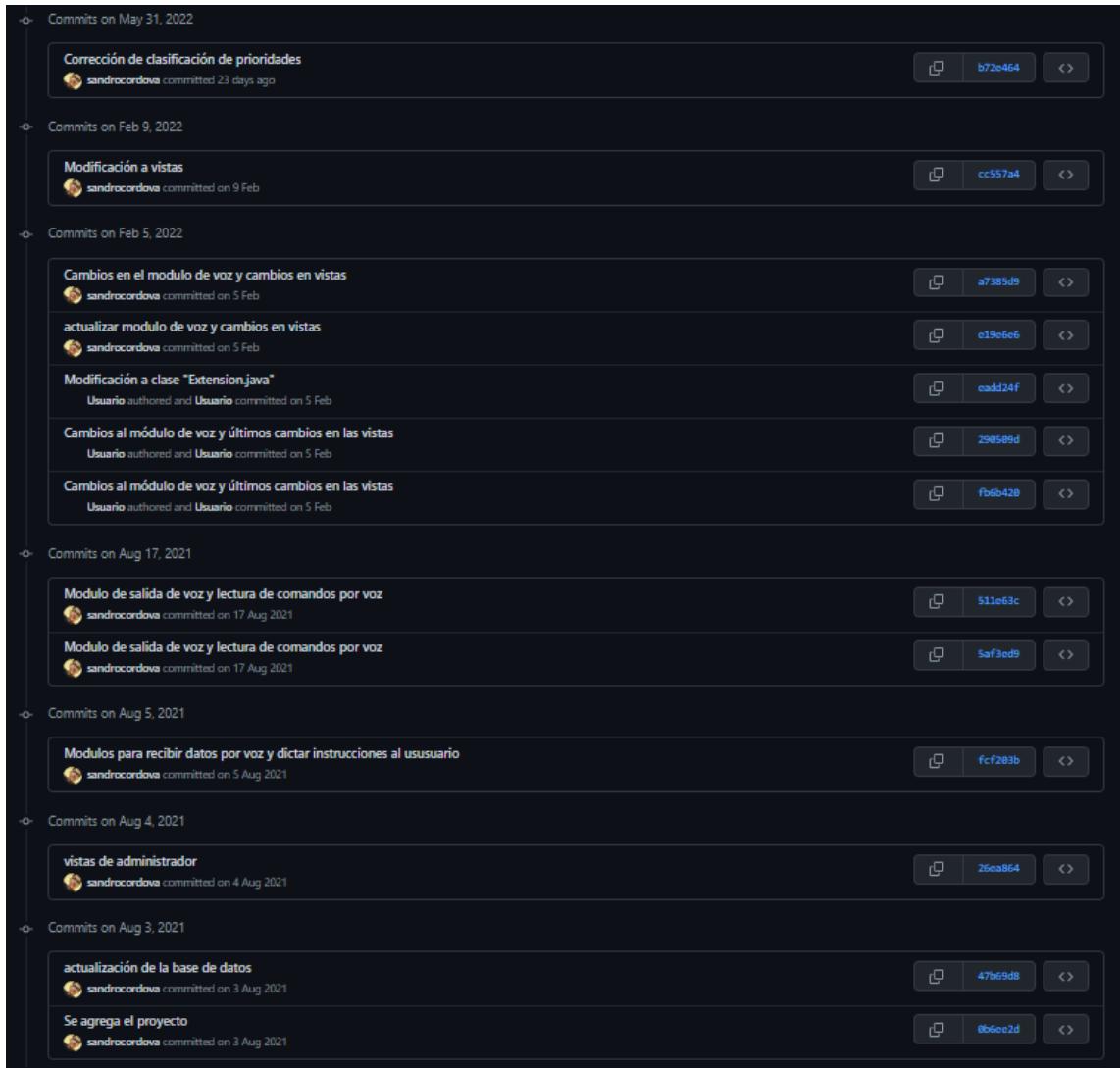


Figura 17. Control de versiones en GitHub.

#### 6.1.16. Estructura del módulo de software

La estructura del módulo de software se observa en la Figura 18 y en la Tabla 20 se detallan los diferentes componentes. Para poder acceder a cada uno de los paquetes y clases junto con sus métodos revisar

Anexo 4. Desarrollo y documentación o en el repositorio:

<https://github.com/sandrocordova/AgenteInteligenteJava.git>

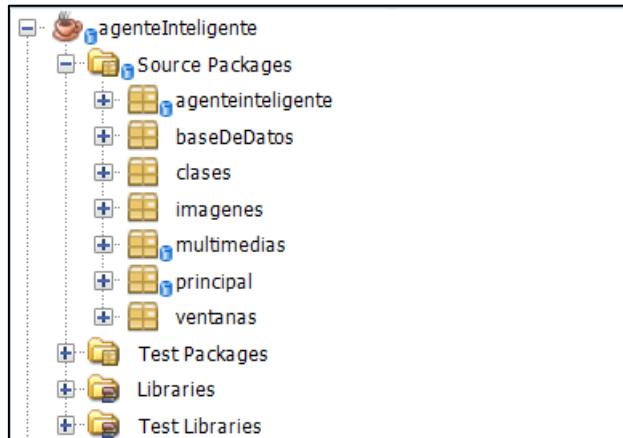


Figura 18. Estructura general.

Tabla 20. Detalles de la estructura general

Nombre	Descripción
<b>Agente Inteligente.</b>	Paquete que contiene la clase del agente inteligente junto con sus tareas.
<b>Base de Datos.</b>	Paquete que contiene la clase con la conexión a la base de datos, junto con los métodos de leer, escribir, editar y eliminar.
<b>Clases.</b>	Paquete que contiene las clases correspondientes a cada uno de los objetos que interactúan en el módulo de software.
<b>Imágenes.</b>	Paquete que contiene las imágenes utilizadas para las vistas.
<b>Multimedias.</b>	Paquete que contiene las clases del módulo de voz y el módulo para escucha.
<b>Principal.</b>	Paquete que contiene la clase principal para ejecutar el programa.
<b>Ventanas.</b>	Paquete que contiene las vistas.
<b>Test packages.</b>	Paquete que contiene las pruebas unitarias.
<b>Libraries.</b>	Contiene todas las librerías utilizadas.
<b>Test Libraries.</b>	Contiene las librerías de TestNG para las pruebas unitarias.

#### 6.1.17. Codificación del módulo de software

Esta sección contiene los resultados obtenidos durante la fase de codificación del módulo de software, para su presentación se tomará en cuenta cada una de las funcionalidades del sistema y se mostrará su interfaz gráfica junto con su código fuente.

Para poder ver el código fuente completo del módulo de software revisar el

Anexo 4. Desarrollo y documentación o en el repositorio:

<https://github.com/sandrocordova/AgenteInteligenteJava.git>

#### 6.1.17.1.1. Fase inicial

Abarca la funcionalidad inicial del sistema en donde el usuario que se comunica con el centro de llamadas tiene dos opciones: la opción uno es transferir su llamada al departamento de atención al cliente y la opción dos es ingresar centro de información de Covid 19.

##### Transferir llamada

Esta funcionalidad se encarga de analizar la selección del usuario, si el usuario selecciona la “Opción 1” se llama al método transferir\_llamada() en donde se transfiere la llamada a una de las diferentes extensiones que se encuentren habilitadas en la base de datos.

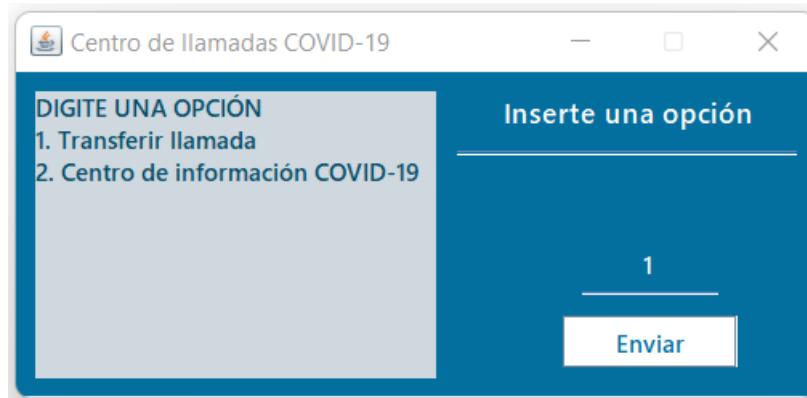


Figura 19. Selección de “Opción 1”

```
@Override  
public void actionPerformed(ActionEvent e) {  
    //Almacena el número de la llamada que ingresa  
    int entrada = Integer.parseInt(vista_opciones.cajaOpcion.getText());  
    llamada.setSeleccion(entrada);  
    //Abre comunicación con la base de datos  
    Conexion coneccion = new Conexion();  
    //Tomar decisión acorde a la selección del usuario  
    if (llamada.getSeleccion() == 1) {  
        //Lo registra en la base de datos  
        coneccion.insertarLlamada(llamada.getNum_telefono(), llamada.getSeleccion());  
        transferir_llamada(llamada);  
        vista_opciones.setVisible(false);  
    } else if (llamada.getSeleccion() == 2) {  
        //Lo registra en la base de datos  
        coneccion.insertarLlamada(llamada.getNum_telefono(), llamada.getSeleccion());  
        crear_agente(llamada);  
        vista_opciones.setVisible(false);  
    } else {  
        vista_opciones.anuncio1.setText("Opción no válida");  
    }  
}
```

Figura 20. Selección del usuario “Opción 1”.

```

public static void transferir_llamada(Llamada llamada) {
    Conexion conexion = new Conexion();
    ArrayList<Extension> listaExtensiones = new ArrayList<Extension>();
    listaExtensiones = conexion.buscarExtension();
    for (int i = 0; i < listaExtensiones.size(); i++) {
        Extension extension = new Extension();
        extension = listaExtensiones.get(i);
        if (extension.getEstado().contains("True")) {
            System.out.println("Su llamada ha sido transferida al:" + extension.getDetalle()
                + " extensión: " + extension.getExtension());
        }
    }
}

```

Figura 21. Método para transferir llamada.

La funcionalidad concluye con la asignación del paciente a una de las extensiones disponibles y notificando que su llamada ha sido transferida.

#### 6.1.17.1.2. Fase de recolección de datos

Si el usuario selecciona la “Opción 2”, se procede a la creación de un agente el cual es asignado al número telefónico de la llamada.

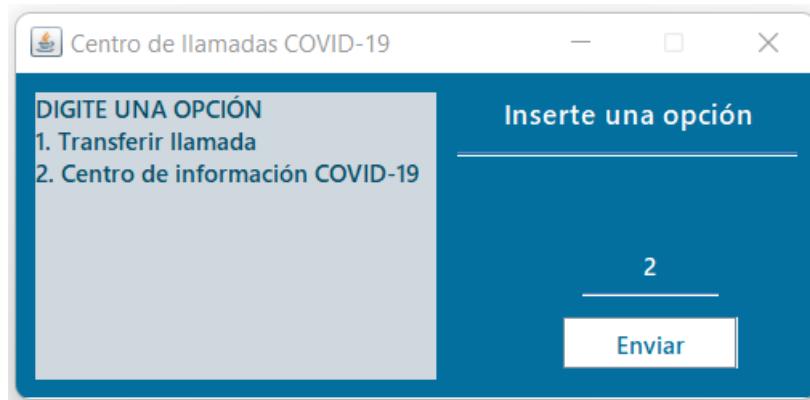


Figura 22. Selección de “Opción 2”

#### Creación y asignación de un agente inteligente

La “Opción 2” llama al método `crear_agente()` en donde se definen las variables requeridas para la creación del agente inteligente, se crea la rutina y el perfil a utilizar, en el perfil se especifica que el agente será levantado en el mismo servidor que la aplicación y que se hará uso de la GUI(interfaz gráfica de usuario). Posteriormente se le asigna un nombre único al agente haciendo uso del número telefónico del paciente y se procede a iniciar el agente con la función `start()`.

```

public static void crear_agente(Llamada llamada) {
    Runtime rt = Runtime.instance();
    Profile p = new ProfileImpl();
    p.setParameter(Profile.MAIN_HOST, "localhost");
    p.setParameter(Profile.GUI, "true");
    ContainerController cc = rt.createMainContainer(p);
    AgentController ac;

    Object[] arguments = new Object[3];
    arguments[0] = llamada.getNum_telefono();
    arguments[1] = llamada.getSeleccion();

    try {
        ac = cc.createNewAgent("Agente" + llamada.getNum_telefono(),
                               "agenteinteligente.AgenteInteligente", arguments);
        ac.start();
    } catch (Exception e) {
        System.out.println("ERRO al CREAR el Agente inteligente");
    }
}

```

Figura 23. Agente inteligente.

Si el agente ha sido creado con éxito se puede verificar mediante el mensaje arrojado en consola, de igual forma se puede evidenciar mediante la GUI que se despliega detallando la información del agente creado y al número telefónico asignado.

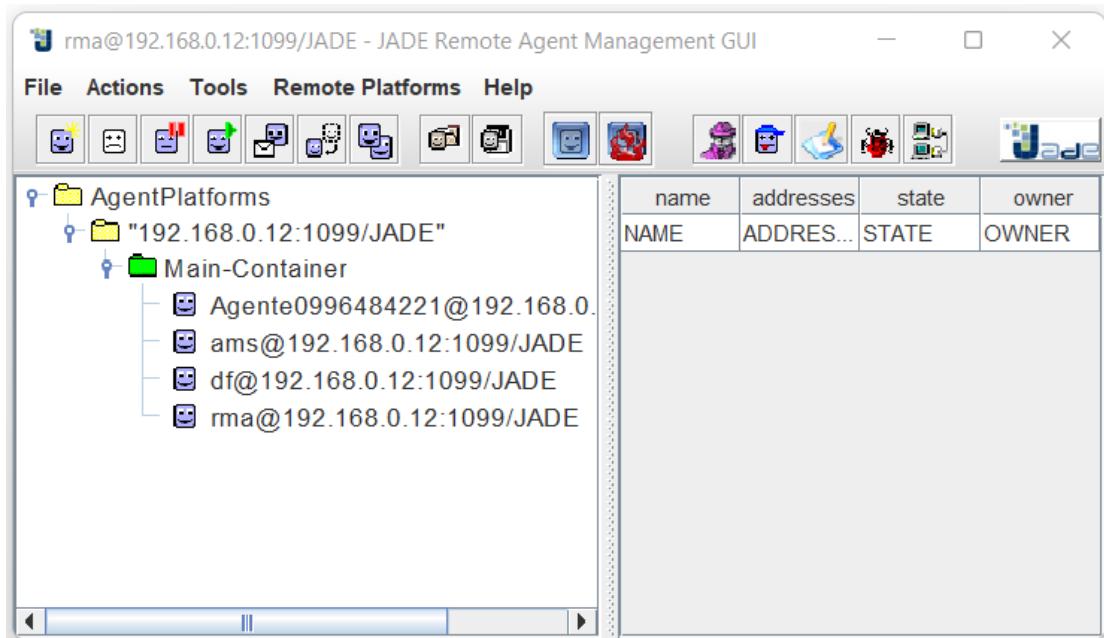


Figura 24. GUI de JADE

## Agente inteligente

Una vez creado el agente inteligente, este tiene un método inicial cuyo objetivo es cumplir con determinadas tareas al ser creado (`setup()`), culminada su primera tarea se procede a programar los comportamientos que este tendrá durante su ciclo de vida (`OneShotBehaviour()`). A continuación, se explica cada uno de estos comportamientos.

### Setup()

Se ejecuta automáticamente al crear el agente, como tarea inicial el agente obtiene los argumentos enviados desde la clase padre y procede a llamar al primer comportamiento programado para el agente (`OneShotBehaviour()`).

```
public void setup() {
    System.out.println("----AGENTE CREADO----");
    Object[] args = getArguments();
    System.out.println("Agente en ejecución: Agente" + args[0]);
    contAgentes = Integer.parseInt(String.valueOf(args[2]));
    //Se da una tarea inicial al agente inteligente
    addBehaviour(new OneShotBehaviour(this) {
        public void action() {
            //Llama al método DatosPaciente
            datosPaciente(String.valueOf(args[0]));
        }
    });
}
```

Figura 25. One Setup

### OneShotBehaviour()

Es un comportamiento que tendrá el agente, este se ejecuta una sola vez durante la vida del agente y es el método principal ya que aquí se solicita los datos del paciente (`datosPaciente()`) y esto desencadena una serie de eventos que culmina con el cierre de la comunicación.

```
//Se da una tarea inicial al agente inteligente
addBehaviour(new OneShotBehaviour(this) {
    public void action() {
        //Llama al método DatosPaciente
        datosPaciente(String.valueOf(args[0]));
    }
});
```

Figura 26. One Shot Behaviour

## Ingreso de datos del paciente

Para tener un registro del paciente se hace uso del método datosPaciente() que es el encargado de solicitar los datos personales del paciente para poder crear el objeto paciente, registrarlos en la base de datos y proceder a solicitar sus síntomas (sintomasPaciente()).



Figura 27. Solicitud de datos al paciente

```
private void datosPaciente(String numero_telefono) {
    //Conexion con la base de datos
    Conexion conexion = new Conexion();
    vista_persona vista_persona = new vista_persona();
    vista_persona.setVisible(true);
    //Termina sección
    Lee lee = new Lee();
    lee.leer("Por favor ingrese sus datos personales..... ");

    paciente = new Paciente();
    ActionListener al = new ActionListener() {
        //Se activa al hacer click en el botón llamar
        @Override
        public void actionPerformed(ActionEvent e) {
            //Almacena el número de la llamada que ingresa
            String nombre = vista_persona.cajaNombre.getText();
            String cedula = vista_persona.cajaCedula.getText();
            String correo = vista_persona.cajaCorreo.getText();
            int edad = Integer.parseInt(vista_persona.cajaEdad.getText());
            vista_persona.setVisible(false);
            paciente.setNum_telefono(numero_telefono);
            paciente.setNombre(nombre);
            paciente.setCedula(cedula);
            paciente.setCorreo(correo);
            paciente.setEdad(edad);
            //Lo registra en la base de datos
            conexion.insertarPaciente(numero_telefono, nombre, cedula, correo, edad);
            sintomasPaciente();
        }
    };
    vista_persona.botonEnviar.addActionListener(al);
}
```

Figura 28. Procesamiento de los datos del paciente

```

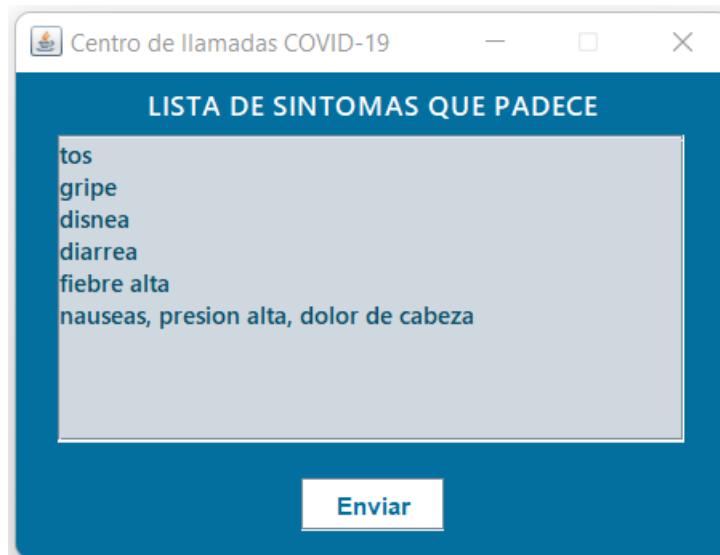
//Registra el paciente ingresado a la base de datos
public boolean insertarPaciente(String telefono,
    String nombre, String cedula, String correo, int edad) {
    documento.put("telefono", telefono);
    documento.put("nombre", nombre);
    documento.put("cedula", cedula);
    documento.put("correo", correo);
    documento.put("edad", edad);
    colecciónPaciente.insert(documento);
    return true;
}

```

*Figura 29. Almacenamiento de los datos del paciente*

### Listar síntomas del paciente

Para esta funcionalidad se desarrolló el método `sintomasPaciente()`, este método solicita al paciente el ingreso de los síntomas que posee, una vez ingresados los síntomas se procede con la clasificación de los mismos (`clasificarSintomas()`). Para poder pasar a la fase de clasificación, el método valida que se haya ingresado por lo menos un síntoma.



*Figura 30. Solicitud de síntomas al paciente*

```

//Método para obtener los síntomas del paciente
private void sintomasPaciente() {
    //Llamamos a la vista para el ingreso de los datos
    vista_sintomas vista_sintomas = new vista_sintomas();
    vista_sintomas.setVisible(true);

    Lee lee = new Lee();
    lee.leer("Por favor... ingrese los síntomas que posee ");

    ActionListener al = new ActionListener() {
        //Se activa al hacer click en el botón llamar
        @Override
        public void actionPerformed(ActionEvent e) {
            //Almacena el número de la llamada que ingresa
            sintomas = vista_sintomas.cajaSintomas.getText();
            if (sintomas.length() > 0) {
                vista_sintomas.setVisible(false);
                //Sección de Código para ubicar las pestañas
                if (contAgentes == 2) {
                    vista_sintomas.setLocation(600, 270);
                } else if (contAgentes == 3) {
                    vista_sintomas.setLocation(600, 540);
                } else {
                    vista_sintomas.setLocation(600, 10);
                }
                //Termina sección
                //Llamamos al método para clasificar los síntomas
                clasificarSintomas();
            } else {
                lee.leer("Información no encontrada ..... por favor intente de nuevo");
            }
        }
    };
    vista_sintomas.botonEnviar.addActionListener(al);
}

```

*Figura 31. Procesamiento de los datos del paciente*

#### **6.1.17.1.3. Clasificación del paciente**

##### **Clasificación del estado de los pacientes**

El método `clasificarSintomas()` se encarga de extraer de la base de datos los síntomas de referencia para realizar la comparativa con los síntomas ingresados por el paciente y asignarle un grado de prioridad.

La clasificación de los pacientes se da en base a los síntomas de referencia obtenidos de las encuestas aplicadas a profesionales en medicina (

Anexo 9. Informe de encuesta de síntomas) y los criterios de clasificación se dan en base al punto 4.18 Identificación de pacientes con Covid-19.

```
//Clasifica los síntomas ingresados por el paciente, los registra en la base de datos
//y toma decisiones acorde al resultado
private void clasificarSintomas() {
    //Se traen los síntomas de la base de datos
    Conexion conexion = new Conexion();
    ArrayList<Sintoma> sintomasReferencia = new ArrayList<Sintoma>();
    sintomasReferencia = conexion.buscarSintomas();
```

Figura 32. Extracción de los síntomas de referencia.

```
//ArrayList para almacenar los síntomas que el paciente tiene
ArrayList<Sintoma> sintomasPaciente = new ArrayList<Sintoma>();
// Comparo los síntomas de la base de datos con los ingresados por el usuario
for (int i = 0; i < sintomasReferencia.size(); i++) {
    Sintoma sintoma = new Sintoma();
    sintoma = sintomasReferencia.get(i);
    if (sintomas.contains(sintoma.getNombre())) {
        //registra los síntomas del paciente a la base de datos
        Conexion conexion2 = new Conexion();
        conexion2.insertarSintomaPaciente(paciente.getCedula(), sintoma.getNombre(),
            sintoma.getPrioridad());
        sintomasPaciente.add(sintoma);
        //Se asigna el grado de prioridad basado en los síntomas del paciente
        if (sintoma.getPrioridad() == 1) {
            contadorPrioridadUno++;
        } else if (sintoma.getPrioridad() == 2) {
            contadorPrioridadDos++;
        } else {
            contadorPrioridadTres++;
        }
    }
}
```

Figura 33. Estableciendo prioridad respecto a los síntomas de referencia.

Una vez obtenido el grado de prioridad de los síntomas se procede a establecer un diagnóstico, este diagnóstico es almacenado en la base de datos y se clasifica al paciente dependiendo del grado de prioridad.

- Si el paciente es clasificado como prioridad uno, se le brindan las medidas de bioseguridad.
- Si el paciente es clasificado como prioridad dos, se le brindan las medidas de bioseguridad y se agenda una cita médica.
- Si el paciente es clasificado como prioridad tres, se le brindan las medidas de bioseguridad, se agenda una cita médica y se receta medicina en caso de emergencia.

```

//Se clasifica al paciente acorde a la prioridad, se almacena el diagnostico en la base de datos
//y se toma decisiones
Conexion conexionDiagnostico = new Conexion();
if (contadorPrioridadTres >= 3) {
    diagnostico.setNumCedula(paciente.getCedula());
    diagnostico.setDiagnostico("Atención medica urgente");
    diagnostico.setDetalles("se recetó medicina");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadTres();
} else if (contadorPrioridadDos < 3) {
    diagnostico.setNumCedula("cedula" + paciente.getCedula());
    diagnostico.setDiagnostico("Atención medica puede esperar");
    diagnostico.setDetalles("No hay observaciones");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadDos();
} else if (contadorPrioridadDos >= 2) {
    diagnostico.setNumCedula(paciente.getCedula());
    diagnostico.setDiagnostico("Atención medica opcional");
    diagnostico.setDetalles("No hay observaciones");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadDos();
} else if (sintomas.length() > 0) {
    diagnostico.setNumCedula(paciente.getCedula());
    diagnostico.setDiagnostico("No requiere atención medica");
    diagnostico.setDetalles("No hay observaciones");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadUno();
}
}

```

Figura 34. Clasificación del paciente.

### Informar sobre medidas de bioseguridad

Si la clasificación del paciente lo requiere, se informan sobre las medidas de bioseguridad que este debe cumplir para prevenir riesgos de contagios.

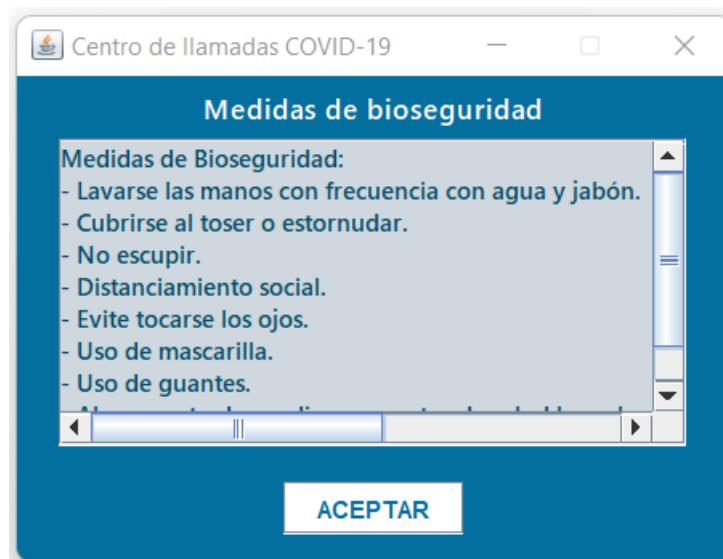


Figura 35. Medidas de bioseguridad

El método `medidasBioseguridad()` presentado en la Figura 36 se encarga de obtener las medidas de bioseguridad registradas en la base de datos y presentarlas al paciente.

```

//Metodo para presentar las medidas de bioseguridad al paciente
private void medidasBioseguridad() {
    String medidas = "Medidas de Bioseguridad: \n";
    //Establece conexión con la base de datos
    Conexion conexion = new Conexion();
    ArrayList<MedidaBioseguridad> listaMedidas = new ArrayList<MedidaBioseguridad>();
    listaMedidas = conexion.buscarMedidasBioseguridad();
    //For se encarga de formar el mensaje que se presentará al usuario
    for (int i = 0; i < listaMedidas.size(); i++) {
        MedidaBioseguridad medida = new MedidaBioseguridad();
        medida = listaMedidas.get(i);
        medidas = medidas + "- " + medida.getNombre() + ". \n";
    }
    //Presentar las medidas de bioseguridad al usuario
    vista_sintomas vista_sintomas = new vista_sintomas();
    vista_sintomas.anuncio.setText("Medidas de bioseguridad");
    vista_sintomas.cajaSintomas.setText(medidas);
    vista_sintomas.botonEnviar.setText("ACEPTAR");
    vista_sintomas.setVisible(true);
}

```

Figura 36. Método medidasBioseguridad()

### Agendar cita médica

Si la clasificación del paciente lo requiere se agenda una cita médica con sus datos y se le brindan los detalles de la cita médica agendada como la fecha, hora, nombre del paciente y su correo electrónico.



Figura 37. Cita generada

El método generarCita() presentado en la Figura 38 se encarga de crear el objeto Cita con los datos del Paciente, le asigna fecha y hora para posteriormente almacenarlo en la base de datos y brindar los detalles de la cita al paciente.

```

//Metodo para generar una cita al paciente
private void generarCita() {
    //Se inicializa el objeto cita
    Cita cita = new Cita();
    cita.setNombres(paciente.getNombre());
    cita.setEdad(paciente.getEdad());
    cita.setCedula(paciente.getCedula());
    cita.setCorreo(paciente.getCorreo());
    cita.setSintomas(cadenaSintomas);
    cita.setDiagnostico(diagnostico.getDiagnostico());
    cita.setNumTelefono(paciente.getNum_telefono());
    //Se establece un formato a la fecha de la cita
    SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy/h:mm");
    Date someDate = new Date();
    String fecha = sdf.format(new Date(someDate.getTime() + TimeUnit.DAYS.toMillis( 1 )));
    cita.setFecha(fecha);
    //Se almacena en la base de datos
    Conexion conexionCita = new Conexion();
    conexionCita.insertarCita(cita.getNombres(), cita.getEdad(), cita.getCedula(),
        cita.getCorreo(), cita.getSintomas(), cita.getDiagnostico(), cita.getFecha());
    //Presentar cita al usuario
    String citaTexto = "Se ha generado una cita\nna Nombre de: " + cita.getNombres()
        + "\n con el Correo electrónico: \n"
        + cita.getCorreo() + "\n Fecha de la cita:\n " + cita.getFecha();
    vista_sintomas vista_sintomas = new vista_sintomas();
    vista_sintomas.anuncio.setText("CITA GENERADA");
    vista_sintomas.cajaSintomas.setText(citaTexto);
    vista_sintomas.botonEnviar.setText("ACEPTAR");
    vista_sintomas.setVisible(true);
}

```

Figura 38. Cita generada

## Recetar en caso de emergencia

Si la clasificación del paciente lo requiere se le receta medicina en caso de emergencia.

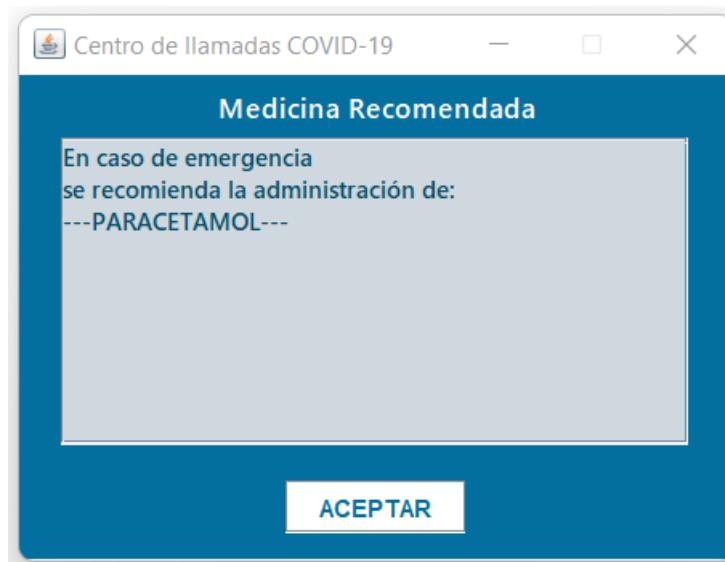


Figura 39. Medicina en caso de emergencia

El método recetarMedicina() presentado en la Figura 40 se encarga de informar al paciente sobre la medicina que puede administrarse en caso de emergencia.

```
//Método para recetar medicina al paciente
private void recetarMedicina() {
    //Presentar el medicamento al usuario
    String medicamentos = "En caso de emergencia ...\\n"
    |   + "se recomienda la administración de: " + "\\nPARACETAMOL";
    vista_sintomas vista_sintomas = new vista_sintomas();
    vista_sintomas.anuncio.setText("Medicina Recomendada");
    vista_sintomas.cajaSintomas.setText(medicamentos);
    vista_sintomas.botonEnviar.setText("ACEPTAR");
    vista_sintomas.setVisible(true);

    //Agregamos al botón el evento de cerrar la ventada
    ActionListener al = new ActionListener() {
        //Se activa al hacer click en el botón llamar
        @Override
        public void actionPerformed(ActionEvent e) {
            vista_sintomas.setVisible(false);
        }
    };
    vista_sintomas.botonEnviar.addActionListener(al);
}
```

Figura 40. Método recetar medicina

#### 6.1.17.1.4. Generar reportes

##### Reportes de pacientes

El administrador puede generar reportes a través de la interfaz gráfica, para generar un porte de los pacientes registrados, el administrador selecciona la opción “Pacientes” botón que llama al método buscarPacientes() el cual busca en la base de datos y proyecta la lista los pacientes registrados.

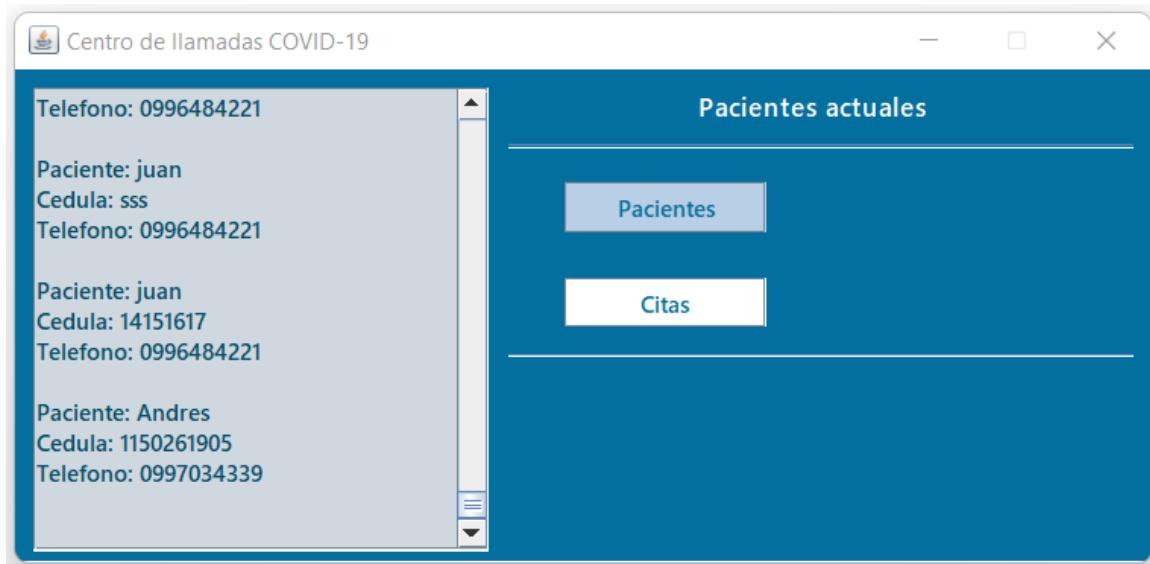


Figura 41. Vista “Generar reporte de pacientes”

```
//Se obtiene la lista de pacientes que se han comunicado con el call center
public ArrayList<Paciente> buscarPacientes() {
    DBCursor cursor = colecciónPaciente.find();
    ArrayList<Paciente> listaPacientes = new ArrayList<Paciente>();
    while (cursor.hasNext()) {
        Paciente paciente = new Paciente();
       DBObject object = cursor.next();
        paciente.setNum_telefono(String.valueOf(object.get("telefono")));
        paciente.setNombre(String.valueOf(object.get("nombre")));
        paciente.setCedula(String.valueOf(object.get("cedula")));
        paciente.setCorreo(String.valueOf(object.get("correo")));
        paciente.setEdad(Integer.parseInt(String.valueOf(object.get("edad"))));
        listaPacientes.add(paciente);
    }
    return listaPacientes;
}
```

Figura 42. Método buscarPacientes()

## Reportes de citas

Para generar un reporte de las citas agendadas, el administrador selecciona la opción “Citas” botón que llama al método buscarCitas() el cual busca en la base de datos y proyecta la lista de citas agendadas.

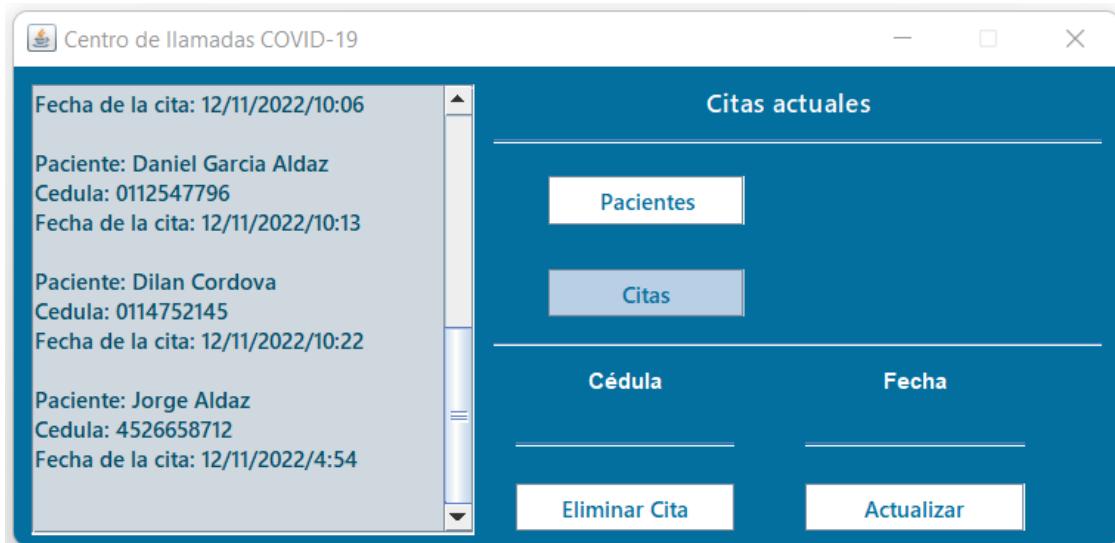


Figura 43. Vista “Generar reporte de citas”

```
//Se obtiene la lista de citas que se han comunicado con el call center
public ArrayList<Cita> buscarCitas() {
    DBCursor cursor = colecciónCita.find();
    ArrayList<Cita> listaCitas = new ArrayList<Cita>();
    while (cursor.hasNext()) {
        Cita cita = new Cita();
       DBObject object = cursor.next();
        cita.setNombres(String.valueOf(object.get("nombres")));
        cita.setEdad(Integer.parseInt(String.valueOf(object.get("edad"))));
        cita.setCedula(String.valueOf(object.get("cedula")));
        cita.setCorreo(String.valueOf(object.get("correo")));
        cita.setSintomas(String.valueOf(object.get("sintomas")));
        cita.setDiagnóstico(String.valueOf(object.get("diagnóstico")));
        cita.setFecha(String.valueOf(object.get("fecha")));
        listaCitas.add(cita);
    }
    return listaCitas;
}
```

Figura 44. Método buscarCitas ()

## Gestión de citas

Además, en el reporte de citas el administrador tiene la opción de modificar/eliminar las citas que se encuentren agendadas, para ello debe ingresar la cédula a la que pertenece la cita y colocar los valores a modificar. O si se desea eliminar la cita basta con colocar su cédula y presionar el botón “Eliminar Cita”.



Figura 45. Vista “actualizar/eliminar cita”

```
public boolean actualizarCita(Cita cita, String fechaNueva) {
    documento.put("cedula", cita.getCedula());

    BasicDBObject documentoNuevo = new BasicDBObject();
    documentoNuevo.put("nombres", cita.getNombres());
    documentoNuevo.put("edad", cita.getEdad());
    documentoNuevo.put("cedula", cita.getCedula());
    documentoNuevo.put("correo", cita.getCorreo());
    documentoNuevo.put("sintomas", cita.getSintomas());
    documentoNuevo.put("diagnostico", cita.getDiagnostico());
    documentoNuevo.put("fecha", fechaNueva);

   leccionCita.findAndModify(documento, documentoNuevo);
System.out.println("Cita con número de cédula: " + cita.getCedula()
    + " ha sido actualizada");
return true;
}
```

Figura 46. Método actualizarCita()

```
//Elimina la cita que coincide con el numero de cedula ingresado
public boolean eliminarCita(String cedula) {
    documento.put("cedula", cedula);
   leccionCita.remove(documento);
System.out.println("Cita con número de cédula: " + cedula + " ha sido eliminada");
return true;
}
```

Figura 47. Método eliminarCita()

#### **6.1.17.1.5. Aprendizaje del agente inteligente**

El agente inteligente debe ser entrenado por lo que es necesario disponer de una interfaz de administrador en donde se pueda mejorar la precisión de la interpretación hecha por el agente.

Para el entrenamiento del agente se desarrollaron dos módulos, el primer módulo es para mejorar el reconocimiento de símbolos y números, aquí las interfaces gráficas llaman al método menuSimbolos() el cual recibe como parámetro “numero” si se trata de números o vacío si se trata de símbolos, estos valores se almacenan en la base de datos para posteriormente ser usados cuando el agente procese la información.

The screenshot shows a window titled "Centro de llamadas COVID...". Inside, a blue header bar says "REGISTRAR NUMERO". Below it, there are two input fields: "Nombre:" followed by a text input containing "voz", and "Numero:" followed by a text input containing "2". At the bottom is a blue "REGISTRAR" button.

*Figura 48. Entrenamiento del agente (numeros)*

The screenshot shows a window titled "Centro de llamadas COVID...". Inside, a blue header bar says "REGISTRAR SÍMBOLO". Below it, there are two input fields: "Nombre:" followed by a text input containing "punto", and "Simbolo:" followed by a text input containing a single dot character. At the bottom is a blue "REGISTRAR" button.

*Figura 49. Entrenamiento del agente (símbolos)*

```

//Inserta los simbolos de referencia para poder comparar con valores ingresados
public boolean insertarSimbolo(String nombre, String simbolo, String condicion) {
    if (condicion == "Número") {
        documento.put("nombre", nombre);
        documento.put("numero", simbolo);
        colecciónNumeros.insert(documento);
    } else {
        documento.put("nombre", nombre);
        documento.put("simbolo", simbolo);
        colecciónSimbolos.insert(documento);
    }
    return true;
}

```

Figura 50. Almacenamiento de los números/símbolos

El segundo módulo es para mejorar el reconocimiento de los dominios web, aquí la interfaz gráfica llama al método menuDominio() el cual almacenada en la base de datos el dominio web que posteriormente será usado cuando el agente procese la información.



Figura 51. Entrenamiento del agente (dominios web)

```

public boolean insertarDominio(String dominioCompleto, String identificador) {
    documento.put("dominio", dominioCompleto);
    documento.put("identificador", identificador);
    colecciónDominio.insert(documento);
    return true;
}

```

Figura 52. Almacenamiento de los dominios

## Interpretación de datos

Para la interpretación de los datos ingresados por el usuario, el agente accede a los datos de entrenamiento, extrae los símbolos, números o dominios y los compara con la información ingresada por el usuario, busca similitudes entre las palabras y si estas coinciden total o parcialmente, las identifica y retorna ese valor como el correcto.

```
try {
    edad = Integer.parseInt(escuchar.leerTxt("numeros", 2));
} catch (Exception e) {
    lee.leer("Valor erróneo .... Por favor ingrese un valor numérico");
    break;
}
```

Figura 53. Llamada al método identificador de números

```
escuchar.escucharPython();
seleccionUsuario = escuchar.leerTxt("correo", 3);
if (!seleccionUsuario.contains("@")) {
    lee.leer("Correo electrónico no válido, por favor intente nuevamente");
    break;
}
```

Figura 54. Llamada al método identificador de dominios

```
public String buscarNumeros(String texto) {
    //obtiene los Numeros de la base de datos
    Conexion conexion = new Conexion();
    ArrayList<Numero> listaNumeros = new ArrayList<Numero>();
    listaNumeros = conexion.buscarNumeros();
    for (int i = 0; i < listaNumeros.size(); i++) {
        texto = texto.replace(listaNumeros.get(i).getNombre(),
            listaNumeros.get(i).getNumero());
    }
    return texto;
}
```

Figura 55. Método buscarNumeros()

```
public String buscarSimbolos(String texto) {
    //obtiene los Numeros de la base de datos
    Conexion conexion = new Conexion();
    ArrayList<Simbolo> listaSimbolos = new ArrayList<Simbolo>();
    listaSimbolos = conexion.buscarSimbolos();
    for (int i = 0; i < listaSimbolos.size(); i++) {
        texto = texto.replace(listaSimbolos.get(i).getNombre(),
            listaSimbolos.get(i).getSimbolo());
    }
    return texto;
}
```

Figura 56. Método buscarSimbolos()

```

public String correo(String correo) {
    correo = buscarNumeros(correo);
    String correoReturn = "";
    String[] arreglo = correo.split(" ");
    //obtiene los dominios de la base de datos
    Conexion conexion = new Conexion();
    ArrayList<Dominios> listaDominios = new ArrayList<Dominios>();
    listaDominios = conexion.buscarDominios();

    int encontrado = 0;
    for (int i = 0; i < arreglo.length; i++) {
        for (int j = 0; j < listaDominios.size(); j++) {
            if (arreglo[i].contains(String.valueOf(listaDominios.get(j).getIdentificador()))) {
                arreglo[i - 1] = "@";
                arreglo[i] = listaDominios.get(j).getDominioCompleto();
                for (int k = i + 1; k < arreglo.length; k++) {
                    arreglo[k] = "";
                }
            }
            break;
        }
    }
    for (int i = 0; i < arreglo.length; i++) {
        correoReturn = correoReturn + arreglo[i];
    }
    return correoReturn;
}

```

*Figura 57. Método identificador de dominios*

## Rendimiento

Para garantizar el rendimiento ya sea de forma individual del módulo de software o colectivo dentro del sistema del centro de llamadas telefónicas, la codificación del módulo de software se desarrolló con la implementación de buenas prácticas de programación, código limpio, reutilizable y escalable, realizando únicamente consultas necesarias y optimizadas a la base de datos haciendo uso de las funcionalidades de Java.

## **6.2. Objetivo II. Implementar el módulo de software del agente inteligente al sistema de llamadas telefónicas**

### **6.2.1. Costos de implementación de un centro de llamadas telefónicas**

La Tabla 21 presenta los costos de implementación de un centro de llamadas telefónicas, los cuales están estimados en el lapso de un año y se basan en los precios definidos por la corporación nacional de telecomunicaciones [36] y los valores de equipos acorde a los precios dados por tiendas en línea [37].

**Tabla 21. Costos de implementación de un centro de llamadas telefónicas.**

Detalle	Valor	Descripción	Total parcial
<b>Costo de línea telefónica</b>	\$60 + IVA	Al menos cuatro líneas para un centro de llamadas.	\$274
<b>Mensualidad</b>	\$12	Mínimo un año de servicio.	\$96
<b>Valor por consumo</b>	\$0.03	Aproximadamente \$250 mensuales.	\$1500
<b>Costo de instalación</b>	\$150	Valor por línea telefónica.	\$600
<b>Central telefónica</b>	\$500	Equipo encargado de recibir llamadas y asignar a una de las líneas.	\$500
<b>Terminal de comunicación fija</b>	\$50	Terminales fijas de entrada y salida.	\$200
<b>Computadora</b>	\$900	Computadora básica para la administración y gestión del módulo.	\$900
<b>Técnicos especializados</b>	\$500	Mano de obra para la instalación de equipos y adecuación del ambiente.	\$500
<b>Gastos varios</b>	\$400	Licencias, mantenimiento, daños estructurales, infraestructura.	\$400
<b>Total</b>			<b>\$4 900</b>

La implementación del módulo de software se dio en un entorno local, con la ayuda del diagrama general de la aplicación, diagrama de paquetes y diagrama de despliegue.

Los resultados obtenidos durante esta implementación son: el manual de despliegue de uso técnico en donde se especifican los requerimientos que debe tener el equipo para que pueda soportar el software, lista de librerías y dependencias necesarias, lista de herramientas utilizadas y los pasos a seguir para cumplir con el despliegue de la aplicación (

Anexo 8. Manual de despliegue para desarrolladores). Las siguientes secciones detallan cada uno de los resultados obtenidos.

### **6.2.2. Elementos y librerías a implantar**

#### **Portabilidad**

Para garantizar la portabilidad del módulo desarrollado, todo el software incluyendo las librerías utilizadas son de código abierto y son usadas en sus versiones estables según sus creadores. De igual forma, el hardware utilizado cumple con especificaciones básicas que tiene cualquier ordenador moderno.

La Tabla 22 muestra la lista de los elementos de Hardware y Software utilizados. Para garantizar la portabilidad

**Tabla 22. Requerimientos y características**

<b>Software</b>	<b>Hardware</b>
IDE NetBeans v12.6	Computador con mínimo 4GB de RAM.
JDK v17.0.2	Procesador Core i3, similar o superior.
Java v17.0.2	Altavoz conectado al equipo.
Mongo DB v5.0.1	Micrófono conectado al equipo.

La Tabla 23 muestra la lista de librerías y dependencias utilizadas.

**Tabla 23. Librerías y dependencias**

<b>Elemento/librería</b>	<b>Descripción</b>	<b>Ruta</b>
TalkingJavaSDK-170.	Librería para el módulo de voz.	Su instalación se detalla en la siguiente sección.
JADE v4.3.	Librería de Java para la gestión de agentes inteligentes.	Se carga directamente al proyecto.
MongoDB-connection v2.9.3.	Librería para la conexión y gestión de Mongo DB.	Se carga directamente al proyecto.
Speech v2.0	Librería para convertir texto a voz y voz a texto.	Viene por defecto con el JDK.
Swing v1.0.3	Librería para interfaz gráfica.	Viene por defecto con el JDK.

### **TalkingJavaSDK-170**

Esta librería consta de cuatro archivos principales, los mismos que deben instalarse de la siguiente manera:

1. El archivo “cgjsapi.jar” debe ser instalando como cualquier librería de Java en el proyecto principal.
2. De la misma forma el archivo “packet.jar” debe ser instalado como una librería dentro del proyecto principal.
3. Si el sistema utilizado no especifica una arquitectura de 64 bits, se debe mover el archivo “cgjsapi170.dll” a la carpeta bin del directorio raíz del JDK que usualmente tienen el siguiente directorio “C:\Program Files\Java\jdk-17.0.2\bin”.
4. Si el sistema utilizado cumple con una arquitectura de 64 bits se debe mover el archivo “cgjsapi170\_x64” a la carpeta bin del directorio raíz del JDK que usualmente tienen el siguiente directorio “C:\Program Files\Java\jdk-17.0.2\bin”.

### **JADE y MongoDB-connection**

Estas dos librerías cuentan con la instalación global de cualquier librería utilizadas por Java. Se descarga el archivo .jar desde cualquier repositorio en internet y se procede a añadir como librería del proyecto.

### 6.2.3. Implementación del módulo de software

A continuación, se presenta la instalación de los diferentes componentes utilizados en el módulo de software, para poder obtener detalles de versiones o la instalación paso a paso ver

Anexo 8. Manual de despliegue para desarrolladores.

#### Instalación de Java

```
Product Version: Apache NetBeans IDE 12.6
Java: 17.0.2; Java HotSpot(TM) 64-Bit Server VM 17.0.2+8-LTS-86
Runtime: Java(TM) SE Runtime Environment 17.0.2+8-LTS-86
System: Windows 11 version 10.0 running on amd64; Cp1252; es_MX (nb)
User directory: C:\Users\Usuario\AppData\Roaming\NetBeans\12.6
Cache directory: C:\Users\Usuario\AppData\Local\NetBeans\Cache\12.6
```

Figura 58. Instalación de Java.

#### Instalación de la base de datos

```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
MongoDB shell version v5.0.1
Connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("120a6222-45f1-4768-b2de-1c044ab56d02") }
MongoDB server version: 5.0.1
=====
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility. The "mongo" shell has been deprecated and
will be removed in
an upcoming release.
We recommend you begin using "mongosh".
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
=====
-- 
The server generated these startup warnings when booting:
  2021-08-03T12:26:18.988-05:00: Access control is not enabled for the database. Read an
d write access to data and configuration is unrestricted
-- 
-- 
  Enable MongoDB's free cloud-based monitoring service, which will then receive and disp v
```

Figura 59. Instalación de la base de datos Mongo DB.

## Instalación del JDK de Java y las librerías especificadas en la sección anterior.

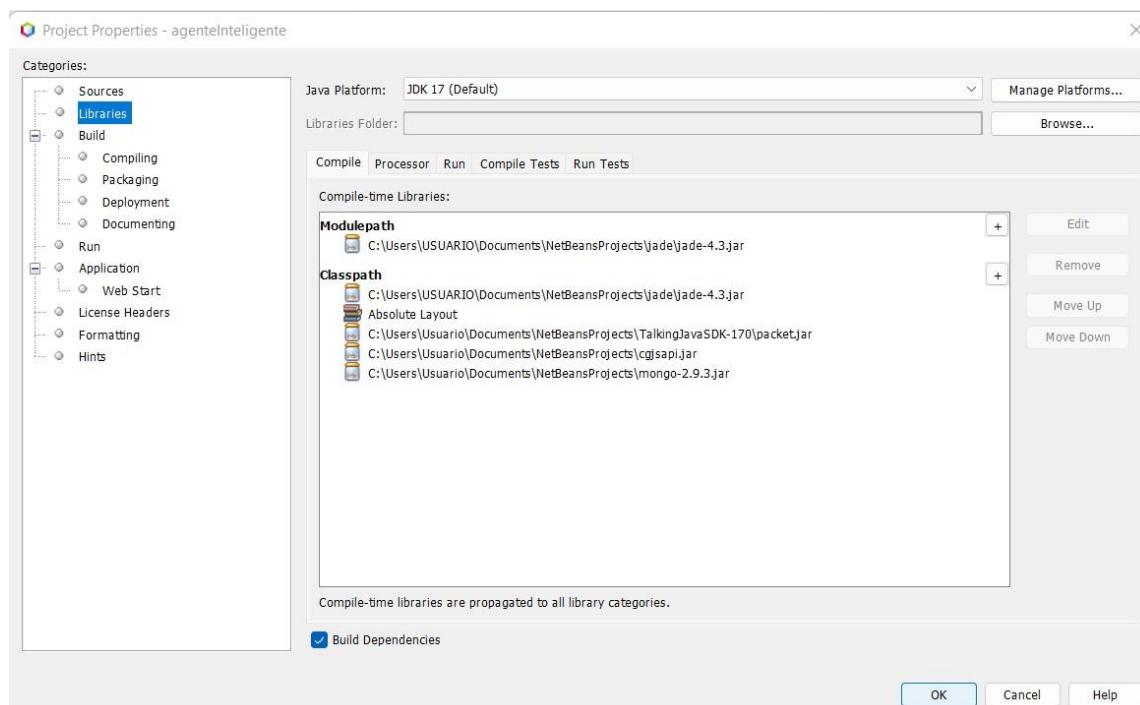


Figura 60. JDK y librerías.

## Instalar y ejecutar el módulo de software

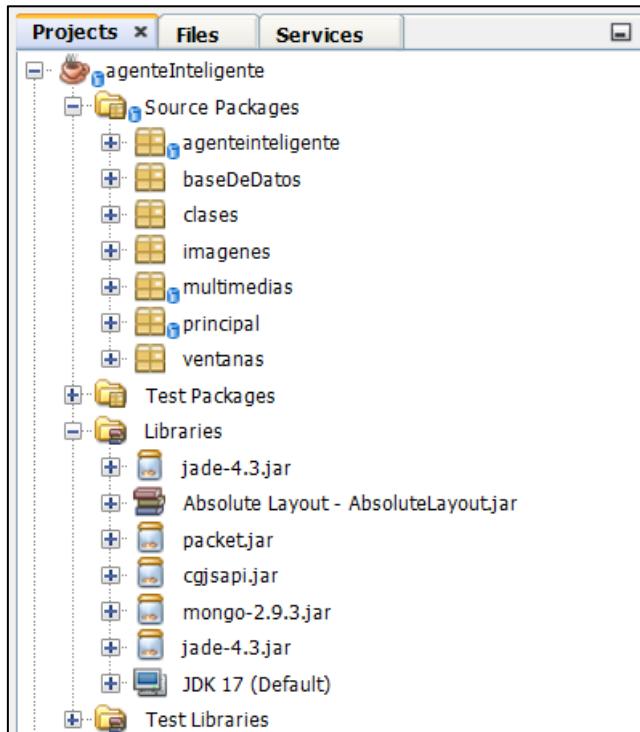


Figura 61. Instalación del módulo de software

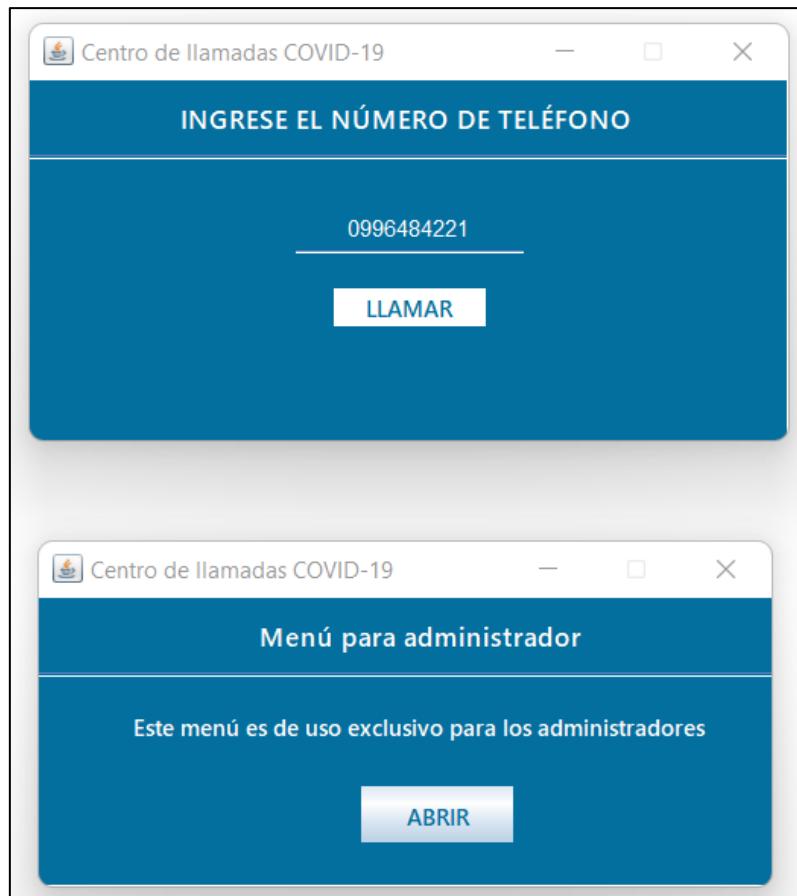


Figura 62. Pantalla principal del módulo de software.

#### 6.2.4. Pruebas unitarias

Las pruebas unitarias fueron aplicadas con la finalidad de corroborar el correcto funcionamiento del módulo de software a nivel de código, así como para identificar y solucionar los problemas que este tenga, se hizo uso del framework TestNG el cual viene implementado y es facilitado por Java debido a sus funcionalidades (

Anexo 5. Plan de pruebas Unitarias).

A continuación, la Tabla 24 presenta la lista de módulos y funcionalidades del módulo de software que fueron evaluados en las pruebas desarrolladas (

Anexo 4. Desarrollo y documentación).

**Tabla 24. Lista de módulos**

Módulo	Descripción
Módulo de transferencia	Transferir llamada
Módulo de clasificación	Clasificación del paciente
Módulo de decisiones	Medidas de bioseguridad
	Generar cita médica
	Recetar en casos de emergencia
Módulo de reportes de pacientes	Generar reporte de Pacientes
Módulo de gestión de citas	Visualizar cita
	Actualizar cita
	Borrar cita

**Funcionalidades a probar**

Las pruebas unitarias realizadas forman parte de las primeras pruebas aplicadas al sistema desarrollado, por lo tanto, se evaluaron todas las funcionalidades directamente involucradas con el fin de asegurar el correcto funcionamiento del módulo, estas pruebas fueron efectuadas desde el punto de vista del desarrollador.

**Participantes**

El encargado de efectuar las pruebas fue el desarrollador, quién también estuvo encargado de aplicar cada una de las correcciones necesarias para que el módulo software pueda cumplir con los criterios de aceptación.

**Objetivo**

Asegurar que cada uno de los métodos del código desarrollado brinde los resultados esperados.

**Desarrollo**

El desarrollo de estas pruebas se dará de forma general a cada uno de los módulos detallados en la Tabla 24. La evaluación de cada módulo debe cumplir con la siguiente secuencia:

1. Revisión de la especificación del componente en la documentación.
2. Revisión del código.
3. Revisión de la documentación del desarrollo del código.
4. Generar un caso de prueba.
5. Aplicación del caso de prueba.
6. Aplicación de criterios de aceptación o de rechazo en base a los resultados.
7. Documentación de los resultados.

**Pruebas unitarias con TestNG**

A continuación, se presentan las pruebas aplicadas a los diferentes métodos de las clases pertenecientes al módulo de software desarrollado, si el método es llamado de forma exitosa, la prueba se considera aprobada, si el método no es llamado de forma exitosa, se realiza el análisis y corrección del método y se vuelve aplicar la prueba.

### Prueba unitaria al método principal

Las siguientes figuras muestran los resultados de las pruebas unitarias realizadas al método principal que desencadena la secuencia de eventos tales como la transferencia de llamada y la creación de agentes.

```
@Test  
public void testMain() {  
    System.out.println("main");  
    String[] args = null;  
    Main.main(args);  
}
```

Figura 63. Prueba unitaria al método principal.

<b>Tests passed: 100.00 %</b> The test passed. (0.391 s)	main ===== Command line suite Total tests run: 1, Failures: 0, Skips: 0 =====
---	---

Figura 64. Resultado de prueba unitaria al método principal.

### Prueba unitaria al método “hay\_llamada”

Las siguientes figuras muestran los resultados de las pruebas unitarias realizadas al método “hay\_llamada” que detecta el ingreso de una llamada.

```
@Test  
public void testHay_llamada() {  
    System.out.println("hay_llamada");  
    Llamada llamada = null;  
    Main.hay_llamada(llamada);  
}
```

Figura 65. Prueba unitaria al método hay\_llamada

```

Tests passed: 100.00 %
The test passed. (0.913 s)

hay_llamada
Using 64-bit native code - SAPI4 is NOT supported
CloudGarden's JSAPI1.0 implementation
Version 1.7.0_x64
Implementation contained in files cgjsapi.jar and cgjsapi170_x64.dll
Esperando respuesta del usuario
Lectura del archivo
=====

```

Figura 66. Resultado de prueba unitaria al método `hay_llamada`

### Prueba unitaria al método “transferir\_llamada”

Las siguientes figuras muestran los resultados de las pruebas unitarias realizadas al método “`transferir_llamada`” que redirige al usuario en caso de que este lo requiera.

```

@Test
public void testTransferir_llamada() {
    System.out.println("transferir_llamada");
    Llamada llamada = null;
    boolean expResult = true;
    boolean result = Main.transferir_llamada(llamada);
    assertEquals(result, expResult);
}

```

Figura 67. Prueba unitaria al método `transferir_llamada`

```

Tests passed: 100.00 %
The test passed. (0.13 s)

transferir_llamada
Base de datos conectada...
Su llamada ha sido transferida al:Atención al cliente extensión: 12
=====
Command line suite
Total tests run: 1, Failures: 0, Skips: 0
=====
```

Figura 68. Resultado de prueba unitaria al método `transferir_llamada`

### Prueba unitaria al método “leer”

Las siguientes figuras muestran los resultados de las pruebas unitarias realizadas al método “`leer`” que lee la información al usuario.

```

@Test
public void testLeer() {
    System.out.println("leer");
    String texto = "Este es un ejemplo";
    Lee instance = new Lee();
    instance.leer(texto);
}

```

Figura 69. Prueba unitaria al método `leer`

The screenshot shows a terminal window with the following output:

```
Tests passed: 100.00 %
The test passed. (2.796 s)

leer
Using 64-bit native code - SAPI4 is NOT supported
CloudGarden's JSAPI1.0 implementation
Version 1.7.0_x64
Implementation contained in files cgjsapi.jar and cgjsapi170_x64.dll
=====
Command line suite
Total tests run: 1, Failures: 0, Skips: 0
```

Figura 70. Resultado de prueba unitaria al método leer

## Resultados

La lista de cambios aplicados al sistema después de efectuar las pruebas, se encuentran en la Tabla 25, así mismo, los detalles de las pruebas aplicadas a cada uno de los módulos, junto con los casos de prueba y los resultados detallados arrojados se presentan en el

Anexo 5. Plan de pruebas Unitarias.

**Tabla 25. Lista de correcciones aplicadas**

Corrección	Módulo	Detalle	Solución
C1	Todos los módulos	Mejora de las vistas presentadas al usuario.	Mejorar las vistas del usuario.
C2	Módulo de clasificación	La clasificación del paciente acorde a sus síntomas es errónea.	Revisión y corrección de los métodos encargados de la clasificación.
C3	Módulo de decisiones	Las decisiones tomadas acorde a la clasificación del paciente, se encuentran invertidas.	Analizar y corregir el orden de llamada de cada método.
C4	Módulo de clasificación	Los síntomas de la base de datos no coinciden con los resultados de la encuesta aplicada a los profesionales en la salud.	Revisar y corregir la base de datos.

### **6.3. Objetivo III: Evaluar la aceptación y correcto funcionamiento del módulo de software en el sistema de llamadas telefónicas**

En el presente apartado se indica el proceso que se llevó a cabo para el cumplimiento del tercer objetivo, para ello se efectuaron pruebas de aceptación a profesionales en medicina, con el fin de evaluar la aceptación del módulo de software desde el punto de vista médico, asimismo se aplicaron las pruebas de aceptación a los usuarios finales que son el grupo objetivo a quien está destinado el módulo de software.

En conjunto, se presenta en esta fase el manual de usuario con el fin de proporcionar la información necesaria para que el usuario final tenga los conocimientos requeridos para que pueda hacer uso del módulo software. En el mismo manual, además, se especifica cuál es la secuencia a seguir dentro del software y cuáles son los datos que serán necesarios para la interacción con el mismo (

Anexo 7. Manual de usuario).

### 6.3.1. Funcionamiento del módulo de software

Esta sección muestra cuál es la secuencia en la interfaz gráfica que sigue el usuario final al momento de interactuar con el módulo de software.

#### Caso de prueba

El caso de prueba presentado para observar el funcionamiento del módulo consiste en la clasificación de un usuario de prioridad alta, en donde el agente inteligente toma los síntomas ingresados por el usuario y se encarga de darles un grado de prioridad, ya sea para generar una cita médica, recetar en caso de emergencia o brindar medidas de bioseguridad.

#### Proceso

Una vez ingresados los datos personales del usuario Figura 71, el sistema le solicita listar los síntomas que posee Figura 72.

Centro de Llamadas COVID-19

DATOS PERSONALES

Nombre: Sandro

Cédula: 1150261905

Correo: sandro@gmail.com

Edad: 22

Pasaporte

ENVIAR

Figura 71. Ingreso de datos personales del usuario.

Centro de Llamadas COVID-19

LISTA DE SINTOMAS QUE PADECE

Tos  
Insomnio  
Gripe  
Dolor de cabeza

Enviar

Figura 72. Listar síntomas del usuario.

En este punto el sistema debe clasificar al usuario acorde al grado de prioridad que le corresponda según sus síntomas.

Una vez obtenido el grado de prioridad de los síntomas paciente, se procede a establecer un diagnóstico, este diagnóstico es almacenado en la base de datos y se clasifica al paciente dependiendo de su prioridad.

### Respuesta del módulo de software

Para el caso de prueba planteado el sistema debe clasificar al paciente como prioridad máxima, generando de manera automática una cita médica Figura 73, presentando las medidas de bioseguridad Figura 74 y presentando la medicina que puede ingerir en caso de emergencia Figura 75.



Figura 73. Cita generada.

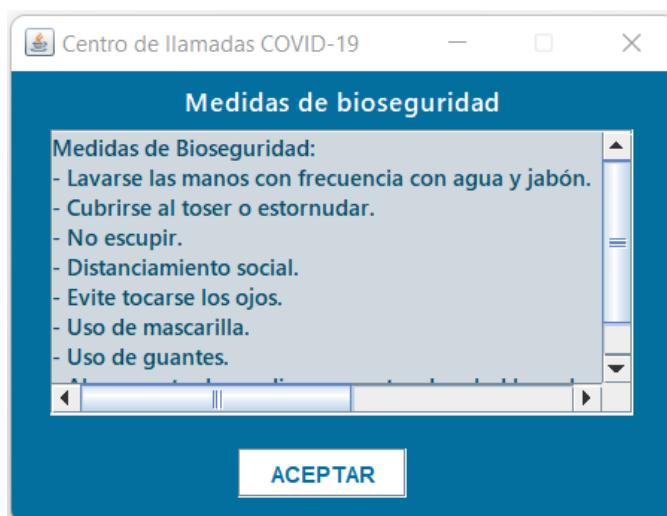
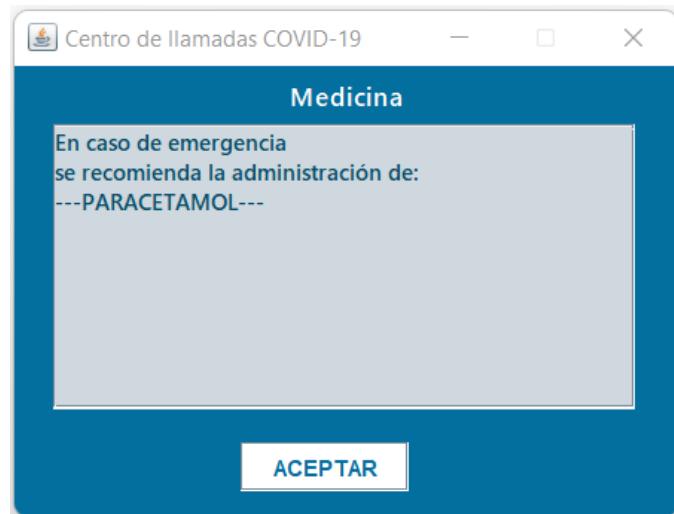


Figura 74. Medidas de bioseguridad.



*Figura 75. Recetar en caso de emergencia.*

Para el caso de prueba planteado, el módulo de software cumplió de manera exitosa con la clasificación del paciente, dando como resultado el correcto funcionamiento del método y permitiendo que el mismo sea sometido a los siguientes casos de pruebas.

Los detalles de las pruebas aplicadas a cada uno de los módulos, junto con los casos de prueba y los resultados arrojados se presentan en el

Anexo 4. Desarrollo y documentación.

### **6.3.2. Pruebas de aceptación a usuarios finales**

El presente punto muestra los resultados obtenidos de las pruebas de aceptación a usuarios finales, la información detallada de la encuesta aplicada como su diseño y análisis de resultados se encuentran en el

Anexo 11. Pruebas de aceptación a usuarios finales.

Las pruebas de aceptación fueron realizadas mediante el método de muestreo no probabilístico por conveniencia a dieciocho personas comunes entre los 20 a 49 años de edad que validan el funcionamiento de la aplicación.

#### **Objetivo**

Determinar el nivel de funcionalidad, facilidad de uso, y aceptación del módulo de voz implementado en el sistema desarrollado.

#### **Aplicación**

La presentación del módulo de software desarrollado fue aplicada de forma presencial a dieciocho estudiantes de la universidad nacional de Loja, aquí se procedió a informar a los encuestados sobre el funcionamiento de la aplicación, datos necesarios para su uso, la secuencia a seguir y cuál es el funcionamiento interno para la clasificación de los pacientes.

Después de presentar el sistema desarrollado, se procedió a la aplicación de la encuesta cuyas preguntas se encuentran listadas en la Tabla 26, la misma constó de 9 preguntas.

**Tabla 26. Lista de preguntas aplicadas**

Nº	Pregunta
P1	¿La aplicación le da opción a transferir su llamada? a. Sí b. No
P2	¿La aplicación le permite listar sus síntomas con precisión? a. Siempre b. A veces c. Nunca
P3	¿La aplicación le asigna una clasificación según los síntomas que usted ingresó? a. Sí b. No
P4	¿La aplicación le brinda información sobre las medidas de bioseguridad? a. Sí b. No
P5	¿La aplicación le genera una cita médica? (Si usted fue clasificado como prioridad 2 o prioridad 3) a. Sí

	b. No
P6	¿La aplicación le receta medicina de emergencia? (Si usted fue clasificado como prioridad 3) a. Sí b. No
P7	¿La aplicación al momento de usarla fue intuitiva y sencilla de usar? a. Siempre b. Casi siempre c. A veces
P8	¿La voz emitida por la aplicación es agradable y comprensible? a. Totalmente de acuerdo b. De acuerdo c. En desacuerdo
P9	¿Cree que la aplicación ayuda en el proceso de atención médica mediante llamadas telefónicas? a. Totalmente de acuerdo b. De acuerdo c. En desacuerdo

### Resultados de la encuesta

Tabla 27. Resultados pregunta 1

Alternativas	Frecuencia	Porcentaje
Sí	11	100%
No	0	0%
Total	11	100%

Tabla 28. Resultados pregunta 2

Alternativas	Frecuencia	Porcentaje
Sí	11	100%
No	0	0%
Total	11	100%

Tabla 29. Resultados pregunta 3

Alternativas	Frecuencia	Porcentaje
Sí	11	100%
No	0	0%
Total	11	100%

**Tabla 30. Resultados pregunta 4**

Alternativas	Frecuencia	Porcentaje
Sí	11	100%
No	0	0%
Total	11	100%

**Tabla 31. Resultados pregunta 5**

Alternativas	Frecuencia	Porcentaje
Sí	11	100%
No	0	0%
Total	11	100%

**Tabla 32. Resultados pregunta 6**

Alternativas	Frecuencia	Porcentaje
Sí	11	100%
No	0	0%
Total	11	100%

**Tabla 33. Resultados pregunta 7**

Alternativas	Frecuencia	Porcentaje
Siempre	11	95.24%
Casi siempre	0	4.76%
A veces	0	0%
Total	11	100%

**Tabla 34. Resultados pregunta 8**

Alternativas	Frecuencia	Porcentaje
Sí	11	100%
No	0	0%
Total	11	100%

**Tabla 35. Resultados pregunta 9**

Alternativas	Frecuencia	Porcentaje
Totalmente de acuerdo	11	100%
De acuerdo	0	0%
En desacuerdo	0	0%
Total	11	100%

### Evaluación de las hipótesis planteadas

- H1: El usuario puede transferir su llamada telefónica.

Los resultados obtenidos en la pregunta uno, permiten aceptar la hipótesis planteada ya que el cien por ciento de los encuestados manifiestan que el sistema si les permite transferir su llamada telefónica.

- H2: El usuario puede listar los síntomas que posee.

Los resultados obtenidos en la pregunta dos, permiten aceptar la hipótesis planteada ya que el cien por ciento de los encuestados manifiestan que el sistema si les permite enumerar los diferentes síntomas que poseen.

- H3: El sistema clasifica al paciente según los síntomas que posee.

Los resultados obtenidos en la pregunta tres, permiten aceptar la hipótesis planteada ya que el cien por ciento de los encuestados manifiestan que el sistema les brinda una clasificación según los síntomas ingresados.

- H4: El sistema brinda información de las medidas de bioseguridad, genera una cita médica o receta en caso de emergencia según la prioridad asignada al usuario.

La hipótesis es aceptada ya que los resultados obtenidos en las preguntas cuatro, cinco y seis, muestran que los 18 encuestados recibieron información sobre las medidas de bioseguridad, a 11 de ellos clasificados como prioridad 2 se les agendó una cita médica y a 2 de los 18 encuestados clasificados como prioridad 3 se les recetó medicina en caso de emergencia.

- H5: El sistema cuenta con una interfaz intuitiva y sencilla de usar.

La hipótesis es aceptada ya que los resultados obtenidos en la pregunta siete, muestran que el cien por ciento de los encuestados consideran que la aplicación es intuitiva y sencilla de usar.

- H6: El sistema cuenta con una voz aguda y agradable al oído.

La hipótesis se acepta ya que los resultados obtenidos en la pregunta ocho evidencian que el cien por ciento de los encuestados, afirman que la voz emitida por parte del programa es una voz aguda y agradable al oído.

- H7: La aplicación desarrollada ayuda en el proceso de atención médica mediante llamadas telefónicas.

La hipótesis se acepta ya que los resultados obtenidos en la pregunta nueve evidencian que el cien por ciento de los encuestados, consideran que la aplicación si ayuda en el proceso de atención médica mediante llamadas telefónicas.

## **Usabilidad**

La hipótesis 5 afirma que el sistema cuenta con una interfaz de uso intuitiva y sencilla dando por cumplido el requisito no funcional de usabilidad.

## **Voz aguda y agradable**

La hipótesis 6 afirma que el sistema cuenta con una voz aguda y agradable al oído del usuario durante la interacción, dando por cumplido el requisito no funcional de voz aguda y agradable.

### **6.3.3. Pruebas de aceptación a expertos en medicina**

El presente punto muestra los resultados obtenidos de las pruebas de aceptación, la información detallada de la encuesta aplicada como su diseño y análisis de resultados se encuentran en el Anexo 10. Pruebas de aceptación a expertos.

Las pruebas de aceptación fueron realizadas mediante el método de muestreo no probabilístico a seis expertos en el área de la salud pertenecientes al distrito 11D04 Celica-Pindal-Puyango cuyos datos se listan en la Tabla 36.

**Tabla 36. Lista de encuestados**

Nombres	Apellidos	Profesión	Cédula
Liliana Elizabeth	Sanmartín Cruz	Medico	1900752237
Hugo Ramiro	Córdova Córdova	Enfermero	0703296608
Johnny	Minga Tapia	Médico	1103410542
Bertha Beatriz	Narváez Briceño	Enfermera	1900523810
Andrés	Rivas	Enfermero	0706459518
Maricela Jesús	Tenesaca Álvarez	Enfermera	0350151593

## **Objetivo**

Determinar el nivel de funcionalidad, facilidad de uso, tiempos de respuesta y aceptación del módulo de voz implementado en el sistema desarrollado.

## **Aplicación**

La presentación del módulo de software desarrollado se dio mediante la herramienta Zoom, aquí se procedió a informar a los encuestados sobre el funcionamiento de la aplicación, datos necesarios para su uso, la secuencia a seguir y cuál es el funcionamiento interno para la clasificación de los pacientes.

Después de presentar el sistema desarrollado, se procedió a la aplicación de la encuesta cuyas preguntas se encuentran listadas en la Tabla 37, la misma constó de 13 preguntas y fue aplicada mediante la herramienta web Google Forms para facilitar la aplicación de la encuesta y a su vez el análisis de los datos obtenidos.

**Tabla 37. Lista de preguntas aplicadas**

Nº	Pregunta
P1	¿La aplicación al momento de usarla fue? d. Muy fácil de usar e. Fácil de usar f. Normal g. Difícil de usar h. Muy difícil de usar
P2	¿Requirió ayuda para realizar las tareas solicitadas? d. Siempre e. A veces f. Casi nunca g. Nunca
P3	¿Los tiempos de respuesta al realizar las tareas por parte de la aplicación, han sido adecuados? a. Adecuado b. Poco adecuado c. Muy adecuado
P4	¿Considera que la aplicación presentada es fácil de usar para cualquier persona con la capacidad de realizar una llamada? a. Totalmente de acuerdo b. De acuerdo c. En desacuerdo
P5	¿Las instrucciones de ingreso de datos solicitados por la aplicación son fáciles de seguir? a. Totalmente de acuerdo b. De acuerdo c. En desacuerdo
P6	¿La secuencia durante la interacción con la aplicación es intuitiva? a. Totalmente de acuerdo b. De acuerdo c. En desacuerdo
P7	¿La voz emitida por la aplicación es clara y comprensible? d. Totalmente de acuerdo e. De acuerdo f. En desacuerdo
P8	¿Indique el rango de tiempo estimado de interacción con la aplicación? a. 1 a 3 minutos b. 3 a 6 minutos c. 6 a 10 minutos d. Mayor a 10 minutos
P9	¿Cree que la clasificación de pacientes según sus síntomas realizada por el sistema es correcta? d. Totalmente de acuerdo e. De acuerdo f. En desacuerdo
P10	Indique el porcentaje estimado de efectividad en las tareas de funcionalidad de la aplicación. a. 20% b. 20% y 40% c. 40% y 60% d. 60% y 80% e. 80% y 100%

P11	¿Cree usted que la aplicación puede ser de ayuda para las personas que poseen Covid-19? c. Sí d. No
P12	¿Cree usted que la aplicación puede ser de ayuda para las instituciones de salud que reciben llamadas de personas con Covid-19? a. Sí b. No
P13	¿Tiene alguna sugerencia para la mejora de la aplicación?

## Resultados de la encuesta

Tabla 38. Resultados pregunta 1

Alternativas	Frecuencia	Porcentaje
Muy fácil de usar	2	33,3%
Fácil de usar	3	50%
Normal	1	16,7%
Difícil de usar	0	0%
Muy difícil de usar	0	0%
Total	6	100%

Tabla 39. Resultados pregunta 2

Alternativas	Frecuencia	Porcentaje
Siempre	0	0%
A veces	2	33.3%
Casi nunca	1	16,7%
Nunca	3	50%
Total	6	100%

Tabla 40. Resultados pregunta 3

Alternativas	Frecuencia	Porcentaje
Muy adecuado	2	33,3%
Adecuado	3	50%
Poco adecuado	1	16,7%
Total	6	100%

**Tabla 41. Resultados pregunta 4**

<b>Alternativas</b>	<b>Frecuencia</b>	<b>Porcentaje</b>
<b>Totalmente de acuerdo</b>	3	50%
<b>De acuerdo</b>	3	50%
<b>En desacuerdo</b>	0	0%
<b>Total</b>	6	100%

**Tabla 42. Resultados pregunta 5**

<b>Alternativas</b>	<b>Frecuencia</b>	<b>Porcentaje</b>
<b>Totalmente de acuerdo</b>	4	66.7%
<b>De acuerdo</b>	2	33.3%
<b>En desacuerdo</b>	0	0%
<b>Total</b>	6	100%

**Tabla 43. Resultados pregunta 6**

<b>Alternativas</b>	<b>Frecuencia</b>	<b>Porcentaje</b>
<b>Totalmente de acuerdo</b>	2	33.3%
<b>De acuerdo</b>	4	66.7%
<b>En desacuerdo</b>	0	0%
<b>Total</b>	6	100%

**Tabla 44. Resultados pregunta 7**

<b>Alternativas</b>	<b>Frecuencia</b>	<b>Porcentaje</b>
<b>Totalmente de acuerdo</b>	4	66.7%
<b>De acuerdo</b>	2	33.3%
<b>En desacuerdo</b>	0	0%
<b>Total</b>	6	100%

**Tabla 45. Resultados pregunta 8**

<b>Alternativas</b>	<b>Frecuencia</b>	<b>Porcentaje</b>
<b>1 a 3 min</b>	1	16.7%
<b>3 a 6 min</b>	2	33.3%
<b>6 a 10 min</b>	3	50%
<b>Más de 10 min</b>	0	0%
<b>Total</b>	6	100%

**Tabla 46. Resultados pregunta 9**

Alternativas	Frecuencia	Porcentaje
Totalmente de acuerdo	3	50%
De acuerdo	3	50%
En desacuerdo	0	0%
Total	6	100%

**Tabla 47. Resultados pregunta 10**

Alternativas	Frecuencia	Porcentaje
20%	0	0%
20% y 40%	0	0%
40% y 60%	1	16%
60% y 80%	2	33.3%
80% y 100%	3	50%
Total	6	100%

**Tabla 48. Resultados pregunta 11**

Alternativas	Frecuencia	Porcentaje
Sí	6	100%
No	0	0%
Total	6	100%

**Tabla 49. Resultados pregunta 12**

Alternativas	Frecuencia	Porcentaje
Sí	6	100%
No	0	0%
Total	6	100%

### Evaluación de las hipótesis planteadas

- H1: Es fácil e intuitivo de usar para el usuario.

De acuerdo a los resultados obtenidos en las preguntas uno y dos se pudo aceptar la hipótesis planteada ya que los encuestados consideran que la aplicación es fácil o muy fácil de usar y en su mayoría nunca necesitaron de ayuda para su manipulación.

- H2: Tiene una buena aceptación por parte de los expertos en medicina.

La hipótesis es aceptada, ya que en su mayoría los resultados arrojados por la encuesta han sido favorables, dejando claro que existe un alto nivel de aceptación, especialmente en las preguntas once y doce que permiten medir el grado de aceptación por parte de los profesionales en medicina.

- H3: La comunicación con la aplicación clara y comprensible.

La hipótesis es acepta, los resultados obtenidos de la pregunta siete reflejan que la mayoría, con cuatro encuestados, afirma que la voz emitida por parte del programa es clara y comprensible.

## 7. Discusión

**Objetivo I: Diseñar el módulo de software del agente inteligente para el sistema de llamadas telefónicas utilizando el lenguaje de programación Java y la metodología de desarrollo Kendall y Kendall.**

Para cumplir con el primer objetivo se empleó la metodología Kendall & Kendall junto con el modelo Scrum para adoptar una estrategia de desarrollo incremental y facilitar la organización de las tareas presentadas en las distintas fases de la metodología. Como se esperaba Kendall & Kendall fue de gran ayuda durante todo el proyecto ya que gracias a su documentación permitió llevar el proceso de forma rápida y eficiente, lo que coincide con los trabajos analizados [38] en donde se desarrolla un sistema de información y [39] en donde se presenta el diseño de un portal web, ambos trabajos cumplieron con éxito sus objetivos gracias al uso de la metodología mencionada.

Durante el desarrollo del proyecto se presentaron algunas limitaciones al utilizar la metodología Kendall & Kendall ya que requiere un alto nivel de interacción entre el usuario y el desarrollador, lo cual no fue posible debido a la falta de reuniones con el personal de los centros médicos, por tiempo limitado de los funcionarios o la no disponibilidad de los mismos. Para solventar la falta de reuniones requeridas, se optó por la realización de historias de usuario, con el fin de adquirir una idea general de las funcionalidades del módulo de software y así obtener los requerimientos del mismo.

Por otro lado, en la fase de desarrollo uno de los principales retos al integrar un agente inteligente, es su codificación, asignación de tareas y su integración con el entorno, lo cual fue solucionado gracias a la implementación de JADE (Java Agent Development Framework) coincidiendo con el análisis inicial de esta librería ya que brindó todas las herramientas necesarias para agilizar el proceso de desarrollo del módulo de software.

Limitaciones durante la codificación del módulo de software debido a las escasas librerías de código abierto para el reconocimiento de voz en JAVA, lo cual fue solventado al reciclar librerías encontradas en repositorios de otros proyectos.

**Objetivo II: Implementar el módulo de software del agente inteligente al sistema de llamadas telefónicas.**

Para la implementación del módulo de software se realizó la solicitud de implementación al Dr. Victor Hugo Tinoco Montaño, director distrital 11D04 Celica-Pindal-Puyango (

Anexo 12. Solicitud de implementación del módulo de software), quién luego de la presentación y validación del módulo de software, manifestó que su implementación no es posible debido a los recursos limitados que poseen las instituciones a su cargo. De igual forma, su implementación requiere de gran inversión por lo que no puede ser financiada netamente por el tesista (punto 6.2.1).

Debido a los factores mencionados, la implementación del módulo de software se hizo de forma simulada en un entorno local y planteando como trabajo a futuro la implementación del módulo de software en un centro médico.

Con el modelo 4+1 (

Anexo 3. Arquitectura de software) se logró la implementación del módulo de software ya que facilitó la representación de los diferentes tipos de modelos utilizados logrando que la arquitectura de software se documente de forma simple y eficiente.

Durante la implementación del módulo de software se observó que el módulo de voz implementado tenía muchas falencias durante el reconocimiento de voz ya que comparaba la voz ingresada con diferentes idiomas dando como resultado una interpretación imprecisa, esto fue solucionado al especificar que los datos ingresados serán únicamente en idioma español, por lo cual no se recomienda el uso del módulo de voz para detectar varios idiomas a la vez.

**Objetivo III: Evaluar la aceptación y correcto funcionamiento del módulo de software en el sistema de llamadas telefónicas.**

Para la validación a nivel técnico, inicialmente se consideró el uso de JUnit para efectuar las pruebas unitarias, sin embargo, se presentaron limitaciones como el no permitir el uso de objetos como parámetros ni la ejecución de pruebas en paralelo. Debido a esto se optó por el uso del marco de pruebas TestNG ya que logra solventar las limitaciones de JUnit facilitando la aplicación de los diferentes casos de prueba y evaluar el funcionamiento de cada uno de los componentes del módulo desarrollado.

Las pruebas funcionales permitieron determinar que los requerimientos planteados se cumplieron a cabalidad, evaluando su funcionamiento en diversas situaciones generadas y obteniendo resultados positivos en la detección y clasificación de pacientes con Covid-19.

Así mismo, las pruebas de aceptación fueron claves para determinar que el módulo de software tiene un alto grado de aceptación, ya que tanto los expertos en medicina como los usuarios finales concuerdan en que el módulo de software es de ayuda para el proceso de atención médica mediante llamadas telefónicas de un centro médico. Las pruebas de aceptación se efectuaron mediante el muestreo no probabilístico por conveniencia a seis expertos en medicina pertenecientes al distrito 11D04 Celica-Pindal-Puyango (Anexo 10. Pruebas de aceptación a expertos) y a veintidós usuarios finales (

Anexo 11. Pruebas de aceptación a usuarios finales) permitiendo así definir que el módulo desarrollado cumple con las necesidades y requerimientos planteados.

## **8. Conclusiones**

Una vez realizado el módulo software planteado en el presente trabajo de titulación se concluye:

- El uso de agentes inteligentes sirven en el proceso de atención médica mediante llamadas telefónicas ayudando a disminuir la saturación de los centros de salud, ya que realiza la identificación y clasificación de pacientes en menos de tres minutos, duración que es conveniente comparada con el proceso normal que puede tardar minutos, horas e incluso días hasta la asignación de una cita.
- La metodología Kendall & Kendall debe ser usada en proyectos que requieren gran cantidad de documentación y de una interacción constante con el cliente. Por otro lado, al combinarla con el modelo SCRUM ayuda en la planificación, diseño y evaluación de las diferentes etapas de codificación, logrando en conjunto un proceso organizado y coordinado que fue fundamental para el cumplimiento de los objetivos planteados.
- El uso del estándar IEEE 830 en la fase de especificación de requerimientos de información, permitió documentar de forma idónea los 9 requisitos funcionales y 7 requisitos no funcionales dados por los centros de atención médica, brindando así el punto de partida para el desarrollo del módulo de software.
- Los modelos desarrollados en el modelo de arquitectura 4+1 y el manual de despliegue para desarrolladores aportaron significativamente en la fase de implementación del módulo de software, mejorando la representación de su arquitectura y por ende facilitando el despliegue de sus componentes.
- Los resultados de las pruebas de aceptación determinaron que el módulo desarrollado es aprobado por los usuarios finales, ya que cumple con los requerimientos planteados por los centros de atención médica. De igual forma, las mismas pruebas permitieron validar el módulo desarrollado desde el punto de vista profesional para corroborar el cumplimiento de los requerimientos.

## **9. Recomendaciones**

Después de completar el trabajo de titulación, las recomendaciones son:

- Utilizar la metodología Kendall y Kendall para proyectos que requieran de un alto nivel de documentación y una interacción constante con el cliente sobre todo en la fase de obtención de requerimientos.
- Hacer uso de JADE (Java Agent Development Framework) para la implementación de agentes inteligentes en Java, ya que posee funcionalidades que ayudan a reducir los tiempos de planificación y codificación.
- Utilizar módulos de voz en un solo idioma para mejorar tiempos de interpretación y mejorar su precisión en el reconocimiento de voz.
- Hacer uso del estándar IEEE para la documentación de los requerimientos.

Se recomienda para trabajos futuros:

- Expandir la base de datos utilizada por el módulo software desarrollado para mejorar la precisión durante la clasificación de los pacientes.
- Mejorar el entrenamiento del agente inteligente para que este pueda aprender de forma no supervisada.
- La implementación del módulo de software en un centro de atención médica, con el equipo de un centro de llamadas necesario y los dispositivos de entrada y salida requeridos.

## 10. Bibliografía

- [1] Ministerio de Salud Pública del Ecuador (MSP), “Evidencia sobre el tratamiento de Covid-19, con medicina ancestral,” *Journal of Clinical Investigation*, vol. 130, no. 5, p. 192, 2020.
- [2] M. de salud Publica, “GacetaEpidemiológicaSemanalNo.34,” vol. 148, pp. 148–162.
- [3] Ministerio de salud pública, “Agendamiento de Citas Médicas – Ministerio de Salud Pública”.
- [4] P. Boucher, “Artificial intelligence: How does it work, why does it matter, and what can we do about it?,” 2020.
- [5] L. Rouhiainen, *Inteligencia artificial: 101 cosas que debes saber hoy sobre nuestro futuro*.
- [6] S. Badaró, L. Javier Ibañez, and M. J. Agüero, “Sistemas Expertos: Fundamentos, Metodologías y Aplicaciones.”
- [7] B. López, *Inteligencia Artificial*. Reading: Nuevo Laredo, Tam, 2005.
- [8] A. Samarin, *AGENTES SOFTWARE Y REDES ACTIVAS*. Reading, 2017.
- [9] Jorge Martínez Ladrón de Guevara, “Fundamentos de programación en Java”.
- [10] A. Franco Barrios, “MANUAL DE JADE PARA PRINCIPIANTES.”
- [11] J. Ubertino Jaramillo Zhingre Director, “CONSTRUCCIÓN DE UN SISTEMA DE DIÁLOGO COMPLETO, EN LENGUA ESPAÑOLA, PARA LA BIBLIOTECA BRAILLE DEL HONORABLE CONSEJO PROVINCIAL DE LOJA, EL CUAL SE ENCARGARA DE PROPORCIONAR INFORMACIÓN BIBLIOGRÁFICA A LAS ESTUDIANTES NO VIDENTES.”
- [12] Karl Seguin, “The little MongoDB book.”
- [13] C. Beust and H. Suleiman, *Next Generation Java Testing: TestNG and Advanced Concepts*. Addison-Wesley Professional, 2007.
- [14] A. N. Cadavid, J. Daniel Fernández Martínez, and J. Morales Vélez, “Revisión de metodologías ágiles para el desarrollo de software A review of agile methodologies for software development.”
- [15] “Modelado e implementación de un sistema multiagente”.

- [16] F. González, S. Calero, and D. Loaiza, “Comparación de las metodologías cascada y ágil para el aumento de la productividad en el desarrollo de software ,” *Universidad Santiago de Cali, Facultad de Ingeniería, Programa de Ingeniería Industrial*, vol. 11, 2019.
- [17] J. G. Navarro, “Estudio Comparativo De Metodologías, Herramientas Y Wiki De Soporte Para La Gestión De Proyectos De Desarrollo De Software,” *Universitat Oberta de Catalunya*, p. 90, 2018.
- [18] J. M. Q. Xavier Quiñónez-Ku, Juan Casierra Cavada, Luis Herrera-Izquierdo, “Análisis Comparativo de metodologías Agiles de desarrollo de software: una revisión bibliográfica Comparative analysis of agile software development methodologies : a bibliographic review,” *ResearchGate*, no. June, 2019.
- [19] Adriana. Peralta, “Metodología scrum,” *Universidad ORT Uruguay*, p. 12, 2003, [Online]. Available: <https://fi.ort.edu.uy/innovaportal/file/2021/1/scrum.pdf>
- [20] K. Kendall and J. Kendall, *Analisis Y Diseño De Sistemas*. 2011. [Online]. Available: [www.FreeLibros.me](http://www.FreeLibros.me)
- [21] T. Otzen and C. Manterola, “Técnicas de Muestreo sobre una Población a Estudio Sampling Techniques on a Population Study,” 2017.
- [22] Organización Mundial de la Salud, “Manejo clínico de la COVID-19 27 de mayo de 2020 Orientaciones provisionales.”
- [23] Organización Panamericana de la Salud, “Definiciones de casos para la vigilancia COVID-19.”
- [24] A. A. Hidalgo, “Lineamientos para la prevención COVID-19 e Inocuidad de los Alimentos REGISTRO DE REVISIÓN Y APROBACIÓN DEL DOCUMENTO Acción Nombre / Cargo Institución Firma y Fecha Elaborado por,” p. 23, 2020.
- [25] Q. Qian, H. Yan, and Z. Wang, “Application Prospect of Artificial Intelligence Technology Basted on the Large Call Center,” Jun. 2020, pp. 1–6. doi: 10.1145/3424978.3425012.
- [26] Michael McTear, “Conversational\_AI,” 2020.
- [27] S. Hanke, E. Sandner, S. Kadyrov, and A. Stainer-Hochgatterer, “Daily life support at home through a virtual support partner,” 2016.

- [28] S. Hussain and A. Ginige, "Extending a Conventional Chatbot Knowledge Base to External Knowledge Source and Introducing User Based Sessions for Diabetes Education," Jun. 2018, pp. 698–703. doi: 10.1109/WAINA.2018.00170.
- [29] A. de La and H. Manotas, "Inge-CUC-Revista de la Facultad de Ingeniería MULTI-AGENT SYSTEM DESIGN FOR NETWORK MONITORING USING JADE AND JPCAP," 2010.
- [30] Sandro Michael Córdova Carrión, "REVISIÓN SISTEMÁTICA DE LITERATURA PARA UN SISTEMA DE LLAMADAS TELEFÓNICAS PARA LA PRESTACIÓN DE SERVICIOS MEDIANTE UN AGENTE INTELIGENTE," 2021.
- [31] Sandro Córdova, "Sistema de información sobre COVID-19 mediante llamadas telefónicas con la ayuda de un agente inteligente," 2021.
- [32] A. Rodríguez Jiménez and A. Omar Pérez Jacinto, "Métodos científicos de indagación y de construcción del conocimiento," pp. 179–200, 2017, doi: 10.21158/01208160.n82.2017.1647.
- [33] R. Ruiz, "El Metodo Científico y sus Etapas," *Mexico*, vol. 2, p. 79, 2011.
- [34] Ministerio de salud pública del Ecuador, "Establecimientos de salud de primer nivel autorizados."
- [35] EDINA, "Clinicas y hospitales en Loja - Ecuador," *clinicas y hospitales en Loja - Ecuador*, 2022. <https://www.edina.com.ec/Buscador?b=clinicas+y+hospitales&c=Loja> (accessed Jun. 29, 2022).
- [36] CNT, "CNT Empresarial," 2022. <https://empresas.cnt.com.ec/> (accessed Jul. 12, 2022).
- [37] Mercado Libre, "Equipo para Call Center," 2022. <https://listado.mercadolibre.com.ec/equipo-para-call-center> (accessed Jul. 12, 2022).
- [38] L. Iván Suárez Orduña and M. Patiño Ortiz Julián Patiño Ortiz, "Sistema de información para el control de los derechos de autor."
- [39] Córdova Eder, Pérez Hugo, and Tabares Julio, "PORTAL WEB PARA LA GERENCIA DE INFORMATICA", doi: 10.944.276.

## **11. Anexos**

**Anexo 1.** Especificación de requisitos de software IEEE 830

---

### **Especificación de requisitos de software IEEE 830**

**Proyecto:** Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente

---

## Ficha de documento

**Tabla 1. Ficha de documento**

<b>Fecha</b>	<b>Revisión</b>	<b>Autor</b>	<b>Verificado y Validado</b>
26/06/2021	0.1	Sandro Michael Córdova Carrión	Ing. José Oswaldo Guamán Quinche
29/03/2022	0.2	Sandro Michael Córdova Carrión	Ing. José Oswaldo Guamán Quinche
04/07/2022	0.2	Sandro Michael Córdova Carrión	Ing. José Oswaldo Guamán Quinche
07/07/2022	0.3	Sandro Michael Córdova Carrión	Ing. José Oswaldo Guamán Quinche

## **1. Introducción**

Este documento es una Especificación de Requisitos Software (ERS) para el “Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente”. Esta especificación se ha estructurado basándose en las directrices dadas por el estándar IEEE 830.

### **1.1. Propósito**

El presente documento tiene como propósito definir las especificaciones funcionales y no funcionales para el diseño del “Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente”.

### **1.2. Alcance**

La solución informática tiene como objetivo realizar el diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas, por tal motivo los requisitos están dirigidos a los usuarios del sistema, para continuar con el desarrollo de la aplicación y profundizar en la automatización de ésta.

### **1.3. Personal involucrado**

**Tabla 2. Desarrollador del proyecto**

<b>Nombre</b>	Sandro Michael Córdova Carrión
<b>Rol</b>	Analista, diseñador y programador
<b>Categoría Profesional</b>	Estudiante
<b>Responsabilidad</b>	Análisis de información, diseño y programación de la solución informática.
<b>Información de contacto</b>	sandro.cordova@unl.edu.ec

**Tabla 3. Director del trabajo de titulación**

<b>Nombre</b>	José Oswaldo Guamán Quinche
<b>Rol</b>	Director del Trabajo de Titulación
<b>Categoría Profesional</b>	Ingeniero en Sistemas
<b>Responsabilidad</b>	Supervisar y asesorar en el desarrollo del Trabajo de Titulación.
<b>Información de contacto</b>	jose.o.guaman@unl.edu.ec

#### **1.4. Definiciones, acrónimos y abreviaturas**

**Tabla 4. Definiciones, acrónimos y abreviaturas**

Nombre	Descripción
<b>Usuario</b>	Persona que usará el sistema.
<b>ERS</b>	Especificación de Requisitos Software.
<b>RF</b>	Requerimiento Funcional.
<b>RNF</b>	Requerimiento No Funcional.

#### **1.5. Referencias**

**Tabla 5. Referencias**

Título del Documento	Referencia
Estándar IEEE 830 - 1998	IEEE

#### **1.6. Resumen**

Este documento consta de tres secciones. En la primera sección se realiza una introducción al mismo y se proporciona una visión general de la especificación de recursos del sistema.

En la segunda sección del documento se realiza una descripción general del sistema, con el fin de conocer las principales funciones que éste debe realizar, los datos asociados, así como los factores, restricciones, supuestos y dependencias que afectan al desarrollo, sin entrar en detalles excesivos.

Por último, la tercera sección del documento es aquella en la que se definen detalladamente los requisitos que debe satisfacer el sistema.

## **2. Descripción general**

### **2.1. Perspectiva del producto**

La solución informática será diseñada para trabajar en entornos de escritorio, lo que permitirá su utilización de forma rápida y eficaz.

## 2.2. Funcionalidad del producto

**Tabla 6. Historia de usuario 01**

<b>Historia de Usuario</b>	
<b>Número:</b> 01	<b>Usuario:</b> Paciente
<b>Nombre de la historia:</b> Transferir llamada	
<b>Prioridad:</b> Alta	<b>Riesgo en desarrollo:</b> Medio
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b>  Como paciente quiero que mi llamada sea transferida al departamento de atención al cliente del centro médico para comunicarme directamente con una persona.	
<b>Criterio de aceptación:</b>  Asignación de la llamada al departamento de atención al cliente.	
<b>Observaciones:</b> s/n	

**Tabla 7. Historia de usuario 02**

<b>Historia de Usuario</b>	
<b>Número:</b> 02	<b>Usuario:</b> Sistema
<b>Nombre de la historia:</b> Clasificación del paciente.	
<b>Prioridad:</b> Alta	<b>Riesgo en desarrollo:</b> Alta
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b>  Como paciente quiero que el sistema me clasifique acorde a mis síntomas para poder recibir la atención médica requerida.	
<b>Criterio de aceptación:</b>  El paciente es clasificado con un grado de prioridad alta. El paciente es clasificado con un grado de prioridad media. El paciente es clasificado con un grado de prioridad baja.	
<b>Observaciones:</b> <ul style="list-style-type: none"><li>• Clasificaciones posibles: Prioridad baja, prioridad media, prioridad alta.</li><li>• Prioridad baja: El sistema brinda medidas de bioseguridad (HU3).</li><li>• Prioridad media: El agente inteligente brinda medidas de bioseguridad (HU03), es opcional la generación de una cita (HU4).</li><li>• Prioridad alta: El agente inteligente brinda medidas de bioseguridad (HU03), el agente inteligente genera una cita médica (HU4), el agente inteligente receta medicamentos (HU5).</li></ul>	

**Tabla 8. Historia de usuario 03**

<b>Historia de Usuario</b>	
<b>Número:</b> 03	<b>Usuario:</b> Sistema
<b>Nombre de la historia:</b> Medidas de bioseguridad	
<b>Prioridad:</b> Alta	<b>Riesgo en desarrollo:</b> Medio
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b>  Como paciente quiero que se me brinden las medidas de bioseguridad para prevenir contagios.	
<b>Criterio de aceptación:</b>  El paciente recibe una lista de medidas de bioseguridad.	

<b>Observaciones:</b> s/n
---------------------------

**Tabla 9. Historia de usuario 04**

<b>Historia de Usuario</b>	
<b>Número:</b> 04	<b>Usuario:</b> Sistema
<b>Nombre de la historia:</b> Generar cita médica	
<b>Prioridad:</b> Alta	<b>Riesgo en desarrollo:</b> Alto
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b> Como paciente quiero generar una cita médica para atenderme en el centro médico.	
<b>Criterio de aceptación:</b> Se genera una cita médica con los datos del paciente. Se presenta los datos de la cita médica al paciente.	
<b>Observaciones:</b> Únicamente los pacientes clasificados como prioridad alta pueden generar una cita médica. Información requerida: Número de cédula, nombres, apellidos y correo electrónico.	

**Tabla 10. Historia de usuario 05**

<b>Historia de Usuario</b>	
<b>Número:</b> 05	<b>Usuario:</b> Sistema
<b>Nombre de la historia:</b> Recetar en caso de emergencia	
<b>Prioridad:</b> Media	<b>Riesgo en desarrollo:</b> Medio
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b> Como paciente quiero que se me receten medicamentos de emergencia como medida preventiva hasta poder ser atendido en el centro médico.	
<b>Criterio de aceptación:</b> El paciente recibe los medicamentos que puede consumir en caso de emergencia.	
<b>Observaciones:</b> s/n	

**Tabla 11. Historia de usuario 06**

<b>Historia de Usuario</b>	
<b>Número:</b> 06	<b>Usuario:</b> Administrador
<b>Nombre de la historia:</b> Visualizar pacientes	
<b>Prioridad:</b> Baja	<b>Riesgo en desarrollo:</b> Medio
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b> Cómo administrador quiero visualizar la lista de pacientes que se han comunicado con el centro médico para generar informes.	
<b>Criterio de aceptación:</b> Despliegue del listado de pacientes que se han comunicado con el centro médico.	

<b>Observaciones:</b> s/n
---------------------------

**Tabla 12. Historia de usuario 07**

<b>Historia de Usuario</b>	
<b>Número:</b> 07	<b>Usuario:</b> Administrador
<b>Nombre de la historia:</b> Visualizar citas	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Medio
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b> Cómo administrador quiero visualizar la lista de citas que se han agendado en el centro médico para generar informes.	
<b>Criterio de aceptación:</b> Despliegue del listado de citas agendadas en el centro médico.	

**Tabla 13. Historia de usuario 08**

<b>Historia de Usuario</b>	
<b>Número:</b> 08	<b>Usuario:</b> Administrador
<b>Nombre de la historia:</b> Actualizar cita	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Medio
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b> Cómo administrador quiero actualizar la información de las citas agendadas en el centro médico para corregir errores durante el proceso de agendamiento.	
<b>Criterio de aceptación:</b> Actualización exitosa de los datos de la cita.	
<b>Observaciones:</b> <ul style="list-style-type: none"> <li>• La información actualizada no puede coincidir con otra cita.</li> <li>• Únicamente se puede actualizar la cédula y la fecha de la cita.</li> </ul>	

**Tabla 14. Historia de usuario 09**

<b>Historia de Usuario</b>	
<b>Número:</b> 09	<b>Usuario:</b> Administrador
<b>Nombre de la historia:</b> Borrar cita	
<b>Prioridad en negocio:</b> Alta	<b>Riesgo en desarrollo:</b> Medio
<b>Programador responsable:</b> Sandro Córdova	
<b>Descripción:</b> Cómo administrador quiero eliminar una cita de forma lógica para cancelar la cita agendada por el paciente.	
<b>Criterio de aceptación:</b> Eliminación lógica de la cita agendada.	
<b>Observaciones:</b> La eliminación debe ser lógica, no física.	

### **2.3. Características de los usuarios**

**Tabla 15. Características del administrador**

<b>Tipo de usuario</b>	Administrador
<b>Formación</b>	Ninguna
<b>Actividades</b>	Control y manejo total del sistema en general.

**Tabla 16. Características del usuario**

<b>Tipo de usuario</b>	Usuario
<b>Formación</b>	Ninguna
<b>Actividades</b>	<ul style="list-style-type: none"><li>• Acceder a la información presentada por el sistema.</li><li>• Acceder a las diferentes funcionalidades del sistema.</li><li>• Solicitar atención personalizada.</li></ul>

### **2.4. Restricciones**

- El sistema se desarrollará en base a la metodología Kendall y Kendall por su adaptabilidad al trabajo de titulación.
- La aplicación será utilizada en un ordenador.
- El sistema deberá tener un diseño e implementación sencilla.
- Lenguajes y tecnologías en uso: Java, JADE.
- La aplicación será desarrollada en el framework Netbeans.

#### **2.4.1. Suposiciones y dependencias**

- Se asume que los requisitos descritos en el presente documento son estables.
- Los equipos en los que se vaya a ejecutar el sistema deben cumplir los requisitos antes indicados para garantizar una ejecución correcta de la misma.
- El administrador del sistema tendrá conocimiento de tecnologías similares.

### **2.5. Requisitos específicos**

#### **Interfaces de usuario**

La interfaz con el usuario consistirá en una serie de opciones planteadas por el sistema en donde, el usuario podrá seleccionar cualquiera de estas opciones mediante su voz. Durante la interacción sistema-usuario la comunicación se da únicamente mediante llamada telefónica desde cualquier dispositivo móvil o fijo.

## **Interfaces de hardware**

Será necesario disponer de lo siguiente:

- Computador.
- Micrófono.
- Altavoces.

## **Interfaces de software**

- Sistema Operativo: Windows 7 o superior.

## **Interfaces de comunicación**

El computador se comunicará con el teléfono fijo encargado de receptar las entradas de voz y la salida de audio del sistema.

## **2.6. Requisitos funcionales**

En el presente apartado se detalla los requisitos funcionales que serán utilizados en el sistema.

**Tabla 17. Requisito funcional 01**

<b>Identificación del requisito</b>	RF01
<b>Nombre del requisito</b>	Transferir llamada
<b>Descripción del requisito</b>	El usuario puede solicitar al sistema comunicación directa con el departamento de atención al cliente del centro médico.
<b>Prioridad del requisito</b>	Media

**Tabla 18. Requisito funcional 02**

<b>Identificación del requisito</b>	RF02
<b>Nombre del requisito</b>	Listar síntomas
<b>Descripción del requisito</b>	El sistema permitirá al usuario listar los síntomas que este posee para poder determinar el estado del paciente.
<b>Prioridad del requisito</b>	Alta

**Tabla 19. Requisito funcional 03**

<b>Identificación del requisito</b>	RF03
<b>Nombre del requisito</b>	Clasificación del estado de los pacientes
<b>Descripción del requisito</b>	El sistema compara y clasifica los síntomas dados por el paciente y toma decisiones dependiendo de la clasificación.
<b>Prioridad del requisito</b>	Alta

**Tabla 20. Requisito funcional 04**

<b>Identificación del requisito</b>	RF04
<b>Nombre del requisito</b>	Informar sobre medidas de bioseguridad
<b>Descripción del requisito</b>	El sistema debe presentar al usuario las medidas de bioseguridad necesarias para poder acercarse al centro médico.

<b>Prioridad del requisito</b>	Media
--------------------------------	-------

**Tabla 21. Requerimiento funcional 05**

<b>Identificación del requisito</b>	RF05
<b>Nombre del requisito</b>	Agendar citas
<b>Descripción del requisito</b>	El usuario podrá agendar una cita médica a través del sistema. Los datos para agendar una cita son: Especialidad médica, doctor, número de cédula, nombres, apellidos, fecha y hora.
<b>Prioridad del requisito</b>	baja

**Tabla 22. Requerimiento funcional 06**

<b>Identificación del requisito</b>	RF06
<b>Nombre del requisito</b>	Recetar en caso de emergencia
<b>Descripción del requisito</b>	En caso de emergencia, el sistema debe recomendar al usuario el consumo exclusivo de Paracetamol, hasta que este pueda ser atendido en el centro médico.
<b>Prioridad del requisito</b>	Alta

**Tabla 23. Requerimiento funcional 07**

<b>Identificación del requisito</b>	RF07
<b>Nombre del requisito</b>	Gestión de citas.
<b>Descripción del requisito</b>	El administrador podrá visualizar, actualizar y cancelar las citas médicas agendadas.
<b>Prioridad del requisito</b>	Media

**Tabla 24. Requerimiento funcional 08**

<b>Identificación del requisito</b>	RF08
<b>Nombre del requisito</b>	Generar reportes de pacientes.
<b>Descripción del requisito</b>	El administrador podrá visualizar la información consolidada de los pacientes que se han comunicado con el centro médico.
<b>Prioridad del requisito</b>	media

**Tabla 25. Requerimiento funcional 09**

<b>Identificación del requisito</b>	RF12
<b>Nombre del requisito</b>	Generar reportes de citas.
<b>Descripción del requisito</b>	El administrador podrá visualizar la información consolidada de las citas que se han generado en el centro médico.
<b>Prioridad del requisito</b>	media

## 2.7. Requisitos no funcionales

**Tabla 26. Requisitos no funcionales**

Código	Requisitos	Descripción
01	Rendimiento.	El sistema debe garantizar que las consultas al sistema u otro proceso no afecte el desempeño de la base de datos, ni a la respuesta del sistema de forma general. Monitoreando el tiempo de respuesta de la base de datos y del sistema en general, antes y después de la implementación del módulo de software.
02	Seguridad.	El sistema garantizará la seguridad de la información y datos personales que el usuario proporcione al momento de generar una cita médica. Aplicando mecanismos de protección a los datos personales de tipo confidencial (encriptación).
03	Usabilidad.	El sistema deberá contar con una interfaz de uso intuitiva y sencilla, evaluada a través de pruebas de aceptación del usuario en el uso del sistema.
04	Disponibilidad.	La disponibilidad del sistema deberá ser continua con un nivel de servicio mínimo del 95% durante los 365 días del año 24/7, exceptuando horarios de mantenimiento y fallos no planificados. Al medir el nivel de satisfacción del usuario respecto a la disponibilidad del módulo de software.
05	Mantenibilidad.	El sistema debe disponer de documentación que permita realizar operaciones de mantenimiento.
06	Portabilidad.	El sistema deberá ser compatible con la mayoría de los sistemas operativos en los que se ejecute. A través del uso de herramientas de código abierto y en sus versiones estables para el desarrollo del módulo.
07	Voz aguda y agradable.	A través de pruebas de aceptación el sistema debe garantizar una voz aguda y agradable al oído del usuario durante la interacción.

**Anexo 2.** Informe de entrevistas a centros médicos

**Informe de entrevistas a centros médicos**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

## **1. Elicitación de requerimientos**

En este proceso se presenta el proceso de elicitation de requerimientos para la captura de los requisitos de sistema de software. A continuación, se describe la adquisición de requisitos por medio de la técnica de la entrevista, el análisis y validación de los requerimientos finales.

## **2. Entrevista**

La entrevista se realizó a la/las personas a cargo del departamento de coordinación, que son quienes se encargan de la atención personal y telefónica de los pacientes de los diferentes establecimientos de asistencia y tratamiento médico en la ciudad de Loja. A continuación, se listan los nombres de las personas entrevistadas (PE) de los diferentes establecimientos médicos:

- Sueanny Vélez, “Consorcio médico” Atención al cliente. (PE01)
- Ruth Ojeda, “Clínica hospital municipal ‘Julia Esther González Delgado’”, Departamento de Coordinación. (PE02)
- Mary Orellana, “Clínica hospital municipal ‘Julia Esther González Delgado’”, Dispensario médico. (PE03)
- Alexandra Quizhpe, “Clínica Abendaño”, Departamento de coordinación. (PE04)
- Carolina Guerrero, “Clínica Abendaño”, Departamento de pediatría. (PE05)
- María Fernanda Celi, “Clínica Mogrovejo”, Departamento de coordinación. (PE06)
- Jessica Ruiz, “Clínica San José”, Departamento de triaje. (PE07)

El objetivo de la entrevista es obtener información sobre el proceso seguido cuando ingresa una llamada al centro médico, además de información general de interés para el usuario, recomendaciones generales y preguntas comunes por parte de los usuarios.

**PE: Persona Entrevistada**

**RP: Respuesta Pregunta**

**1) (RP01) ¿Cuál es el proceso que se sigue cuando una persona llama a la institución en busca de información o soporte médico?**

**PE01:**

No hay un protocolo definido, los pacientes preguntan directamente por un médico.

**PE02:**

Se recibe directamente la llamada, y luego se la asigna a un departamento.

**PE03:**

No se recibe directamente la llamada, él que tiene el protocolo es en coordinación que es donde receptan la llamada y luego se la asigna a este departamento.

**PE04:**

Asignación a la secretaría de cada médico mediante call center para que ella le ayude al paciente.

**PE05:**

- Costo
- Agendar
- Re agenda la cita

**PE06:**

- Solicitar cita médica.
- Pedir el número de teléfono de algún doctor si es una emergencia.
- Costos.

**EP07:**

Sin respuesta.

**2) (RP02) ¿El Proceso seguido es el definido por la institución o se sigue un estándar general para atención médica?**

**PE01:**

El establecido por la clínica.

**PE02:**

Es por la institución y se cumplen con las normas del ministerio de salud pública.

**PE03:**

Es por la institución y se cumplen con las normas del ministerio de salud pública.

**PE04:**

El establecido por la clínica.

**PE05:**

El establecido por la clínica.

**PE06:**

El establecido por la clínica.

**PE07:**

El establecido por la clínica.

**3) (RP03) ¿Qué opciones tiene la persona una vez que se comunica mediante llamada telefónica con la institución?**

**PE01:**

- Reservas de citas.
- Precio de la consulta de los doctores dependiendo la especialidad.
- Agenda citas para consultas médicas.

**PE02:**

- Consultas de precios.
- Horarios de atención.
- Asignación o reserva de turnos.
- Exámenes de laboratorio.
- Se maneja por diferentes departamentos.

**PE03:**

Como es un área farmacología, directamente con nosotros no se comunican para ese tipo de información, pero si es que se diera el caso se da la información de horarios, atención médica y por supuesto cuando se trata de algún medicamento en general.

**PE04:**

Asignación de la llamada a cada uno de los asistentes.

**PE05:**

- Costo.
- Agendar.
- Re agenda la cita.

**PE06:**

- Solicitar cita médica.
- Pedir el número de teléfono de algún doctor si es una emergencia.
- Costos.

**PE07:**

- Con qué doctor desea atenderse.
- El precio del doctor.
- Horarios de doctores.
- Asignación de turnos

**4) (RP04) Si la sintomatología del paciente o el motivo de la llamada es por contagio de Covid-19. ¿Se acepta en la institución este tipo de pacientes?**

**PE01:**

Para ingresos en la clínica se maneja, en el consorcio solo los pacientes que posiblemente presentan Covid-19 con el oftalmólogo, el doctor hace las pruebas para confirmar un diagnóstico y se los acepta siguiendo el debido protocolo de bioseguridad.

**PE02:**

No se aceptan pacientes con Covid-19.

**PE03:**

Realmente nosotros no tenemos ningún caso Covid-19, de ser el caso tendrían que comunicarse directamente con el médico, no se recomienda ningún medicamento.

**PE04:**

Claro, no se puede negar la atención a ningún paciente.

**PE05:**

Si.

**PE06:**

No, solo es estudio de tomografía, pero no para hospitalización.

**PE07:**

Si.

**5) (RP05) ¿Cuáles son las recomendaciones que se brinda a las personas que se comunican con síntomas de Covid-19?**

**PE01:**

- Que vengan cumpliendo las medidas de bioseguridad, si es posible con doble mascarilla, que mantengan el distanciamiento y esperen a su doctor.
- Se siguen todas las medidas de bioseguridad establecidas.
- Solo con un acompañante.

**PE02:**

Solo en el caso de que tengan alguna duda de algún medicamento que les hayan recetado, se le comunica con el médico que le dio la receta, pero no se atiende a personas con Covid-19, se les recomienda que vayan a una casa de salud donde asistan pacientes con Covid-19.

**PE03:**

Nosotros como farmacia interna de la clínica receptamos recetas médicas, si es que el caso de un paciente que viniera, lo que nosotros podemos ayudar es reservar un turno con el médico y es él facultativo quien determina que pasos puede seguir.

**PE04:**

El médico que atiende el Covid-19 tiene su secretaria personal, pero se solicita traje de bioseguridad, alcohol, doble mascarilla.

**PE05:**

Las medidas de bioseguridad que están establecidas, visor, distanciamiento, mascarilla y lavado de manos uso de alcohol, se le restringe a un acompañante, se receta solo paracetamol.

**PE06:**

Venir de preferencia solo o acompañado de un solo familiar, traer mascarilla NK95 y un protector facial, venir en horas donde no haya mucha afluencia de gente, en horarios de la tarde.

**PE07:**

Que vengan protegidos, con mascarilla, si han estado en contacto con otros familiares para que también se realicen la prueba y aislamiento.

**6) (RP06) ¿Cuáles son las dudas por las que generalmente se comunican las personas con el centro médico?**

**PE01:**

Para agendar citas con los diferentes especialistas del consorcio médico.

**PE02:**

- Horarios de atención de los médicos.
- Preguntando los servicios que ofrece la casa de salud.

**PE03:**

- Dudas con respecto a farmacia.
- Medicamento que se puede administrar con otro nombre comercial o genérico.
- Otros medicamentos que tengan la misma composición sirvan para el mismo síntoma o enfermedad.

**PE04:**

- Para saber de determinado médico y que especialidad tiene.
- El horario.
- Costo de consulta.
- Información general.

**PE05:**

- Si sus niños están enfermos tienen alguna patología.
- Si necesitan de algún tratamiento.
- Si ya finalizan el tratamiento.
- Preguntando costos de la consulta.

**PE06:**

Agendar turnos.

**PE07:**

El valor que cobran.

**7) (RP07) Piensa usted que sería de gran ayuda un sistema de agentes inteligentes que brinde al paciente este tipo de recomendaciones y solucione dudas comunes, dejando así solo aquellas llamadas de personas que realmente necesitan asistencia por parte del personal del centro médico.**

**PE01:**

Sin respuesta.

**PE02:**

Si, nos ahorraría tiempo.

**PE03:**

Por el momento no le vemos necesario ya que solo se maneja directamente farmacia.

**PE04:**

Es complicado ya que cada médico tiene horarios diferentes.

**PE05:**

Sí, sería importante para optimizar tiempo de pacientes.

**PE06:**

Sí, sería necesario, útil y así poder brindar un trato o asistencia mejor a los pacientes que vienen.

**PE07:**

Ayudaría informando horarios y agendando turnos.

8) (RP08) Si este tipo de tecnología se aplica al centro médico:

PE01:

- a. ¿Qué otra funcionalidad se le podría añadir?  
Sin respuesta
- b. ¿Cuál sería el tiempo estimado para la interacción entre el sistema que gestione la llamada y el paciente?  
Generalmente un minuto máximo tres
- c. ¿La voz que escucharía el paciente al momento de comunicarse debe ser masculina o femenina? ¿Grave o aguda?  
Creo que femenina, suave.

PE02:

- a. ¿Qué otra funcionalidad se le podría añadir?  
Sin respuesta
- b. ¿Cuál sería el tiempo estimado para la interacción entre el sistema que gestione la llamada y el paciente?  
Unos dos minutos máximo 4 minutos
- c. ¿La voz que escucharía el paciente al momento de comunicarse debe ser masculina o femenina? ¿Grave o aguda?  
Femenina, grave

PE03:

- a. ¿Qué otra funcionalidad se le podría añadir?  
Sería más un sistema interno.
- b. ¿Cuál sería el tiempo estimado para la interacción entre el sistema que gestione la llamada y el paciente?  
Generalmente de dos minutos y máximo hasta que el paciente pueda despejar su duda.
- c. ¿La voz que escucharía el paciente al momento de comunicarse debe ser masculina o femenina? ¿Grave o aguda?  
Debe ser una voz tranquila, serena sin importar si es masculina o femenina.

**PE04:**

- a. **¿Qué otra funcionalidad se le podría añadir?**

No, se me ocurre nada.

- b. **¿Cuál sería el tiempo estimado para la interacción entre el sistema que gestione la llamada y el paciente?**

Depende puede ser de un minuto, hasta diez.

- c. **¿La voz que escucharía el paciente al momento de comunicarse debe ser masculina o femenina? ¿Grave o aguda?**

Voz grave.

**PE05:**

- a. **¿Qué otra funcionalidad se le podría añadir?**

Clasificar a los pacientes, y a que consulta va a ir.

- b. **¿Cuál sería el tiempo estimado para la interacción entre el sistema que gestione la llamada y el paciente?**

No más de treinta segundos.

- c. **¿La voz que escucharía el paciente al momento de comunicarse debe ser masculina o femenina? ¿Grave o aguda?**

Masculina, grave.

**PE06:**

- a. **¿Qué otra funcionalidad se le podría añadir?**

Cirugías para que los doctores puedan agendarlas.

- b. **¿Cuál sería el tiempo estimado para la interacción entre el sistema que gestione la llamada y el paciente?**

Tres minutos.

- c. **¿La voz que escucharía el paciente al momento de comunicarse debe ser masculina o femenina? ¿Grave o aguda?**

Tono de voz aguda.

**PE07:**

- a. **¿Qué otra funcionalidad se le podría añadir?**

Enviándoles un mensaje cuando les toque la cita para confirmación.

- b. **¿Cuál sería el tiempo estimado para la interacción entre el sistema que gestione la llamada y el paciente?**

5 min máximo 20 min.

- c. **¿La voz que escucharía el paciente al momento de comunicarse debe ser masculina o femenina? ¿Grave o aguda?**

No afectaría solo sería un buen trato una voz normal.

### 3. Lista Preliminar de requisitos

**Tabla 1. Lista preliminar de requisitos**

Nº	Requisito	Solicitado/ Inferido	Referencia
01	El sistema debe permitir comunicarse con la secretaría de un médico en específico	Inferido	RP01: PE04
02	El sistema debe permitir comunicarse con un departamento específico	Inferido	RP01: PE03, PE02
03	El sistema debe permitir agendar citas	Inferido	RP01: PE05, PE06
04	El sistema debe permitir consultar el precio de la consulta médica	Inferido	RP3: PE01
05	El sistema debe permitir acceder al número de teléfono de los médicos	Inferido	RP3: PE06
06	El sistema debe tener un apartado para pacientes con Covid-19	Inferido	RP04: PE01, PE03, PE04, PE05, PE07
07	El sistema debe informar si el centro médico admite pacientes con Covid-19	Inferido	RP04: PE01, PE03, PE04, PE05, PE07
08	El sistema debe informar sobre las medidas de bioseguridad a cumplir el paciente para poder asistir al centro médico.	Inferido	RP05: PE01, PE02, PE03, PE04, PE05, PE06, PE07
09	El sistema solo puede recetar paracetamol en caso de requerir algún medicamento	Inferido	RP05: PE05
10	El tiempo máximo de iteración entre el paciente y el sistema debe ser de 6min	Inferido	RP08: PE01 - PE07(b)
11	El sistema debe informar con un tono de voz agudo y agradable	Solicitado	RP08: PE01(c), PE03(c), PE06(c), PE07(c)
12	El administrador podrá gestionar la información general del centro médico. (crear, consultar, modificar, buscar y eliminar)	Inferido	RP01: PE01
13	El administrador podrá gestionar la información de las citas médicas, como precios, días, horarios (crear, consultar, modificar, buscar y eliminar).	Inferido	RP03: PE01
14	El administrador podrá gestionar la información cada uno de los médicos que trabajen en el centro médico como disponibilidad y números de teléfono (crear, consultar, modificar, buscar y eliminar).	Inferido	RP01: PE04 RP03: PE06
15	El administrador podrá gestionar módulos del sistema (crear, consultar, modificar, buscar y eliminar)	Inferido	

#### 4. Matriz de iteración

Por medio de la matriz de iteración se evalúa los requisitos para identificar conflictos o solapamientos. La matriz presenta dos entradas, donde cada entrada contiene todos los requisitos, obteniendo que estos relacionen entre sí.

- Solapamiento: Si entre  $r_1$  y  $r_2$  tratan los mismos aspectos del sistema, dando redundancia. Se lo representa con la letra S.
- Conflicto: Si entre  $r_1$  y  $r_2$  son contradictorios, dando problemas de consistencia interna. Se lo representa con la letra C.

En la Tabla 2 se presenta el solapamiento o conflicto que puede existir entre los requisitos anteriormente obtenidos.

**Tabla 2. Matriz de iteración**

R	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
01					C										
02															
03															
04															
05	C														
06							C								
07						C									
08															
09															
10															
11															
12															
13															
14															
15															

La descripción de los conflictos encontrados en la matriz realizada se presenta en la Tabla 2, con ello se puede hacer un análisis y brindar una solución al requisito en conflicto por medio de la creación de un nuevo requisito.

## 5. Solapamiento y conflictos

**Tabla 3. Solapamiento y conflictos**

Nº	Requisito	Hallazgo	Descripción	Requisito final
1	01, 05	C	El requisito 01 tiene el mismo propósito del requisito 05.	El sistema debe permitir comunicarse con un médico en específico (número personal o la extensión de su secretaria).
2	06, 07	C	La negación del requisito 07 contradice al requisito 06.	El sistema debe tener un apartado para pacientes con Covid-19.

## 6. Lista final de requisitos

Una vez solucionado los conflictos encontrados, en la Tabla 3 se presenta los requisitos finales que serán aplicados en el desarrollo de la aplicación web.

**Tabla 4. Lista final de requisitos**

Código	Requisitos
01	Comunicación con los médicos
02	Comunicación los diferentes departamentos
03	Agendar citas
04	Informar costos
05	Apartado para pacientes con Covid-19
06	Informar sobre medidas de bioseguridad
07	Recetar en caso de emergencia
08	Tiempo de iteración de 6min
09	Voz aguda y agradable
10	Gestión de módulos
11	Gestión de información general
12	Gestión de citas
13	Gestión de médicos

## 7. Entrevistados

*Figura 1. Registro de personas entrevistadas.*



Figura 2. Entrevista a Sueanny Vélez, “Consorcio médico” Atención al cliente.



Figura 3. Entrevista a Ruth Ojeda, “Clínica hospital municipal ‘Julia Esther González Delgado’” Departamento de Coordinación.



Figura 4. Entrevista a Mary Orellana, “Clínica hospital municipal ‘Julia Esther González Delgado’”, Dispensario médico.



Figura 5. Entrevista a Carolina Guerrero, “Clínica Abendaño”, Departamento de pediatría.



Figura 6. Entrevista a Alexandra Quizhpe, “Clínica Abendaño”, Departamento de coordinación.

**Anexo 3.** Arquitectura de software

**Documento de Arquitectura de Software**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

## **1. Introducción**

Este documento es un resumen general sobre la arquitectura del software para el “Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente”. Este documento se ha estructurado basándose en el modelo 4+1, en donde se utilizan las vistas necesarias de arquitectura para describir los aspectos del sistema.

### **1.1. Propósito**

El presente documento tiene como propósito describir el diseño de la arquitectura de software para el Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente a través del modelo 4+1.

### **1.2. Alcance**

El documento de arquitectura abarca la definición de la arquitectura de la aplicación a través de las vistas de lógica, de despliegue, de procesos, física y de escenarios.

### **1.3. Referencias**

**Tabla 1. Referencias**

Título del Documento	Referencia
Arquitectura de Software 4+1	Modelo 4+1
Especificación de Requerimientos de Software IEEE 830	Anexo 1. Especificación de requisitos de software IEEE 830

### **1.4. Resumen**

Este documento consta de cuatro secciones. En la primera sección se realiza una introducción al mismo y se proporciona una visión general del documento de arquitectura de software.

En la segunda sección del documento se realiza una descripción general de la arquitectura, con el fin de conocer el modelo 4+1 y los objetivos.

En la tercera sección del documento, se definen las vistas y entregables de acuerdo al modelo 4+1, que son: vista de escenarios (diagrama de Casos de Uso), vista lógica (Diagrama de Clases), vista de despliegue (Diagrama de Componentes), vista de procesos (Diagrama de Actividad) y vista física (Diagrama de Despliegue).

Por último, la cuarta sección del documento es aquella en la que se define la arquitectura de la aplicación.

## 2. Descripción General

### 2.1. Representación de la arquitectura

La solución informática utiliza una arquitectura diseñada a través del modelo 4+1, misma que propone cuatro vistas bien diferenciadas y que se relacionan entre sí con una vista más, que se denomina vista “+1” o vista de escenarios. En la Tabla 2 se describe cada una de las vistas.

**Tabla 2. Arquitectura 4+1.**

Vista	Perspectiva	Elemento	Descripción
Vista Lógica.	Usuario Final.	Diagrama de Clases y representación de base de datos.	Representa la funcionalidad que el sistema proporcionara a los usuarios finales.
Vista de Despliegue.	Administrador de Software.	Diagrama de Paquetes.	Se muestra como está dividido el software en componentes y las dependencias que hay entre esos componentes.
Vista Física.	Administrador de Software.	Diagrama de Despliegue.	Se muestra todos los componentes físicos del sistema, así como todas las conexiones físicas entre esos componentes que conforman la solución.
Vista de Procesos.	Programador.	Diagrama de Actividad.	Se muestra los procesos que hay en el sistema y la forma en la que se comunican estos procesos.
Vista de Escenarios (+1).	General.	Diagrama de Casos de Uso.	Esta vista tiene como objetivo unir y relacionar las 4 vistas. Presentada en la Figura X sección X.

### 2.2. Objetivos de la arquitectura

El desarrollo del sistema cumplirá con las siguientes características:

- **Fiabilidad:** Esta vista va ser representada por los casos de uso, teniendo como objetivo unir y relacionar las otras 4 vistas.
- **Disponibilidad:** La disponibilidad del sistema deberá ser continua con un nivel de servicio para los usuarios de 24 horas del día durante los 365 días del año.
- **Funcionabilidad:** El sistema debe responder al usuario en menos de 10 segundos.
- **Rendimiento:** La aplicación web proporcionara de forma rápida y precisa la información solicitada por el usuario.
- **Seguridad:** El sistema contará con autenticación de usuario mediante el correo electrónico y contraseña; además garantizará la seguridad con respecto a la información y datos que se mantengan (cédula, diagnóstico médico).
- **Usabilidad:** El sistema debe presentar una interfaz de ayuda para que los usuarios se les facilite el uso del sistema.



### 3. Modelo 4+1

#### 3.1. Vista de Escenarios

En esta vista se representan los casos de uso, donde se muestra la interacción de los diferentes actores con el sistema, teniendo como objetivo unir y relacionar las otras 4 vistas.

##### 3.1.1. Actores del sistema

Tabla 3. Actor del sistema 01

Identificador.	AS-01
Nombre.	Paciente.
Descripción.	Actor que realiza la llamada y accede al centro de información sobre Covid-19. Se asume que la persona que se contacta con el centro de atención médica es el paciente.

Tabla 4. Actor del sistema 01

Identificador.	AS-02
Nombre.	Administrador
Descripción.	Encargado de actualizar, eliminar y generar reportes de las citas médicas agendadas, así como generar reportes de los pacientes que se comunican con el centro de atención médica.

##### 3.1.2. Diagrama de casos de uso

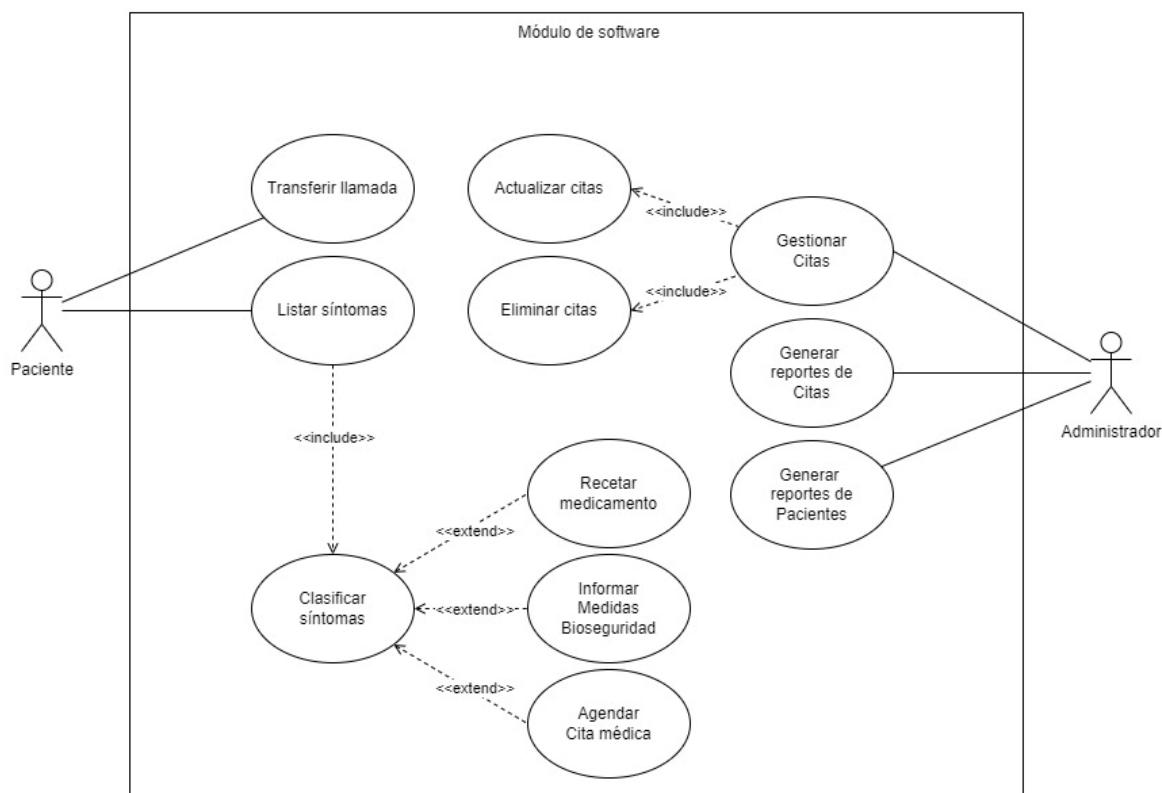


Figura 1. Diagrama de casos de uso.

### 3.1.3. Especificación de Casos de Uso

**Tabla 5. Caso de uso 01**

<b>Identificador</b>	UC-01
<b>Nombre</b>	Transferir llamada.
<b>Descripción</b>	El paciente solicita que se transfiera su llamada a la línea designada para atención al cliente del centro médico.
<b>Actores</b>	Paciente
<b>Precondiciones</b>	El paciente se comunicó con el centro de atención médica.
<b>Postcondiciones</b>	Se da inicio al caso de uso CU-02.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El paciente se comunica con el centro de atención médica.</li> <li>2. El paciente solicita mediante que su llamada sea transferida.</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El paciente cierra la llamada.</li> <li>2. El paciente ingresa el centro de información de Covid-19.</li> </ol>

**Tabla 6. Caso de uso 02**

<b>Identificador</b>	UC-02
<b>Nombre</b>	Listar síntomas
<b>Descripción</b>	El paciente brinda una lista de los síntomas que posee.
<b>Actores</b>	Paciente
<b>Precondiciones</b>	El paciente no ha seleccionado la opción de "transferir llamada (CU-02)
<b>Postcondiciones</b>	El agente realiza una clasificación del estado del paciente basado en sus síntomas CU-03.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El paciente ingresa el centro de información de Covid-19.</li> <li>2. El paciente lista los síntomas de forma verbal.</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El paciente cierra la llamada.</li> </ol>

**Tabla 7. Caso de uso 03**

<b>Identificador</b>	UC-03
<b>Nombre</b>	Clasificar síntomas
<b>Descripción</b>	El agente inteligente compara y clasifica los síntomas dados por el paciente, esta clasificación puede ser de prioridad alta, prioridad media y prioridad baja.
<b>Actores</b>	Paciente
<b>Precondiciones</b>	El paciente ha dado los síntomas que posee CU-02.
<b>Postcondiciones</b>	El agente toma decisiones basado en los síntomas del paciente (CU-04, CU-05, CU-06).
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El paciente ingresa el centro de información de Covid-19.</li> <li>2. El paciente lista los síntomas de forma verbal.</li> <li>3. El agente clasifica al paciente según sus síntomas.</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El paciente cierra la llamada.</li> </ol>

**Tabla 8. Caso de uso 04**

<b>Identificador</b>	UC-04
<b>Nombre</b>	Informar Medidas de bioseguridad.
<b>Descripción</b>	En caso de que los síntomas del paciente lo ameriten, el agente inteligente brinda al paciente las medidas de bioseguridad que debe cumplir.
<b>Actores</b>	Paciente
<b>Precondiciones</b>	El paciente ha dado los síntomas que posee CU-02.
<b>Postcondiciones</b>	El agente brinda la información al paciente basado en sus síntomas.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El paciente ingresa el centro de información de Covid-19.</li> <li>2. El paciente lista los síntomas de forma verbal.</li> <li>3. El agente clasifica al paciente según sus síntomas.</li> <li>4. El agente brinda la información al paciente basado en sus síntomas.</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El paciente cierra la llamada.</li> </ol>

**Tabla 9. Caso de uso 05**

<b>Identificador</b>	UC-05
<b>Nombre</b>	Agendar cita médica.
<b>Descripción</b>	En caso de que los síntomas del paciente lo ameriten, el agente inteligente genera una cita médica con los datos personales del paciente.
<b>Actores</b>	Paciente
<b>Precondiciones</b>	El paciente ha dado los síntomas que posee CU-02.
<b>Postcondiciones</b>	El agente brinda la información al paciente basado en sus síntomas.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El paciente ingresa el centro de información de Covid-19.</li> <li>2. El paciente lista los síntomas de forma verbal.</li> <li>3. El agente clasifica al paciente según sus síntomas.</li> <li>4. El agente brinda la información al paciente basado en sus síntomas.</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El paciente cierra la llamada.</li> </ol>

**Tabla 10. Caso de uso 06**

<b>Identificador</b>	UC-06
<b>Nombre</b>	Recetar medicamento.
<b>Descripción</b>	En el caso de que los síntomas del paciente lo ameriten, el agente inteligente indica al paciente los medicamentos que puede tomar de manera preventiva.
<b>Actores</b>	Paciente
<b>Precondiciones</b>	El paciente ha dado los síntomas que posee CU-02.
<b>Postcondiciones</b>	El agente brinda la información al paciente basado en sus síntomas.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El paciente ingresa el centro de información de Covid-19.</li> <li>2. El paciente lista los síntomas de forma verbal.</li> <li>3. El agente clasifica al paciente según sus síntomas.</li> <li>4. El agente brinda la información al paciente basado en sus síntomas.</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El paciente cierra la llamada.</li> </ol>

**Tabla 11. Caso de uso 07**

<b>Identificador</b>	UC-07
<b>Nombre</b>	Gestionar citas
<b>Descripción</b>	Permite al administrador actualizar la fecha de la cita médica agendada. Permite la eliminación lógica de la cita médica agendada.
<b>Actores</b>	Administrador
<b>Precondiciones</b>	El administrador debe ingresar a la interfaz gráfica del módulo de administrador.
<b>Postcondiciones</b>	Se modificará la información de las citas agendadas.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El administrador selecciona la opción “Generar reportes”.</li> <li>2. El sistema despliega los diferentes reportes que se pueden generar.</li> <li>3. El administrador hace clic en el botón “Listar citas”.</li> <li>4. El administrador ingresará el número de cédula asignado a la cita que desea modificar.</li> <li>5. El administrador ingresará la nueva fecha de la cita</li> <li>6. El administrador seleccionará la opción “Modificar cita”</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El administrador cancela la modificación de la cita.</li> </ol>

**Tabla 12. Caso de uso 08**

<b>Identificador</b>	UC-08
<b>Nombre</b>	Generar reportes de citas.
<b>Descripción</b>	Permite al administrador del sistema generar reportes de las citas médicas agendadas en el centro médico.
<b>Actores</b>	Administrador
<b>Precondiciones</b>	Debe existir citas agendadas previamente para poder visualizarlas.
<b>Postcondiciones</b>	Se entregará un reporte con la lista de citas agendadas solicitada.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El administrador selecciona la opción “Generar reportes”.</li> <li>2. El sistema despliega los diferentes reportes que se pueden generar.</li> <li>3. El administrador hace clic en el botón “Listar citas”.</li> <li>4. El sistema genera y despliega una vista con el reporte seleccionado.</li> </ol>
<b>Flujos alternos y excepciones</b>	<ol style="list-style-type: none"> <li>1. El administrador cancela la generación del reporte.</li> </ol>

**Tabla 13. Caso de uso 09**

<b>Identificador</b>	UC-09
<b>Nombre</b>	Generar reportes de pacientes.
<b>Descripción</b>	Permite al administrador del sistema generar reportes de los pacientes que se han comunicado con en el centro médico.
<b>Actores</b>	Administrador
<b>Precondiciones</b>	Debe existir pacientes registrados en la base de datos.
<b>Postcondiciones</b>	Se entregará un reporte con la lista de pacientes que se han comunicado con el centro médico.
<b>Flujo normal de eventos</b>	<ol style="list-style-type: none"> <li>1. El administrador selecciona la opción “Generar reportes”.</li> <li>2. El sistema despliega los diferentes reportes que se pueden generar.</li> <li>3. El administrador hace clic en el botón “Listar pacientes”.</li> <li>4. El sistema genera y despliega una vista con el reporte seleccionado.</li> </ol>
<b>Flujos alternos y excepciones</b>	

1. El administrador cancela la generación del reporte.

### 3.2. Vista lógica

En esta vista se representa la funcionalidad que el sistema proporcionara a los usuarios finales.

#### 3.2.1. Diagrama de clases

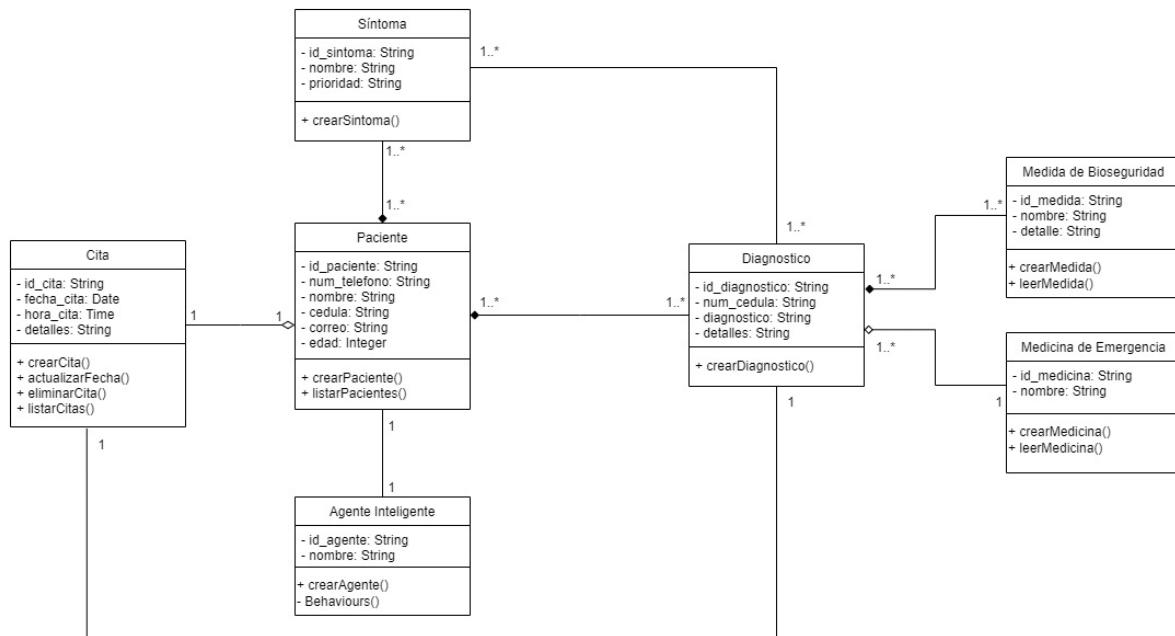


Figura 2. Diagrama de clases.

#### 3.2.2. Base de datos

Fase de Recolección	Fase de Clasificación	Fase de Resultados	Entrenamiento del agente inteligente
llamada	diagnóstico	medida_bioseguridad	dominio
paciente	sintoma_paciente	medicamentos	extension
extensiones		cita	numeros simbolos

Figura 3. Representación de la base de datos.

### 3.3. Vista de procesos

En esta vista se muestra los procesos que hay en el sistema y la forma en la que se comunican estos procesos.

#### 3.3.1. Diagrama de actividad

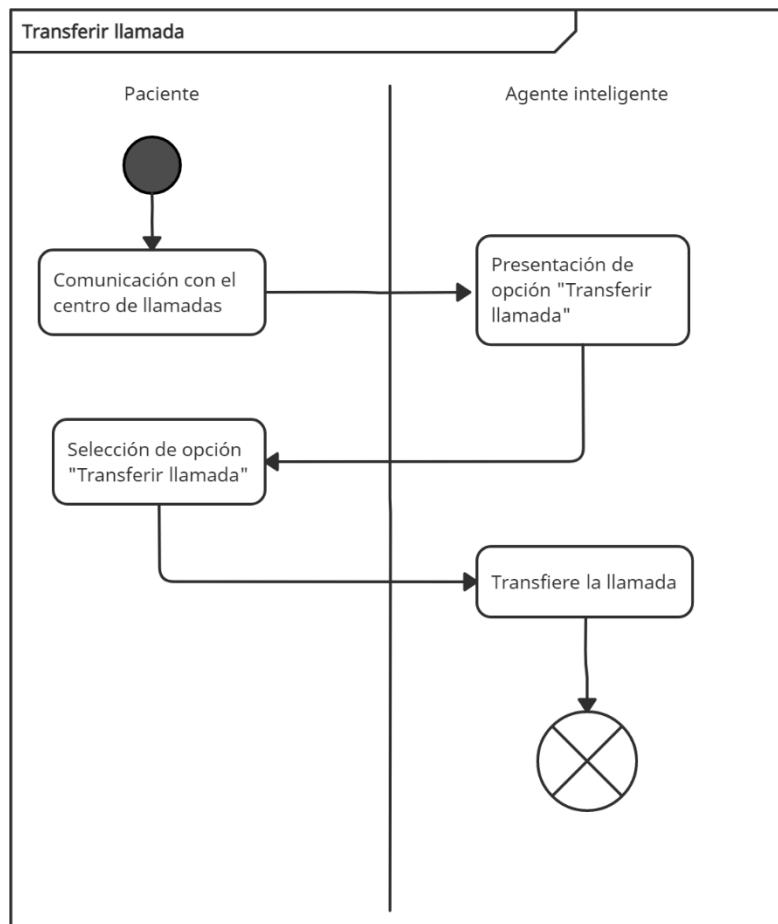


Figura 4. Diagrama de actividades, transferir llamada.

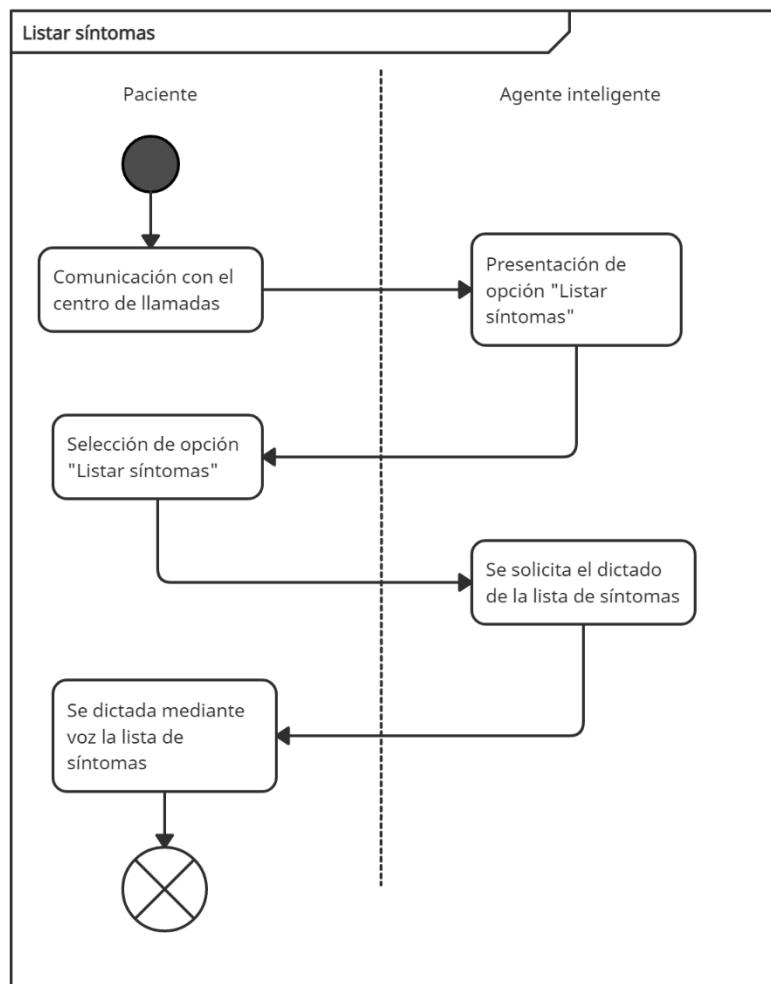


Figura 5. Diagrama de actividades, listar síntomas.

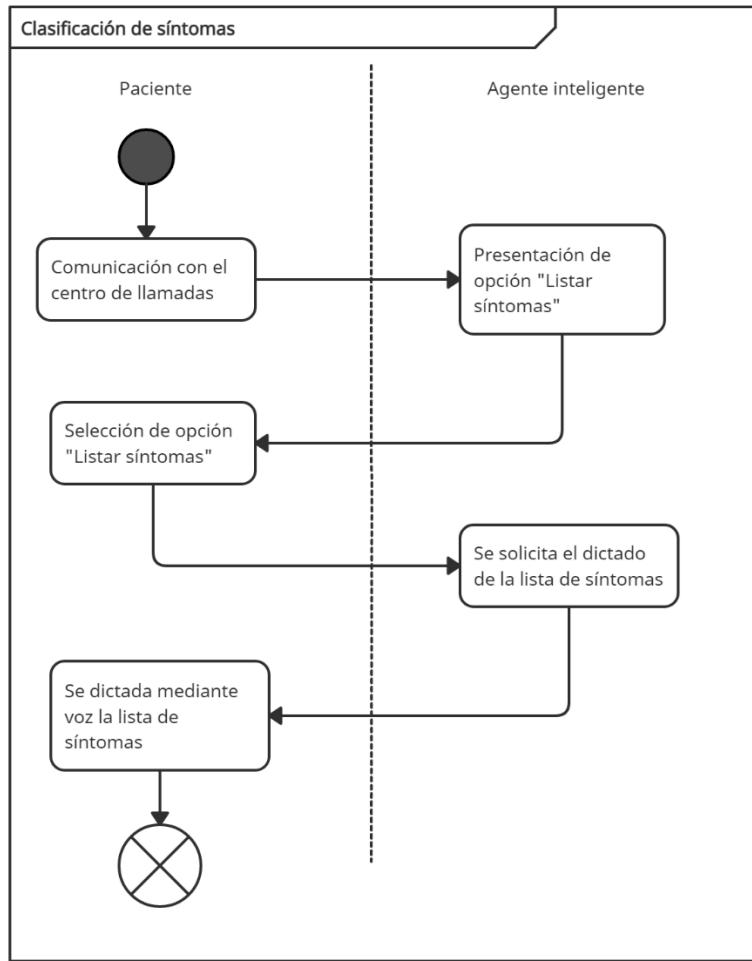


Figura 6. Diagrama de actividades, clasificar síntomas.

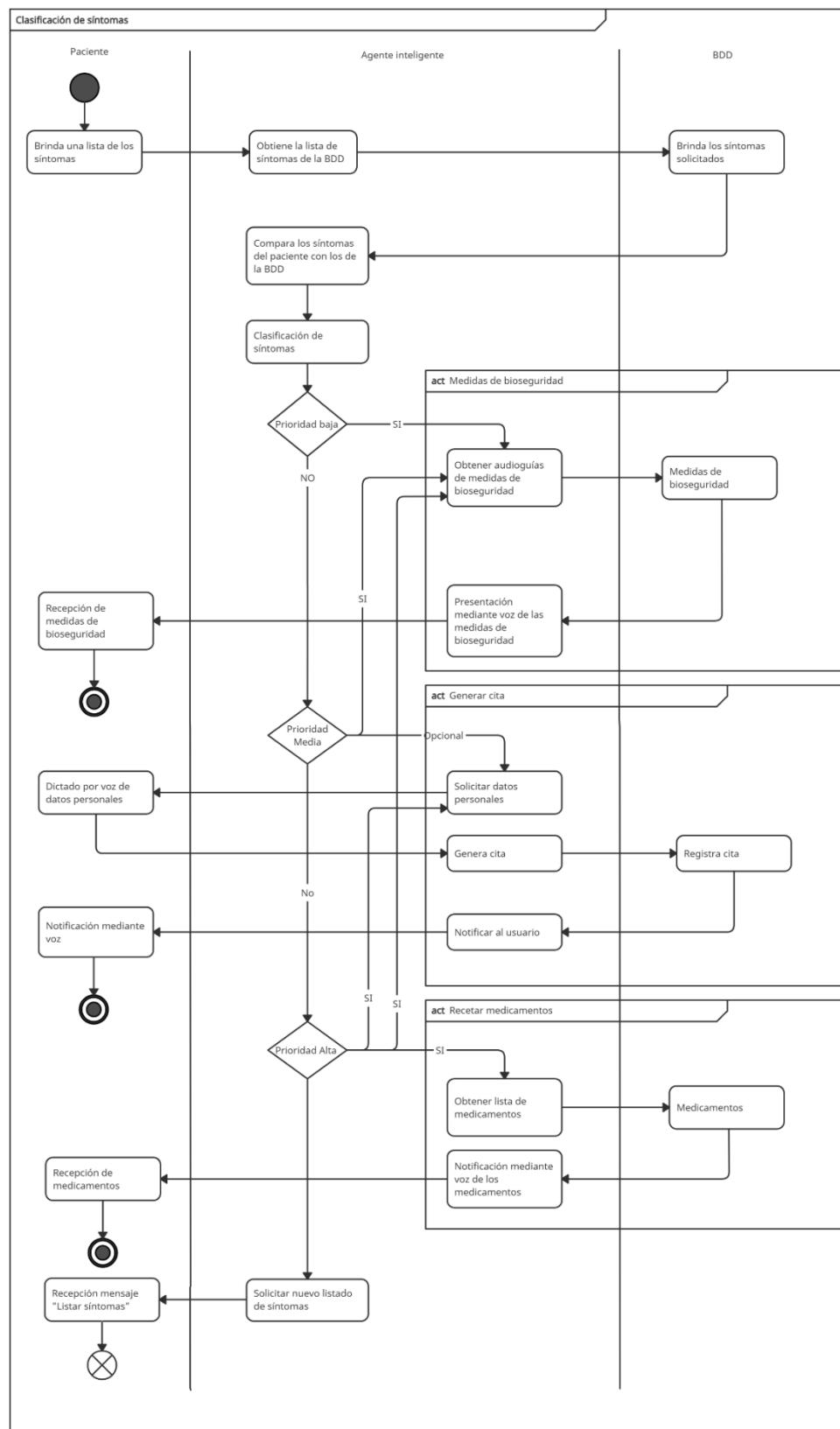


Figura 7. Diagrama de actividades, clasificación de síntomas.



### 3.4. Vista de Despliegue

En esta vista se muestra cómo está dividido el sistema de software en componentes y las dependencias que hay entre los componentes.

#### 3.4.1. Diagrama de paquetes

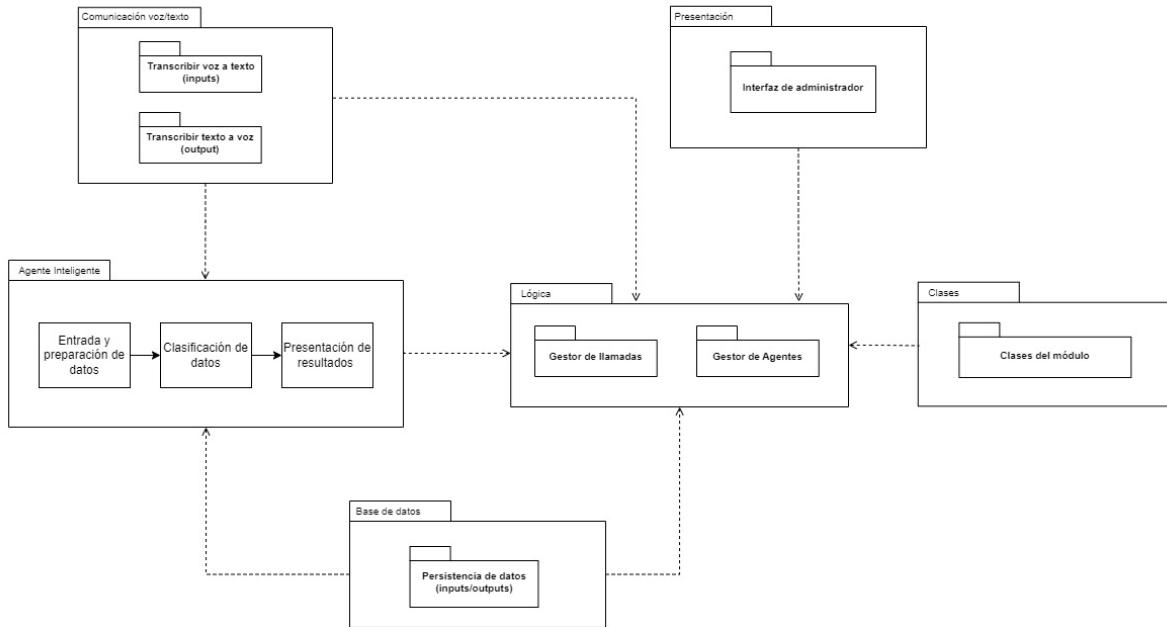


Figura 8. Diagrama de paquetes.

### 3.5. Vista de Física

En esta vista se muestra todos los componentes físicos del sistema, así como todas las conexiones físicas entre esos componentes que conforman la solución.

#### 3.5.1. Diagrama de despliegue

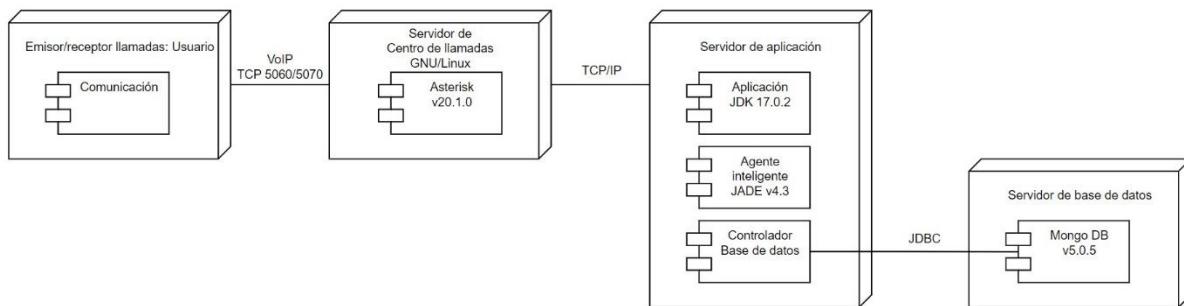


Figura 9. Diagrama de despliegue.

### 4. Esquema general

#### Módulo de software del Agente Inteligente

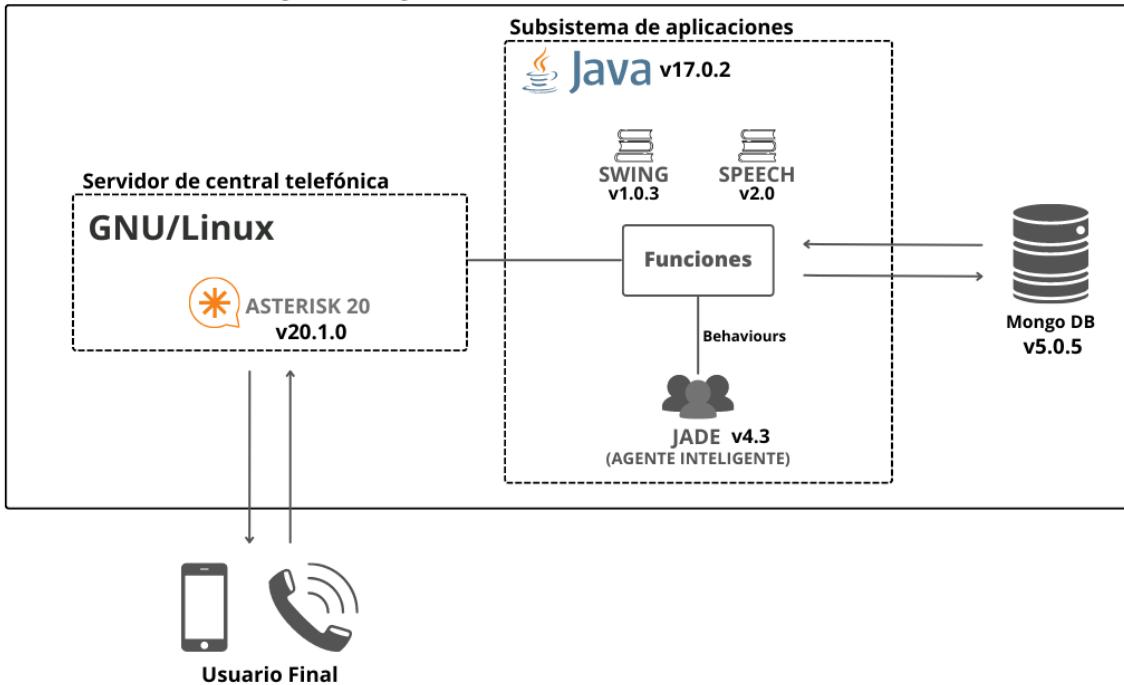


Figura 10. Esquema general.

## **Anexo 4. Desarrollo y documentación**

### **Desarrollo y documentación**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

## **Desarrollo y documentación del software**

El presente documento, contiene la ejecución de la fase V de la metodología para el proceso de desarrollo planteado, esta fase comprende la ejecución y la documentación del software, para poder cumplir con lo mencionado se hizo uso del modelo SCRUM, en donde se considera la siguiente estructura: Planificación, diseño, codificación, pruebas y resultados.

Dentro de la planificación se busca dar cumplimiento a los requerimientos en base a las historias de usuario, por lo tanto, se toma un grupo de aquellas que puedan desarrollarse en conjunto y durante el periodo establecido para la iteración, estos requerimientos son listados en forma de actividades en donde se presenta: Un resumen de la actividad a desarrollar, historia de usuario a la que dará cumplimiento una vez terminado, responsable y el estado de la actividad.

Para el diseño, se hace uso del documento arquitectura de software elaborado en la Fase IV de la metodología planteada (

Anexo 3. Arquitectura de software), de este documento se toma aquella información que es relevante, para poder cumplir con las actividades planteadas en la planificación, si se debe aclarar conceptos o recalcar el uso de tecnologías se debe especificar en esta sección.

Para esta sección de desarrollo se detalla mediante pseudocódigo cómo se ha dado cumplimiento a cada una de las actividades planteadas, se especifican mediante imágenes el uso de las diferentes funciones y métodos para cada una de las actividades.

Como punto final se tiene la sección de pruebas, en donde se plantean los resultados arrojados por la fase de desarrollo, aquí se pueden detallar los resultados obtenidos ya sea en documentación, registros de la base de datos, vistas y toda prueba que valide el cumplimiento de dicha actividad.

Cada iteración concluye con una tabla en donde se especifican los resultados obtenidos, esta tabla es una réplica de las actividades planteadas en la planificación, ya que el objetivo de esta sección es registrar el estado de las actividades, si están pendientes deben ser desarrolladas en la siguiente iteración, aquellas actividades que han ido surgiendo durante el desarrollo de la presente iteración y en el caso de que surjan problemas en el desarrollo de actividades deben ser detalladas.

Para poder ver todo el código fuente del módulo de software se puede ver en el repositorio:

**<https://github.com/sandrocordova/AgentelInteligenteJava.git>**

## 1. Módulos de desarrollo

Para poder cumplir con la fase de desarrollo y documentación de software, se hizo una agrupación por módulos de las historias de usuario y con sus respectivos requerimientos, logrando así agrupar aquellas historias de usuario que tienen actividades similares.

**Tabla 1. Historias de usuario y requerimientos.**

Módulo	Historia de usuario	Descripción	Requerimiento
Módulo de transferencia	HU01	Transferir llamada	RF01
Módulo de clasificación	HU02	Clasificación del paciente	RF03
Módulo de decisiones	HU03	Medidas de bioseguridad	RF04
	HU04	Generar cita médica	RF05
	HU05	Recetar en casos de emergencia	RF06
Módulo de reportes de pacientes	HU06	Generar reporte de Pacientes	RF08
Módulo de gestión de citas	HU07	Visualizar cita	RF07 – RF09
	HU08	Actualizar cita	RF07
	HU09	Borrar cita	RF07

Una vez analizada la tabla por módulos de las historias de usuario, se logró determinar que se hará uso de cuatro iteraciones en donde se abarcan todas las actividades planteadas para cada uno de los módulos, la primera iteración tendrá una duración de una semana, la segunda iteración tendrá una duración de dos semanas y las iteraciones tres y cuatro tendrán una duración de una semana cada una. A continuación, se detalla cada una de las iteraciones.

## **2. Iteración 1**

### **2.1. Planificación**

En la Tabla 1 se presenta las incidencias para la primera iteración.

**Tabla 2. Tareas de la iteración 1**

<b>Resumen</b>	<b>Responsable</b>	<b>Estado</b>
Instalación del framework NetBeans	Sandro Córdova	Pendiente
Instalación del JDK de Java	Sandro Córdova	Pendiente
Creación del proyecto	Sandro Córdova	Pendiente
Agregar la libreta JADE dentro del proyecto	Sandro Córdova	Pendiente
Crear repositorio de GitHub para el proyecto	Sandro Córdova	Pendiente
Instalación de la base de datos	Sandro Córdova	Pendiente
Creación de la base de datos	Sandro Córdova	Pendiente
Creación de las colecciones en la base de datos	Sandro Córdova	Pendiente

### **2.2. Diseño**

La documentación para el diseño del software, se encuentra presentado a detalle en el documento de arquitectura de software (

Anexo 3. Arquitectura de software), en esta sección únicamente se presentarán los diagramas que se han utilizado para poder desarrollar las actividades propuestas en la planificación de la iteración.

### Esquema de base de datos

Para el desarrollo de la base de datos se tomó como referencia el presente diagrama, en donde se especifica las colecciones y los tipos de datos que tendrán cada uno de los documentos dentro de la colección.

Fase de Recolección	Fase de Clasificación	Fase de Resultados	Entrenamiento del agente inteligente
llamada	diagnostico	medida_bioseguridad	dominio
paciente	sintoma_paciente	medicamentos	extension
extensiones		cita	numeros simbolos

Figura 1. Esquema de base de datos.

Para el desarrollo de las actividades planteadas en esta iteración se hará uso de versiones actuales y estables que tengan las diferentes tecnologías a utilizar. Todas estas tecnologías serán implementadas en el sistema operativo Windows 10 con una arquitectura de 64 bits.

### 2.3. Codificación

#### 2.3.1. Instalación de Java

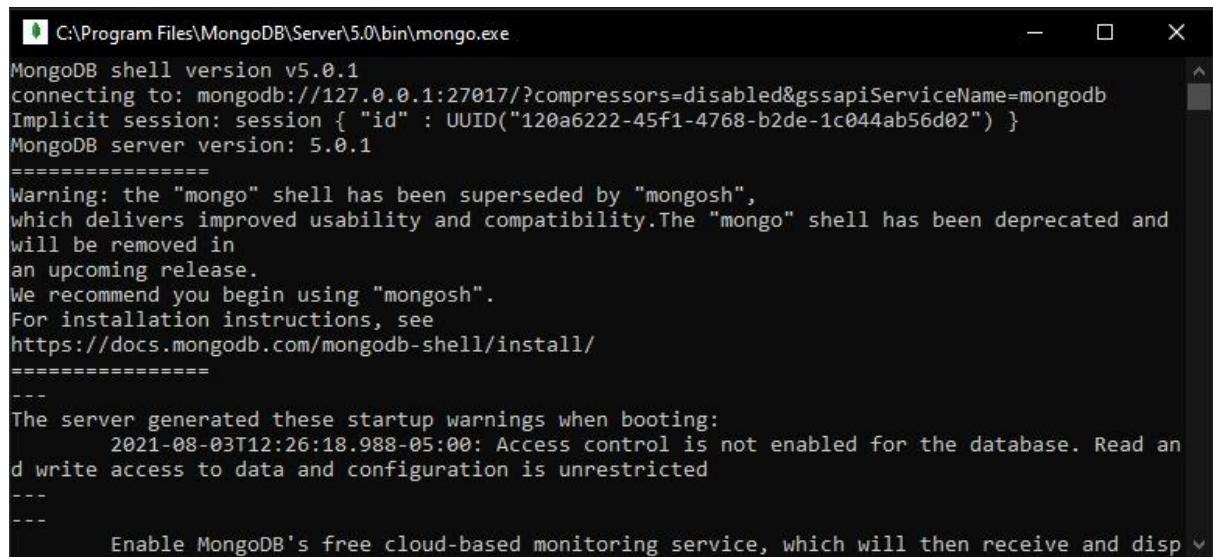
En la presente imagen se puede observar que se encuentra instalada la versión de JAVA 17.0.2 en su versión lanzada el día miércoles, veinte de julio del dos mil veintiuno.

**Product Version:** Apache NetBeans IDE 12.6  
**Java:** 17.0.2; Java HotSpot(TM) 64-Bit Server VM 17.0.2+8-LTS-86  
**Runtime:** Java(TM) SE Runtime Environment 17.0.2+8-LTS-86  
**System:** Windows 11 version 10.0 running on amd64; Cp1252; es\_MX (nb)  
**User directory:** C:\Users\Usuario\AppData\Roaming\NetBeans\12.6  
**Cache directory:** C:\Users\Usuario\AppData\Local\NetBeans\Cache\12.6

Figura 2. Instalación de JAVA.

#### 2.3.2. Instalación de la base de datos Mongo DB

Se instaló la base de datos de Mongo DB en su versión 5.0.1

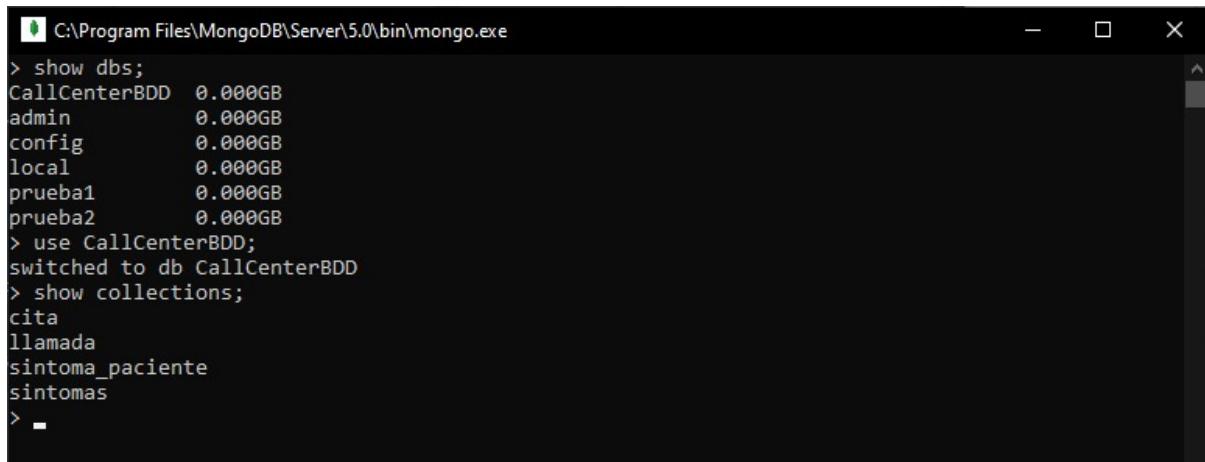


```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
MongoDB shell version v5.0.1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("120a6222-45f1-4768-b2de-1c044ab56d02") }
MongoDB server version: 5.0.1
=====
Warning: the "mongo" shell has been superseded by "mongosh",
which delivers improved usability and compatibility.The "mongo" shell has been deprecated and
will be removed in
an upcoming release.
We recommend you begin using "mongosh".
For installation instructions, see
https://docs.mongodb.com/mongodb-shell/install/
=====
The server generated these startup warnings when booting:
  2021-08-03T12:26:18.988-05:00: Access control is not enabled for the database. Read an
d write access to data and configuration is unrestricted
-----
Enable MongoDB's free cloud-based monitoring service, which will then receive and disp v
```

Figura 3. Instalación de la base de datos Mongo DB.

### 2.3.3. Creación de las colecciones en la base de datos

Se creó la base de datos con el nombre de “CallCenterBDD”, posteriormente se crearon las colecciones necesarias para la fase inicial del proyecto, en donde constan las colecciones: cita, llamada, síntoma\_paciente, síntoma.



```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> show dbs;
CallCenterBDD 0.000GB
admin          0.000GB
config         0.000GB
local          0.000GB
prueba1        0.000GB
prueba2        0.000GB
> use CallCenterBDD;
switched to db CallCenterBDD
> show collections;
cita
llamada
sintoma_paciente
sintomas
>
```

Figura 4. Colecciones en la base de datos.

### 2.3.4. Instalación del JDK y librerías

Posteriormente, se procedió a crear el proyecto de JAVA con el nombre de “callCenter” en donde se encuentra asociado el JDK por defecto, así como también se encuentran cargadas las librerías de JADE, “jade-4.3” en su versión 4.3 y las librerías necesarias para la base de datos “mongodb”.

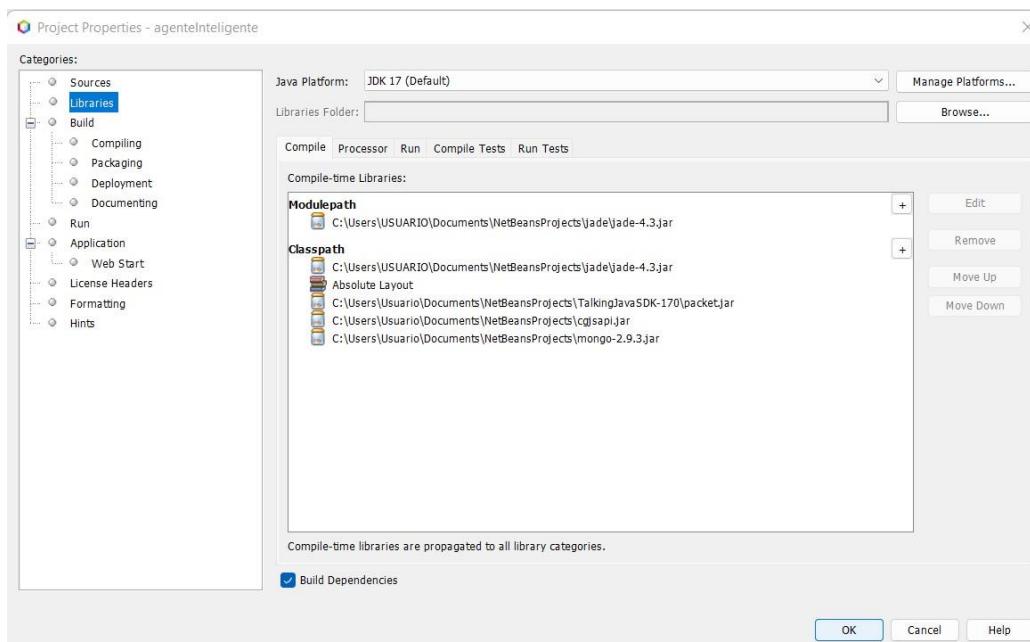
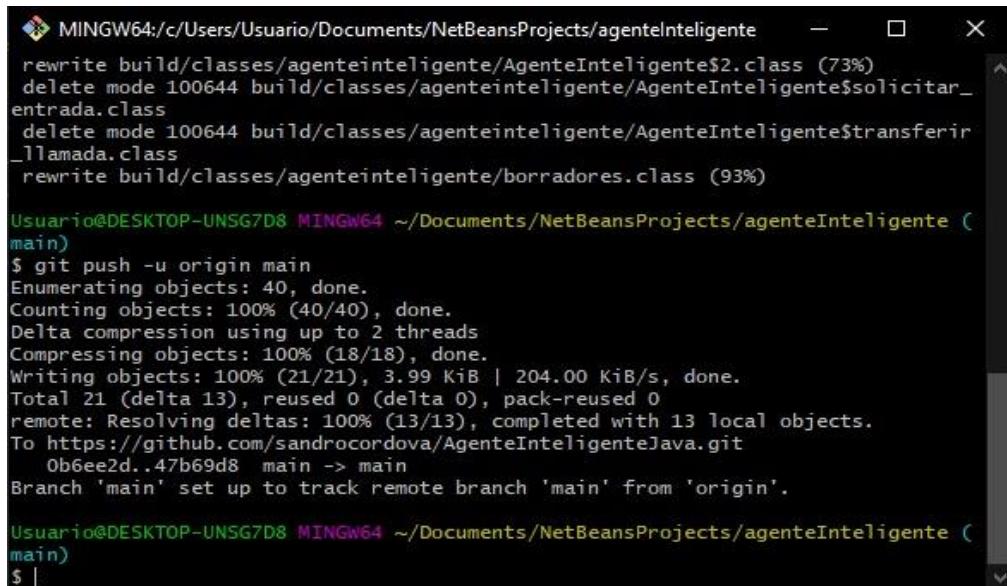


Figura 5. JDK y librerías.



### 2.3.5. Crear repositorio en GitHub para el proyecto

El proyecto se subió al repositorio de GitHub con el nombre de “AgenteInteligenteJava”.



```
MINGW64:/c/Users/Usuario/Documents/NetBeansProjects/agenteInteligente - □ X
rewrite build/classes/agenteinteligente/AgenteInteligente$2.class (73%)
delete mode 100644 build/classes/agenteinteligente/AgenteInteligente$solicitar_
entrada.class
delete mode 100644 build/classes/agenteinteligente/AgenteInteligente$transferir_
_llamada.class
rewrite build/classes/agenteinteligente/borradores.class (93%)

Usuario@DESKTOP-UNSG7D8 MINGW64 ~/Documents/NetBeansProjects/agenteInteligente (main)
$ git push -u origin main
Enumerating objects: 40, done.
Counting objects: 100% (40/40), done.
Delta compression using up to 2 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (21/21), 3.99 KiB | 204.00 KiB/s, done.
Total 21 (delta 13), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (13/13), completed with 13 local objects.
To https://github.com/sandrocordova/AgenteInteligenteJava.git
  0b6ee2d..47b69d8  main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.

Usuario@DESKTOP-UNSG7D8 MINGW64 ~/Documents/NetBeansProjects/agenteInteligente (main)
$ |
```

Figura 6. Repositorio en GitHub.

## 2.4. Revisión de la iteración

En la Tabla 3 se presenta el estado de las actividades desarrolladas durante la primera iteración.

Tabla 3. Revisión de iteración

Resumen	Responsable	Estado
Instalación del framework NetBeans	Sandro Córdova	Terminado
Instalación del JDK de Java	Sandro Córdova	Terminado
Creación del proyecto	Sandro Córdova	Terminado
Agregar la libreta JADE dentro del proyecto	Sandro Córdova	Terminado
Crear repositorio de GitHub para el proyecto	Sandro Córdova	Terminado
Instalación de la base de datos	Sandro Córdova	Terminado
Creación de la base de datos	Sandro Córdova	Terminado
Creación de las colecciones en la base de datos	Sandro Córdova	Terminado

### **3. Iteración 2**

#### **3.1. Planificación**

En la Tabla 4. se presenta las incidencias para la segunda iteración.

**Tabla 4. Tareas de la iteración 2**

Resumen	Clave HU	Responsable	Estado
Transferir llamada	HU01	Sandro Córdova	Pendiente
Listar síntomas	HU02	Sandro Córdova	Pendiente
Clasificación del paciente	HU02	Sandro Córdova	Pendiente
Generar cita médica	HU03	Sandro Córdova	Pendiente
Medidas de bioseguridad	HU04	Sandro Córdova	Pendiente
Recetar en casos de emergencia	HU05	Sandro Córdova	Pendiente

#### **3.2. Diseño**

La documentación para el diseño del software, se encuentra presentado a detalle en el documento de arquitectura de software (

Anexo 3. Arquitectura de software), en esta sección únicamente se presentarán los diagramas que se han utilizado para poder desarrollar las actividades propuestas en la planificación de la iteración.

Para el diseño de las actividades planteadas en esta iteración es necesario hacer uso del diagrama de clases especificado en la arquitectura de software (

Anexo 3. Arquitectura de software), para poder observar la funcionalidad que el sistema proporcionará a los usuarios, además; es necesario el diagrama de casos de uso para poder analizar los diferentes escenarios por los que el paciente pasara durante la interacción con el sistema.

### 3.2.1. Diagrama de casos de uso

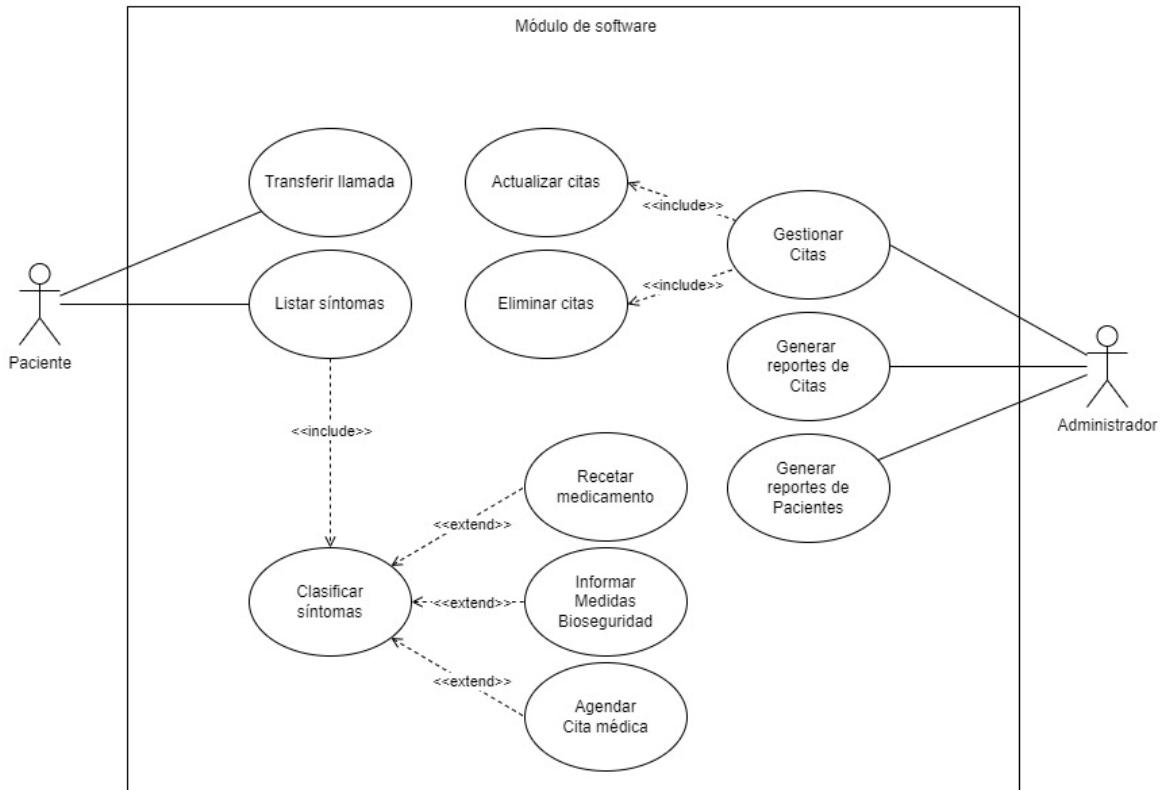


Figura 7. Diagrama de usos.

### 3.2.2. Esquema de clases

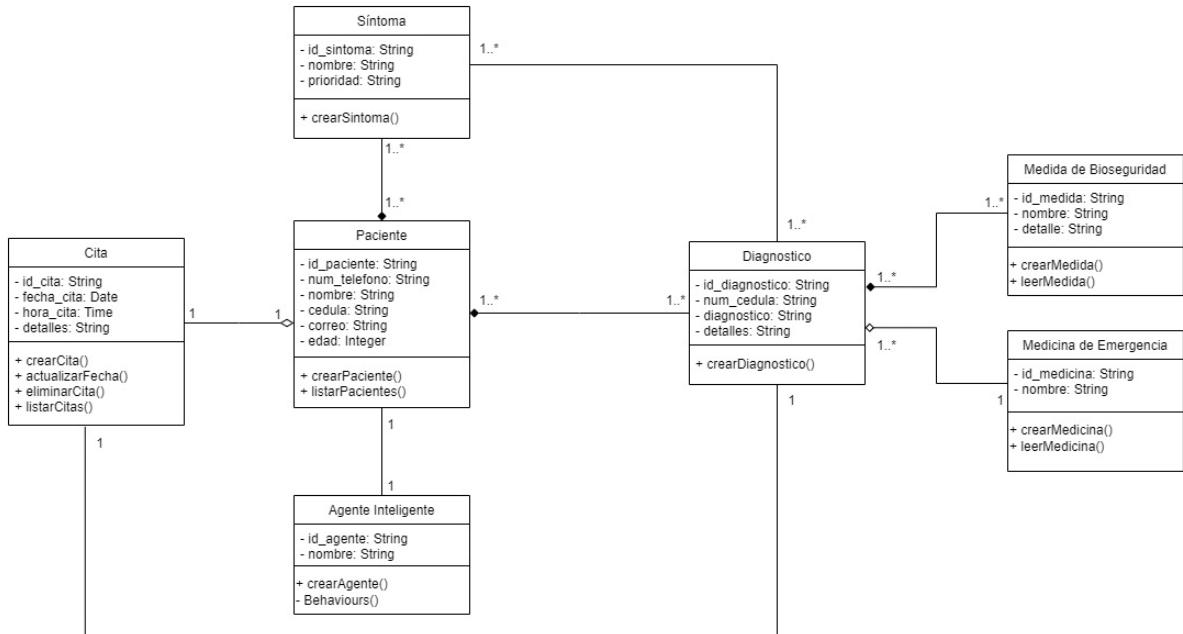


Figura 8. Diagrama de clases.

### 3.3. Codificación

#### 3.3.1. Transferir llamada

El presente método se encuentra escuchando constantemente al paciente, cuando este selecciona la “Opción 1” se procede a llamar el método “transferir\_llamada” el cual se encargará de transferir al paciente con una de las extensiones habilitadas para atención personalizada.

```
@Override
public void actionPerformed(ActionEvent e) {
    //Almacena el número de la llamada que ingresa
    int entrada = Integer.parseInt(vista_opciones.cajaOpcion.getText());
    llamada.setSeleccion(entrada);
    //Abre comunicación con la base de datos
    Conexion coneccion = new Conexion();
    //Tomar decisión acorde a la selección del usuario
    if (llamada.getSeleccion() == 1) {
        //Lo registra en la base de datos
        coneccion.insertarLlamada(llamada.getNum_telefono(), llamada.getSeleccion());
        transferir_llamada(llamada);
        vista_opciones.setVisible(false);
    } else if (llamada.getSeleccion() == 2) {
        //Lo registra en la base de datos
        coneccion.insertarLlamada(llamada.getNum_telefono(), llamada.getSeleccion());
        crear_agente(llamada);
        vista_opciones.setVisible(false);
    } else {
        vista_opciones.anuncio1.setText("Opción no válida");
    }
}
```

Figura 9. Transferir llamada.

#### 3.3.2. Método transferir llamada

Si el paciente solicita transferir su llamada, se habilita el siguiente método, en donde, el sistema obtiene la información de las diferentes extensiones que se encuentran registradas en la base de datos, revisa que ésta se encuentre habilitada con el condicional “if” y se la asigna al paciente.

```
public static void transferir_llamada(Llamada llamada) {
    Conexion conexion = new Conexion();
    ArrayList<Extension> listaExtensiones = new ArrayList<Extension>();
    listaExtensiones = conexion.buscarExtension();
    for (int i = 0; i < listaExtensiones.size(); i++) {
        Extension extension = new Extension();
        extension = listaExtensiones.get(i);
        if (extension.getEstado().contains("True")) {
            System.out.println("Su llamada ha sido transferida al:" + extension.getDetalle()
                + " extensión: " + extension.getExtension());
        }
    }
}
```

Figura 10. Método transferir llamada.

### 3.3.3. Listar síntomas

Se crea y asigna un agente inteligente, el cual a partir de este punto será el encargado de atender todas las solicitudes por parte del paciente, tomar decisiones, interactuar con la base de datos y presentar la información solicitada.

```
public static void crear_agente(Llamada llamada) {
    Runtime rt = Runtime.instance();
    Profile p = new ProfileImpl();
    p.setParameter(Profile.MAIN_HOST, "localhost");
    p.setParameter(Profile.GUI, "true");
    ContainerController cc = rt.createMainContainer(p);
    AgentController ac;

    Object[] arguments = new Object[3];
    arguments[0] = llamada.getNum_telefono();
    arguments[1] = llamada.getSeleccion();

    try {
        ac = cc.createNewAgent("Agente" + llamada.getNum_telefono(),
                               "agenteinteligente.AgenteInteligente", arguments);
        ac.start();
    } catch (Exception e) {
        System.out.println("ERRO al CREAR el Agente inteligente");
    }
}
```

Figura 11. Agente inteligente.

El agente inteligente se levanta y tiene como tarea inicial el solicitar los datos del paciente, los datos solicitados por parte del agente son: nombres, cédula, correo y la edad.

```
public void setup() {
    System.out.println("----AGENTE CREADO----");
    Object[] args = getArguments();
    System.out.println("Agente en ejecución: Agente" + args[0]);

    //Se da una tarea inicial al agente inteligente
    addBehaviour(new OneShotBehaviour(this) {
        public void action() {
            //Llama al método DatosPaciente
            datosPaciente(String.valueOf(args[0]));
        }
    });

    //Tarea que el agente inteligente desarrollará de manera repetitiva
    addBehaviour(new CyclicBehaviour(this) {
        public void action() {

            if (paciente.getNombre() == null) {
                System.out.println("vacio");
            }
        }
    });
}
```

Figura 12. Datos del paciente.

```

private void datosPaciente(String numero_telefono) {
    //Conexion con la base de datos
    Conexion conexion = new Conexion();
    vista_persona vista_persona = new vista_persona();
    vista_persona.setVisible(true);
    paciente = new Paciente();
    ActionListener al = new ActionListener() {
        //Se activa al hacer click en el botón llamar
        @Override
        public void actionPerformed(ActionEvent e) {
            //Almacena el número de la llamada que ingresa
            String nombre = vista_persona.cajaNombre.getText();
            String cedula = vista_persona.cajaCedula.getText();
            String correo = vista_persona.cajaCorreo.getText();
            int edad = Integer.parseInt(vista_persona.cajaEdad.getText());
            vista_persona.setVisible(false);
            paciente.setNum_telefono(numero_telefono);
            paciente.setNombre(nombre);
            paciente.setCedula(cedula);
            paciente.setCorreo(correo);
            paciente.setEdad(edad);
            //Lo registra en la base de datos
            conexion.insertarPaciente(numero_telefono, nombre, cedula, correo, edad);
            sintomasPaciente();
        }
    };
    vista_persona.botonEnviar.addActionListener(al);
}

```

Figura 13. Método datos del paciente.

Una vez ingresados los datos personales del paciente, este los almacena en la base de datos y solicita al paciente ingresar la lista de síntomas que este posee para poder realizar un análisis y clasificación del estado del paciente.

```

//Método para obtener los síntomas del paciente
private void sintomasPaciente() {
    //Llamamos a la vista para el ingreso de los datos
    vista_sintomas vista_sintomas = new vista_sintomas();
    vista_sintomas.setVisible(true);
    ActionListener al = new ActionListener() {
        //Se activa al hacer click en el botón llamar
        @Override
        public void actionPerformed(ActionEvent e) {
            //Almacena el número de la llamada que ingresa
            sintomas = vista_sintomas.cajaSintomas.getText();
            vista_sintomas.setVisible(false);
            //Llamamos al método para clasificar los síntomas
            clasificarSintomas();
        }
    };
    vista_sintomas.botonEnviar.addActionListener(al);
}

```

Figura 14. Síntomas del paciente.

### **3.3.4. Clasificación del paciente**

En la Figura 15. se inicializa el método "clasificarSintomas", el mismo que comienza extrayendo los síntomas que se encuentran cargados en la base de datos para poder realizar la comparación con los síntomas ingresados por el paciente como se muestra en la Figura 16, aquí se le asigna un grado de prioridad al paciente, basado en los síntomas que este posea. La clasificación de los pacientes se da en base a los síntomas obtenidos de las encuestas aplicadas a los profesionales en medicina (ver

#### Anexo 9. Informe de encuesta de síntomas).

```
//Clasifica los síntomas ingresados por el paciente, los registra en la base de datos
//y toma decisiones acorde al resultado
private void clasificarSintomas() {
    //Se traen los síntomas de la base de datos
    Conexion conexion = new Conexion();
    ArrayList<Sintoma> sintomasReferencia = new ArrayList<Sintoma>();
    sintomasReferencia = conexion.buscarSintomas();
```

Figura 15. Síntomas del paciente.

```
//ArrayList para almacenar los síntomas que el paciente tiene
ArrayList<Sintoma> sintomasPaciente = new ArrayList<Sintoma>();
// Comparo los síntomas de la base de datos con los ingresados por el usuario
for (int i = 0; i < sintomasReferencia.size(); i++) {
    Sintoma sintoma = new Sintoma();
    sintoma = sintomasReferencia.get(i);
    if (sintomas.contains(sintoma.getNombre())) {
        //registra los síntomas del paciente a la base de datos
        Conexion conexion2 = new Conexion();
        conexion2.insertarSintomaPaciente(paciente.getCedula(), sintoma.getNombre(),
            sintoma.getPrioridad());
        sintomasPaciente.add(sintoma);
        //Se asigna el grado de prioridad basado en los síntomas del paciente
        if (sintoma.getPrioridad() == 1) {
            contadorPrioridadUno++;
        } else if (sintoma.getPrioridad() == 2) {
            contadorPrioridadDos++;
        } else {
            contadorPrioridadTres++;
        }
    }
}
```

Figura 16. Clasificar síntomas.

Una vez obtenido el grado de prioridad del paciente, se procede a establecer un diagnóstico dependiendo de la prioridad del paciente, este diagnóstico es almacenado en la base de datos y se procede a clasificar al paciente dependiendo de su prioridad, existen tres clasificaciones y cada una desata distintos eventos.

```

//Se clasifica al paciente acorde a la prioridad, se almacena el diagnostico en la base de datos
//y se toma decisiones
Conexion conexionDiagnostico = new Conexion();
if (contadorPrioridadTres >= 3) {
    diagnostico.setNumCedula(paciente.getCedula());
    diagnostico.setDiagnostico("Atención medica urgente");
    diagnostico.setDetalles("se recetó medicina");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadTres();
} else if (contadorPrioridadDos < 3) {
    diagnostico.setNumCedula("cedula" + paciente.getCedula());
    diagnostico.setDiagnostico("Atención medica puede esperar");
    diagnostico.setDetalles("No hay observaciones");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadDos();
} else if (contadorPrioridadDos >= 2) {
    diagnostico.setNumCedula(paciente.getCedula());
    diagnostico.setDiagnostico("Atención medica opcional");
    diagnostico.setDetalles("No hay observaciones");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadDos();
} else if (sintomas.length() > 0) {
    diagnostico.setNumCedula(paciente.getCedula());
    diagnostico.setDiagnostico("No requiere atención medica");
    diagnostico.setDetalles("No hay observaciones");
    conexionDiagnostico.insertarDiagnosticoPaciente(diagnostico.getNumCedula(),
        diagnostico.getDiagnostico(), diagnostico.getDetalles());
    pacientePrioridadUno();
}

```

*Figura 17. Comparar síntomas.*

Cada una de las clasificaciones del paciente, desencadena un conjunto de eventos. La clasificación del paciente como prioridad uno, llama al método "medidasBiosegurdiad", la clasificación del paciente como prioridad dos, llama a los métodos, "generarCita" y "medidasBiosegurdiad", por último, si el paciente es clasificado como prioridad tres, se llama a los métodos "generarCita", "recetarMedicina" y "medidasBiosegurdiad".

```

private void pacientePrioridadTres() {
    generarCita();
    recetarMedicina();
    medidasBioseguridad();
}

private void pacientePrioridadDos() {
    generarCita();
    medidasBioseguridad();
}

private void pacientePrioridadUno() {
    medidasBioseguridad();
}

```

*Figura 18. Clasificación del paciente.*

### 3.3.5. Generar cita médica

El siguiente método es el encargado de tomar los datos del paciente, genera una cita médica, la registra en la base de datos y es presentada al paciente.

```
private void generarCita() {
    //Se inicializa el objeto cita
    Cita cita = new Cita();
    cita.setNombres(paciente.getNombre());
    cita.setEdad(paciente.getEdad());
    cita.setCedula(paciente.getCedula());
    cita.setCorreo(paciente.getCorreo());
    cita.setSintomas(cadenaSintomas);
    cita.setDiagnostico(diagnostico.getDiagnostico());
    cita.setNumTelefono(paciente.getNum_telefono());
    //Se establece un formato a la fecha de la cita
    SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy/h:mm");
    String fecha = sdf.format(new Date());
    cita.setFecha(fecha);
    //Se almacena en la base de datos
    Conexion conexionCita = new Conexion();
    conexionCita.insertarCita(cita.getNombres(), cita.getEdad(), cita.getCedula(),
    cita.getCorreo(), cita.getSintomas(), cita.getDiagnostico(), cita.getFecha());
    //Presentar cita al usuario
    String citaTexto = "Nombre: " + cita.getNombres() + "\nCorreo: "
    + cita.getCorreo() + "\nFecha: " + cita.getFecha();
    vista_sintomas.vista_sintomas = new vista_sintomas();
    vista_sintomas.anuncio.setText("CITA GENERADA");
    vista_sintomas.cajaSintomas.setText(citaTexto);
    vista_sintomas.botonEnviar.setText("ACEPTAR");
    vista_sintomas.setVisible(true);
    //Se lee la información de la cita médica al usuario
    Lee lee = new Lee();
    lee.leer(citaTexto);
}
```

Figura 19. Generar cita.

### 3.3.6. Medidas de bioseguridad

El presente método muestra la conexión con la base de datos para poder obtener las medidas de bioseguridad que se encuentran almacenadas, lo almacena en una cadena de caracteres y esta es presentada al paciente de forma visual y de forma verbal.

```

private void medidasBioseguridad() {
    String medidas = "Medidas de Bioseguridad: \n";
    //Establece conexión con la base de datos
    Conexion conexion = new Conexion();
    ArrayList<MedidaBioseguridad> listaMedidas = new ArrayList<MedidaBioseguridad>();
    listaMedidas = conexion.buscarMedidasBioseguridad();
    //For se encarga de formar el mensaje que se presentará al usuario
    for (int i = 0; i < listaMedidas.size(); i++) {
        MedidaBioseguridad medida = new MedidaBioseguridad();
        medida = listaMedidas.get(i);
        medidas += "- "+medida.getNombre()+"\n";
    }
    //Presentar las medidas de bioseguridad al usuario
    vista_sintomas vista_sintomas = new vista_sintomas();
    vista_sintomas.anuncio.setText("Medidas de bioseguridad");
    vista_sintomas.cajaSintomas.setText(medidas);
    vista_sintomas.botonEnviar.setText("ACEPTAR");
    vista_sintomas.setVisible(true);
    //Se leen las medidas de bioseguridad al usuario
    Lee lee = new Lee();
    lee.leer(medidas);
    System.out.println("MULTIMEDIA DE MEDIDAS BIOSEGURIDAD");
}

```

Figura 20. Método de medidas de bioseguridad.

### 3.3.7. Recetar en casos de emergencia

El siguiente método es el encargado de presentar al paciente los medicamentos de los que este puede hacer uso en caso de emergencia, esta información es presentada mediante audio y de forma visual.

```

private void generarCita() {
    //Se inicializa el objeto cita
    Cita cita = new Cita();
    cita.setNombres(paciente.getNombre());
    cita.setEdad(paciente.getEdad());
    cita.setCedula(paciente.getCedula());
    cita.setCorreo(paciente.getCorreo());
    cita.setSintomas(cadenaSintomas);
    cita.setDiagnostico(diagnostico.getDiagnostico());
    cita.setNumTelefono(paciente.getNum_telefono());
    //Se establece un formato a la fecha de la cita
    SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy/h:mm");
    String fecha = sdf.format(new Date());
    cita.setFecha(fecha);
    //Se almacena en la base de datos
    Conexion conexionCita = new Conexion();
    conexionCita.insertarCita(cita.getNombres(), cita.getEdad(), cita.getCedula(),
    cita.getCorreo(), cita.getSintomas(), cita.getDiagnostico(), cita.getFecha());
    //Presentar cita al usuario
    String citaTexto = "Nombre: " + cita.getNombres() + "\nCorreo: "
    + cita.getCorreo() + "\nFecha: " + cita.getFecha();
    vista_sintomas vista_sintomas = new vista_sintomas();
    vista_sintomas.anuncio.setText("CITA GENERADA");
    vista_sintomas.cajaSintomas.setText(citaTexto);
    vista_sintomas.botonEnviar.setText("ACEPTAR");
    vista_sintomas.setVisible(true);
    //Se lee la información de la cita médica al usuario
    Lee lee = new Lee();
    lee.leer(citaTexto);
}

```

Figura 21. Método recetar en caso de emergencia.

### 3.4. Pruebas

#### 3.4.1. Transferir llamada

Para poder emular la llamada de ingreso al sistema se ha hecho uso de la librería Swing de Java para poder realizar el ingreso manual del número de teléfono o celular que se comunique con el sistema, la llamada inicia cuando se haga clic en el botón "Llamar".



Figura 22. Ingreso del número telefónico.

El paciente al momento de realizar la llamada tiene dos opciones, la primera es transferir su llamada y la segunda es ingresar al centro de información de Covid-19, para poder transferir su llamada, el paciente debe seleccionar la opción uno, para este ejemplo se debe digitar manualmente el número uno, sin embargo, también puede hacerlo mediante su voz, diciendo la palabra “uno” u “opción uno”.

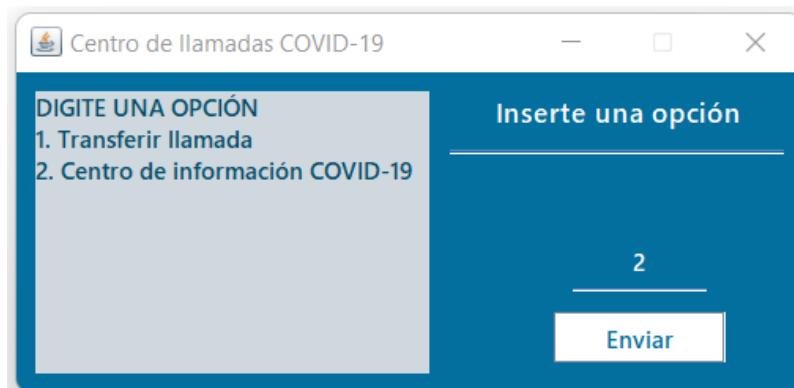


Figura 23. Menú principal.

Si el paciente selecciona la opción uno “transferir llamada”, el sistema automáticamente enlaza la llamada con la extensión que se encuentre disponible en la base de datos, en este caso el “departamento de atención al cliente, extensión 12” y se termina la comunicación con el paciente.

```

: Output - agenteInteligente (run)
run:
0997034339
Base de datos conectada....
Base de datos conectada....
Su llamada ha sido transferida al:Atención al cliente extensión: 12

```

Figura 24. Resultados en consola de opción "transferir llamada".

### 3.4.2. Listar síntomas

Si el paciente selecciona la opción dos “Centro de información Covid-19”, el sistema le asigna un agente, el cual se encargará desde este punto, de la gestión de toda la interacción con el paciente. Como se puede evidenciar en la imagen, para este ejemplo se ha creado el “Agente0997034339”, el mismo que se encuentra en estado activo.

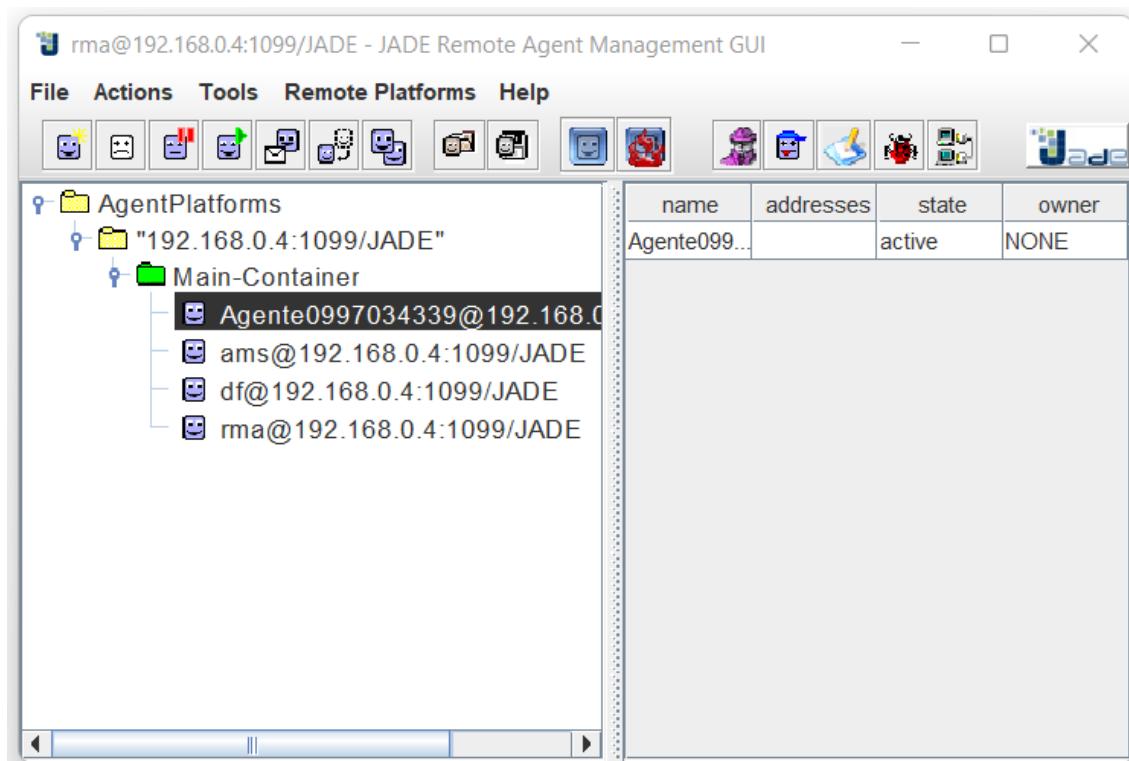


Figura 25. Agente asignado al paciente.

Cada agente creado tiene sus actividades por efectuar, la primera actividad es la de solicitar la información personal al paciente, los datos que debe ingresar el paciente son: nombre, cédula, correo, edad, una vez ingresados los datos el paciente envía la información al sistema.

Centro de Llamadas COVID-19

DATOS PERSONALES

Nombre: Andres

Cédula: 1150261905

Correo: sandro@gmail.com

Edad: 22

Pasaporte

ENVIAR

Figura 26. Ingreso de datos personales del paciente.

Una vez ingresados los datos personales del paciente el sistema le solicita listar los síntomas que padece, estos no tienen un límite de palabras.

Centro de Llamadas COVID-19

LISTA DE SINTOMAS QUE PADECE

tos  
gripe  
disnea  
diarrea  
fiebre alta  
nauseas, presion alta, dolor de cabeza

Enviar

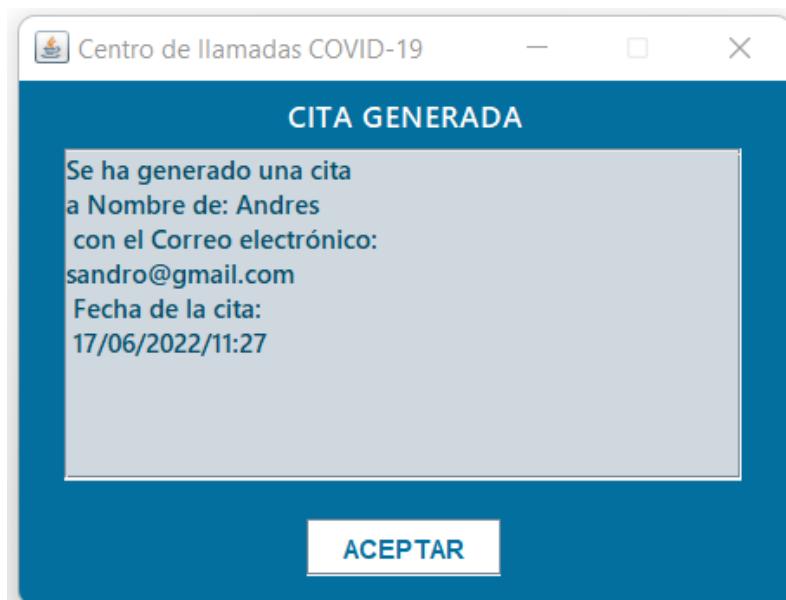
Figura 27. Lista de síntomas que padece.

### 3.4.3. Clasificación del paciente

El agente inteligente toma los síntomas ingresados por el paciente y se encarga de darles un grado de prioridad, dependiendo el grado de prioridad puede generar una cita médica, recetar en caso de emergencia dar las medidas de bioseguridad.

#### **3.4.4. Generar cita médica**

Una vez que el sistema clasifica los síntomas genera una cita médica si los síntomas del paciente son de grado alto.



*Figura 28 Cita generada.*

#### **3.4.5. Medidas de bioseguridad**

A continuación, se presenta una lista con las medidas de protección básicas contra el nuevo coronavirus, las mismas que han sido obtenidas del Protocolo de prevención de Covid-19 presentado por parte del ministerio de salud pública de la república del Ecuador (ver Marco teórico), y también de las entrevistas realizadas a los diferentes centros de salud, que constan en el Informe de la entrevista (

Anexo 2. Informe de entrevista).

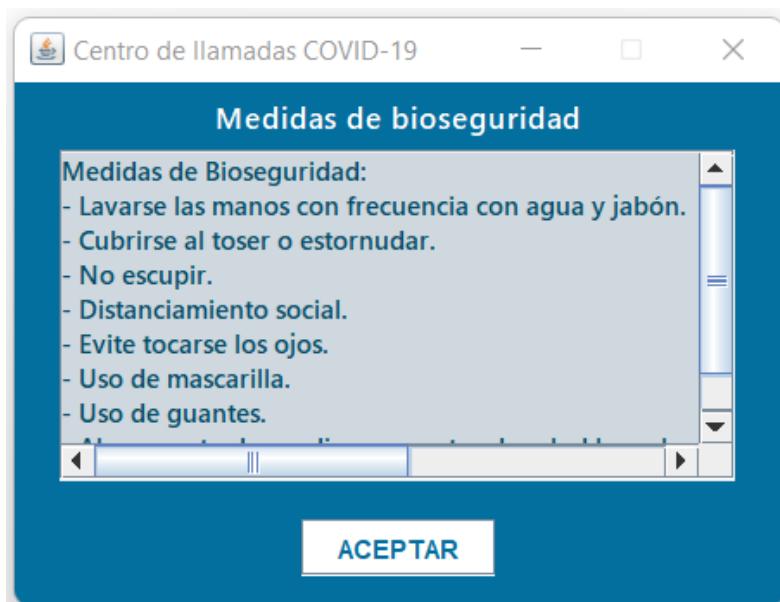


Figura 29. Medidas de bioseguridad.

#### 3.4.6. Recetar en casos de emergencia

En caso que el sistema considere una emergencia, recetará al paciente con paracetamol hasta que pueda llegar a la institución médica. La medicina presentada al paciente se obtuvo en las encuestas a profesionales en medicina (

Anexo 9. Informe de encuesta de síntomas).

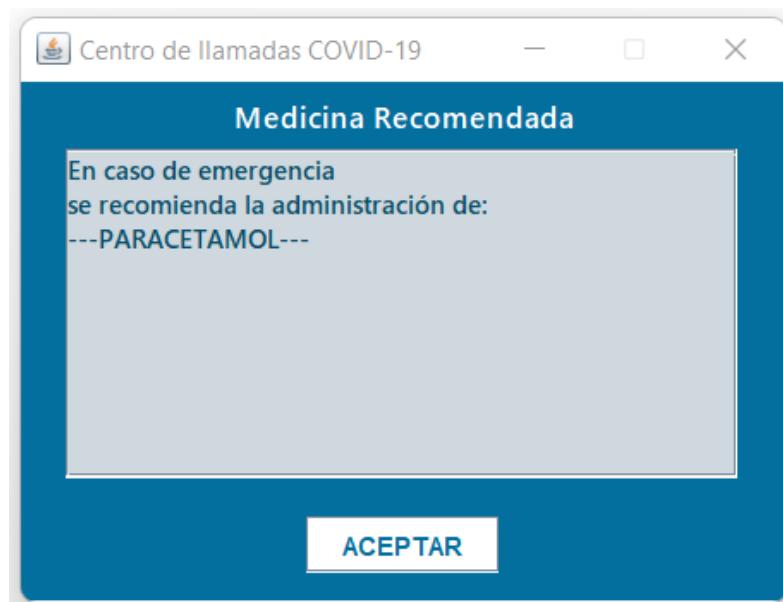


Figura 30. Recetar en caso de emergencia.

### **3.5. Revisión de iteración**

En la Tabla 6. Se presenta el estado de las actividades desarrolladas de la segunda iteración.

**Tabla 6. Revisión de la iteración 2**

Resumen	Clave HU	Responsable	Estado
Transferir llamada	HU01	Sandro Córdova	Terminado
Listar síntomas	HU02	Sandro Córdova	Terminado
Clasificación del paciente	HU02	Sandro Córdova	Terminado
Generar cita médica	HU03	Sandro Córdova	Terminado
Medidas de bioseguridad	HU04	Sandro Córdova	Pendiente
Recetar en casos de emergencia	HU05	Sandro Córdova	Pendiente

### **4. Iteración 3**

#### **4.1. Planificación**

En la Tabla 7. Se presenta las incidencias para la tercera iteración.

**Tabla 7. Tareas de la iteración 3**

Resumen	Clave HU	Responsable	Estado
Agregar cita	HU04	Sandro Córdova	Pendiente
Visualizar cita	HU07	Sandro Córdova	Pendiente
Actualizar cita	HU08	Sandro Córdova	Pendiente
Borrar cita	HU09	Sandro Córdova	Pendiente
Reporte de pacientes	HU06	Sandro Córdova	Pendiente

#### **4.2. Diseño**

La documentación para el diseño del software, se encuentra presentado a detalle en el documento de arquitectura de software (

Anexo 3. Arquitectura de software), en esta sección únicamente se presentarán los diagramas que se han utilizado para poder desarrollar las actividades propuestas en la planificación de la iteración.

#### 4.2.1. Esquema de base de datos

Para el desarrollo de la base de datos se tomó como referencia el presente diagrama, en donde se especifica las colecciones y los tipos de datos que tendrán cada uno de los documentos dentro de la colección.

Fase de Recolección	Fase de Clasificación	Fase de Resultados	Entrenamiento del agente inteligente
llamada	diagnostico	medida_bioseguridad	dominio
paciente	sintoma_paciente	medicamentos	extension
extensiones		cita	numeros simbolos

Figura 31. Esquema de base de datos.

Para el diseño de las actividades planteadas en esta iteración es necesario hacer uso del diagrama de clases especificado en la arquitectura de software (

Anexo 3. Arquitectura de software), para poder observar la funcionalidad que el sistema proporcionará a los usuarios, así también es necesario el diagrama de casos de uso para poder analizar los diferentes escenarios por los que el paciente pasara durante la interacción con el sistema.

#### 4.2.2. Esquema de usos

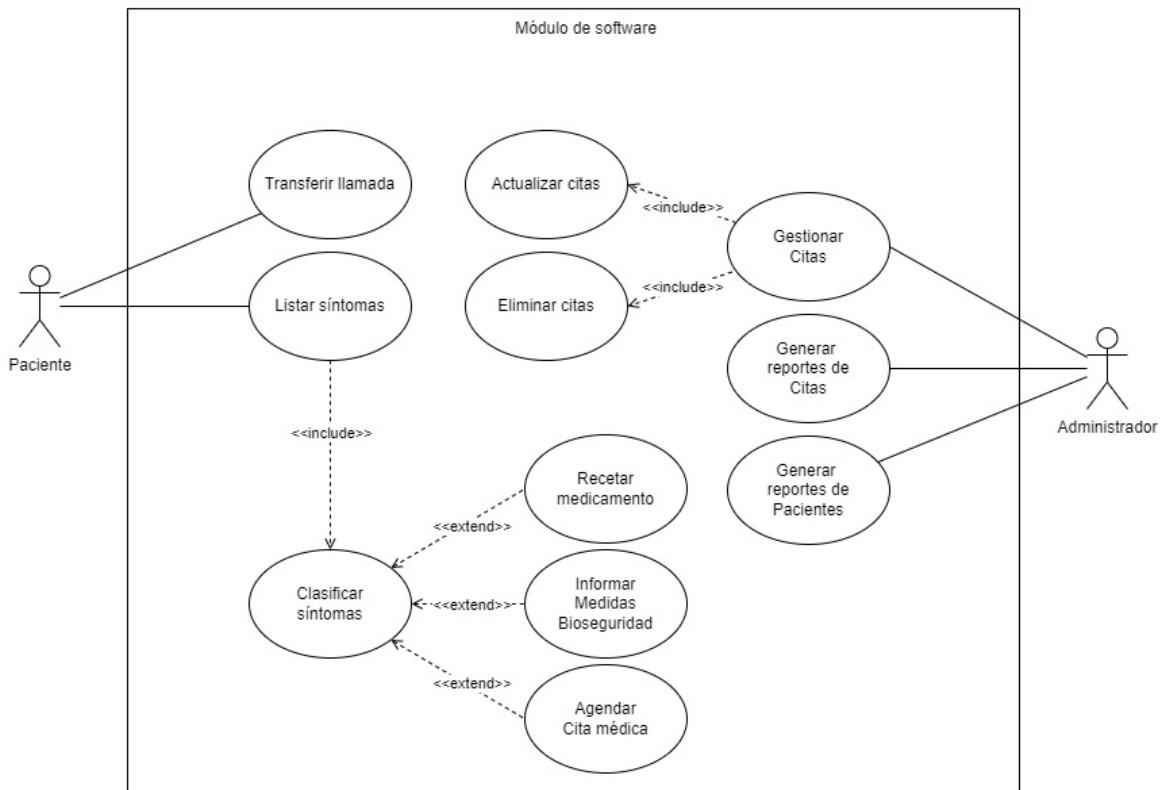


Figura 32. Diagrama de usos.

#### 4.2.3. Esquema de clases

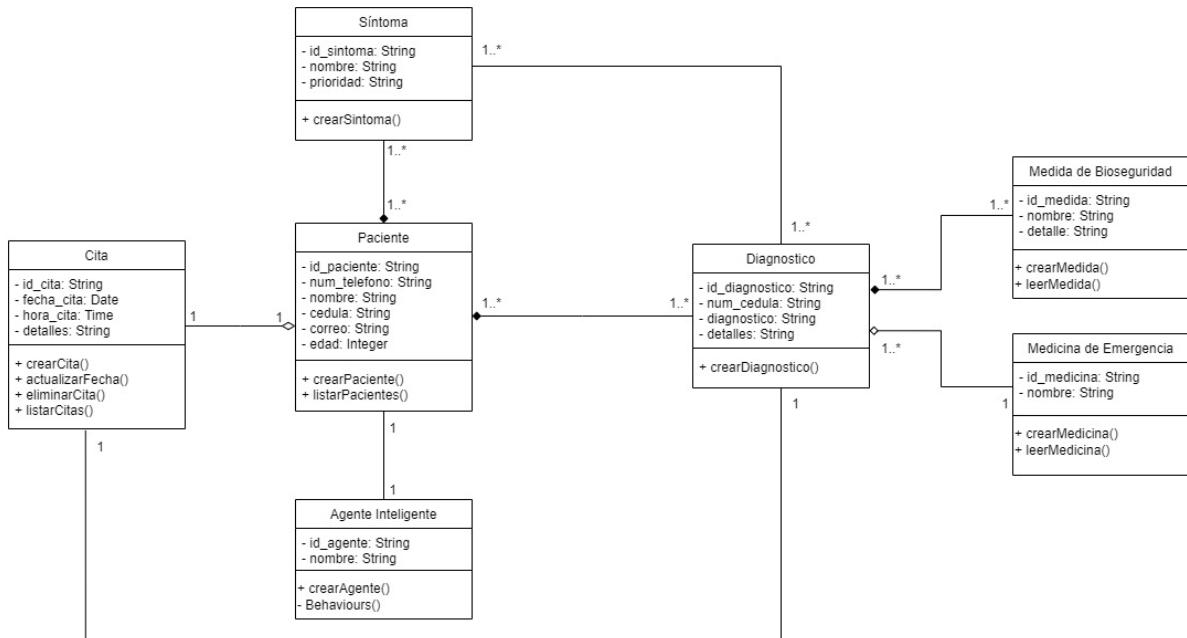


Figura 33. Diagrama de clases.

## 4.3. Codificación

### 4.3.1. Agendar cita

El presente método se encargar de crear el objeto cita médica con los datos ingresados por el paciente.

```
//Inserta la cita generada para el paciente
public boolean insertarCita(String nombres, int edad, String cedula,
    String correo, String sintomas, String diagnostico, String fecha) {
    documento.put("nombres", nombres);
    documento.put("edad", edad);
    documento.put("cedula", cedula);
    documento.put("correo", correo);
    documento.put("sintomas", sintomas);
    documento.put("diagnostico", diagnostico);
    documento.put("fecha", fecha);
   leccionCita.insert(documento);
    return true;
}
```

Figura 34. Agendar cita.

### 4.3.2. Visualizar cita

El presente método se encargar de presentar la cita al usuario y registrarla en la base de datos.

```
//Escuchador para listar las citas de la base de datos
ActionListener alCita = new ActionListener() {
    //Se activa al hacer click en el botón llamar
    @Override
    public void actionPerformed(ActionEvent e) {
        vista_gestion.anuncio1.setText("Citas actuales");
        Conexion conexion = new Conexion();
        ArrayList<Cita> listaCitas = new ArrayList<Cita>();
        listaCitas = conexion.buscarCitas();
        String str = "";
        for (int i = 0; i < listaCitas.size(); i++) {
            Cita cita = new Cita();
            cita = listaCitas.get(i);
            str = "Paciente: " + cita.getNombres()
                + "\nCedula: " + cita.getCedula()
                + "\nFecha de la cita: " + cita.getFecha() + "\n\n";
        }
    }
};
vista_gestion.botonCitas.addActionListener(alCita);
```

Figura 35. Escuchador de listar citas.

```

//Se obtiene la lista de citas que se han comunicado con el call center
public ArrayList<Cita> buscarCitas() {
    DBCursor cursor = colecciónCita.find();
    ArrayList<Cita> listaCitas = new ArrayList<Cita>();
    while (cursor.hasNext()) {
        Cita cita = new Cita();
        DBObject object = cursor.next();
        cita.setNombres(String.valueOf(object.get("nombres")));
        cita.setEdad(Integer.parseInt(String.valueOf(object.get("edad"))));
        cita.setCedula(String.valueOf(object.get("cedula")));
        cita.setCorreo(String.valueOf(object.get("correo")));
        cita.setSintomas(String.valueOf(object.get("sintomas")));
        cita.setDiagnóstico(String.valueOf(object.get("diagnóstico")));
        cita.setFecha(String.valueOf(object.get("fecha")));
        listaCitas.add(cita);
    }
    return listaCitas;
}

```

Figura 36. Buscar cita.

```

//Se obtiene la cita que coincide con el número de cédula ingresado
public Cita buscarCita(ArrayList<Cita> listaCitas, String cedula) {
    for (int i = 0; i < listaCitas.size(); i++) {
        Cita cita = new Cita();
        cita = listaCitas.get(i);
        System.out.println(cita.getCedula());
        if (cita.getCedula().contains(cedula)) {
            return cita;
        }
    }
    return null;
}

```

Figura 37. Visualizar cita.

#### 4.3.3. Actualizar cita

El presente método se encarga de obtener el número de cédula del paciente cuya cita se desea actualizar, se constata que el número de cédula ingresado le pertenezca a un paciente con cita registrada en la base de datos, si cumple con este requisito, el método llama a la función de “ActualizarCita” y se envían los datos que serán actualizados. Para este método, únicamente se permite la actualización de la fecha para la cita.

```
//Escuchador para actualizar una cita de la base de datos
ActionListener alActualizarCita = new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        vista_gestion.anuncio1.setText("Actualizar Cita");
        Conexion conexion = new Conexion();
        ArrayList<Cita> listaCitas = new ArrayList<Cita>();
        listaCitas = conexion.buscarCitas();

        Cita cita = new Cita();
        cita = conexion.buscarCita(listaCitas,
            vista_gestion.cajaCedulaCita.getText());
        if (cita != null) {
            if (conexion.actualizarCita(cita, vista_gestion.cajaFechaCita.getText())) {
                vista_gestion.anuncio1.setText("Cita Actualizada");
            } else {
                vista_gestion.anuncio1.setText("Cita no encontrada");
            }
        } else {
            vista_gestion.anuncio1.setText("Cita no encontrada");
        }
    }
};
vista_gestion.btnActualizarFecha.addActionListener(alActualizarCita);
```

Figura 38. Escuchador para actualizar cita.

```
public boolean actualizarCita(Cita cita, String fechaNueva) {
    documento.put("cedula", cita.getCedula());

    BasicDBObject documentoNuevo = new BasicDBObject();
    documentoNuevo.put("nombres", cita.getNombres());
    documentoNuevo.put("edad", cita.getEdad());
    documentoNuevo.put("cedula", cita.getCedula());
    documentoNuevo.put("correo", cita.getCorreo());
    documentoNuevo.put("sintomas", cita.getSintomas());
    documentoNuevo.put("diagnostico", cita.getDiagnostico());
    documentoNuevo.put("fecha", fechaNueva);

    colecciónCita.findAndModify(documento, documentoNuevo);
    return true;
}
```

Figura 39. Actualizar cita.

#### 4.3.4. Borrar cita

El presente método se encarga de obtener el número de cédula del paciente cuya cita se desea eliminar, se constata que el número de cédula ingresado le pertenezca a un paciente registrado en la base de datos, si cumple con este requisito, el método llama a la función de eliminar cita de la base de datos.

```
//Escuchador para eliminar a una cita de la base de datos
ActionListener alEliminarCita = new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        vista_gestion.anuncio1.setText("Eliminar Cita");
        Conexion conexion = new Conexion();
        ArrayList<Cita> listaCitas = new ArrayList<Cita>();
        listaCitas = conexion.buscarCitas();

        Cita cita = new Cita();
        cita = conexion.buscarCita(listaCitas,
            vista_gestion.cajaCedulaCita.getText());
        if (cita != null) {
            conexion.eliminarCita(cita.getCedula());
            vista_gestion.anuncio1.setText("Cita Eliminada");
        } else {
            vista_gestion.anuncio1.setText("Cita no encontrada");
        }
    }
};
vista_gestion.btnEliminarCita.addActionListener(alEliminarCita);
```

Figura 40. Escuchador de eliminar cita.

```
//Elimina la cita que coincide con el numero de cedula ingresado
public boolean eliminarCita(String cedula) {
    documento.put("cedula", cedula);
    colecciónCita.remove(documento);
    System.out.println("Cita con número de cédula: "+cedula+" ha sido eliminada");
    return true;
}
```

Figura 41. Eliminar cita.

#### 4.3.5. Visualizar pacientes

```
//Se obtiene el paciente que coincide con el número de cédula ingresado
public Paciente buscarPaciente(ArrayList<Paciente> listaPacientes, String cedula) {
    for (int i = 0; i < listaPacientes.size(); i++) {
        Paciente paciente = new Paciente();
        paciente = listaPacientes.get(i);
        if (paciente.getCedula().contains(cedula)) {
            return paciente;
        }
    }
    return null;
}
```

Figura 42. Buscar paciente.

#### 4.3.6. Actualizar pacientes

```
//Busca el paciente y actualiza sus datos
public boolean actualizarPaciente(Paciente paciente, Paciente pacienteNuevo) {
    documento.put("cedula", paciente.getCedula());

    BasicDBObject documentoNuevo = new BasicDBObject();
    documentoNuevo.put("telefono", pacienteNuevo.getNum_telefono());
    documentoNuevo.put("nombre", pacienteNuevo.getNombre());
    documentoNuevo.put("cedula", pacienteNuevo.getCedula());
    documentoNuevo.put("correo", pacienteNuevo.getCorreo());
    documentoNuevo.put("edad", pacienteNuevo.getEdad());

   leccionPaciente.findAndModify(documento, documentoNuevo);
    return true;
}
```

Figura 43. Actualizar paciente.

#### 4.3.7. Borrar módulos

```
//Elimina al paciente con el numero de cedula ingresado
public boolean eliminarPaciente(String cedula) {
    documento.put("cedula", cedula);
   leccionPaciente.remove(documento);
    System.out.println("Paciente con número de cédula: "+cedula+" ha sido eliminado");
    return true;
}
```

Figura 44. Eliminar paciente.

### 4.4. Pruebas

#### 4.4.1. Visualizar cita

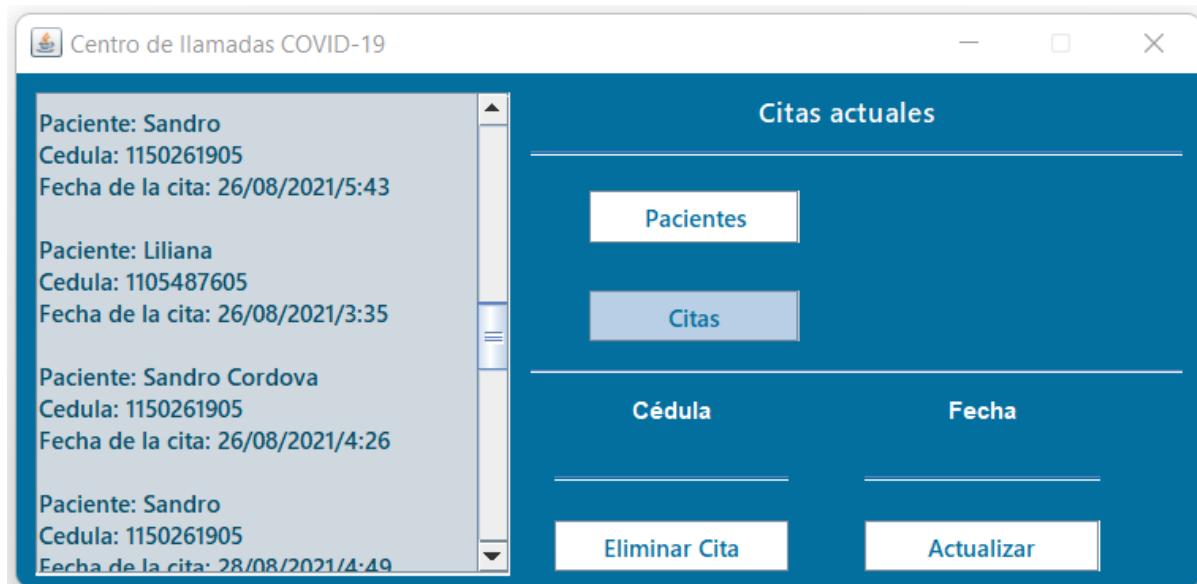


Figura 45. Lista de las citas.

#### 4.4.2. Actualizar cita



Figura 46. Vista de actualizar cita.

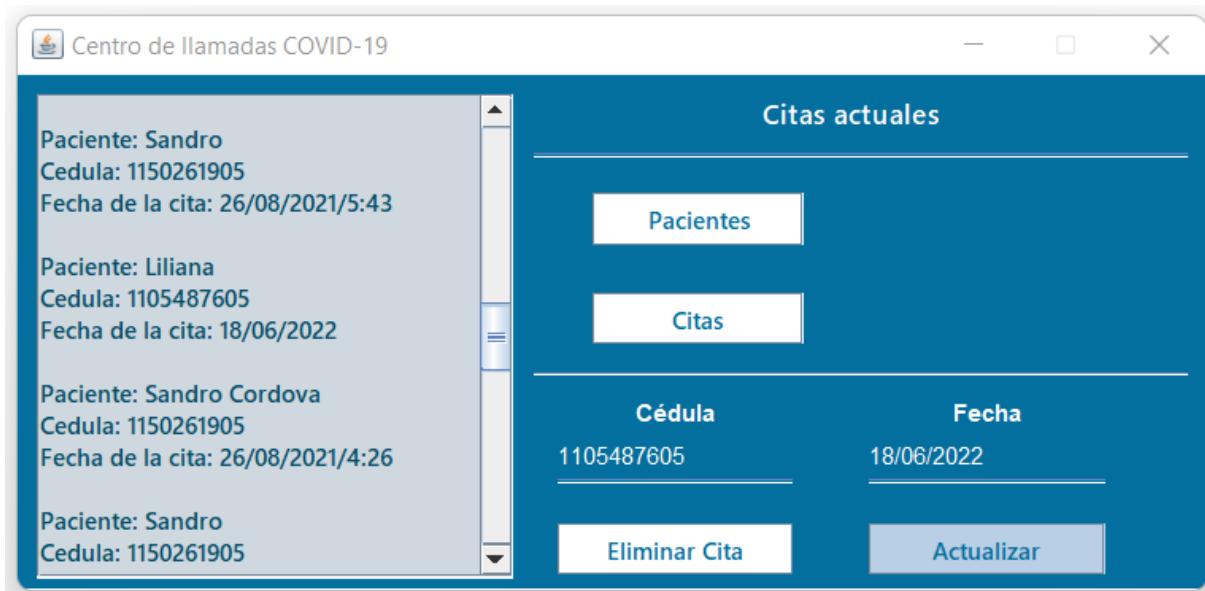


Figura 47. Vista de cita actualizada.

#### 4.4.3. Borrar cita

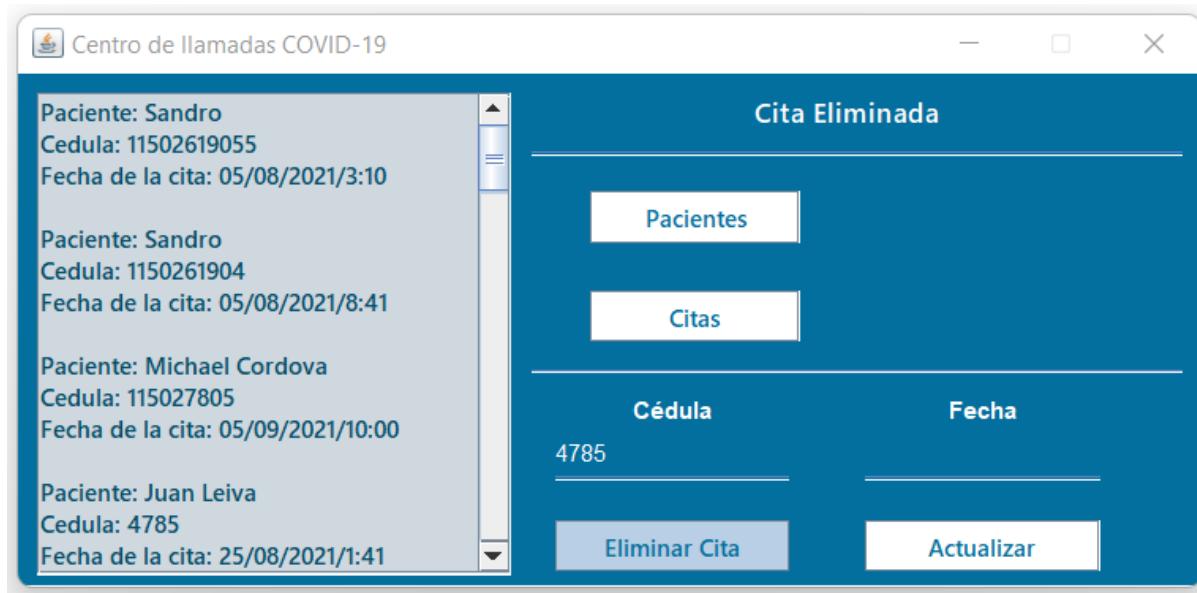


Figura 48. Vista de eliminar cita.

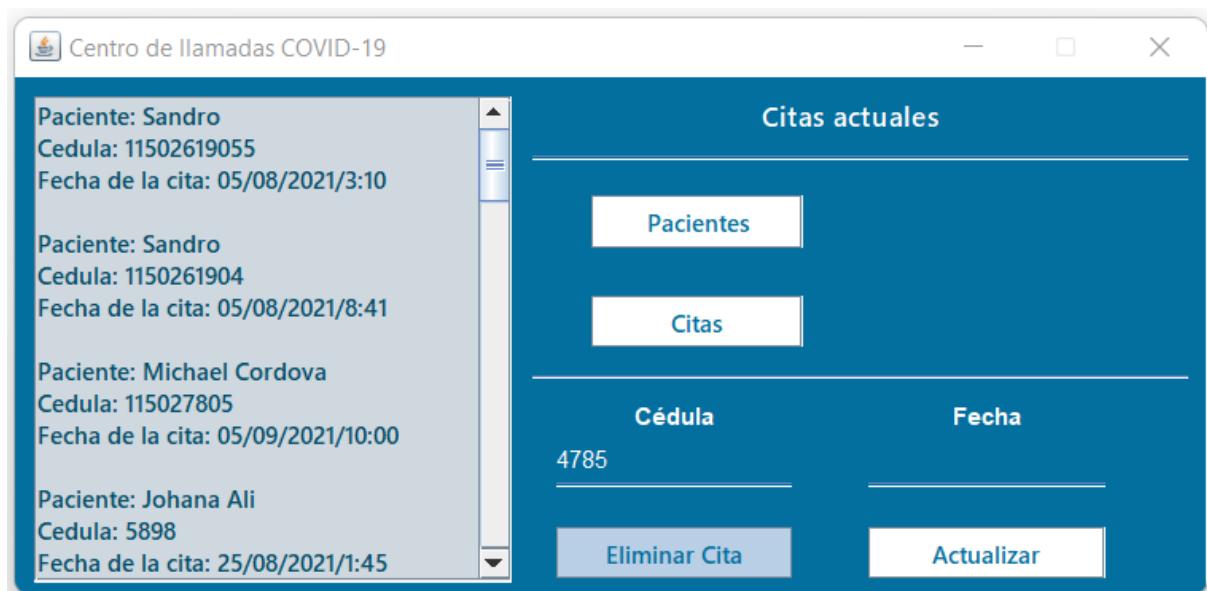
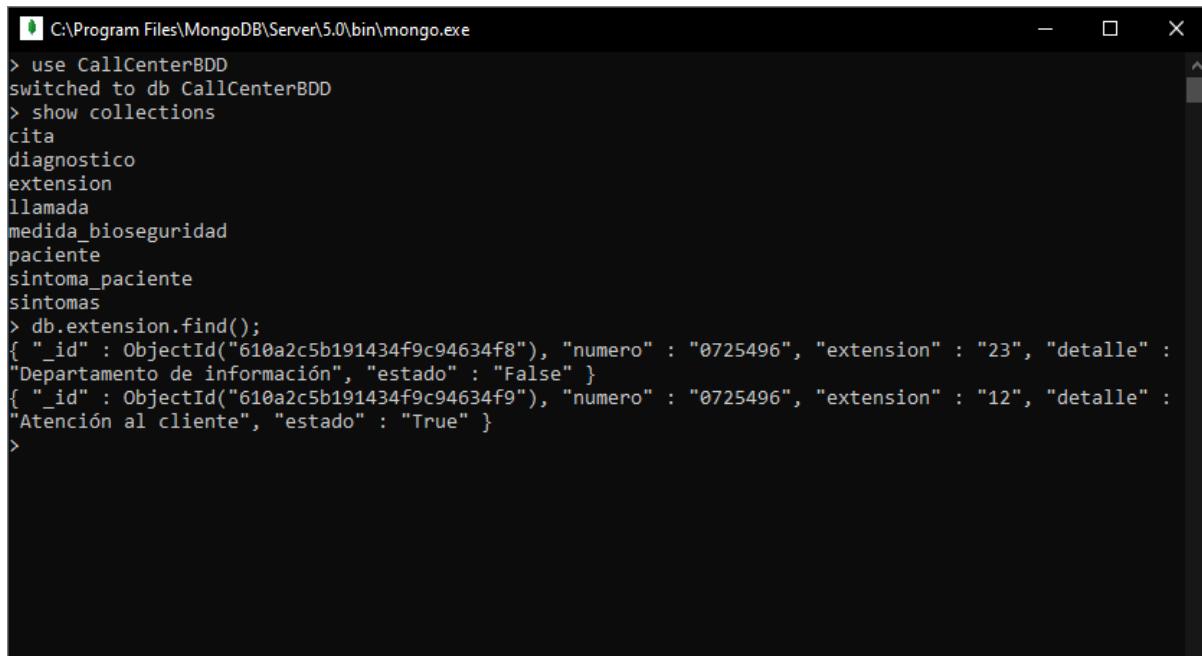


Figura 49. Vista de cita eliminada.

#### 4.4.4. Gestión de información general

Las siguientes imágenes muestran los registros de la base de datos del call center, utilizados por el sistema para poder llevar a cabo las diferentes actividades.

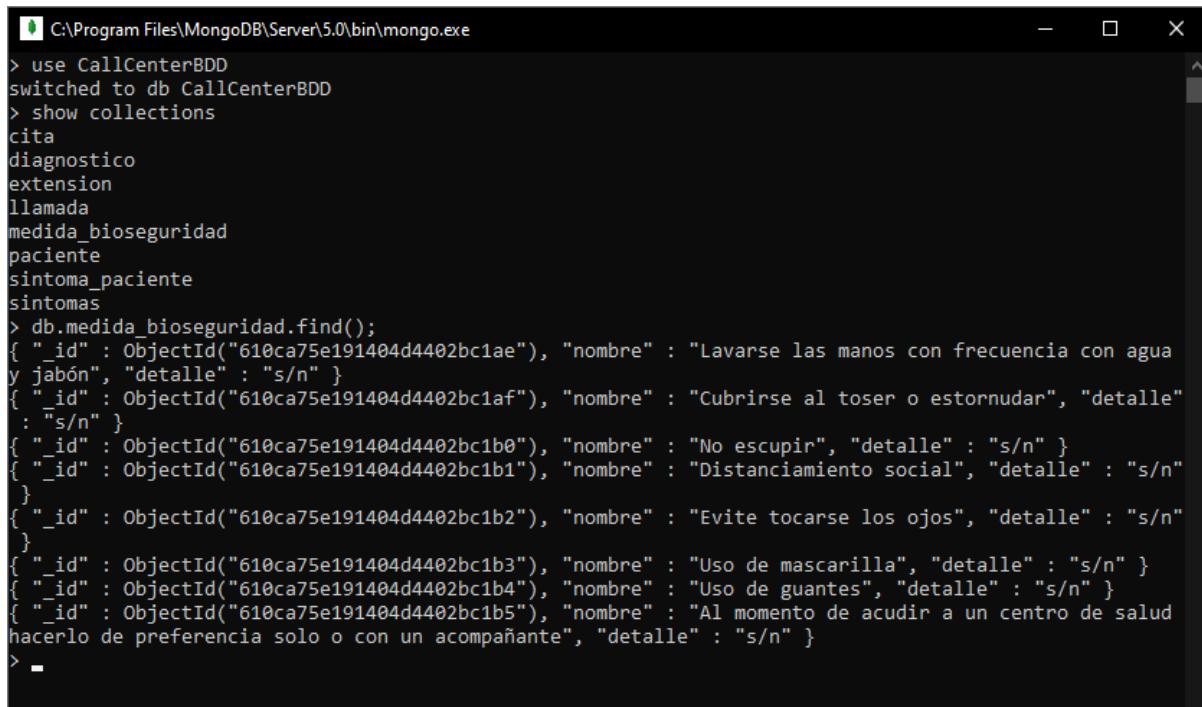
##### Extensiones registradas en la base de datos



```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> use CallCenterBDD
switched to db CallCenterBDD
> show collections
cita
diagnostico
extension
llamada
medida_bioseguridad
paciente
sintoma_paciente
sintomas
> db.extension.find();
{ "_id" : ObjectId("610a2c5b191434f9c94634f8"), "numero" : "0725496", "extension" : "23", "detalle" : "Departamento de información", "estado" : "False" }
{ "_id" : ObjectId("610a2c5b191434f9c94634f9"), "numero" : "0725496", "extension" : "12", "detalle" : "Atención al cliente", "estado" : "True" }
>
```

Figura 50. Extensiones registradas.

##### Medidas de bioseguridad



```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> use CallCenterBDD
switched to db CallCenterBDD
> show collections
cita
diagnostico
extension
llamada
medida_bioseguridad
paciente
sintoma_paciente
sintomas
> db.medida_bioseguridad.find();
{ "_id" : ObjectId("610ca75e191404d4402bc1ae"), "nombre" : "Lavarse las manos con frecuencia con agua y jabón", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1af"), "nombre" : "Cubrirse al toser o estornudar", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b0"), "nombre" : "No escupir", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b1"), "nombre" : "Distanciamiento social", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b2"), "nombre" : "Evite tocarse los ojos", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b3"), "nombre" : "Uso de mascarilla", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b4"), "nombre" : "Uso de guantes", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b5"), "nombre" : "Al momento de acudir a un centro de salud hacerlo de preferencia solo o con un acompañante", "detalle" : "s/n" }
>
```

Figura 51. Medidas de bioseguridad.

#### 4.4.5. Visualizar módulos

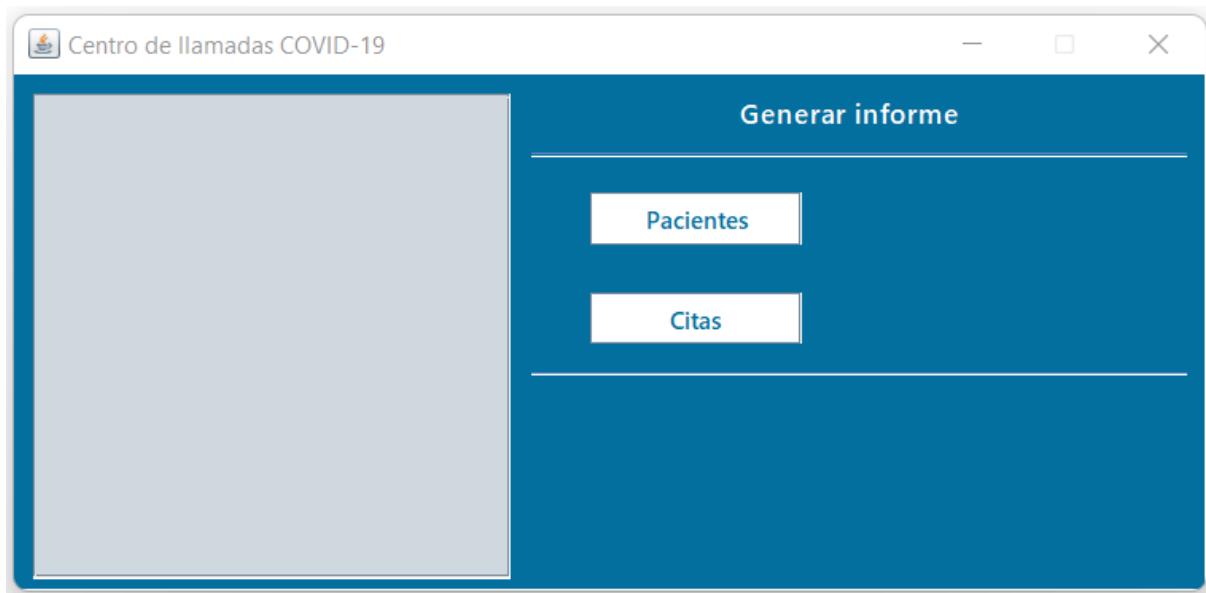


Figura 52. Vista generar informe.

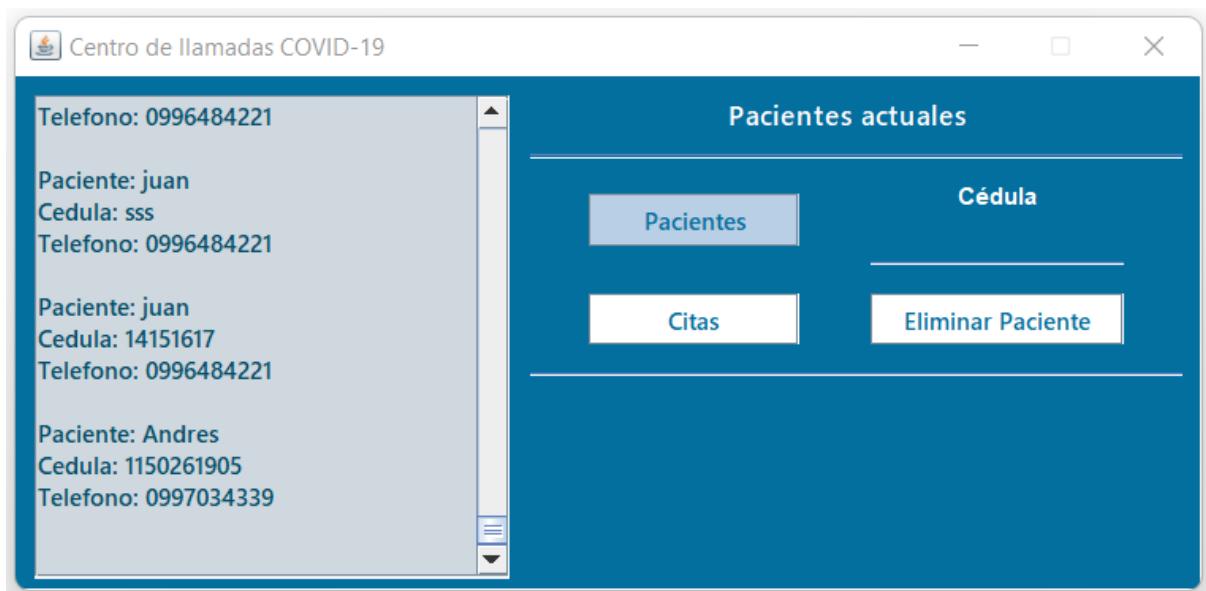


Figura 53. Vista de lista de pacientes actuales.

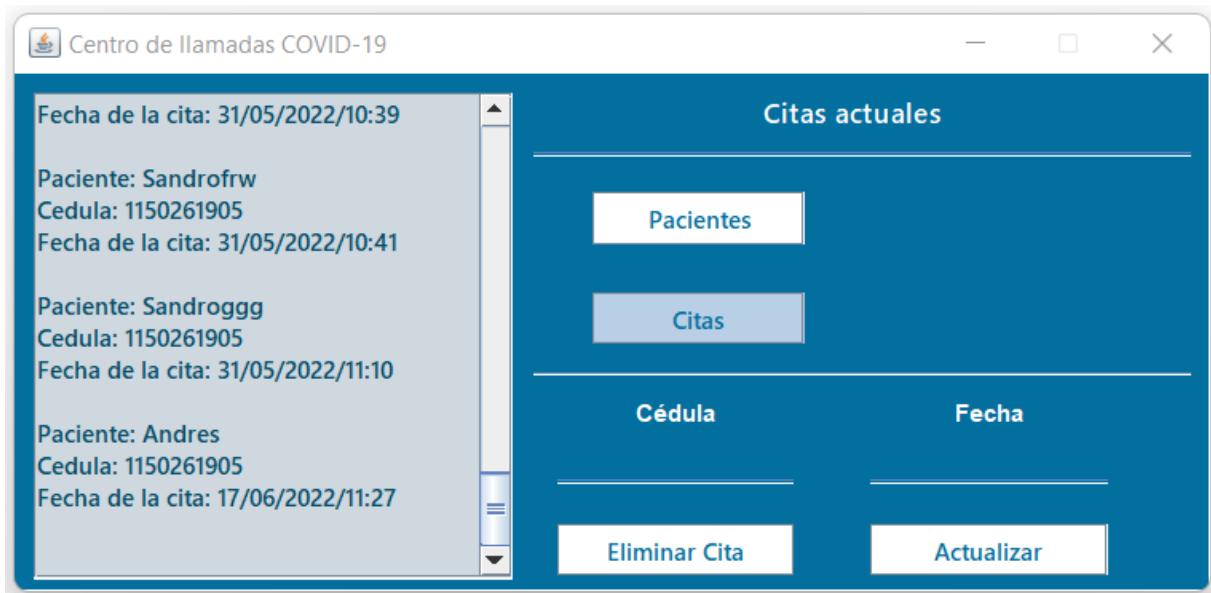


Figura 54. Vista de lista de citas actuales.

#### 4.4.6. Borrar módulos

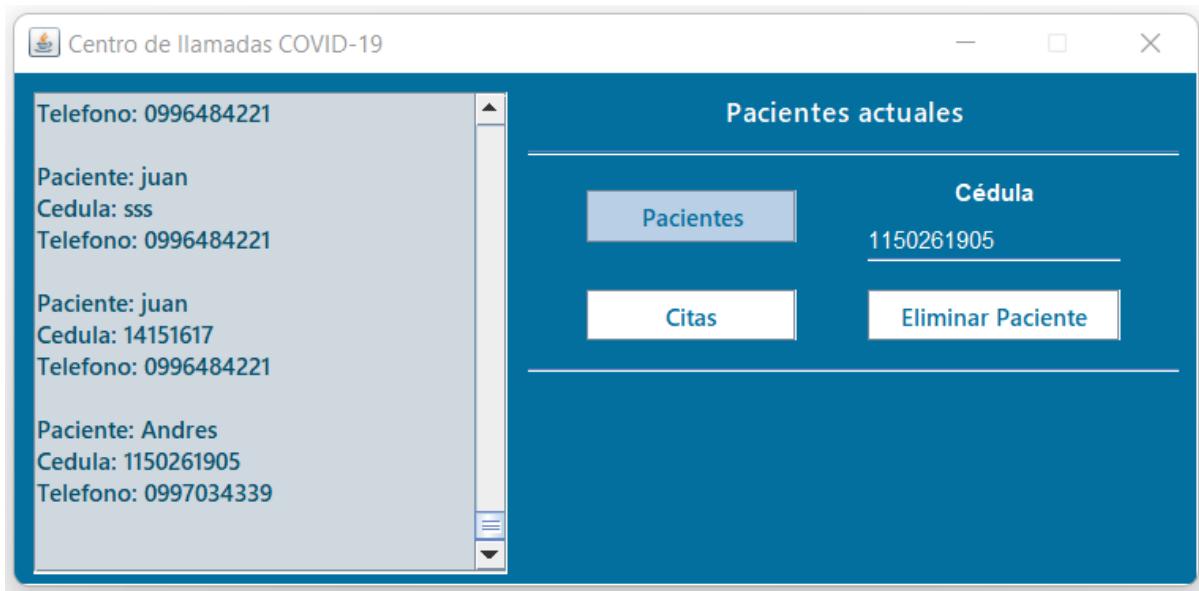


Figura 55. Vista de eliminar paciente.

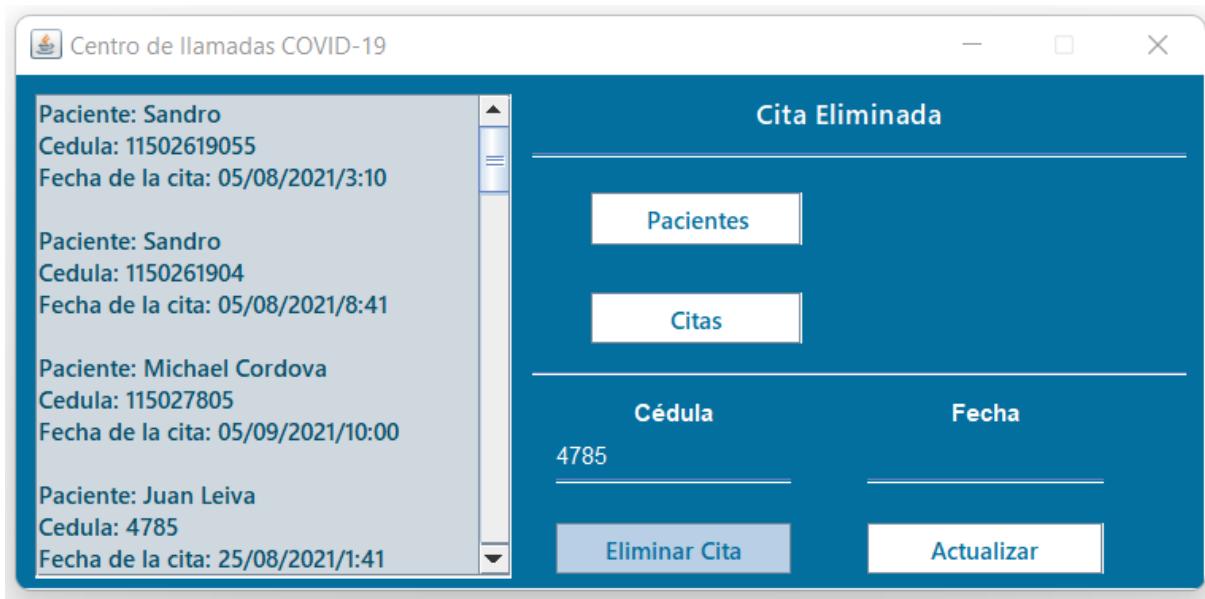


Figura 56. Vista de eliminar cita.

#### 4.5. Revisión de iteración

En la Tabla 8 se presenta el estado de las actividades desarrolladas de la tercera iteración.

**Tabla 8. Revisión de la iteración 3**

Resumen	Clave HU	Responsable	Estado
Agregar cita	HU04	Sandro Córdova	Terminado
Visualizar cita	HU07	Sandro Córdova	Terminado
Actualizar cita	HU08	Sandro Córdova	Terminado
Borrar cita	HU09	Sandro Córdova	Terminado
Reporte de pacientes	HU06	Sandro Córdova	Terminado

## **5. Iteración 4**

### **5.1. Planificación**

En la Tabla 9 se presenta las incidencias para la cuarta iteración.

**Tabla 9. Tablas de la iteración 4**

Resumen	Responsable	Estado
Voz aguda y agradable	Sandro Córdova	Pendiente
Diseño de interfaces gráficas	Sandro Córdova	Terminado

### **5.2. Diseño**

La documentación para el diseño del software, se encuentra presentado a detalle en el documento de arquitectura de software (

Anexo 3. Arquitectura de software), en esta sección únicamente se presentarán los diagramas que se han utilizado para poder desarrollar las actividades propuestas en la planificación de la iteración.

Para el diseño de las actividades planteadas en esta iteración es necesario hacer uso del diagrama de clases especificado en la arquitectura de software (

Anexo 3. Arquitectura de software), para poder observar la funcionalidad que el sistema proporcionará a los usuarios, así también es necesario el diagrama de casos de uso para poder analizar los diferentes escenarios por los que el paciente pasara durante la interacción con el sistema.

### 5.3. Codificación

#### 5.3.1. Generar reportes

En las siguientes imágenes se puede evidenciar que el administrador puede obtener la lista de los pacientes que se han registrado al sistema, así como también las citas generadas para los mismos.

#### Visualizar cita

El presente método se encargará de visualizar cita y registrarla en la base de datos

```
//Escuchador para listar las citas de la base de datos
ActionListener alCita = new ActionListener() {
    //Se activa al hacer click en el botón llamar
    @Override
    public void actionPerformed(ActionEvent e) {
        vista_gestion.anuncio1.setText("Citas actuales");
        Conexion conexion = new Conexion();
        ArrayList<Cita> listaCitas = new ArrayList<Cita>();
        listaCitas = conexion.buscarCitas();
        String str = "";
        for (int i = 0; i < listaCitas.size(); i++) {
            Cita cita = new Cita();
            cita = listaCitas.get(i);
            str = "Paciente: " + cita.getNombres()
                + "\nCedula: " + cita.getCedula()
                + "\nFecha de la cita: " + cita.getFecha() + "\n\n";
        }
    }
};
vista_gestion.botonCitas.addActionListener(alCita);
```

Figura 57. Escuchador de listar citas.

```
//Se obtiene la lista de citas que se han comunicado con el call center
public ArrayList<Cita> buscarCitas() {
    DBCursor cursor = colecciónCita.find();
    ArrayList<Cita> listaCitas = new ArrayList<Cita>();
    while (cursor.hasNext()) {
        Cita cita = new Cita();
       DBObject object = cursor.next();
        cita.setNombres(String.valueOf(object.get("nombres")));
        cita.setEdad(Integer.parseInt(String.valueOf(object.get("edad"))));
        cita.setCedula(String.valueOf(object.get("cedula")));
        cita.setCorreo(String.valueOf(object.get("correo")));
        cita.setSintomas(String.valueOf(object.get("sintomas")));
        cita.setDiagnóstico(String.valueOf(object.get("diagnóstico")));
        cita.setFecha(String.valueOf(object.get("fecha")));
        listaCitas.add(cita);
    }
    return listaCitas;
}
```

Figura 58. Buscar cita.

```
//Se obtiene la cita que coincide con el número de cédula ingresado
public Cita buscarCita(ArrayList<Cita> listaCitas, String cedula) {
    for (int i = 0; i < listaCitas.size(); i++) {
        Cita cita = new Cita();
        cita = listaCitas.get(i);
        System.out.println(cita.getCedula());
        if (cita.getCedula().contains(cedula)) {
            return cita;
        }
    }
    return null;
}
```

Figura 59. Visualizar cita.

## Visualizar pacientes

```
//Se obtiene el paciente que coincide con el número de cédula ingresado
public Paciente buscarPaciente(ArrayList<Paciente> listaPacientes, String cedula) {
    for (int i = 0; i < listaPacientes.size(); i++) {
        Paciente paciente = new Paciente();
        paciente = listaPacientes.get(i);
        if (paciente.getCedula().contains(cedula)) {
            return paciente;
        }
    }
    return null;
}
```

Figura 60. Buscar paciente.

### 5.3.2. Voz aguda y agradable

#### Leer información

Los siguientes métodos son los encargados de cargar las configuraciones de voz para poder comunicarse con el paciente, se hizo uso de la librería "speech" de java y se usó una voz de género femenino para poder cumplir con los requisitos establecidos.

```
import javax.speech.*;
import javax.speech.synthesis.*;
import java.util.*;

public class Lee {
    //Se inicializa el sintetizador de voz
    Synthesizer synth;
    public Lee() {
        //Se cargan las configuraciones de la voz
        SynthesizerModeDesc required = new SynthesizerModeDesc();
        required.setLocale(Locale.ROOT);
        Voice voice = new Voice(null, Voice.GENDER_FEMALE, Voice.GENDER_FEMALE, null);
        required.addVoice(voice);
    }
}
```

Figura 61. Cargar configuraciones de voz.

```
//Metodo para leer el texto ingresado
public void leer(String texto) {
    try {
        synth = Central.createSynthesizer(null);
        synth.allocate();
        synth.resume();
        synth.speakPlainText(texto, null);
        synth.waitEngineState(Synthesizer.QUEUE_EMPTY);
        synth.deallocate();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Figura 62. Lectura del texto ingresado.

## 5.4. Pruebas

### 5.4.1. Generar reportes

La presente vista fue desarrollada para poder listar la información de los pacientes y de las citas que se encuentran registrados en la base de datos.

#### Visualizar cita

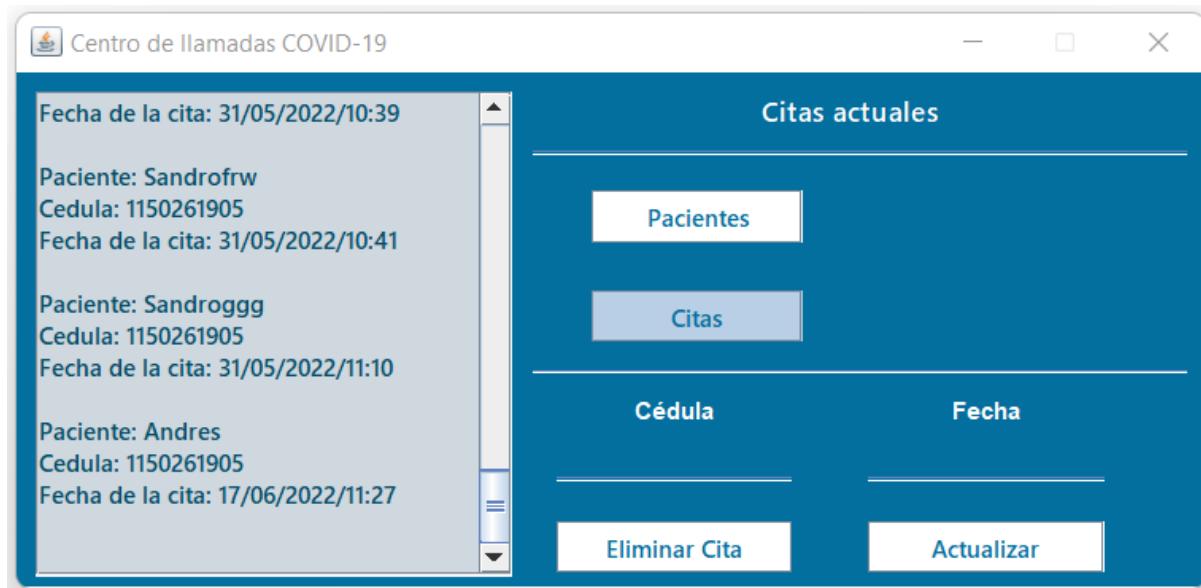


Figura 63. Lista de las citas.

#### Visualizar módulos

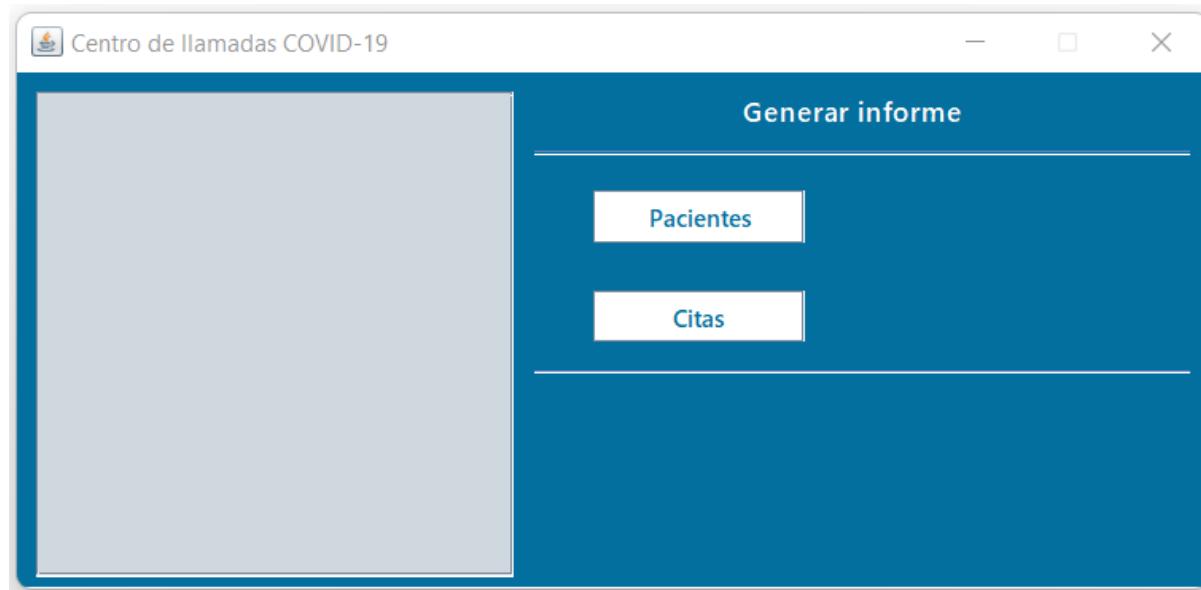


Figura 64. Vista generar informe.

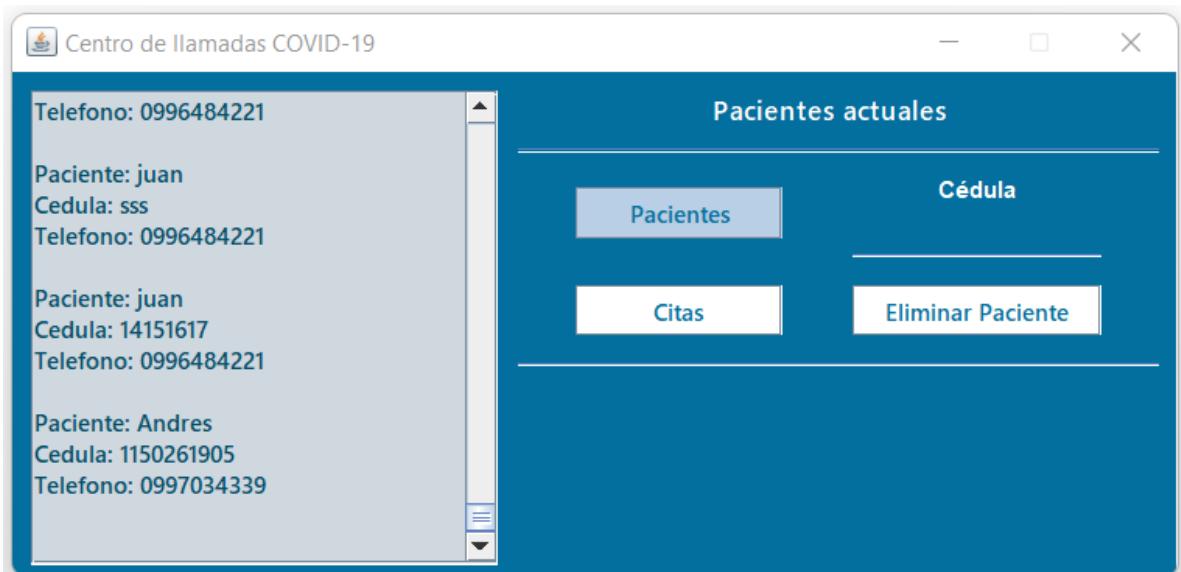


Figura 65. Vista de lista de pacientes actuales.

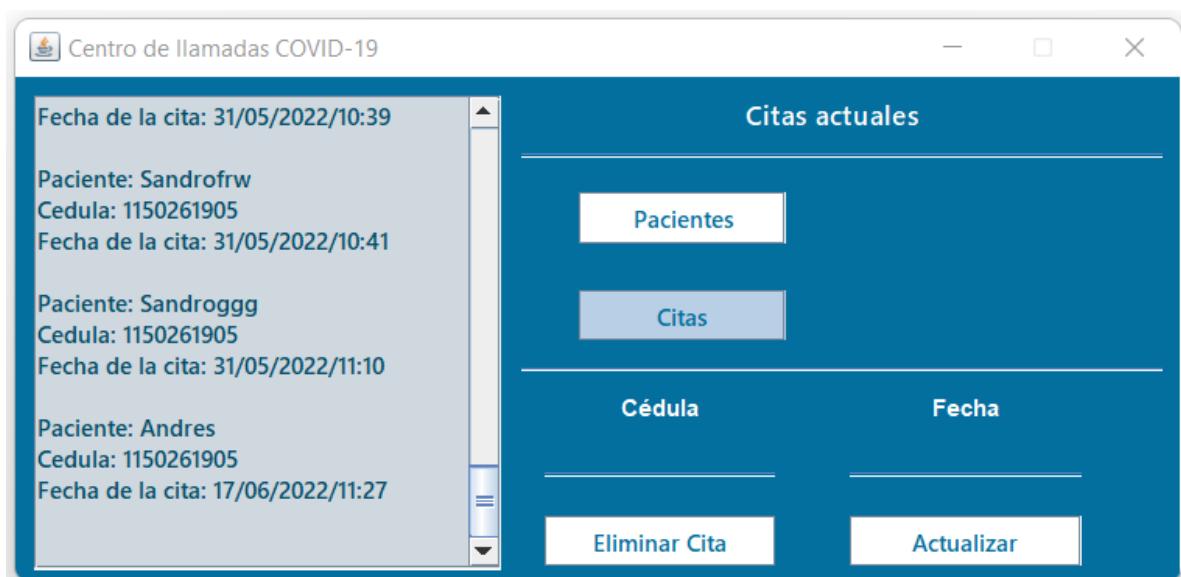


Figura 66. Vista de lista de citas actuales.

#### **5.4.2. Tiempo de interacción**

El tiempo de la interacción entre el sistema y el paciente debe ser validada por el usuario final en la fase de pruebas.

#### **5.4.3. Voz aguda y agradable**

Durante la fase de codificación se hizo uso de una voz femenina para la comunicación con el paciente, para el criterio de la persona encargada del módulo, dicha voz cumple con las especificaciones requeridas, sin embargo, estas características serán aceptadas y rechazadas por parte del usuario final en la fase de pruebas del sistema. La voz agradable y aguda utilizada para el software desarrollado se pudo inferir de las entrevistas iniciales aplicadas a los diferentes centros de salud de la ciudad de Loja (

Anexo 3. Arquitectura de software).

### 5.5. Revisión de iteración

En la Tabla10 se presenta el estado de las actividades desarrolladas de la cuarta iteración.

**Tabla 10. Revisión de la iteración 4.**

Resumen	Responsable	Estado
Voz aguda y agradable	Sandro Córdova	Terminado
Diseño de interfaces gráficas	Sandro Córdova	Terminado

## **Anexo 5. Plan de pruebas Unitarias**

### **Plan de pruebas unitarias**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

## **1. Establecer los objetivos**

El objetivo de este documento es verificar la funcionalidad correcta del cada uno de los métodos implementados en el sistema, aislando cada parte del código y mostrar que las partes individuales son correctas.

## **2. Alcance**

Los casos de pruebas están dirigidos para el docente Ing. José Oswaldo Guamán Quinché (director del Trabajo de Titulación), docente de la Carrera de Ingeniería en Sistemas / Computación de la Universidad Nacional de Loja, quien valida los diferentes casos de pruebas. Y el estudiante Sandro Michael Córdova Carrión que es el que genera y registra los diferentes casos de pruebas.

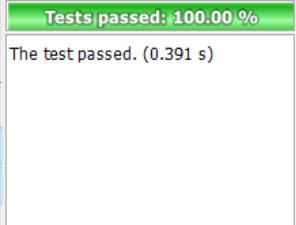
## **3. Definición de los casos de pruebas**

En este apartado se describe en detalle cada uno de los casos de pruebas (CP) que se identificaron para verificar la funcionalidad de cada uno de los métodos del sistema haciendo uso de la librería TestNG brindada por Java.

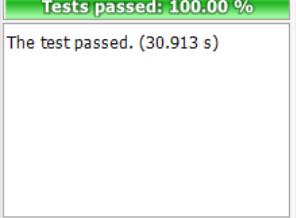
**Tabla 1. Casos de pruebas unitarias**

Código	Descripción	Método
CP01	Se corrobora el correcto funcionamiento del método principal encargado de crear el agente inteligente que tomará el control del sistema.	Main()
CP02	Se corrobora el correcto funcionamiento del método encargado de comunicar al agente inteligente que se ha registrado una nueva llamada.	hay_llamada()
CP03	Se corrobora el correcto funcionamiento del método encargado de transferir la llamada del usuario a una de las extensiones habilitadas.	transferir_llamada()
CP04	Se corrobora el correcto funcionamiento del método encargado de presentar la información brindada por el sistema mediante voz.	Leer()

**Tabla 2. Caso de prueba 1**

CP01				
Nº	Descripción	Método	¿Aceptado ?	Observaciones
1	Se corrobora el correcto funcionamiento del método principal encargado de crear el agente inteligente que tomará el control del sistema.	Main()	Sí	N/A
<b>Capturas TestNG</b>				
<pre>@Test public void testMain() {     System.out.println("main");     String[] args = null;     Main.main(args); }</pre>				
<b>Resultado</b>				
 <pre>Tests passed: 100.00 % The test passed. (0.391 s)</pre>				
<pre>main ===== Command line suite Total tests run: 1, Failures: 0, Skips: 0 =====</pre>				

**Tabla 3. Caso de prueba 2**

CP02				
Nº	Descripción	Método	¿Aceptado ?	Observaciones
2	Se corrobora el correcto funcionamiento del método encargado de comunicar al agente inteligente que se ha registrado una nueva llamada.	hay_llamada()	Sí	N/A
<b>Capturas TestNG</b>				
<pre>@Test public void testHay_llamada() {     System.out.println("hay_llamada");     Llamada llamada = null;     Main.hay_llamada(llamada); }</pre>				
<b>Resultado</b>				
 <pre>Tests passed: 100.00 % The test passed. (30.913 s)</pre>				
<pre>hay_llamada Using 64-bit native code - SAPI4 is NOT supported CloudGarden's JSAPI1.0 implementation Version 1.7.0_x64 Implementation contained in files cgjsapi.jar and cgjsapi170_x64.dll Esperando respuesta del usuario Lectura del archivo =====</pre>				

--

Tabla 4. Caso de prueba 3

CP03				
Nº	Descripción	Método	¿Aceptado?	Observaciones
3	Se corrobora el correcto funcionamiento del método encargado de transferir la llamada del usuario a una de las extensiones habilitadas.	transferir_llamada()	Sí	N/A

#### Capturas TestNG

```

@Test
public void testTransferir_llamada() {
    System.out.println("transferir_llamada");
    Llamada llamada = null;
    boolean expResult = true;
    boolean result = Main.transferir_llamada(llamada);
    assertEquals(result, expResult);
}

```

#### Resultado

Tests passed: 100.00 %	
The test passed. (0.13 s)	<pre> transferir_llamada Base de datos conectada.... Su llamada ha sido transferida al:Atención al cliente extensión: 12 ===== Command line suite Total tests run: 1, Failures: 0, Skips: 0 =====</pre>

Tabla 5. Caso de prueba 4.

CP04				
Nº	Descripción	Método	¿Aceptado?	Observaciones
4	Se corrobora el correcto funcionamiento del método encargado de presentar la información brindada por el sistema mediante voz.	leer()	Sí	N/A

#### Capturas TestNG

```

@Test
public void testLeer() {
    System.out.println("leer");
    String texto = "Este es un ejemplo";
    Lee instance = new Lee();
    instance.leer(texto);
}

```

## Resultado

<b>Tests passed: 100.00 %</b>	
The test passed. (2.796 s)	leer Using 64-bit native code - SAPI4 is NOT supported CloudGarden's JSAPI1.0 implementation Version 1.7.0_x64 Implementation contained in files cgjsapi.jar and cgjsapi170_x64.dll ===== Command line suite Total tests run: 1, Failures: 0, Skips: 0

## **Anexo 6. Plan de pruebas funcionales**

### **Plan de pruebas funcionales**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente

#### **4. Establecer los objetivos**

El presente documento forma parte del Trabajo de Titulación denominado “Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente” y tiene como objetivo recoger los casos de pruebas que verifican que el módulo de software satisface los requisitos especificados.

#### **5. Alcance**

Los casos de pruebas están dirigidos para el docente Ing. José Oswaldo Guamán Quinche (director del Trabajo de Titulación), docente de la Carrera de Ingeniería en Sistemas / Computación de la Universidad Nacional de Loja, quien valida los diferentes casos de pruebas. Y el estudiante Sandro Michael Córdova Carrión que es el que genera y registra los diferentes casos de pruebas.

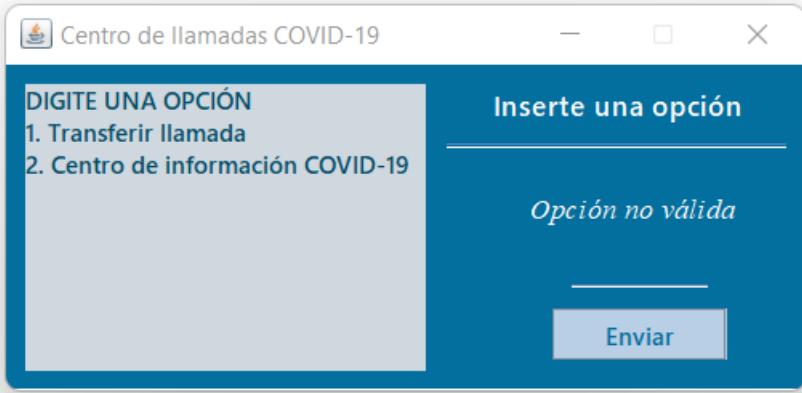
#### **6. Definición de los casos de pruebas**

En este apartado se describe en detalle cada uno de los casos de pruebas (CP) que se identificaron para verificar la funcionalidad del sistema. Además de indicarse aquellos que deben realizarse para asegurar el correcto despliegue del módulo de software.

**Tabla 1. Casos de pruebas funcionales**

Código	Descripción	Resultado esperado
CP01	Se probará la respuesta del módulo de software con los campos vacíos.	El sistema no debe permitir el envío de datos erróneos, debe mostrar un mensaje que indique que los campos deben ser llenados de forma correcta.
CP02	Se probará la respuesta del módulo de software “Transferir llamada”.	El sistema debe transferir la llamada del usuario a una de las extensiones registradas en la base de datos.
CP03	Se probará la creación de una cita médica al usuario basado en sus síntomas.	El sistema debe crear una cita médica con los datos brindados por el usuario.
CP04	Se probará la presentación de medidas de bioseguridad al usuario basado en sus síntomas.	El sistema debe listar las medidas de bioseguridad al usuario.
CP05	Se probará si el sistema presentará los medicamentos que el usuario puede usar en caso de emergencia.	El sistema debe presentar al usuario la medicina recomendada en caso de emergencia.
CP06	Se probará si el sistema permite la modificación de los datos de una cita médica.	El sistema debe actualizar los datos de la cita médica del usuario seleccionado.
CP07	Se probará si el sistema permite la eliminar de los datos de una cita médica.	El sistema debe eliminar los datos de la cita médica del usuario seleccionado.

**Tabla 2. Caso de prueba 1**

Registrar con campos en blanco	CP01	
	¿Aceptado?	Sí
<b>Descripción:</b> Se probará la respuesta del módulo de software con los campos vacíos.		
<b>Prerrequisitos</b> <ul style="list-style-type: none"> <li>• Ingresar al sistema</li> <li>• Iniciar la comunicación con el centro de información de Covid-19.</li> </ul>		
<b>Pasos:</b> <ol style="list-style-type: none"> <li>1. Intentar ingresar al sistema sin insertar los datos requeridos por el mismo.</li> <li>2. Intentar enviar datos erróneos al sistema.</li> </ol>		
<b>Resultado esperado:</b> El sistema no debe permitir el envío de datos erróneos, debe mostrar un mensaje que indique que los campos deben ser llenados de forma correcta.		
<b>Resultado obtenido:</b> <ul style="list-style-type: none"> <li>• El sistema no permite el ingreso de campos en blanco.</li> </ul>  <p>The screenshot shows a window titled "Centro de llamadas COVID-19". The main message is "INGRESE EL NÚMERO DE TELÉFONO" with an empty input field below it. A blue error message at the bottom states: "El número ingresado no es válido".</p>  <p>The screenshot shows a window titled "Centro de llamadas COVID-19". On the left, there's a sidebar with options: "DIGITE UNA OPCIÓN" (1. Transferir llamada, 2. Centro de información COVID-19). The main area has a message "Inserte una opción" with an empty input field below it. A blue error message at the bottom states: "Opción no válida".</p>		
<ul style="list-style-type: none"> <li>• El sistema no permite el ingreso de datos erróneos.</li> </ul>		

Centro de Llamadas COVID-19

### DATOS PERSONALES

Nombre: Sandro

Cédula: \_\_\_\_\_  Pasaporte

Correo: \_\_\_\_\_

Edad: \_\_\_\_\_

*Cédula No Válida*

**ENVIAR**

Centro de Llamadas COVID-19

### DATOS PERSONALES

Nombre: Xavier

Cédula: 1150261905  Pasaporte

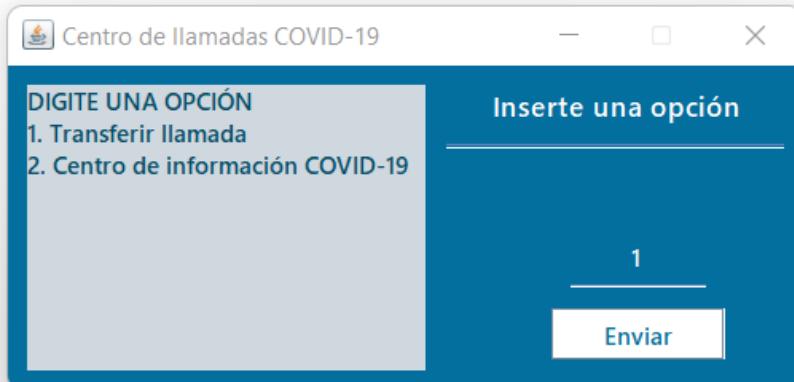
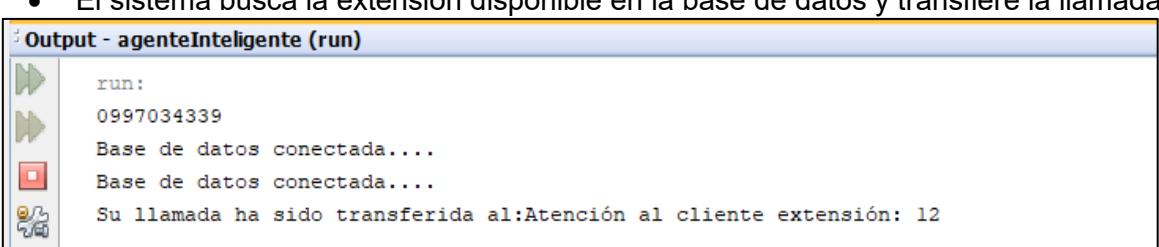
Correo: 2 \_\_\_\_\_

Edad: 38 \_\_\_\_\_

*Correo No Válido*

**ENVIAR**

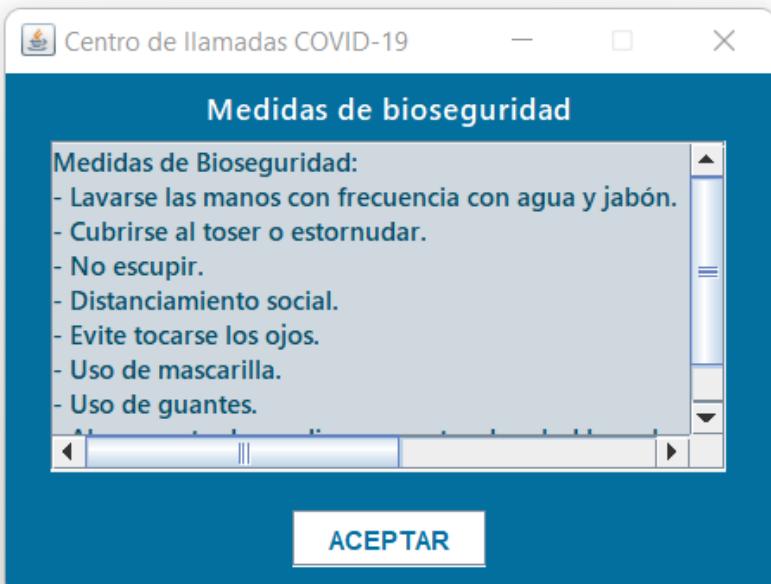
**Tabla 3. Caso de prueba 2**

Transferir llamada	CP02	
	¿Aceptado?	Sí
<b>Descripción:</b> Se probará la respuesta del módulo de software “Transferir llamada”.		
<b>Prerrequisitos</b> <ul style="list-style-type: none"> <li>Ingresar al sistema</li> </ul>		
<b>Pasos:</b> <ol style="list-style-type: none"> <li>Intentar ingresar al sistema.</li> <li>Seleccionar la opción “Transferir llamada”.</li> </ol>		
<b>Resultado esperado:</b> El sistema debe transferir la llamada del usuario a una de las extensiones registradas en la base de datos.		
<b>Resultado obtenido:</b> <ul style="list-style-type: none"> <li>El usuario selecciona la opción “Transferir llamada”.</li> </ul>  <ul style="list-style-type: none"> <li>El sistema busca la extensión disponible en la base de datos y transfiere la llamada.</li> </ul> 		

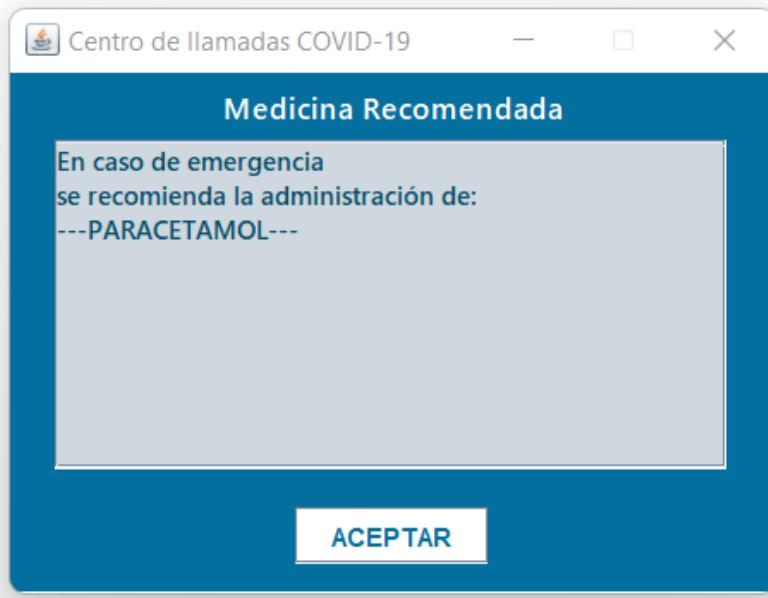
**Tabla 4. Caso de prueba 3**

Generar cita médica	CP03	
	¿Aceptado?	Sí
<b>Descripción:</b> Se probará la creación de una cita médica al usuario basado en sus síntomas.		
<b>Prerrequisitos</b> <ul style="list-style-type: none"> <li>• Ingresar al sistema</li> <li>• El usuario debe ingresar al centro de información de Covid-19</li> <li>• El usuario debe ingresar los síntomas que posee</li> </ul>		
<b>Pasos:</b> <ol style="list-style-type: none"> <li>1. Listar los síntomas que posee</li> <li>2. Enviar la lista de síntomas que posee</li> </ol>		
<b>Resultado esperado:</b> El sistema debe crear una cita médica con los datos brindados por el usuario.		
<b>Resultado obtenido:</b> 		

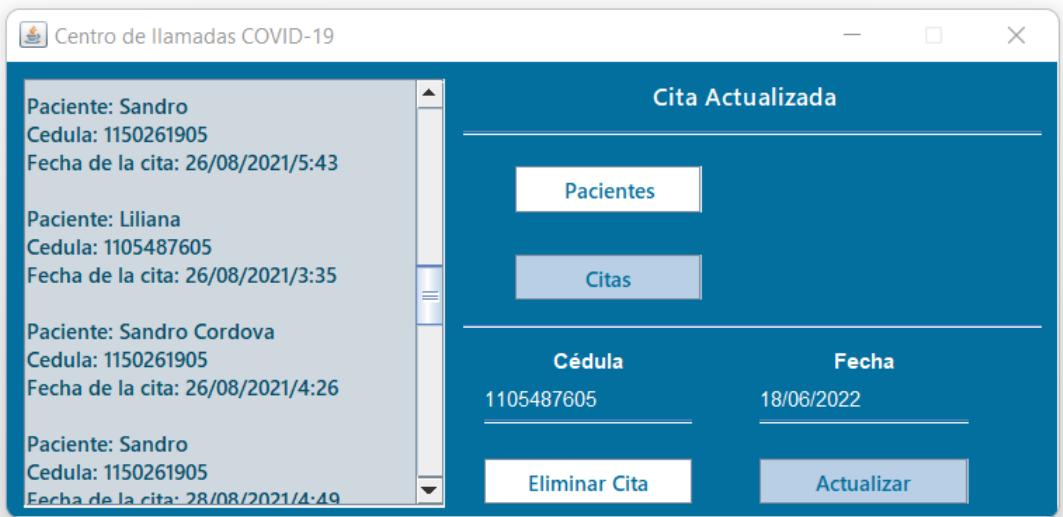
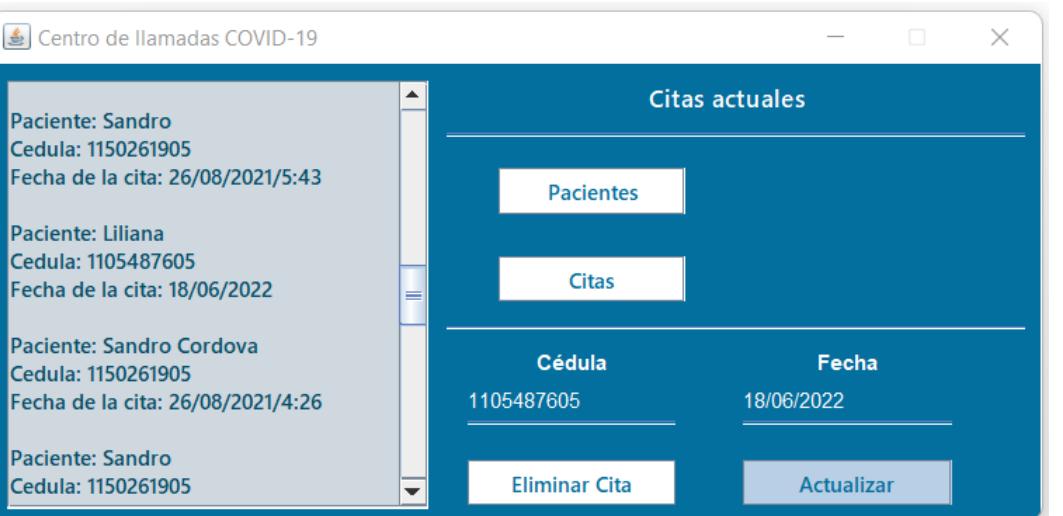
**Tabla 5. Caso de prueba 4**

Generar lista de medidas de bioseguridad	CP04	
	¿Aceptado?	Sí
<b>Descripción:</b> Se probará la presentación de medidas de bioseguridad al usuario basado en sus síntomas.		
<b>Prerrequisitos</b> <ul style="list-style-type: none"> <li>• Ingresar al sistema</li> <li>• El usuario debe ingresar al centro de información de Covid-19</li> <li>• El usuario debe ingresar los síntomas que posee</li> </ul>		
<b>Pasos:</b> <ol style="list-style-type: none"> <li>1. Listar los síntomas que posee</li> <li>2. Enviar la lista de síntomas que posee</li> </ol>		
<b>Resultado esperado:</b> El sistema debe listar las medidas de bioseguridad al usuario.		
<b>Resultado obtenido:</b> 		

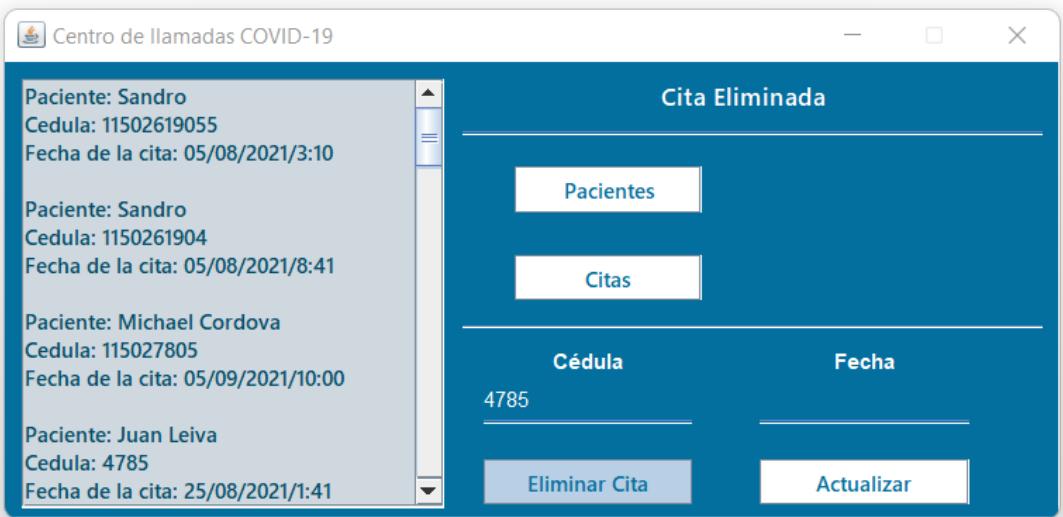
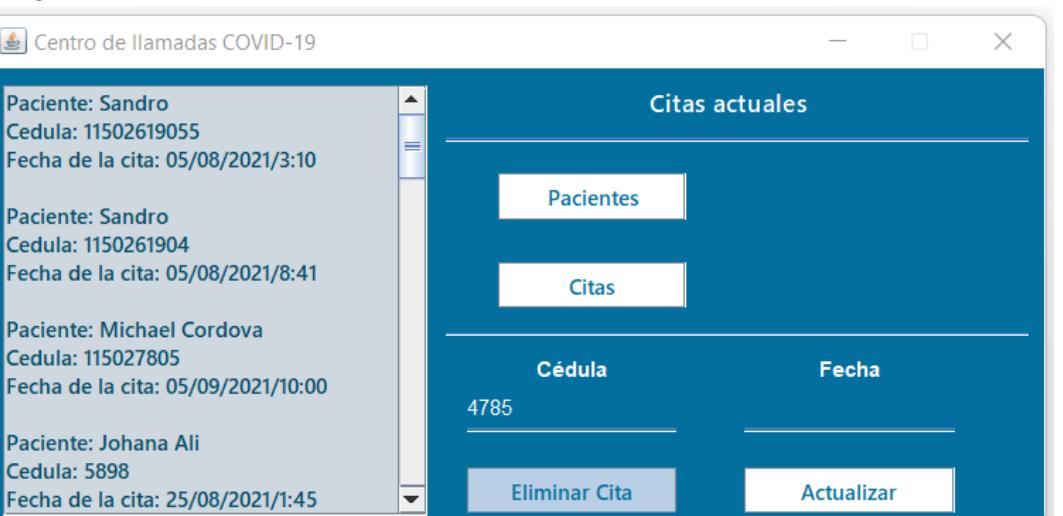
**Tabla 6. Caso de prueba 5**

Recetar medicina en caso de emergencia	CP05	
	¿Aceptado?	Sí
<b>Descripción:</b> Se probará si el sistema presentará los medicamentos que el usuario puede usar en caso de emergencia.		
<b>Prerrequisitos</b> <ul style="list-style-type: none"> <li>• Ingresar al sistema</li> <li>• El usuario debe ingresar al centro de información de Covid-19</li> <li>• El usuario debe ingresar los síntomas que posee</li> </ul>		
<b>Pasos:</b> <ol style="list-style-type: none"> <li>1. Listar los síntomas que posee</li> <li>2. Enviar la lista de síntomas que posee</li> </ol>		
<b>Resultado esperado:</b> El sistema debe presentar al usuario la medicina recomendada en caso de emergencia.		
<b>Resultado obtenido:</b>  <p>The screenshot shows a window titled "Centro de llamadas COVID-19". Inside, a blue box displays the message: "Medicina Recomendada" followed by "En caso de emergencia se recomienda la administración de: ---PARACETAMOL---". At the bottom of this box is a blue button labeled "ACEPTAR".</p>		

**Tabla 7. Caso de prueba 6**

Recetar medicina en caso de emergencia	CP06	
	¿Aceptado?	Sí
<b>Descripción:</b> Se probará si el sistema permite la modificación de los datos de una cita médica.		
<b>Prerrequisitos</b> <ul style="list-style-type: none"> <li>• Ingresar al sistema</li> <li>• Ingresar a la interfaz del administrador del sistema</li> </ul>		
<b>Pasos:</b> <ol style="list-style-type: none"> <li>1. Listar las citas actuales</li> <li>2. Ingresar el número de cédula del usuario cuya cita se quiere modificar</li> <li>3. Ingresar los valores a cambiar de la cita</li> <li>4. Presionar en el botón “Actualizar”</li> </ol>		
<b>Resultado esperado:</b> El sistema debe actualizar los datos de la cita médica del usuario seleccionado.		
<b>Resultado obtenido:</b> <ul style="list-style-type: none"> <li>• Ingreso de los datos del usuario</li> </ul>  <p>The screenshot shows a window titled "Centro de Llamadas COVID-19". On the left, there is a list of patients with their names, IDs, and appointment dates. One appointment is highlighted: "Paciente: Sandro, Cedula: 1150261905, Fecha de la cita: 26/08/2021/5:43". On the right, there is a panel titled "Cita Actualizada" with fields for "Cédula" (1105487605) and "Fecha" (18/06/2022). Below these fields are two buttons: "Eliminar Cita" and "Actualizar".</p> <ul style="list-style-type: none"> <li>• Corroborar la información modificada.</li> </ul>  <p>The screenshot shows a window titled "Centro de Llamadas COVID-19". On the left, the list of patients now includes the updated appointment for patient 1105487605 on 18/06/2022. The right panel is identical to the previous screenshot, showing the "Cita Actualizada" panel with the same data and buttons.</p>		

**Tabla 8. Caso de prueba 7**

Recetar medicina en caso de emergencia	CP07	
	¿Aceptado?	Sí
<b>Descripción:</b> Se probará si el sistema permite la eliminar de los datos de una cita médica.		
<b>Prerrequisitos</b> <ul style="list-style-type: none"> <li>• Ingresar al sistema</li> <li>• Ingresar a la interfaz del administrador del sistema</li> </ul>		
<b>Pasos:</b> <ol style="list-style-type: none"> <li>5. Listar las citas actuales</li> <li>6. Ingresar el número de cédula del usuario cuya cita se quiere eliminar</li> <li>7. Ingresar la cédula del usuario</li> <li>8. Presionar en el botón “Eliminar cita”</li> </ol>		
<b>Resultado esperado:</b> El sistema debe eliminar los datos de la cita médica del usuario seleccionado.		
<b>Resultado obtenido:</b> <ul style="list-style-type: none"> <li>• Ingreso de los datos del usuario</li> </ul> 		
<ul style="list-style-type: none"> <li>• Corroborar la información modificada.</li> </ul> 		

## 7. Anexos

A continuación, se adjunta capturas en las en donde se muestra que la información brindada por el sistema es obtenida de la base de datos. Y de igual manera, se muestra que la información ingresada por el usuario es registrada en la base de datos.

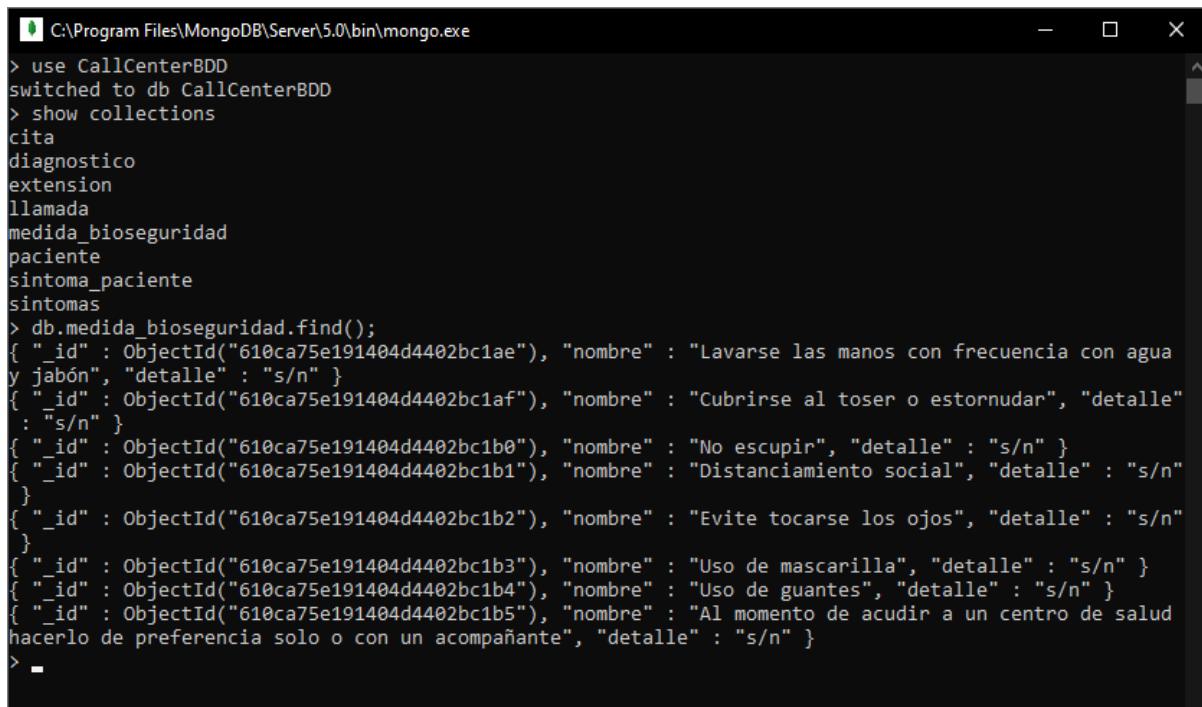
### Extensiones registradas en la base de datos



```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> use CallCenterBDD
switched to db CallCenterBDD
> show collections
cita
diagnostico
extension
llamada
medida_bioseguridad
paciente
sintoma_paciente
sintomas
> db.extension.find();
{ "_id" : ObjectId("610a2c5b191434f9c94634f8"), "numero" : "0725496", "extension" : "23", "detalle" :
"Departamento de información", "estado" : "False" }
{ "_id" : ObjectId("610a2c5b191434f9c94634f9"), "numero" : "0725496", "extension" : "12", "detalle" :
"Atención al cliente", "estado" : "True" }
>
```

Figura 1. Extensiones registradas en la base de datos.

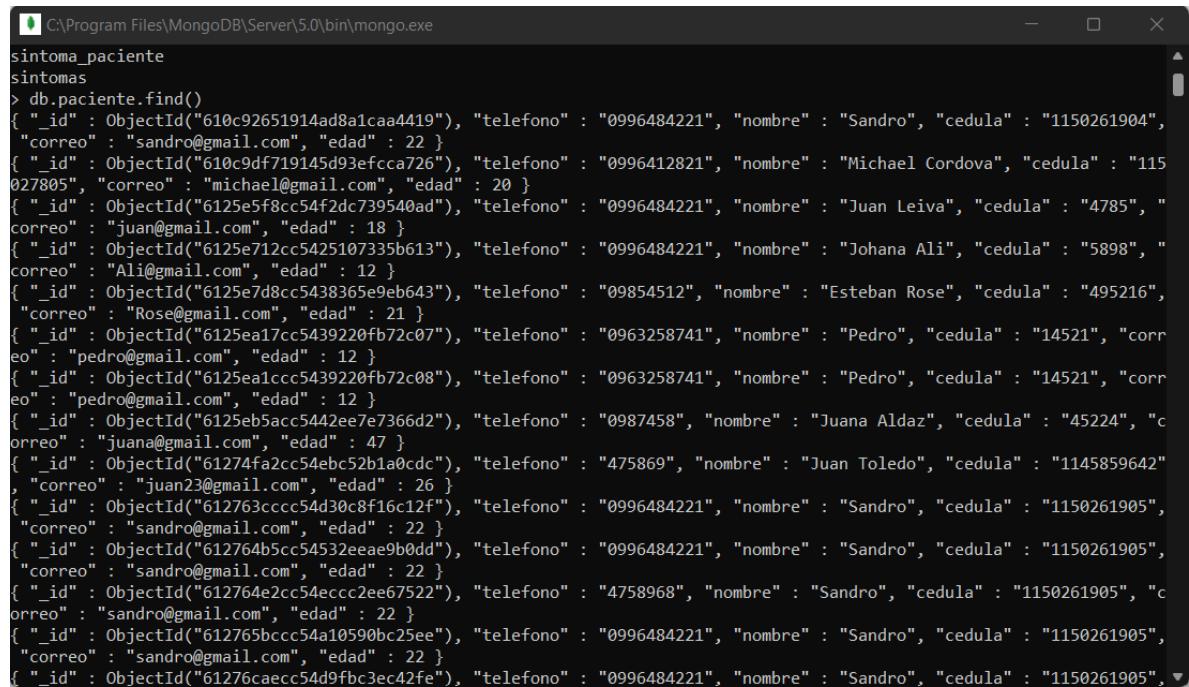
### Medidas de bioseguridad



```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
> use CallCenterBDD
switched to db CallCenterBDD
> show collections
cita
diagnostico
extension
llamada
medida_bioseguridad
paciente
sintoma_paciente
sintomas
> db.medida_bioseguridad.find();
{ "_id" : ObjectId("610ca75e191404d4402bc1ae"), "nombre" : "Lavarse las manos con frecuencia con agua y jabón", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1af"), "nombre" : "Cubrirse al toser o estornudar", "detalle" :
"s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b0"), "nombre" : "No escupir", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b1"), "nombre" : "Distanciamiento social", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b2"), "nombre" : "Evite tocarse los ojos", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b3"), "nombre" : "Uso de mascarilla", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b4"), "nombre" : "Uso de guantes", "detalle" : "s/n" }
{ "_id" : ObjectId("610ca75e191404d4402bc1b5"), "nombre" : "Al momento de acudir a un centro de salud hacerlo de preferencia solo o con un acompañante", "detalle" : "s/n" }
> -
```

Figura 2. Medidas de bioseguridad en la base de datos.

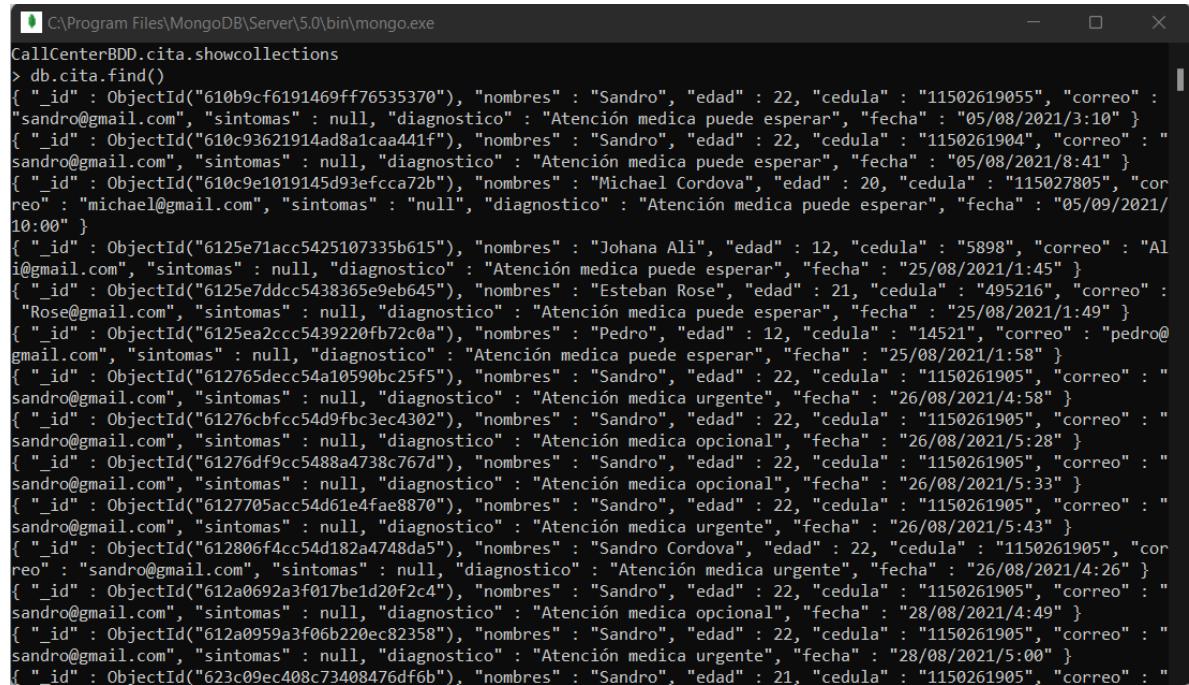
## Pacientes registrados



```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
sintoma_paciente
sintomas
> db.paciente.find()
{ "_id" : ObjectId("610c92651914ad8a1caa4419"), "telefono" : "0996484221", "nombre" : "Sandro", "cedula" : "1150261904", "correo" : "sandro@gmail.com", "edad" : 22 }
{ "_id" : ObjectId("610c9df719145d93efcca726"), "telefono" : "0996412821", "nombre" : "Michael Cordova", "cedula" : "115027805", "correo" : "michael@gmail.com", "edad" : 20 }
{ "_id" : ObjectId("6125e5f8cc54f2dc739540ad"), "telefono" : "0996484221", "nombre" : "Juan Leiva", "cedula" : "4785", "correo" : "juan@gmail.com", "edad" : 18 }
{ "_id" : ObjectId("6125e712cc5425107335b613"), "telefono" : "0996484221", "nombre" : "Johana Ali", "cedula" : "5898", "correo" : "Ali@gmail.com", "edad" : 12 }
{ "_id" : ObjectId("6125e7d8cc5438365e9eb643"), "telefono" : "09854512", "nombre" : "Esteban Rose", "cedula" : "495216", "correo" : "Rose@gmail.com", "edad" : 21 }
{ "_id" : ObjectId("6125ea17cc5439220fb72c07"), "telefono" : "0963258741", "nombre" : "Pedro", "cedula" : "14521", "correo" : "pedro@gmail.com", "edad" : 12 }
{ "_id" : ObjectId("6125ea1ccc5439220fb72c08"), "telefono" : "0963258741", "nombre" : "Pedro", "cedula" : "14521", "correo" : "pedro@gmail.com", "edad" : 12 }
{ "_id" : ObjectId("6125eb5acc5442ee7e7366d2"), "telefono" : "0987458", "nombre" : "Juana Aldaz", "cedula" : "45224", "correo" : "juana@gmail.com", "edad" : 47 }
{ "_id" : ObjectId("61274fa2cc54ebc52b1a0cdc"), "telefono" : "475869", "nombre" : "Juan Toledo", "cedula" : "1145859642", "correo" : "juan23@gmail.com", "edad" : 26 }
{ "_id" : ObjectId("612763cccc54d30c8f16c12f"), "telefono" : "0996484221", "nombre" : "Sandro", "cedula" : "1150261905", "correo" : "sandro@gmail.com", "edad" : 22 }
{ "_id" : ObjectId("612764b5cc54532eeae9b0dd"), "telefono" : "0996484221", "nombre" : "Sandro", "cedula" : "1150261905", "correo" : "sandro@gmail.com", "edad" : 22 }
{ "_id" : ObjectId("612764e2cc54ecc2ee67522"), "telefono" : "4758968", "nombre" : "Sandro", "cedula" : "1150261905", "correo" : "sandro@gmail.com", "edad" : 22 }
{ "_id" : ObjectId("612765bcc54a10590bc25f5"), "telefono" : "0996484221", "nombre" : "Sandro", "cedula" : "1150261905", "correo" : "sandro@gmail.com", "edad" : 22 }
{ "_id" : ObjectId("61276caecc54d9fb3ec42fe"), "telefono" : "0996484221", "nombre" : "Sandro", "cedula" : "1150261905", "correo" : "sandro@gmail.com", "edad" : 22 }
```

Figura 3. Pacientes registrados en la base de datos.

## Citas registradas



```
C:\Program Files\MongoDB\Server\5.0\bin\mongo.exe
CallCenterBDD.cita.showcollections
> db.cita.find()
{ "_id" : ObjectId("610b9cf6191469ff76535370"), "nombres" : "Sandro", "edad" : 22, "cedula" : "11502619055", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica puede esperar", "fecha" : "05/08/2021/3:10" }
{ "_id" : ObjectId("610c93621914ad8a1caa441f"), "nombres" : "Sandro", "edad" : 22, "cedula" : "1150261904", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica puede esperar", "fecha" : "05/08/2021/8:41" }
{ "_id" : ObjectId("610c9e1019145d93efcca72b"), "nombres" : "Michael Cordova", "edad" : 20, "cedula" : "115027805", "correo" : "michael@gmail.com", "sintomas" : "null", "diagnostico" : "Atención medica puede esperar", "fecha" : "05/09/2021/10:00" }
{ "_id" : ObjectId("6125e71acc5425107335b615"), "nombres" : "Johana Ali", "edad" : 12, "cedula" : "5898", "correo" : "Ali@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica puede esperar", "fecha" : "25/08/2021/1:45" }
{ "_id" : ObjectId("6125e7ddcc5438365e9eb645"), "nombres" : "Esteban Rose", "edad" : 21, "cedula" : "495216", "correo" : "Rose@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica puede esperar", "fecha" : "25/08/2021/1:49" }
{ "_id" : ObjectId("6125ea2ccc5439220fb72c0a"), "nombres" : "Pedro", "edad" : 12, "cedula" : "14521", "correo" : "pedro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica puede esperar", "fecha" : "25/08/2021/1:58" }
{ "_id" : ObjectId("612765decc54a10590bc25f5"), "nombres" : "Sandro", "edad" : 22, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica urgente", "fecha" : "26/08/2021/4:58" }
{ "_id" : ObjectId("61276cbfcc54d9fb3ec4302"), "nombres" : "Sandro", "edad" : 22, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica opcional", "fecha" : "26/08/2021/5:28" }
{ "_id" : ObjectId("61276df9cc5488a4738c767d"), "nombres" : "Sandro", "edad" : 22, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica opcional", "fecha" : "26/08/2021/5:33" }
{ "_id" : ObjectId("6127705acc54d61e4fae8870"), "nombres" : "Sandro", "edad" : 22, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica urgente", "fecha" : "26/08/2021/5:43" }
{ "_id" : ObjectId("612806f4cc54d182a4748da5"), "nombres" : "Sandro Cordova", "edad" : 22, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica urgente", "fecha" : "26/08/2021/4:26" }
{ "_id" : ObjectId("612a0692a3f017be1d20f2c4"), "nombres" : "Sandro", "edad" : 22, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica opcional", "fecha" : "28/08/2021/4:49" }
{ "_id" : ObjectId("612a0959a3f06b220ec82358"), "nombres" : "Sandro", "edad" : 22, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica urgente", "fecha" : "28/08/2021/5:00" }
{ "_id" : ObjectId("623c09ec408c73408476df0b"), "nombres" : "Sandro", "edad" : 21, "cedula" : "1150261905", "correo" : "sandro@gmail.com", "sintomas" : null, "diagnostico" : "Atención medica urgente", "fecha" : "28/08/2021/5:00" }
```

Figura 4. Citas registradas en la base de datos.

**Anexo 7.** Manual de usuario

**Manual de usuario**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

## **1. Uso de la aplicación**

Este documento tiene como objetivo el facilitar la comprensión y uso de la aplicación tanto a los usuarios finales como a los administradores del sistema dentro del centro médico, el documento contiene los pasos a seguir para cada una de las acciones que se pueden efectuar dentro de la aplicación, junto con imágenes de la interfaz visual del programa para facilitar la comprensión.

## **2. Sobre la aplicación**

La aplicación detallada en el presente documento se encarga de gestionar a los pacientes que se requieran información sobre Covid-19, la aplicación solicita datos personales de los pacientes, así como lo síntomas que este posea para posteriormente clasificarlos en un grado de prioridad, internamente se toman decisiones según la prioridad del paciente y se procede a efectuar determinadas acciones como la generación de una cita para que el paciente pueda recibir atención médica, una lista de medicamentos que el paciente puede hacer uso como en caso de emergencia y una lista de medidas de bioseguridad que el paciente debe seguir como medida de prevención de contagios.

## **3. Requisitos del sistema**

### **3.1. Hardware**

- Computador con mínimo 4GB de RAM.
- Procesador Core i3, similar o superior.
- Altavoz conectado al equipo.
- Micrófono conectado al equipo.

### **3.2. Software**

- Java Runtime Environment (JRE).
- Mongo DB.
- Windows 7 o superior.

#### 4. Manejo de la aplicación por usuario final

##### 4.1. Opciones principales

**Para acceder al sistema de información de Covid-19:** El usuario que desee acceder al sistema de información, una vez dentro del menú principal de la aplicación, debe seleccionar la opción dos, siguiendo la siguiente secuencia de pasos:

1. Ingreso al menú principal.
2. Identificar el comando que permita el ingreso al sistema de información.
3. Ingreso del comando para acceder al sistema de información (opción dos).
4. Seleccionar la opción “Enviar”.

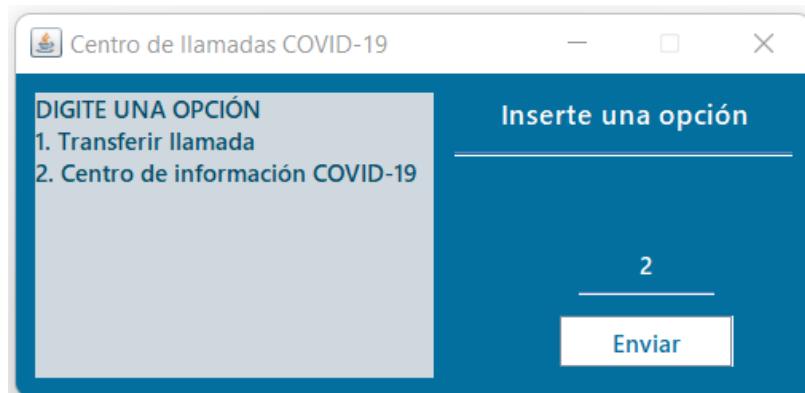


Figura 5. Opción “centro de información Covid-19”.

**Para no acceder al sistema de información de Covid-19:** El usuario que no desee acceder al sistema de información, una vez dentro del menú principal de la aplicación, debe seleccionar cualquiera la opción número uno que será la que enlace la llamada al departamento de atención al cliente. Secuencia normal a seguir:

1. Ingreso al menú principal.
2. Identificar el comando que permita transferir la llamada.
3. Ingreso del comando para transferir la llamada (opción uno).
4. Seleccionar la opción “Enviar”.

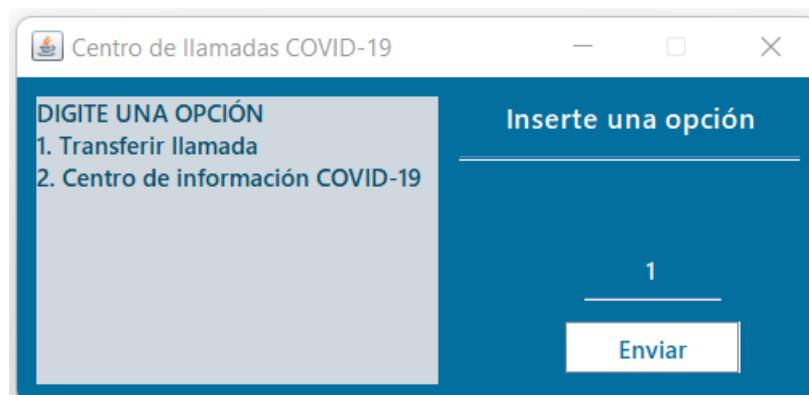


Figura 6. Opción “transferir llamada”.

#### 4.2. Ingreso de datos personales

**Al ingresar datos personales del paciente:** En este punto se considera como paciente a la persona que le pertenezcan los datos ingresados, se debe considerar que los datos personales del paciente serán usados para generar una cita médica en caso de que sea necesario y a su vez, dicha información será guardada en la base de datos. Para poder ingresar los datos se debe seguir la siguiente secuencia:

1. Ingresar el nombre y apellido del paciente.
2. Ingresar el número de cédula del paciente.
3. Ingresar el correo electrónico del paciente.
4. Ingresar la edad del paciente.
5. Seleccionar la opción “Enviar”.

The screenshot shows a software application window titled "Centro de llamadas COVID-19". The main title bar is white with the application name. Below it, a dark blue header bar contains the text "DATOS PERSONALES" in white. The form itself has a light blue background. It includes four text input fields: "Nombre: Andres", "Cédula: 1150261905", "Correo: sandro@gmail.com", and "Edad: 22". To the right of the "Cédula" field is a small checkbox labeled "Pasaporte" which is unchecked. At the bottom center of the form is a blue rectangular button labeled "ENVIAR" in white capital letters.

Figura 7. Ingreso de datos personales.

#### 4.3. Ingreso de síntomas

Al ingresar los síntomas del paciente: En este punto el sistema debe clasificar el estado del paciente según el estado del paciente, por lo tanto, se requiere que el paciente ingrese los síntomas que padezca, siguiendo la siguiente secuencia:

1. Escribir el síntoma.
2. Dar un salto de línea.
3. Repetir los pasos 1 y 2 las veces que sean necesarias.
4. Seleccionar la opción “Enviar”.

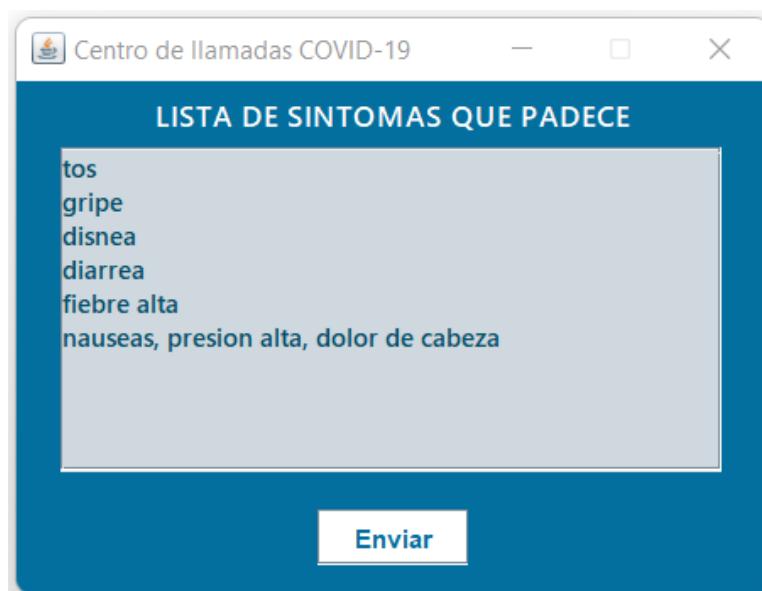


Figura 8. Listar síntomas del paciente.

**Nota:** El número de síntomas ingresados es indefinido y el salto de línea puede o no darse. Si el número de síntomas es menor o igual a 1, la aplicación clasificará al paciente en una de las prioridades más bajas a falta de información.

#### 4.4. Recepción de la información

**Recepción de cita médica:** Si la gravedad de los síntomas del paciente lo ameritan, el sistema va a generar una cita médica, en donde se registran los datos del paciente y se asignará una fecha para la cita.



Gráfico 9. Vista "Cita generada"

**Nota:** Es necesario esperar el resto de información brindada por el sistema.

**Recepción de la medicina recomendada:** Si la gravedad de los síntomas del paciente lo ameritan, el sistema recomendará la administración de los medicamentos que el centro médico maneje acorde a los síntomas del paciente.

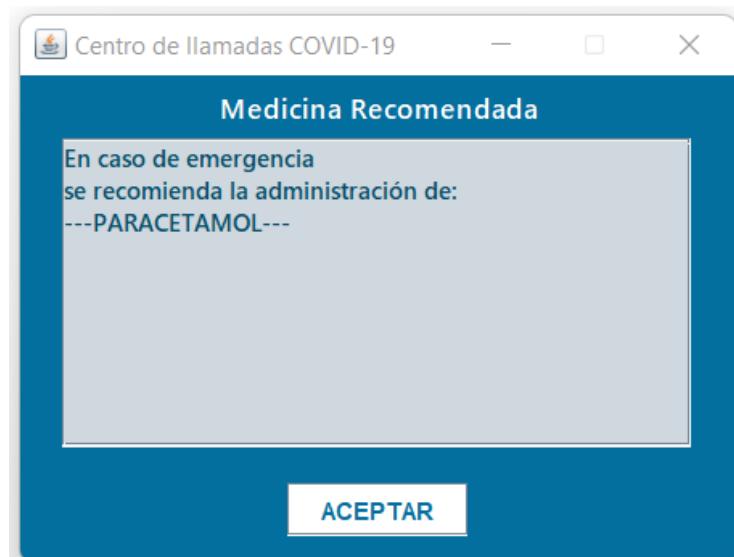


Figura 10. Vista "Medicina recomendada"

**Nota:** Es necesario esperar el resto de información brindada por el sistema.

**Recepción de las medidas de bioseguridad:** Si la gravedad de los síntomas del paciente lo ameritan, el sistema recomendará que se acaten determinadas medidas de bioseguridad.

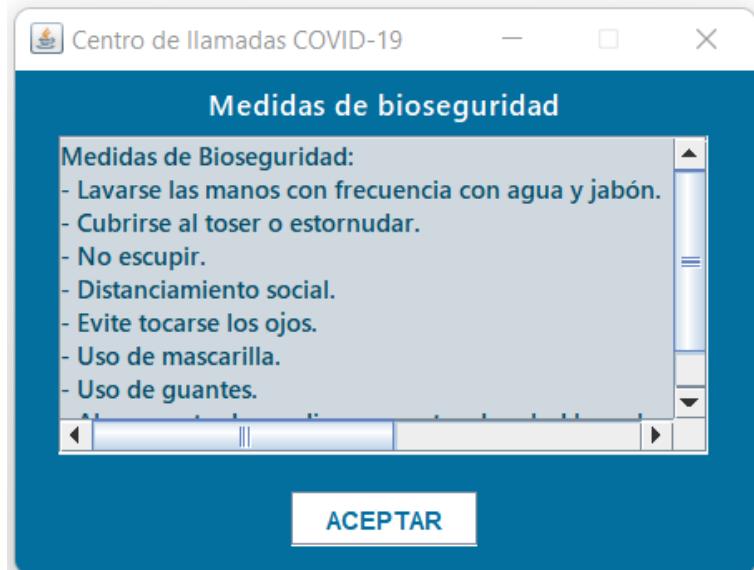


Figura 11. Vista “Medidas de bioseguridad”.

## 5. Manejo de la aplicación por el administrador

### 5.1. Generar informes

**Para generar informes de pacientes:** El administrador del sistema puede obtener una lista de todos los pacientes que se encuentren registrados en la base de datos del centro médico. Para poder tener acceso a la lista de pacientes, el administrador debe presionar el botón “Pacientes” mostrado en la figura X.

**Para generar informes de las citas:** El administrador del sistema puede obtener una lista de todas las citas que se encuentran registradas en la base de datos del centro médico. Para poder tener acceso a la lista de las citas, el administrador debe hacer clic en el botón “Citas” mostrado en la Figura 8.

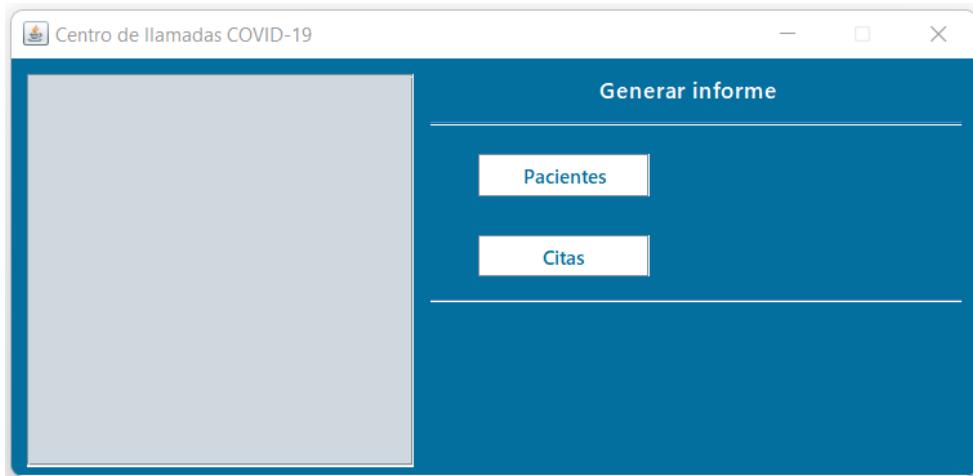


Figura 12. Vista “Generar informes”.

## 5.2. Manipulación de la información

Para eliminar un paciente, el administrador puede eliminar la información de un paciente, en caso de que sea necesario, para ello se deben seguir los siguientes pasos:

1. Identificar el paciente cuya información se desea eliminar.
2. Ingresar el número de cédula del paciente a eliminar.
3. Presionar el botón “Eliminar Paciente”.
4. Actualizar la lista de pacientes actuales.
5. Constatar la eliminación del paciente.

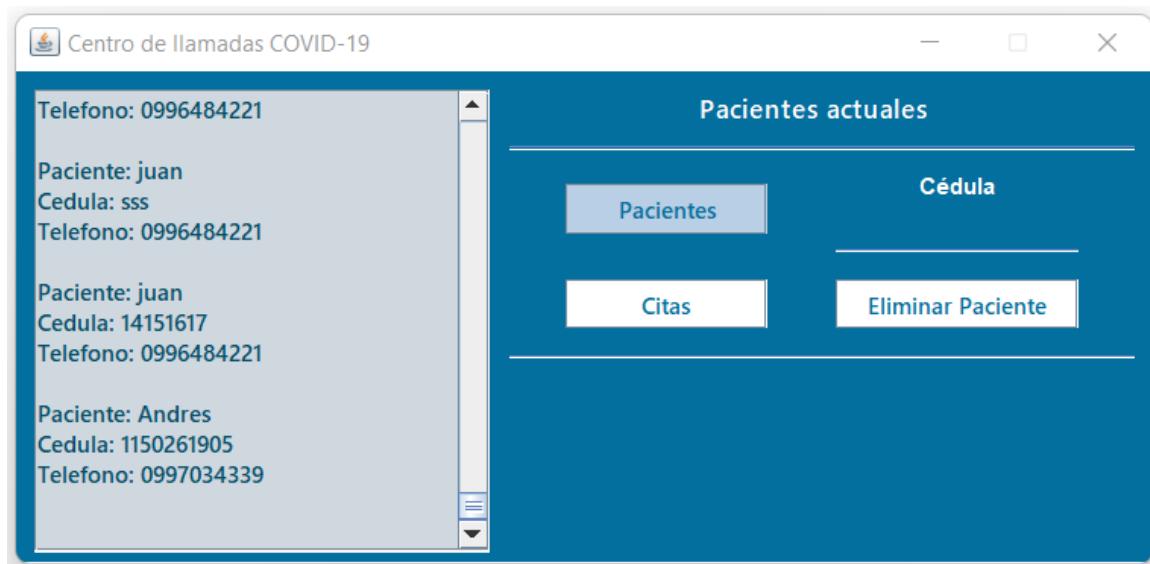


Figura 13. Vista “Eliminar paciente”.

Para eliminar una cita, el administrador puede eliminar la información de la cita de un paciente, en caso de que sea necesario, para ello se deben seguir los siguientes pasos:

1. Identificar el paciente cuya cita se desea eliminar.
2. Ingresar el número de cédula del paciente.
3. Presionar el botón “Eliminar Cita”.
4. Actualizar la lista de citas actuales.
5. Constatar la eliminación de la cita del paciente.

**Para actualizar la fecha de una cita**, el administrador puede modificar la fecha en la que se llevará a cabo la cita de un paciente, en caso de que sea necesario se deben seguir los siguientes pasos:

1. Identificar el paciente cuya cita se desea modificar.
2. Ingresar el número de cédula del paciente.
3. Ingresar la nueva fecha de la cita.
4. Presionar el botón “Actualizar Cita”.
5. Actualizar la lista de citas actuales.
6. Constatar la fecha modificada de la cita del paciente.

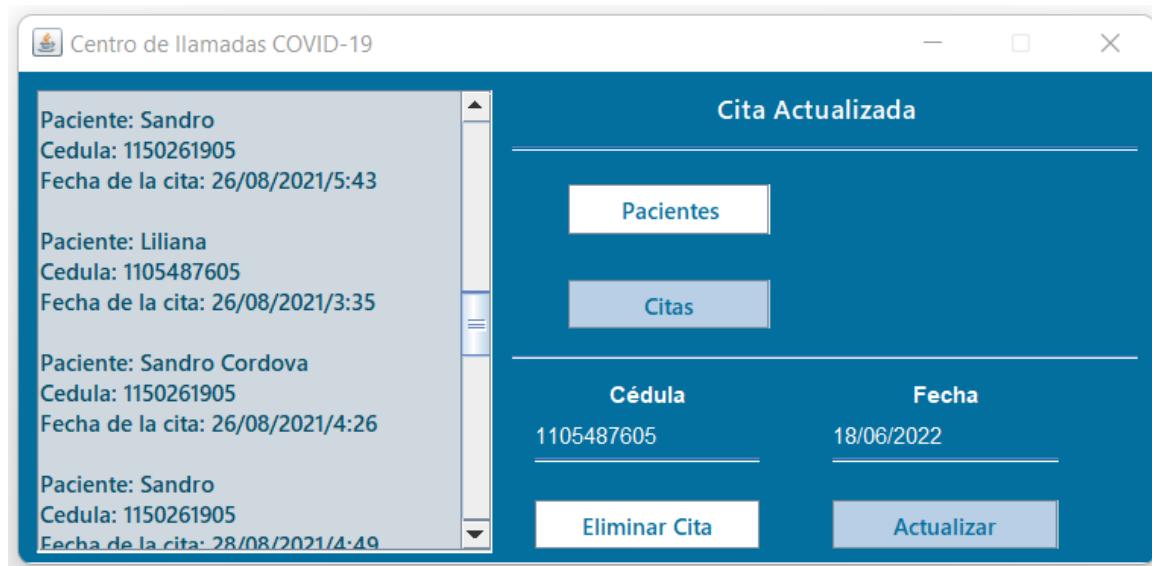


Figura 14. Vista “gestión de citas”.

**Anexo 8. Manual de despliegue para desarrolladores**

**Manual de despliegue para desarrolladores**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

## **1. Introducción**

Este documento presenta a los desarrolladores, la información necesaria para poder hacer un levantamiento de la aplicación desde un punto de vista técnico, aquí se presentarán todas las dependencias del sistema y cómo estas deben encontrarse dentro del mismo para que el programa pueda ejecutarse correctamente. Este es un resumen de todas las actividades desarrolladas, por lo tanto, se omitirá información de conocimiento general como repositorios oficiales para poder obtener los diferentes componentes, sin embargo, todos los archivos mencionados en el presente documento se encuentran en el repositorio: <https://github.com/sandrocordova/AgenteInteligenteJava.git>.

## **2. Objetivo del documento**

El presente documento tiene como objetivo el facilitar la instalación y levantamiento de la aplicación desarrollada, así como la especificación de cada una de sus dependencias para que esta pueda funcionar de forma correcta.

## **3. Documentos relacionados**

**Tabla 1. Documentos relacionados**

Título del Documento	Referencia
Arquitectura de Software	Anexo 3. Arquitectura de software
Desarrollo y documentación	Anexo 4. Desarrollo y documentación

### **Resumen**

Este documento consta de cuatro secciones. En la primera sección se realiza una introducción al mismo y se proporciona una visión general del documento de Arquitectura de Software () .

En la segunda sección del documento se realiza una descripción general de la Arquitectura, con el fin de conocer el modelo 4+1 y los objetivos.

En la tercera sección del documento se definen las vistas y entregables de acuerdo al modelo 4+1, que son: vista de escenarios (diagrama de Casos de Uso), vista lógica (Diagrama de Clases), vista de despliegue (Diagrama de Componentes), vista de procesos (Diagrama de Actividad) y vista física (Diagrama de Despliegue).

Por último, la cuarta sección del documento es aquella en la que se define la Arquitectura de la aplicación. Para obtener mayor información de los diagramas e información utilizada en esta sección ver

**Anexo 3. Arquitectura de software.**

## **4. Requerimientos y características de la implementación**

### **3.1. Hardware**

- Computador con mínimo 4GB de RAM.
- Procesador Core i3, similar o superior.
- Altavoz conectado al equipo.
- Micrófono conectado al equipo.

### **3.2. Software**

- IDE Java.
- JDK.
- Mongo DB.
- Librería TalkingJavaSDK-170
- Librería JADE.
- Librería para conexión de MongoDB con java.

## **4. Documentación necesaria para efectuar el despliegue de la aplicación**

Para facilitar la comprensión de cómo se llevará el despliegue de la aplicación, se hará uso del diagrama general de la aplicación, así como el diagrama de despliegue presentados en el documento de Arquitectura de software (ver

### Anexo 3. Arquitectura de software).

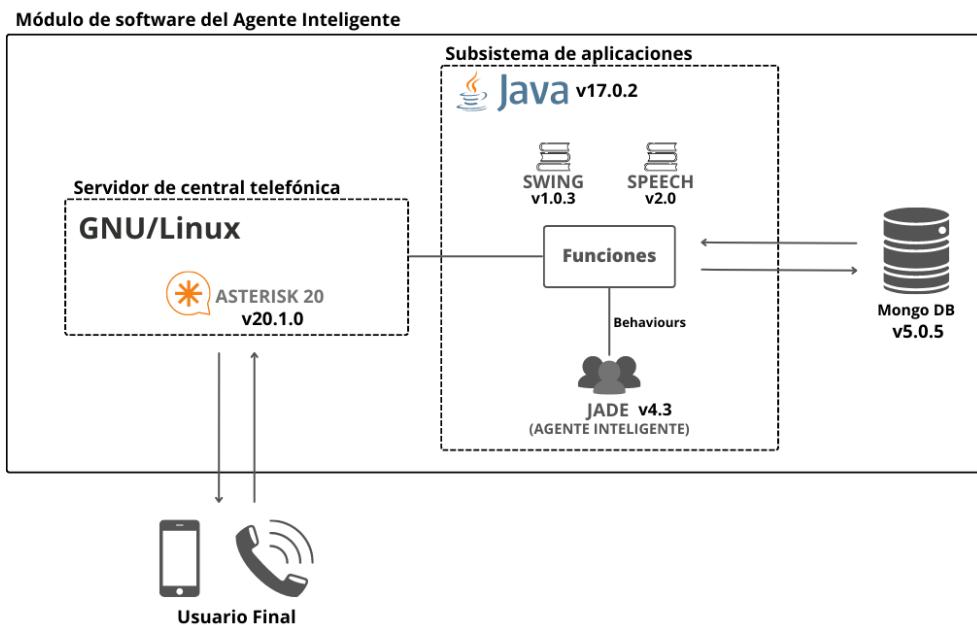


Figura 1. Diagrama General

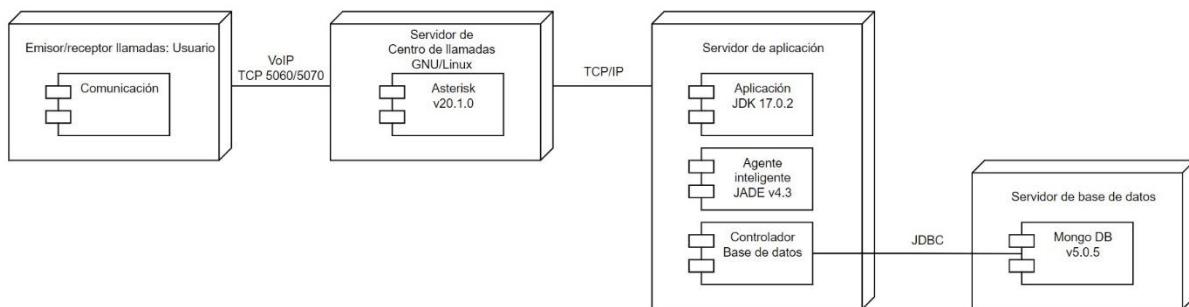


Figura 2. Diagrama de despliegue.

## 5. Instalación y configuración del software Base

### 5.1. Instalación del proyecto principal

El proyecto debe ser descargado y abierto desde su carpeta raíz la cual tiene el nombre de “agenteInteligente”, el proyecto puede ser abierto mediante cualquier IDE de Java, que previamente tenga instalado el JDK.

Una vez cargado el proyecto, se podrá observar las carpetas que contienen las diferentes clases de la aplicación. Entre estas carpetas, la importante es la denominada “principal” ya dicha carpeta contiene la clase “Main.java”, esta es la clase ejecutable que se encargará de iniciar la aplicación.

### 5.2. Instalación de Mongo DB

La instalación de Mongo DB puede darse desde su repositorio oficial, el proyecto no especifica una versión de Mongo DB, siempre y cuando su librería para la conexión con el proyecto principal sea compatible.

Una vez instalado, simplemente debe asegurarse de que dicha acción fue desarrollada exitosamente, ya que el proyecto, se encarga de crear automáticamente la base de datos y las diferentes colecciones utilizadas.

## 6. Elementos y librerías a implantar

**Tabla 2. Elementos y librerías**

Elemento/librería	Descripción	Ruta
TalkingJavaSDK-170	Librería para el módulo voz.	Su instalación se detalla en la siguiente sección.
JADE	Librería de java para la gestión de agentes inteligentes.	Se carga directamente al proyecto.
MongoDB-connection	Librería para la conexión y gestión de Mongo DB.	Se carga directamente al proyecto.

## **7. Detalle de elementos y librerías a implantar**

### **7.1. TalkingJavaSDK-170**

Esta librería consta de cuatro archivos principales, los mismos que deben instalarse de la siguiente manera:

5. El archivo “cgjsapi.jar” debe ser instalando como cualquier librería de java en el proyecto principal.
6. De la misma forma el archivo “packet.jar” debe ser instalado como una librería dentro del proyecto principal.
7. Si el sistema utilizado no especifica una arquitectura de 64 bits, se debe mover el archivo “cgjsapi170.dll” a la carpeta bin del directorio raíz del JDK que usualmente tienen el siguiente directorio “C:\Program Files\Java\jdk-17.0.2\bin”.
8. Si el sistema utilizado cumple con una arquitectura de 64 bits se debe mover el archivo “cgjsapi170\_x64” a la carpeta bin del directorio raíz del JDK que usualmente tienen el siguiente directorio “C:\Program Files\Java\jdk-17.0.2\bin”.

### **7.2. JADE y MongoDB-connection**

Estas dos librerías cuentan con la instalación global de cualquier librería utilizadas por java. Se descarga el archivo .jar desde cualquier repositorio en internet y se procede a añadir como librería del proyecto.

## **8. Procedimiento para carga de datos iniciales**

### **8.1. Síntomas preestablecidos en la base de datos**

La aplicación necesita de determinados síntomas para poder realizar la comparativa y su posterior clasificación de pacientes según sus síntomas. Para ello se deben agregar registros en la base de datos. Esta lista de síntomas fue obtenida de las encuestas a expertos en medicina (ver

**Anexo 9. Informe de encuesta de síntomas).**

Todos estos síntomas se encuentran dentro de la clase ejecutable “DatosIniciales.java” en la carpeta “agenteInteligente”. Por lo tanto, previo a iniciar la clase principal del proyecto se debe ejecutar por separado la clase ejecutable “DatosIniciales.java” y corroborar que la base de datos tenga documentos en su colección “síntomas”.

## **9. Anexo III: síntomas preestablecidos**

### **Síntomas de GRADO 1**

**Tabla 3. Síntomas de GRADO 1**

<b>Código</b>	<b>Síntoma</b>
01	Fiebre leve
02	Tos
03	Dolor de cabeza leve
04	Síntomas de gripe
05	Rinorrea
06	Diarrea
07	Dolor de espalda
08	Dolor muscular

### **Síntomas de GRADO 2**

**Tabla 4. Síntomas de GRADO 2**

<b>Código</b>	<b>Síntoma</b>
01	Dificultad para respirar
02	Perdida del olfato
03	Perdida del gusto
04	Dolor de garganta
05	Fiebre alta
06	Dolor de cuerpo
07	Mialgias
08	Anosmia
09	Tos persistente

### **Síntomas de GRADO 3**

**Tabla 5. Síntomas de GRADO 3**

<b>Código</b>	<b>Síntoma</b>
01	Náuseas
02	Vómito
03	Dolores musculares
04	Disnea
05	Cianosis
06	Hipotensión
07	Confusión
08	Dolor al respirar
09	Fiebre alta
10	Baja saturación
11	Tos productiva
12	Pérdida del gusto
13	Pérdida del olfato

**Anexo 9.** Informe de encuesta de síntomas

**Informe de encuesta de síntomas de Covid-19 a profesionales en medicina**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

## **Síntomas de pacientes con Covid-19**

El presente documento, expone el proceso seguido para poder determinar los síntomas que serán tomados como referencia para la toma decisiones dentro del software realizado. A continuación, se describe la obtención de la lista de síntomas por medio de la técnica de la encuesta, el análisis y validación de los síntomas finales.

### **1. Encuesta**

La encuesta se realizó con la ayuda de la herramienta web “Google forms”, su selección se dio gracias a que esta facilita el acceso a los encuestados, consta dentro del paquete gratuito para editores de documentos de Google, por lo que al momento de aplicarla no se debe cancelar ningún valor, otra característica importante es que facilita la conexión con hojas de cálculo, facilitando así el análisis de la información obtenida. La encuesta fue aplicada a la/las personas que tienen relación directa o tienen experiencia dentro del campo médico. A continuación, se listan los nombres de las personas que respondieron a la encuesta:

- Danny Renato Mora Huiracocha, Estudiante de medicina (PE01).
- Verónica Estefanía Gutiérres Luzon, estudiante de enfermería (PE02).
- Yakeline Cartuche, Médico (PE03).
- Hugo Ramiro Córdova Córdova, Enfermero (PE04).

### **2. Objetivo**

El objetivo de la encuesta es obtener los síntomas que puede poseer un paciente con Covid-19, estos datos serán utilizados como componente para el desarrollo del módulo encargado de clasificar a los pacientes según los síntomas que este posea.

### **3. Términos**

**PE:** Persona Encuestada

**RP:** Respuesta Pregunta

#### **4. Indicaciones generales**

Si un paciente con síntomas de Covid-19 se pudiera clasificar en 3 grados de prioridad.

Siendo:

GRADO 1: Baja probabilidad de COVID-19.

Amerita medidas de bioseguridad como medida de prevención.

GRADO 2: Probabilidad MEDIA de COVID-19.

Amerita medidas de bioseguridad y atención médica opcional.

GRADO 3: Alta probabilidad de COVID-19.

Amerita atención médica inmediata y medicamentos de emergencia.

¿Cuáles serían los síntomas del paciente para cada uno de los diferentes grados?

Procurar usar el menor número de palabras posibles.

#### **5. Encuesta**

**1) (RP01) ¿Cuáles serían los síntomas del paciente para ser clasificado como GRADO 1?**

**PE01:**

- Fiebre.
- Tos.
- Cefalea.

**PE02:**

- Dolor de cabeza leve.
- Febrícula.
- Síntomas de gripe.

**PE03:**

- Cefalea.
- Rинorrea.
- Diarrea.

**PE04:**

- Dolor de espalda.
- Dolor muscular.
- Baja saturación.

**2) (RP02) ¿Cuáles serían los síntomas del paciente para ser clasificado como GRADO 2?**

**PE01:**

- Dificultad para respirar.
- Perdida de olfato y gusto.
- Dolor de garganta.

**PE02:**

- Baja saturación de oxígeno.
- Fiebre.
- Dolor de cuerpo.

**PE03:**

- Fiebre persistente.
- Mialgias.
- Anosmia.

**PE04:**

- Alta temperatura.
- Tos.
- Tos persistente.

**3) (RP03) ¿Cuáles serían los síntomas del paciente para ser clasificado como GRADO 3?**

**PE01:**

- Náuseas.
- Vómito.
- Dolores musculares.
- Disnea.
- Cianosis.
- Hipotensión.
- Confusión.

**PE02:**

- Dolor al respirar.
- Fiebre alta.
- Saturación de oxígeno baja.

**PE03:**

- Tos productiva.
- Baja saturación.
- Disnea de leves esfuerzos.

**PE04:**

- Baja saturación.
- Perdida del gusto.
- Pérdida del olfato.

## 6. Lista preliminar de síntomas

### Síntomas de grado 1

Tabla 1. Síntomas preliminares de grado 1.

Código	Síntoma	Referencia
01	Fiebre	PE01
02	Tos	PE01
03	Cefalea	PE01
04	Dolor de cabeza leve	PE02
05	Fiebricula	PE02
06	Síntomas de gripe	PE02
07	Cefalea	PE03
08	Rinorrea	PE03
09	Diarrea	PE03
10	Dolor de espalda	PE04
11	Dolor muscular	PE04
12	Baja saturación	PE04

### Síntomas de grado 2

Tabla 2. Síntomas preliminares de grado 2.

Código	Síntoma	Referencia
01	Dificultad para respirar	PE01
02	Perdida del olfato	PE01
03	Perdida del gusto	PE01
04	Dolor de garganta	PE01
05	Baja saturación de oxígeno	PE02
06	Fiebre	PE02
07	Dolor de cuerpo	PE02
08	Fiebre persistente	PE03
09	Mialgias	PE03
10	Anosmia	PE03
11	Fiebre alta	PE04
12	Tos	PE04
13	Tos persistente	PE04

### Síntomas de grado 3

**Tabla 3. Síntomas preliminares de grado 3**

Código	Síntoma	Referencia
01	Náuseas	PE01
02	Vómito	PE01
03	Dolores musculares	PE01
04	Disnea	PE01
05	Cianosis	PE01
06	Hipotensión	PE01
07	Confusión	PE01
08	Dolor al respirar	PE02
09	Fiebre alta	PE02
10	Saturación de oxígeno baja	PE02
11	Tos productiva	PE03
12	Baja saturación	PE03
13	Disnea de leves esfuerzos	PE03
14	Baja saturación	PE04
15	Pérdida del gusto	PE04
16	Pérdida del olfato	PE04

### 7. Matriz de iteración

Por medio de la matriz de iteración se evalúa los síntomas para identificar conflictos o solapamientos. La matriz presenta dos entradas, donde cada entrada contiene todos los requisitos, obteniendo que estos relacionen entre sí.

- Solapamiento: Si entre  $r_1$  y  $r_2$  tratan los mismos aspectos del sistema, dando redundancia. Se lo representa con la letra S.
- Conflicto: Si entre  $r_1$  y  $r_2$  son contradictorios, dando problemas de consistencia interna. Se lo representa con la letra C.

En la tabla X se presenta el solapamiento o conflicto que puede existir entre los requisitos anteriormente obtenidos.

### Síntomas de grado 1

**Tabla 4. Matriz de iteración síntomas de grado 1**

S	01	02	03	04	05	06	07	08	09	10	11	12
01					S							
02												
03				S			S					
04			S				S					
05	S											
06												
07			S	S								
08												
09												
10												
11												
12												

## Síntomas de grado 2

Tabla 5. Matriz de iteración síntomas de grado 2.

S	01	02	03	04	05	06	07	08	09	10	11	12	13
01													
02													
03													
04													
05													
06								S			S		
07													
08						S					S		
09													
10													
11						S		S					
12													S
13													S

## Síntomas de grado 3

Tabla 6. Matriz de iteración síntomas de grado 3

S	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16
01																
02																
03																
04													S			
05																
06																
07																
08																
09																
10														S		
11																
12																
13			S													
14										S						
15																
16																

La descripción de los conflictos encontrados en la matriz realizada se presenta en las Tablas VI, V y IV, con ello se puede hacer un análisis y brindar una solución a los síntomas en conflicto o que se solapan por medio de la creación de un nuevo síntoma.

## 8. Solapamiento y conflictos

### Síntomas de grado 1

Tabla 7. Solapamiento y conflictos de síntomas de grado 1

Nº	Síntomas	Error	Descripción	Síntoma final
1	01, 05	S	El síntoma 01 es el mismo que el síntoma 05.	Fiebre
2	03, 04, 07	C	El síntoma 02 es el mismo que el síntoma 04 y que el síntoma 07.	Dolor de cabeza leve

### Síntomas de grado 2

Tabla 8. Solapamiento y conflictos de síntomas de grado 2

Nº	Síntomas	Error	Descripción	Síntoma final
1	06, 08, 11	S	El síntoma 06 es el mismo que el síntoma 08 y que el síntoma 11.	Fiebre alta
2	12, 13	S	El síntoma 12 es el mismo que el síntoma 13.	Tos persistente

### Síntomas de grado 3

Tabla 9. Solapamiento y conflictos de síntomas de grado 3

Nº	Síntomas	Error	Descripción	Síntoma final
1	04, 13	S	El síntoma 04 es el mismo que el síntoma 13.	Disnea
2	10, 14	S	El síntoma 10 es el mismo que el síntoma 14.	Baja saturación

## 9. Lista final de síntomas

Una vez solucionado los conflictos encontrados, en la Tabla 4 se presenta los síntomas finales que serán aplicados en el desarrollo de la aplicación.

### Síntomas de grado 1

Tabla 10. Síntomas de grado 1

Código	Síntoma
01	Fiebre leve
02	Tos
03	Dolor de cabeza leve
04	Síntomas de gripe
05	Rinorrea
06	Diarrea
07	Dolor de espalda
08	Dolor muscular

## Síntomas de grado 2

Tabla 11. Síntomas de grado 2

Código	Síntoma
01	Dificultad para respirar
02	Perdida del olfato
03	Perdida del gusto
04	Dolor de garganta
05	Fiebre alta
06	Dolor de cuerpo
07	Mialgias
08	Anosmia
09	Tos persistente

## Síntomas de grado 3

Tabla 12. Síntomas de grado 3

Código	Síntoma
01	Náuseas
02	Vómito
03	Dolores musculares
04	Disnea
05	Cianosis
06	Hipotensión
07	Confusión
08	Dolor al respirar
09	Fiebre alta
10	Baja saturación
11	Tos productiva
12	Pérdida del gusto
13	Pérdida del olfato

## 10. Entrevistados

	A	K	L	M
1	Marca temporal	Nombres y Apellidos	Número de cédula	Profesión
2	4/08/2021 20:03:19	Danny Renato Mora Huiracocha	1104643851	Ninguna
3	4/08/2021 20:25:07	Verónica Estafania Gutiérrez Luzon	1105139909	Estudiante de enfermería
4	6/08/2021 23:43:48	yakeline Cartuche	0982592222	Médico
5	7/08/2021 21:16:56	Hugo Ramiro Córdova Córdova	703296608	Enfermero
6				

Figura 1. Lista de entrevistados.

## Síntomas de COVID-19

Si un paciente con síntomas de COVID-19 se pudiera clasificar en 3 grados de EMERGENCIA.

Siendo:

GRADO 1: Baja probabilidad de COVID-19.

Amerita medidas de bioseguridad como medida de prevención.

GRADO 2: Probabilidad MEDIA de COVID-19.

Amerita medidas de bioseguridad y atención médica opcional.

GRADO 3: Alta probabilidad de COVID-19.

Amerita atención médica inmediata y medicamentos de emergencia.

¿Cuáles serían los síntomas del paciente para cada uno de los diferentes grados?

Procurar usar el menor número de palabras posibles.

*Figura 2. Indicaciones generales para el desarrollo de la encuesta.*

### Síntoma 1 - GRADO 1

Baja probabilidad de que sea COVID-19. Por ejemplo: "Dolor de cabeza leve"

Texto de respuesta corta

### Síntoma 2 - GRADO 1

Baja probabilidad de que sea COVID-19. Por ejemplo: "Dolor de cabeza leve"

Texto de respuesta corta

### Síntoma 3 - GRADO 1

Baja probabilidad de que sea COVID-19. Por ejemplo: "Dolor de cabeza leve"

Texto de respuesta corta

*Figura 3. Síntomas de grado 1.*

**GRADO 2 -> Probabilidad Media de que sea COVID-19**

¿Cuáles serían los síntomas del paciente para ser clasificado como GRADO 2?

**Síntoma 1 - GRADO 2**

Probabilidad MEDIA de que sea COVID-19. Por ejemplo: "Dolor de cabeza constante"

Texto de respuesta corta

**Síntoma 2 - GRADO 2**

Probabilidad MEDIA de que sea COVID-19. Por ejemplo: "Dolor de cabeza constante"

Texto de respuesta corta

**Síntoma 3 - GRADO 2**

Probabilidad MEDIA de que sea COVID-19. Por ejemplo: "Dolor de cabeza constante"

Texto de respuesta corta

*Figura 4. Síntomas de grado 2*

**GRADO 3 -> Alta probabilidad de que sea COVID-19**

¿Cuáles serían los síntomas del paciente para ser clasificado como GRADO 3?

Síntoma 1 - GRADO 3

Probabilidad ALTA de que sea COVID-19. Por ejemplo: "Problemas para respirar"

Texto de respuesta corta

---

Síntoma 2 - GRADO 3

Probabilidad ALTA de que sea COVID-19. Por ejemplo: "Problemas para respirar"

Texto de respuesta corta

---

Síntoma 3 - GRADO 3

Probabilidad ALTA de que sea COVID-19. Por ejemplo: "Problemas para respirar"

Texto de respuesta corta

---

*Figura 5. Síntomas de grado 3.*

**Anexo 10.** Pruebas de aceptación a expertos

**Informe de encuesta de aceptación a profesionales en medicina**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

Para el desarrollo de la encuesta, se ha seguido el proceso propuesto por Kitchenham.

### 1. Establecer los objetivos de la encuesta

La presente encuesta forma parte del Trabajo de Titulación denominado “Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente”. Por lo cual se desarrolló el siguiente cuestionario con el objetivo de “Determinar el nivel de la funcionalidad, facilidad de uso, tiempos de respuesta y aceptación del módulo de voz implementado en el sistema desarrollado”.

El grupo objetivo de la encuesta son expertos en el área de la salud pertenecientes al distrito 11D04 Celica-Pindal-Puyango.

La Tabla 1 presenta el objetivo de la encuesta en base a la plantilla GQM (Goal-Question-Metric) y en la Tabla 2, se muestra las características a evaluar para el cumplimiento del objetivo.

**Tabla 1. Objetivos de encuesta**

Objetivo de estudio	Aplicación web
Propósito	Determinar el nivel de la facilidad de uso y aceptación de la aplicación
Enfoque de calidad	Facilidad de uso y aceptación
Perspectiva	Investigador
Contexto	Expertos en medicina que validan la aplicación

**Tabla 2. Características a evaluar**

Características	Descripción
Facilidad de uso	Nivel sobre la facilidad de uso de la aplicación web
Aceptación	Nivel de aceptación de la aplicación por parte de los expertos en medicina

Las hipótesis planteadas, son:

- H1: Es fácil e intuitivo de usar para el usuario.
- H2: Tiene una buena aceptación por parte de los expertos en medicina.
- H3: La comunicación con la aplicación clara y comprensible.

## **2. Desarrollar cuestionario**

En este apartado se define las dos actividades que se llevarán a cabo en el desarrollo del cuestionario; comenzando por la definición de las preguntas del cuestionario y finalizando con los tipos de respuesta para cada pregunta.

### **2.1. Diseño de las preguntas**

En la Tabla 3 se presenta el cuestionario, el cual consta de 14 preguntas; mismas que se usó para el cumplimiento del objetivo planteado.

**Tabla 3. Cuestionario de preguntas aplicadas**

Nº	Pregunta
P1	¿La aplicación al momento de usarla fue?
P2	¿Requirió ayuda para realizar las tareas solicitadas?
P3	¿Los tiempos de respuesta al realizar las tareas por parte de la aplicación, han sido adecuados?
P4	¿Considera que la aplicación presentada es fácil de usar para cualquier persona con la capacidad de realizar una llamada?
P5	¿Las instrucciones de ingreso de datos solicitados por la aplicación son fáciles de seguir?
P6	¿La secuencia durante la interacción con la aplicación es intuitiva?
P7	¿La voz emitida por la aplicación es clara y comprensible?
P8	¿Indique el rango de tiempo estimado de interacción con la aplicación?
P9	¿Cree que la clasificación de pacientes según sus síntomas realizada por el sistema es correcta?
P10	Indique el porcentaje estimado de efectividad en las tareas de funcionalidad de la aplicación.
P11	En caso de encontrar algún problema en la funcionalidad de la aplicación, descríbalo a continuación:
P12	¿Cree usted que la aplicación puede ser de ayuda para las personas que poseen Covid-19?
P13	¿Cree usted que la aplicación puede ser de ayuda para las instituciones de salud que reciben llamadas de personas con Covid-19?
P14	¿Tiene alguna sugerencia para la mejora de la aplicación?

## **2.2. Diseño de las respuestas**

La Tabla 4 se muestra las respuestas que se han establecido de acuerdo al tipo de pregunta, dando a elegir al encuestado entre un rango de opciones, para lo cual se han utilizado diferentes escalas de Likert de evaluación, rangos numéricos, además se han definido preguntas abiertas y sugerencias al investigador.

**Tabla 4. Tipo de respuestas**

Pregunta	Respuesta
P1	Opción múltiple
P2	Opción múltiple
P3	Opción múltiple
P4	Opción múltiple
P5	Opción múltiple
P6	Opción múltiple
P7	Opción múltiple
P8	Opción múltiple
P9	Opción múltiple con observaciones
P10	Opción múltiple
P11	Abierta
P12	Dicotómica con observaciones
P13	Dicotómica con observaciones
P14	Abierta

## **3. Evaluar y validar el cuestionario**

La evaluación y validación del cuestionario lo realizó el director del Trabajo de Titulación, con la finalidad de verificar las preguntas planteadas y que permitan el cumplimiento del objetivo; además se estimaron fechas y tiempos adecuados para su ejecución.

## **4. Obtener los datos de la encuesta**

La muestra seleccionada, fue mediante el método no probabilístico por conveniencia, de acuerdo a la accesibilidad con el investigador. La encuesta fue aplicada a seis profesionales en medicina, que actualmente se encuentran ejerciendo en distintas instituciones médicas del distrito 11D04 Celica-Pindal-Puyango.

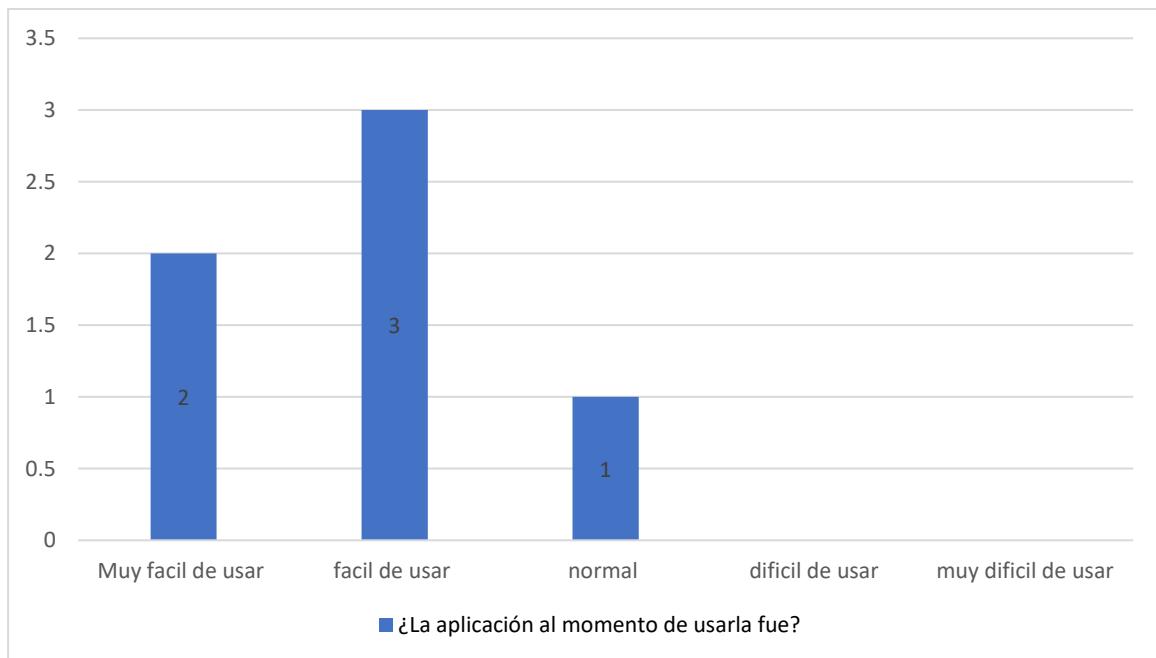
## 5. Analizar los datos obtenidos

### 1. ¿La aplicación al momento de usarla fue?

Tabla 5. Respuestas pregunta 1

Alternativas	Frecuencia	Porcentaje
Muy fácil de usar	2	33,3%
Fácil de usar	3	50%
Normal	1	16,7%
Difícil de usar	0	0%
Muy difícil de usar	0	0%
Total	6	100%

Tabla 6. Gráfica de respuestas pregunta 1



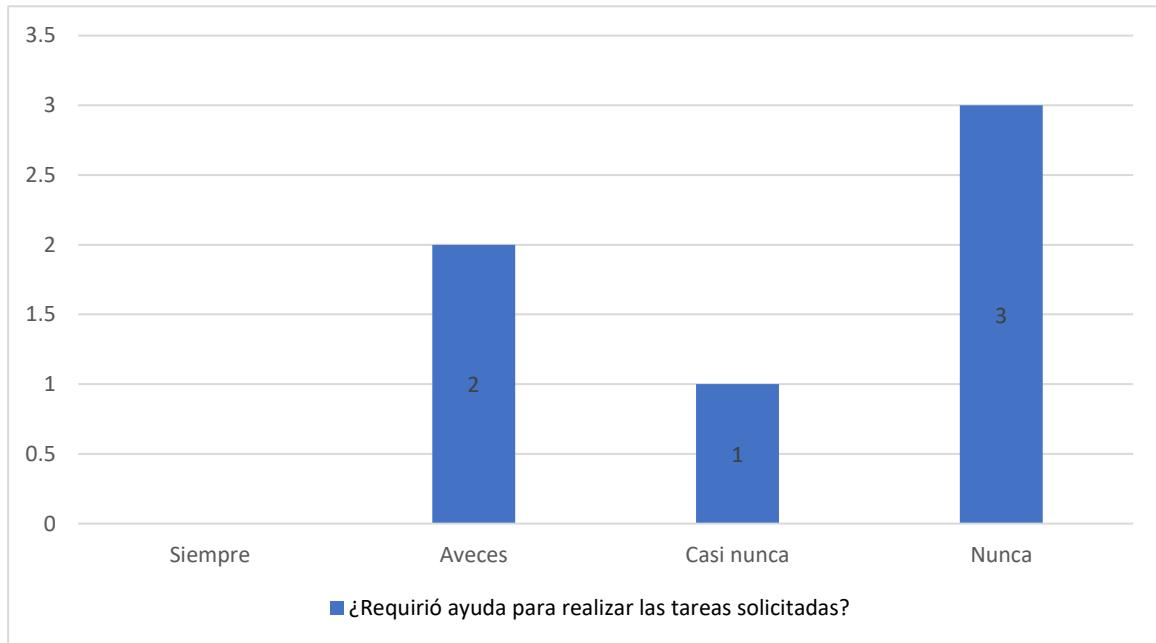
Los resultados obtenidos por esta gráfica indican que dos de las personas encuestadas consideran que la aplicación es muy fácil de usar, en cambio tres de los encuestados cree que la aplicación es fácil de usar y el último de los encuestados considera que la aplicación tiene una complejidad normal.

## 2. ¿Requirió ayuda para realizar las tareas solicitadas?

**Tabla 7. Respuestas pregunta 2**

Alternativas	Frecuencia	Porcentaje
Siempre	0	0%
A veces	2	33.3%
Casi nunca	1	16,7%
Nunca	3	50%
Total	6	100%

**Tabla 8. Gráfica de respuestas pregunta 2**



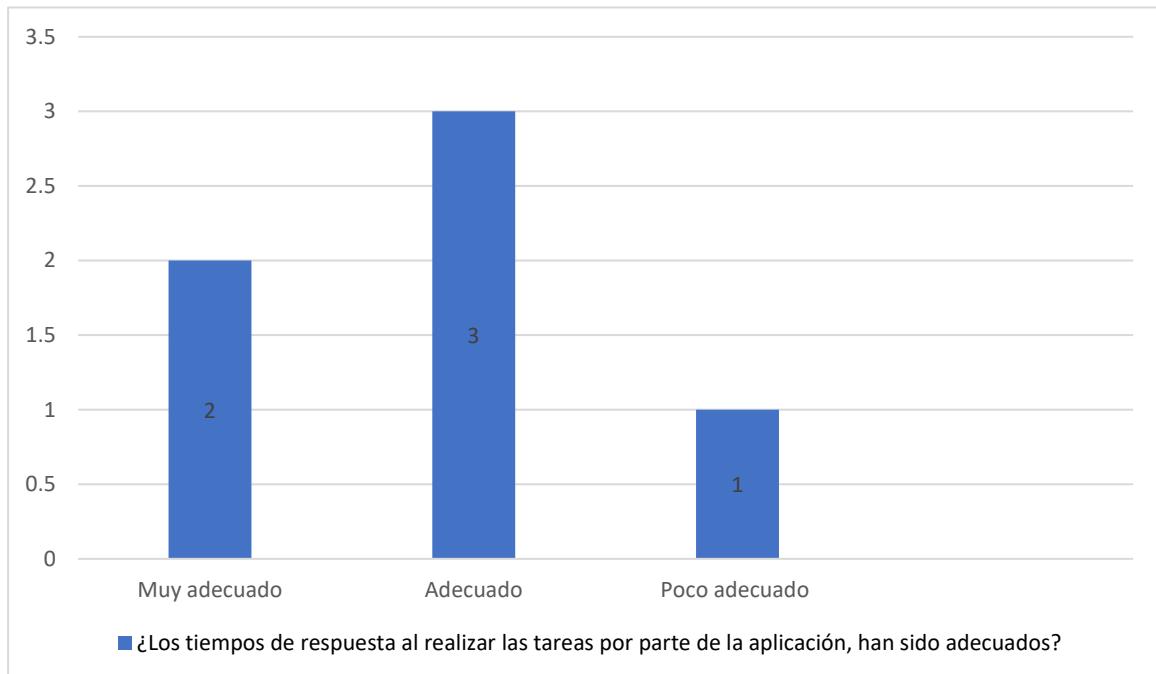
La gráfica muestra que ninguno de los encuestados considera que necesita ayuda para el uso de la aplicación, la mayoría considera que esta ayuda no es necesaria en ningún momento, sin embargo, dos de los encuestados considera que esta ayuda se requiere a veces y un encuestado considera que casi nunca se necesita ayuda.

**3. ¿Los tiempos de respuesta al realizar las tareas por parte de la aplicación, han sido adecuados?**

**Tabla 9. Respuestas pregunta 3**

Alternativas	Frecuencia	Porcentaje
Muy adecuado	2	33,3%
Adecuado	3	50%
Poco adecuado	1	16,7%
Total	6	100%

**Tabla 10. Gráfica de respuestas pregunta 3**



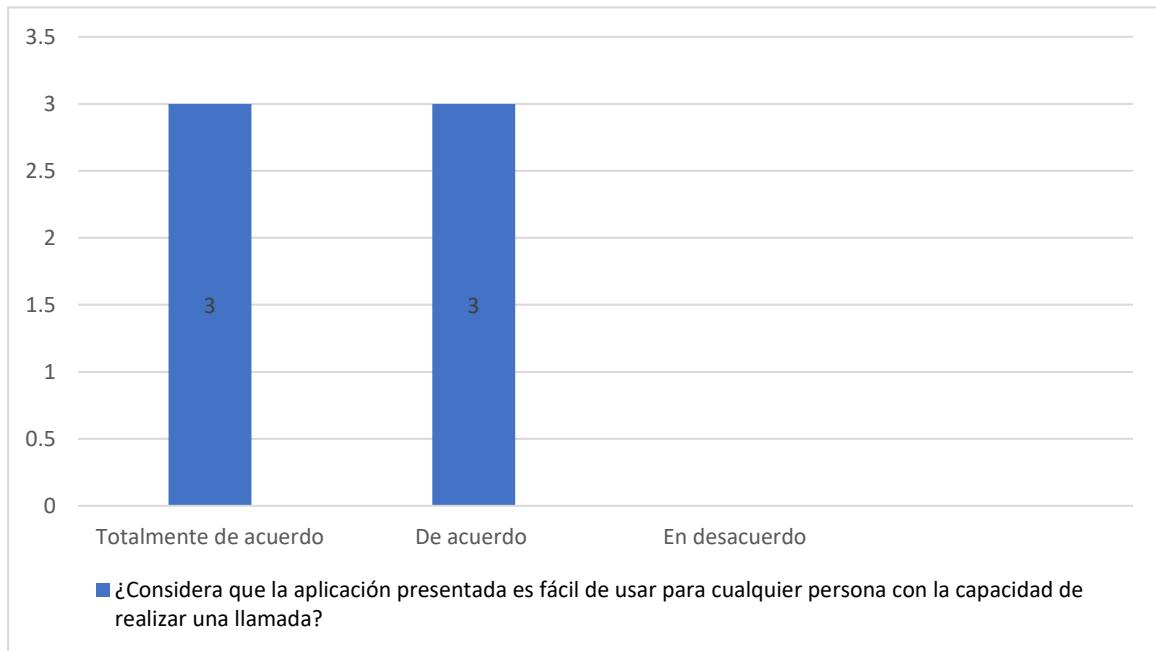
La interpretación de estos datos nos indica que la mayoría de las personas encuestadas considera que la aplicación tiene tiempos de respuesta adecuados, dos de los encuestados considera que son muy adecuados y uno de los encuestados considera que dichos tiempos son poco adecuados.

**4. ¿Considera que la aplicación presentada es fácil de usar para cualquier persona con la capacidad de realizar una llamada?**

**Tabla 11. Respuestas pregunta 4**

Alternativas	Frecuencia	Porcentaje
<b>Totalmente de acuerdo</b>	3	50%
<b>De acuerdo</b>	3	50%
<b>En desacuerdo</b>	0	0%
<b>Total</b>	6	100%

**Tabla 12. Gráfica de respuestas pregunta 4**



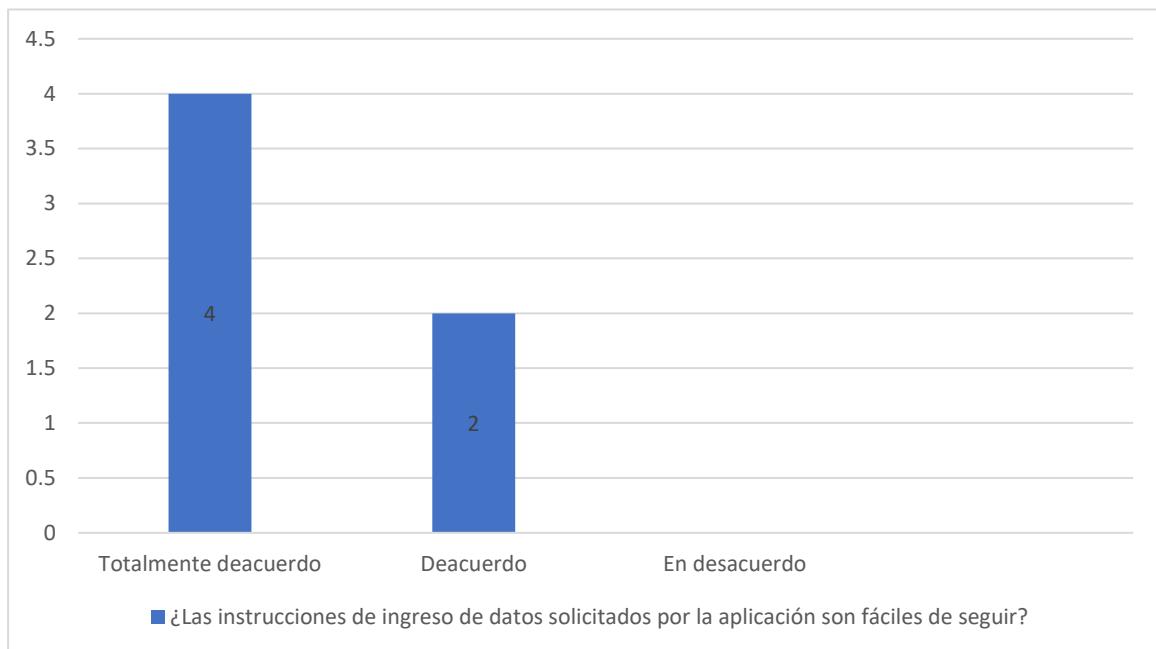
La gráfica nos muestra que la mitad de los encuestados concuerda en que cualquier persona puede hacer uso de la aplicación, la otra mitad concuerda totalmente en que la aplicación puede ser usada por cualquier persona con la capacidad de realizar una llamada.

**5. ¿Las instrucciones de ingreso de datos solicitados por la aplicación son fáciles de seguir?**

**Tabla 13. Respuestas pregunta 5**

Alternativas	Frecuencia	Porcentaje
<b>Totalmente de acuerdo</b>	4	66.7%
<b>De acuerdo</b>	2	33.3%
<b>En desacuerdo</b>	0	0%
<b>Total</b>	6	100%

**Tabla 14. Gráfica de respuestas pregunta 5**



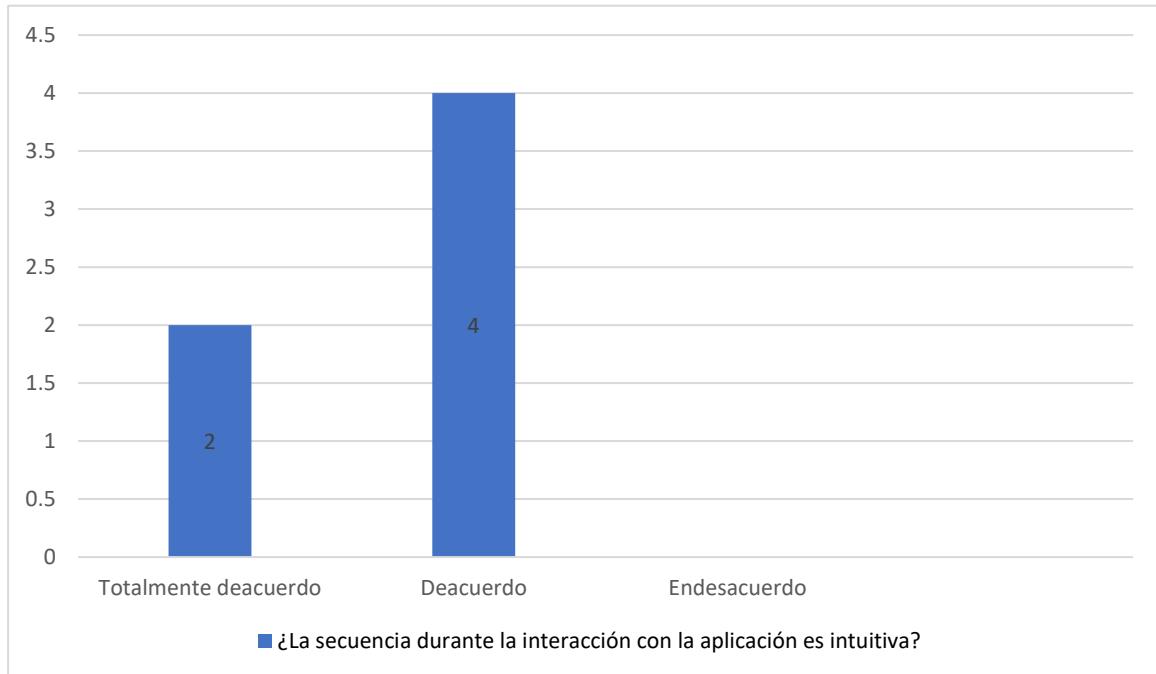
Los resultados obtenidos indican que cuatro de los encuestados concuerdan totalmente en que las instrucciones brindadas por la aplicación son fáciles de seguir, sin embargo, dos de los encuestados expresan que las instrucciones no son al cien por ciento sencillas dejando un grado no tan satisfactorio.

## 6. ¿La secuencia durante la interacción con la aplicación es intuitiva

Tabla 15. Respuestas pregunta 6

Alternativas	Frecuencia	Porcentaje
Totalmente de acuerdo	2	33.3%
De acuerdo	4	66.7%
En desacuerdo	0	0%
Total	6	100%

Tabla 16. Gráfica de respuestas pregunta 6



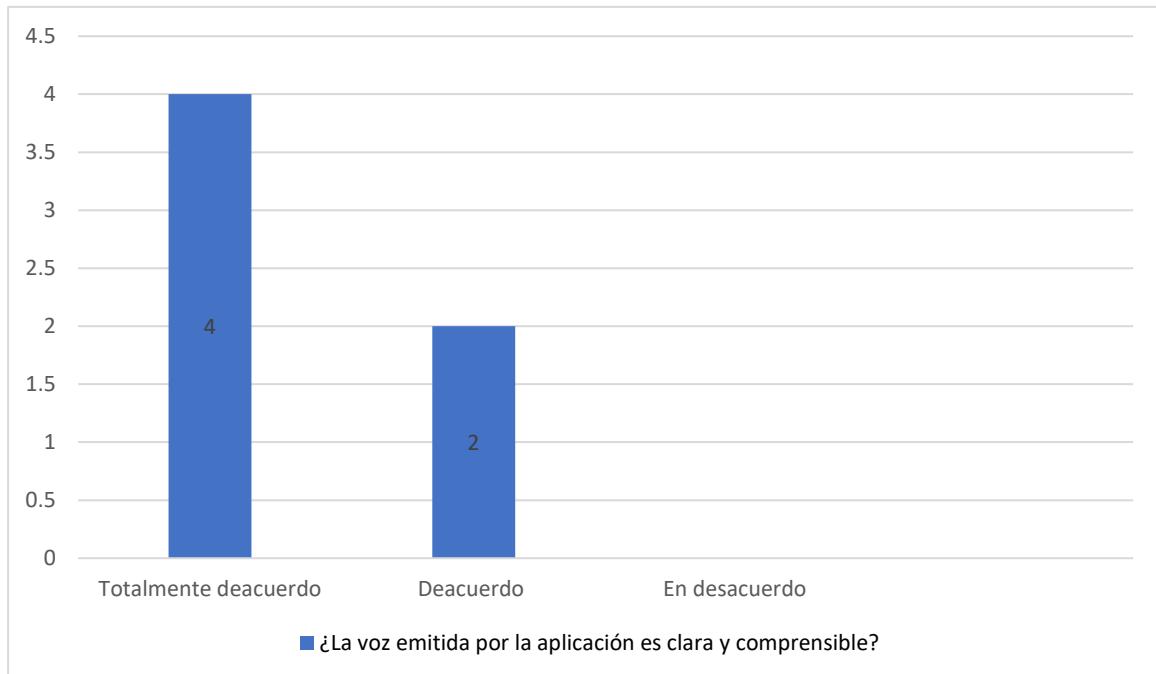
De la gráfica obtenida se puede determinar que la mayoría de los encuestados está de acuerdo en que la interacción con la aplicación es intuitiva, de estos, dos están totalmente de acuerdo con lo expresado y ninguno considera que la aplicación no es intuitiva.

**7. ¿La voz emitida por la aplicación es clara y comprensible?**

**Tabla 17. Respuestas pregunta 7**

Alternativas	Frecuencia	Porcentaje
Totalmente de acuerdo	4	66.7%
De acuerdo	2	33.3%
En desacuerdo	0	0%
Total	6	100%

**Tabla 18. Gráfica de respuestas pregunta 7**



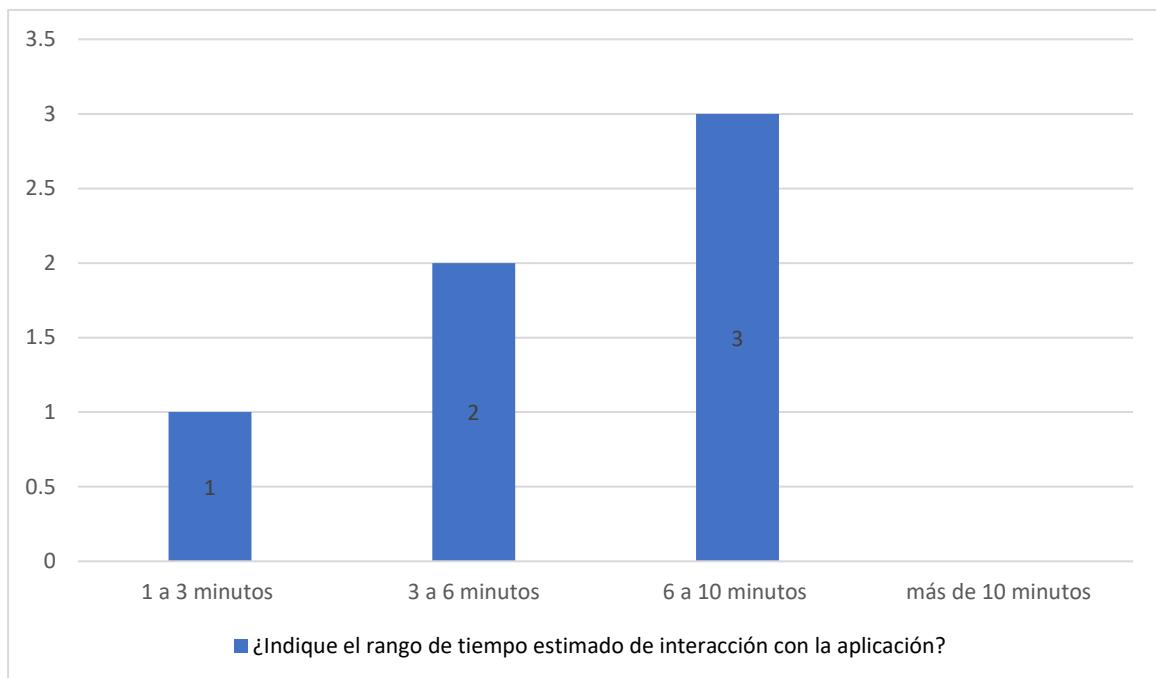
De la gráfica se puede interpretar que cuatro de los encuestados considera que la aplicación es clara y comprensible en su totalidad, dos de estos considera que está de acuerdo con lo mencionado y ninguno considera que la aplicación no es clara y comprensible.

**8. ¿Indique el rango de tiempo estimado de interacción con la aplicación?**

**Tabla 19. Respuestas pregunta 8**

Alternativas	Frecuencia	Porcentaje
1 a 3 min	1	16.7%
3 a 6 min	2	33.3%
6 a 10 min	3	50%
Más de 10 min	0	0%
Total	6	100%

**Tabla 20. Gráfica de respuestas pregunta 8**



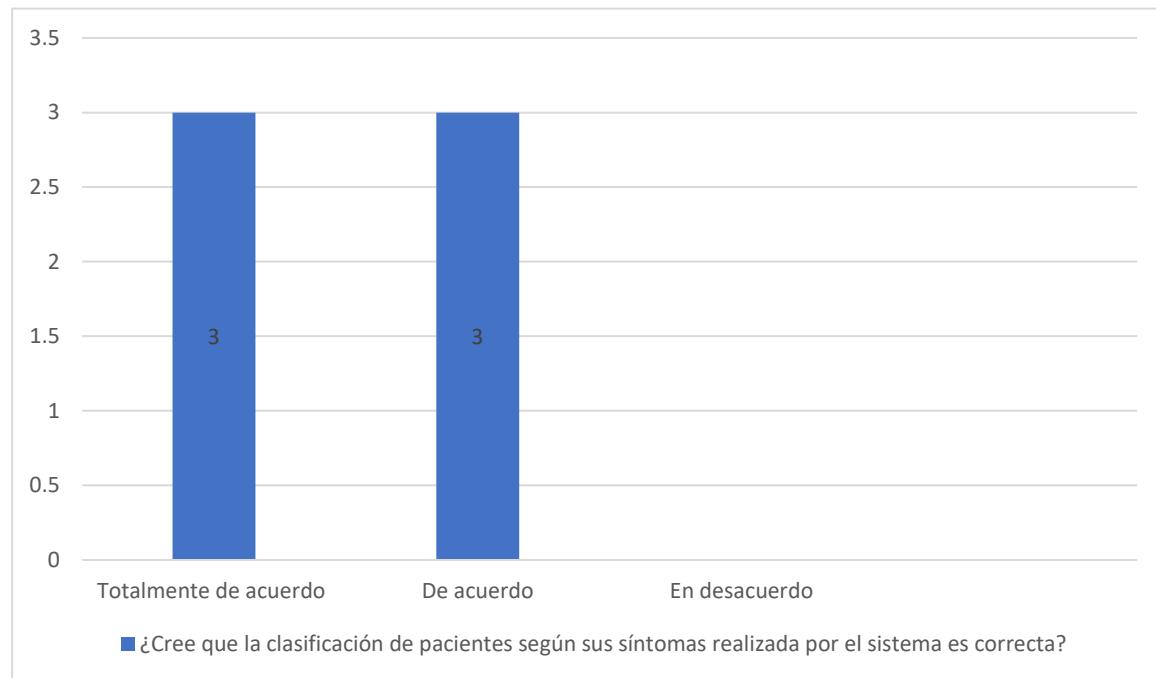
De la gráfica se puede obtener que la mayor parte de los encuestados determina que la interacción con la aplicación es de 6 a 10 minutos, dos estiman que el tiempo de interacción está entre los 3 y 6 minutos y uno considera que el tiempo fluctúa entre 1 a 3 minutos.

**9. ¿Cree que la clasificación de pacientes según sus síntomas realizada por el sistema es correcta?**

**Tabla 21. Respuestas pregunta 9**

Alternativas	Frecuencia	Porcentaje
Totalmente de acuerdo	3	50%
De acuerdo	3	50%
En desacuerdo	0	0%
Total	6	100%

**Tabla 22. Gráfica de respuestas pregunta 9**



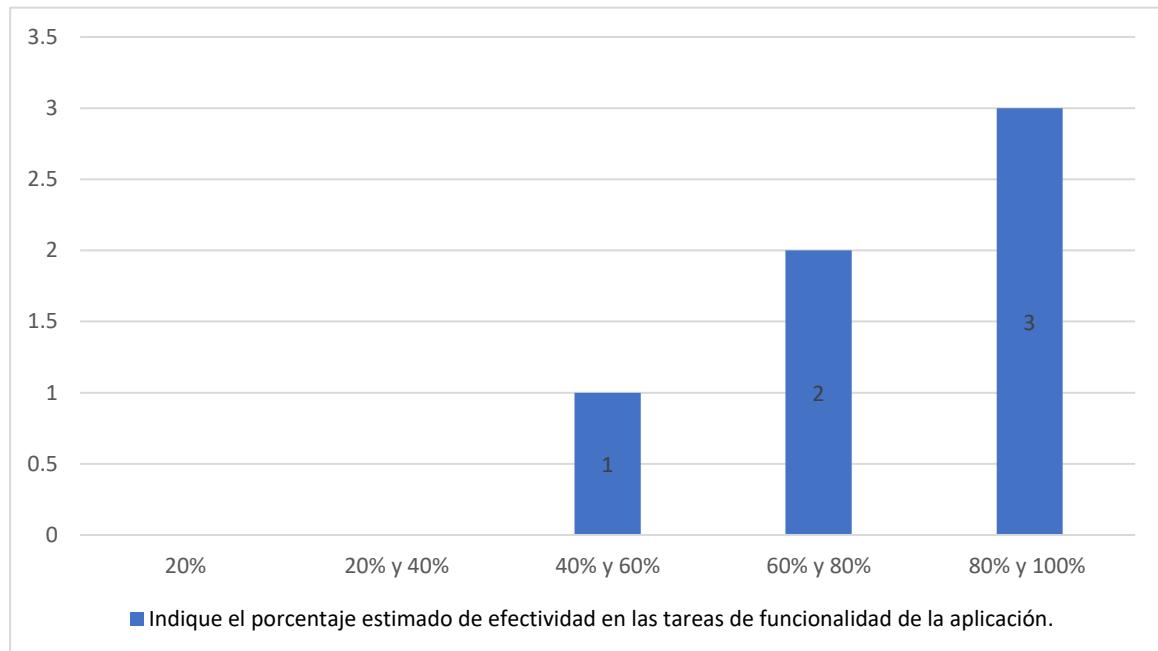
La gráfica obtenida de las respuestas nos indica que tres de los encuestados se encuentran de acuerdo en que la clasificación de los pacientes realizada por el programa es la correcta, así también se obtiene que el resto de los encuestados concuerda totalmente con dicha clasificación.

**10. Indique el porcentaje estimado de efectividad en las tareas de funcionalidad de la aplicación.**

**Tabla 23. Respuestas pregunta 10**

Alternativas	Frecuencia	Porcentaje
20%	0	0%
20% y 40%	0	0%
40% y 60%	1	16%
60% y 80%	2	33.3%
80% y 100%	3	50%
Total	6	100%

**Tabla 24. Gráfica de respuestas pregunta 10**



De la figura se puede obtener que la mayoría de los encuestados establece que las tareas de funcionalidad de la aplicación se han efectuado entre un 80 y 100 por ciento, dos de los encuestados determinan que entre un 60 y 80 por ciento de efectividad y uno de los encuestados expresa que se cumple en un 40 y 60 por ciento.

**11. En caso de encontrar algún problema en la funcionalidad de la aplicación, descríbalo a continuación.**

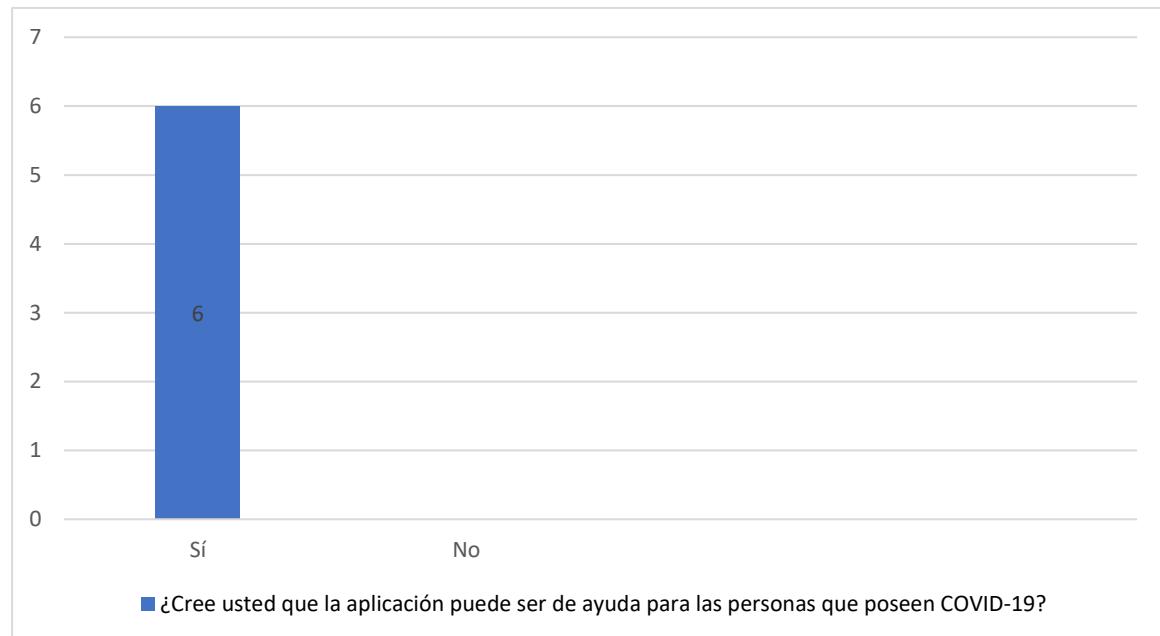
Procurar no usar términos médicos para la recolección de síntomas en la aplicación.

**12. ¿Cree usted que la aplicación puede ser de ayuda para las personas que poseen Covid-19?**

**Tabla 25. Respuestas pregunta 12**

Alternativas	Frecuencia	Porcentaje
Sí	6	100%
No	0	0%
Total	6	100%

**Tabla 26. Gráfica de respuestas pregunta 12**



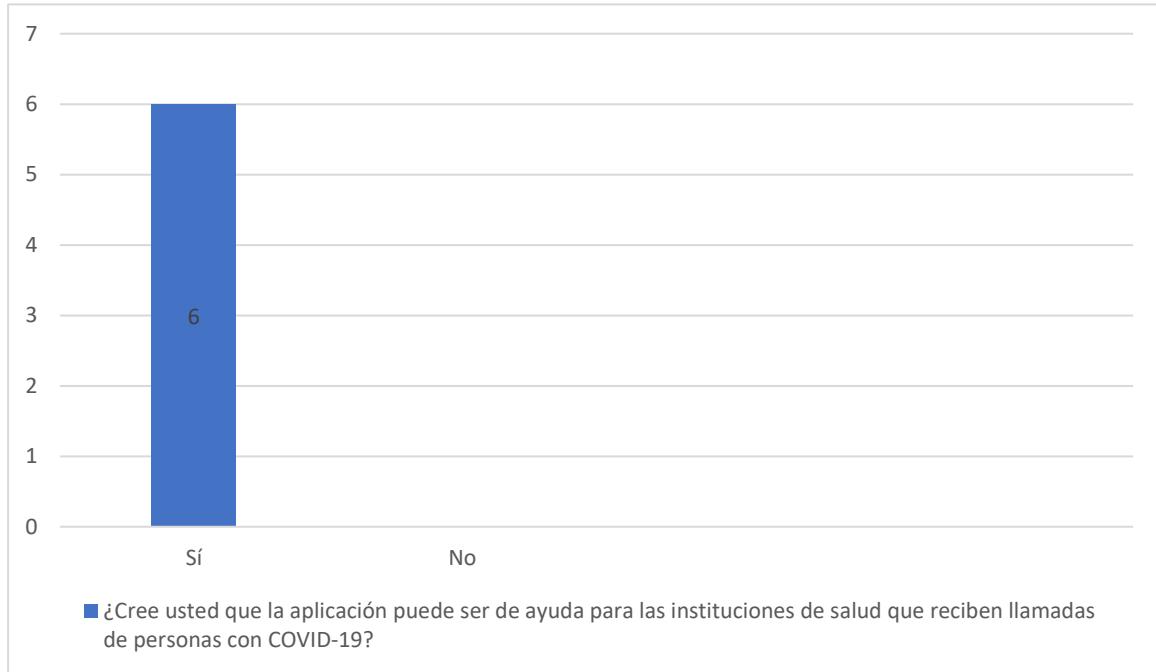
El cien por ciento de los encuestados determina que la aplicación sí es de ayuda para personas que puedan o que posean Covid-19.

**13. ¿Cree usted que la aplicación puede ser de ayuda para las instituciones de salud que reciben llamadas de personas con Covid-19?**

**Tabla 27. Respuestas pregunta 13**

Alternativas	Frecuencia	Porcentaje
Sí	6	100%
No	0	0%
Total	6	100%

**Tabla 28. Gráfica de respuestas pregunta 13**



El cien por ciento de los encuestados determina que la aplicación si es de ayuda para las instituciones que reciben llamadas de pacientes con Covid-19.

**14. ¿Cree usted que la aplicación puede ser de ayuda para las instituciones de salud que reciben llamadas de personas con Covid-19?**

No se recibieron respuestas para esta pregunta.

## **6. Evaluación de hipótesis**

- H1: Es fácil e intuitivo de usar para el usuario.

Gracias a los resultados obtenidos en las preguntas uno y dos se pudo aceptar la hipótesis planteada ya que los encuestados consideran que la aplicación es fácil o muy fácil de usar y en su mayoría nunca necesitaron de ayuda para su manipulación.

- H2: Tiene una buena aceptación por parte de los expertos en medicina.

La hipótesis es aceptada, ya que en su mayoría los resultados arrojados por la encuesta han sido favorables, dejando claro que existe un grado de aceptación elevada, especialmente en las preguntas once y doce que son preguntas claras para expresar la aceptación por parte de los profesionales en medicina.

- H3: La comunicación con la aplicación clara y comprensible.

La hipótesis es acepta, los resultados obtenidos de la pregunta siete evidencian que la mayoría, con cuatro encuestados, afirma que la voz emitida por parte del programa es una voz clara y comprensible.

## **7. Sugerencias al investigador**

Las siguientes sugerencias, han sido obtenidas ya sea por parte de los encuestados o mediante deducción al momento de aplicar las encuestas.

- No se debe manejar un lenguaje médico, ya que es una aplicación para el público y en su mayoría este no comprende términos médicos.
- Se debe procurar no usar un lenguaje técnico al momento de presentar la aplicación para mejorar la comprensión del profesional.
- Evitar el uso de lenguaje técnico en las preguntas planteadas en la encuesta para evitar confusión por parte de los encuestados.
- Evitar el uso de preguntas opcionales y abiertas.

**Anexo 11.** Pruebas de aceptación a usuarios finales

**Informe de encuesta de aceptación a usuarios finales**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

Para el desarrollo de la encuesta, se ha seguido el proceso propuesto por Kitchenham.

### **1. Establecer los objetivos de la encuesta**

La presente encuesta forma parte del Trabajo de Titulación denominado “Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente”. Por lo cual se desarrolló el siguiente cuestionario con el objetivo de “Evaluar el cumplimiento de los requerimientos planteados para el desarrollo del módulo de software”.

El grupo objetivo son personas comunes entre los 20 a 49 años de edad que validan el funcionamiento de la aplicación.

La Tabla 1 presenta el objetivo de la encuesta en base a la plantilla GQM (Goal-Question-Metric) y en la Tabla 2, se muestra las características a evaluar para el cumplimiento del objetivo.

**Tabla 1. Objetivos de encuesta.**

<b>Objetivo de estudio</b>	<b>Aplicación web</b>
Propósito	Evaluar el cumplimiento de los requerimientos planteados para el desarrollo del módulo de software.
Enfoque de calidad	Facilidad de uso y aceptación.
Perspectiva	Investigador.
Contexto	Personas comunes que validan la aplicación.

**Tabla 2. Características a evaluar.**

<b>Características</b>	<b>Descripción</b>
Facilidad de uso	Nivel sobre la facilidad de uso de la aplicación web.
Aceptación	Nivel de aceptación de la aplicación por parte de los usuarios finales.

Las hipótesis planteadas, son:

- H1: El usuario puede transferir su llamada telefónica.
- H2: El usuario puede listar los síntomas que posee.
- H3: El sistema clasifica al paciente según los síntomas que posee.
- H4: El sistema brinda información de las medidas de bioseguridad, genera una cita médica o receta en caso de emergencia según la prioridad asignada al usuario.
- H5: El sistema cuenta con una interfaz intuitiva y sencilla de usar.
- H6: El sistema cuenta con una voz aguda y agradable al oído.
- H7: La aplicación desarrollada ayuda en el proceso de atención médica mediante llamadas telefónicas.

## **2. Desarrollar cuestionario**

En este apartado se define las dos actividades que se llevarán a cabo en el desarrollo del cuestionario; comenzando por la definición de las preguntas del cuestionario y finalizando con los tipos de respuesta para cada pregunta.

### **2.1. Diseño de las preguntas**

En la Tabla 3 se presenta el cuestionario, el cual consta de 9 preguntas; mismas que se usaron para el cumplimiento del objetivo planteado.

**Tabla 3. Preguntas de las pruebas de aceptación aplicadas al usuario final**

Nº	Pregunta
P1	¿La aplicación le da la opción de transferir su llamada? a. Sí b. No
P2	¿La aplicación le permite enumerar sus síntomas? a. Si b. No
P3	¿La aplicación le asigna un grado de prioridad según sus síntomas? (Prioridad baja, media y alta) a. Sí b. No
P4	¿La aplicación le brinda información sobre las medidas de bioseguridad? a. Sí b. No
P5	¿La aplicación le genera una cita médica? (Si usted fue clasificado como prioridad 2 o prioridad 3) a. Sí b. No
P6	¿La aplicación le receta medicina de emergencia? (Si usted fue clasificado como prioridad 3) a. Sí b. No
P7	¿La aplicación al momento de usarla fue intuitiva y sencilla de usar? a. Siempre b. Casi siempre c. A veces
P8	¿La voz emitida por la aplicación es agradable y comprensible? a. Si b. No
P9	¿Considera que la aplicación ayuda en el proceso de atención médica mediante llamadas telefónicas? a. Totalmente de acuerdo b. De acuerdo c. En desacuerdo

## **2.2. Diseño de las respuestas**

La Tabla 4 muestra el tipo de respuesta que se ha designado a cada pregunta, para lo cual se han utilizado dos escalas de evaluación, opción múltiple y dicotómicas.

**Tabla 4. Tipo de respuestas**

Pregunta	Respuesta
P1	Opción dicotómica
P2	Opción dicotómica
P3	Opción dicotómica
P4	Opción dicotómica
P5	Opción dicotómica
P6	Opción dicotómica
P7	Opción múltiple
P8	Opción dicotómica
P9	Opción múltiple

## **3. Evaluar y validar el cuestionario**

La evaluación y validación del cuestionario lo realizó el director del Trabajo de Titulación, con la finalidad de verificar las preguntas planteadas y que permitan el cumplimiento del objetivo.

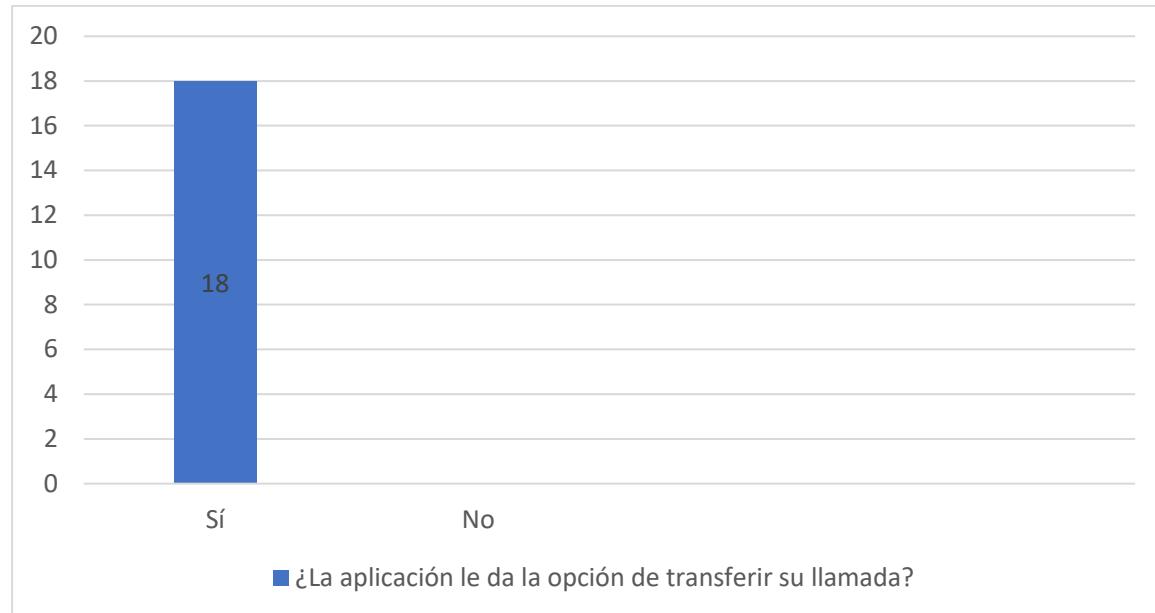
## **4. Obtener los datos de la encuesta**

La muestra seleccionada, fue mediante el método no probabilístico por conveniencia, de acuerdo a la accesibilidad con el investigador. Se seleccionó a dieciocho personas comunes que harán uso de la aplicación desarrollada.

## 5. Analizar los datos obtenidos

### 1. ¿La aplicación le da la opción de transferir su llamada?

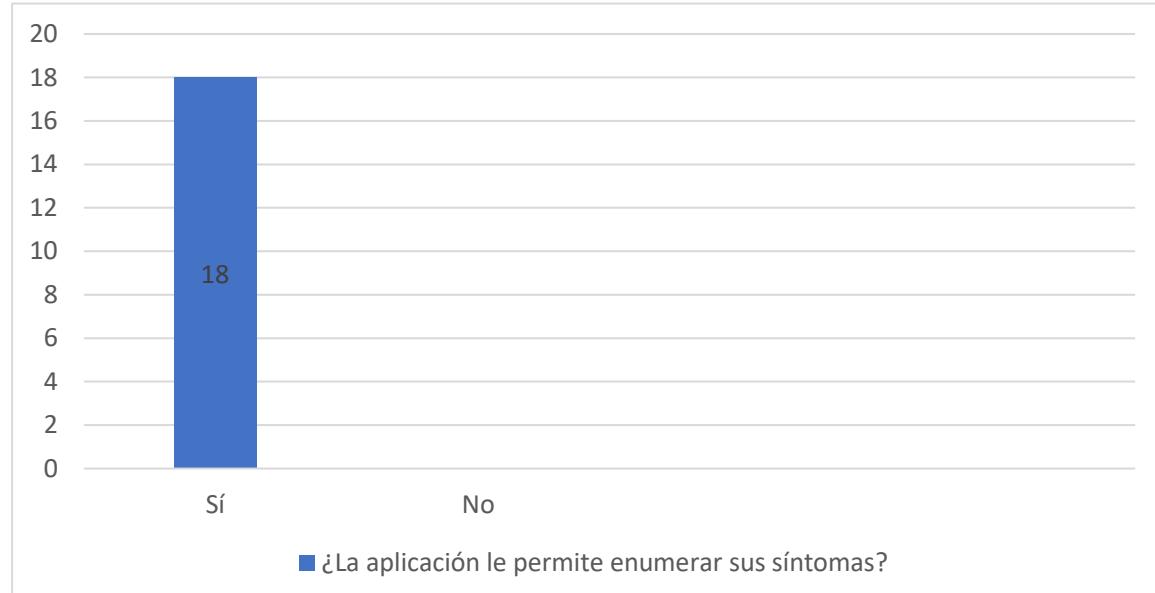
**Tabla 5. Respuestas pregunta 2**



El cien por ciento de los encuestados considera que la aplicación sí les permite transferir su llamada.

### 2. ¿La aplicación le permite enumerar sus síntomas?

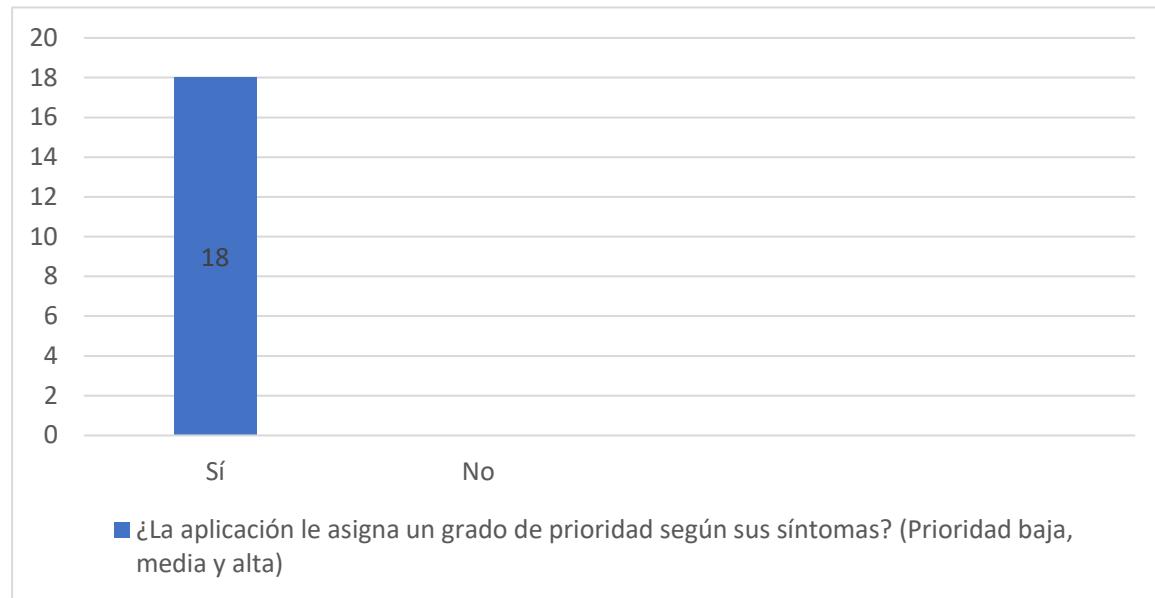
**Tabla 6. Respuestas pregunta 2**



El cien por ciento de los encuestados considera que la aplicación sí les permite enumerar sus síntomas.

**3. ¿La aplicación le asigna un grado de prioridad según sus síntomas? (Prioridad baja, media y alta)**

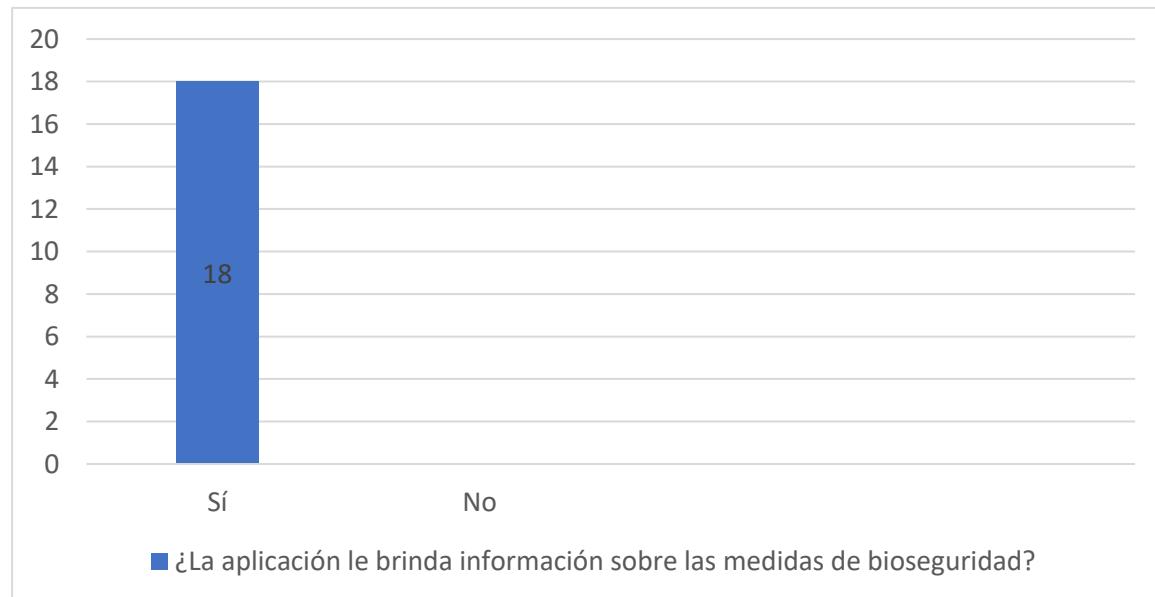
**Tabla 7. Respuestas pregunta 3**



El cien por ciento de los encuestados considera que la aplicación sí les asigna un grado de prioridad en base a síntomas.

**4. ¿La aplicación le brinda información sobre las medidas de bioseguridad?**

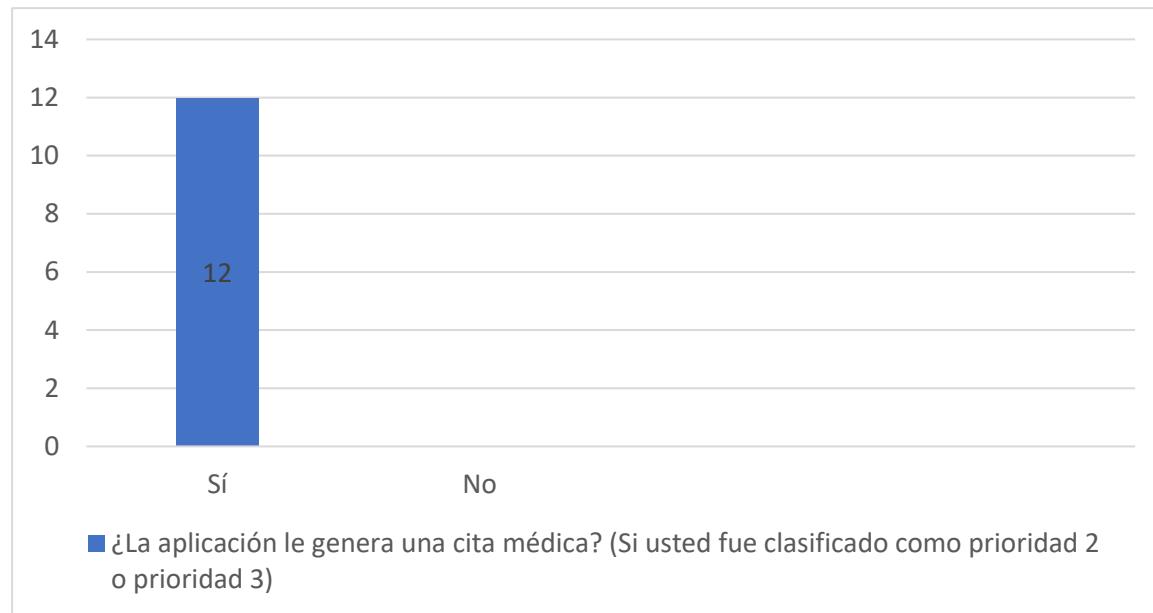
**Tabla 8. Respuestas pregunta 4**



El cien por ciento de los encuestados considera que la aplicación sí les brinda un listado de las medidas de bioseguridad.

**5. ¿La aplicación le genera una cita médica? (Si usted fue clasificado como prioridad 2 o prioridad 3)**

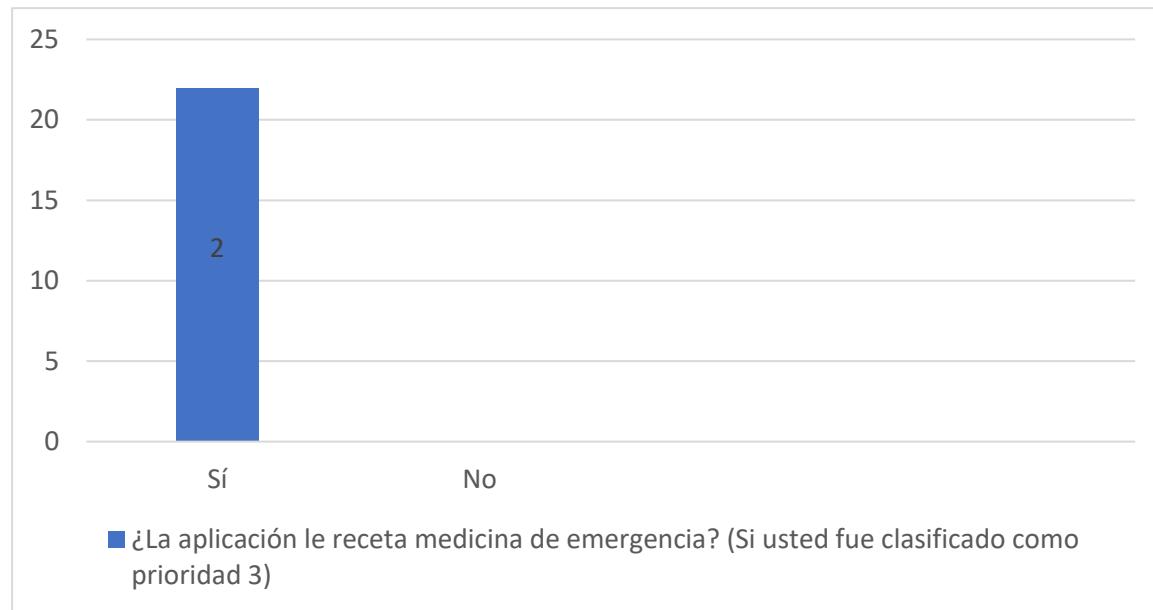
**Tabla 9. Respuestas pregunta 5**



El cien por ciento de los 12 encuestados clasificados como prioridad dos o tres expresa que la aplicación sí les generó una cita médica.

**6. ¿La aplicación le receta medicina de emergencia? (Si usted fue clasificado como prioridad 3)**

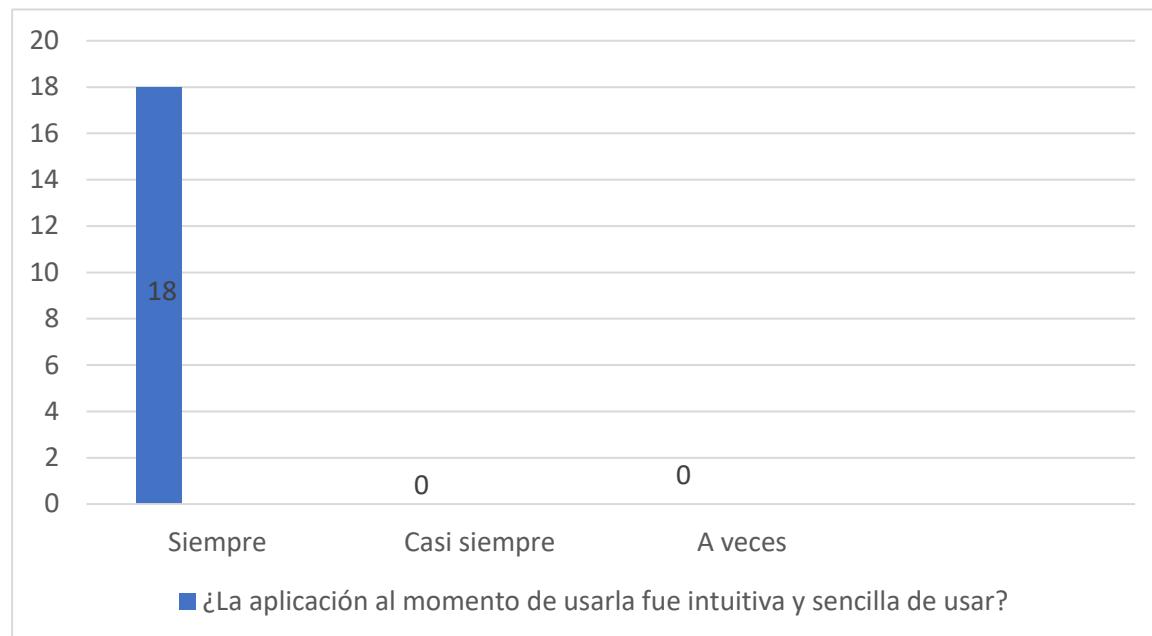
**Tabla 10. Respuestas pregunta 6**



El cien por ciento de los 2 encuestados clasificados como prioridad tres expresa que la aplicación sí les recetó medicina en caso de emergencia.

**7. ¿La aplicación al momento de usarla fue intuitiva y sencilla de usar?**

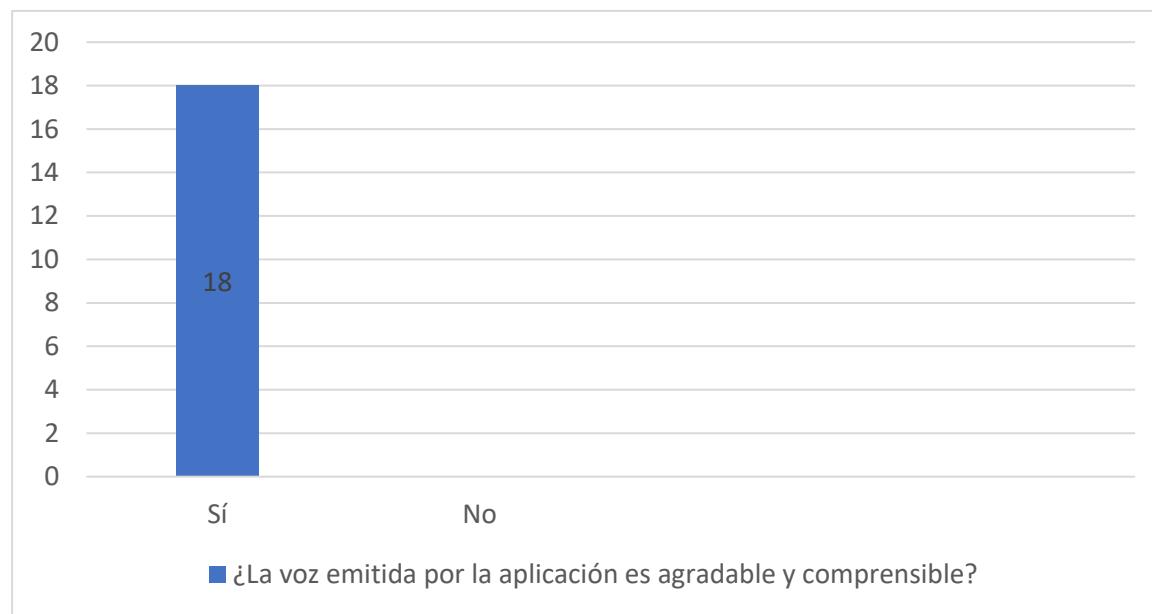
**Tabla 11. Respuestas pregunta 7**



Veintiún de los encuestados considera que la aplicación siempre es intuitiva y sencilla de usar, mientras que uno de los encuestados considera que no siempre lo es.

**8. ¿La voz emitida por la aplicación es agradable y comprensible?**

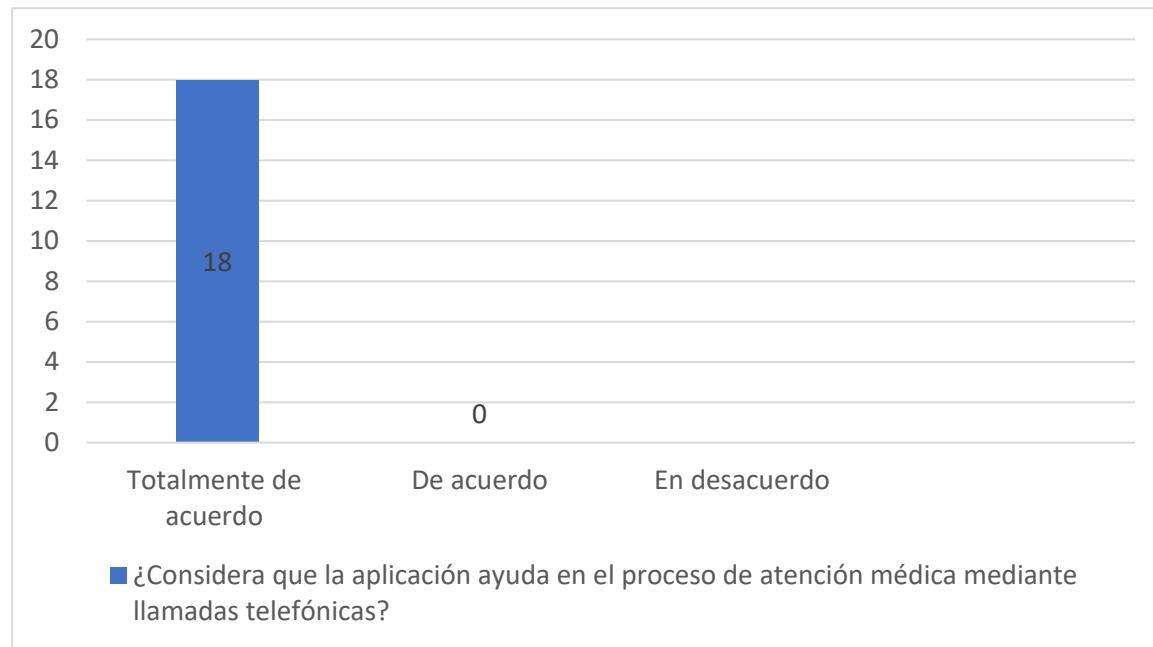
**Tabla 12. Respuestas de la pregunta 8**



El cien por ciento de los encuestados considera que la aplicación emite una voz agradable y comprensible para el oído.

**9. ¿Considera que la aplicación ayuda en el proceso de atención médica mediante llamadas telefónicas?**

**Tabla 13. Respuestas pregunta 9**



El cien por ciento de los encuestados considera que la aplicación aporta en el proceso de atención médica mediante llamadas telefónicas.

**Evaluación de hipótesis**

- H1: El usuario puede transferir su llamada telefónica.

Los resultados obtenidos en la pregunta uno, permiten aceptar la hipótesis planteada ya que el cien por ciento de los encuestados manifiestan que el sistema si les permite transferir su llamada telefónica.

- H2: El usuario puede listar los síntomas que posee.

Los resultados obtenidos en la pregunta dos, permiten aceptar la hipótesis planteada ya que el cien por ciento de los encuestados manifiestan que el sistema si les permite enumerar los diferentes síntomas que poseen.

- H3: El sistema clasifica al paciente según los síntomas que posee.

Los resultados obtenidos en la pregunta tres, permiten aceptar la hipótesis planteada ya que el cien por ciento de los encuestados manifiestan que el sistema les brinda una clasificación según los síntomas ingresados.

- H4: El sistema brinda información de las medidas de bioseguridad, genera una cita médica o receta en caso de emergencia según la prioridad asignada al usuario.

La hipótesis es aceptada ya que los resultados obtenidos en las preguntas cuatro, cinco y seis, muestran que los 11 encuestados recibieron información sobre las medidas de bioseguridad y se les agendó una cita médica y así mismo, a 2 de los 18 encuestados clasificados como prioridad 3 se les recetó medicina en caso de emergencia.

- H5: El sistema cuenta con una interfaz intuitiva y sencilla de usar.

La hipótesis es aceptada ya que los resultados obtenidos en la pregunta siete, muestran que el cien por ciento de los encuestados consideran que la aplicación es intuitiva y sencilla de usar.

- H6: El sistema cuenta con una voz aguda y agradable al oído.

La hipótesis se acepta ya que los resultados obtenidos en la pregunta ocho evidencian que el cien por ciento de los encuestados, afirman que la voz emitida por parte del programa es una voz aguda y agradable al oído.

- H7: La aplicación desarrollada ayuda en el proceso de atención médica mediante llamadas telefónicas.

La hipótesis se acepta ya que los resultados obtenidos en la pregunta nueve evidencian que el cien por ciento de los encuestados, consideran que la aplicación si ayuda en el proceso de atención médica mediante llamadas telefónicas.

### 3. Anexos



CARRERA DE INGENIERÍA  
EN SISTEMAS

#### PRUEBAS DE ACEPTACIÓN A USUARIOS FINALES

##### Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente

- ¿La aplicación le da la opción de transferir su llamada?  
a. Sí  
b. No
- ¿La aplicación le permite enumerar sus síntomas?  
a. Sí  
b. No
- ¿La aplicación le asigna un grado de prioridad según sus síntomas?  
(Prioridad baja, media y alta)  
a. Sí  
b. No
- ¿La aplicación le brinda información sobre las medidas de bioseguridad?  
a. Sí  
b. No
- ¿La aplicación le genera una cita médica?  
(Si usted fue clasificado como prioridad 2 o prioridad 3)  
a. Sí  
b. No
- ¿La aplicación le genera una cita médica?  
(Si usted fue clasificado como prioridad 2 o prioridad 3)  
a. Sí  
b. No
- ¿La aplicación al momento de usarla fue intuitiva y sencilla de usar?  
a. Siempre  
b. Casi siempre  
c. A veces
- ¿La voz emitida por la aplicación es agradable y comprensible?  
a. Sí  
b. No
- ¿Considera que la aplicación ayuda en el proceso de atención médica mediante llamadas telefónicas?  
a. Totalmente de acuerdo  
b. De acuerdo  
c. En desacuerdo

Figura 1. Encuesta aplicada.

Cludad Universitaria "Guillermo Falconi Espinosa" Casilla letra "S"  
Teléfono: 2547 - 252 Ext. 301; 2547-200  
direccion.cls@unl.edu.ec



Figura 2. Aplicación de la encuesta a estudiantes de la UNL.



Figura 3. Aplicación presencial de la encuesta a estudiantes de la UNL.



Figura 4. Aplicación de la encuesta a estudiantes de la UNL.



Figura 5. Aplicación de la encuesta a estudiantes de la UNL.



Figura 6. Aplicación de la encuesta a estudiantes de la UNL.



Figura 7. Aplicación de la encuesta a estudiantes de la UNL.

**Anexo 12.** Solicitud de implementación del módulo de software

**Solicitud de implementación del módulo de software**

Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la  
ayuda de un agente inteligente

Luego de la presentación y socialización del proyecto “**Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente**” al **DIRECTOR DISTRITAL 11D04 CELICA-PINDAL-PUYANGO SALUD Dr. Victor Hugo Tinoco Montaño**, a cargo de 15 unidades de atención de primer nivel (centros de salud) y del “Hospital básico Alamor” se realizó la solicitud escrita de implementación para el módulo de software, la cual se detalla en la Figura 1.

Loja, 28 de Abril del 2022

Sr.  
Dr. Victor Hugo Tinoco Montaño  
DIRECTOR DEL DISTRITO 11D04 CELICA-PINDAL-PUYANGO SALUD  
Alamor

Le saluda cordialmente Sandro Michael Córdova Carrión con CI. 1150261905, estudiante de la carrera de “Ingeniería en sistemas/computación” de la Universidad Nacional de Loja.

Luego de la presentación y validación del correcto funcionamiento del “SISTEMA DE INFORMACION SOBRE COVID-19 MEDIANTE LLAMADAS TELEFONICAS CON LA AVUDA DE UN AGENTE INTELIGENTE”, le envío el presente como solicitud para la implementación del mismo, al centro de llamadas telefónicas o centro de información de los diferentes establecimientos médicos que usted dirige. La implementación es la etapa final de mi proyecto de titulación previo a la obtención del título “Ingeniero en sistemas” y consta en un módulo de software que se encargue de recibir las llamadas telefónicas de los pacientes y en base a sus síntomas, los clasifique y realice determinadas actividades como el agendamiento de citas médicas acorde a los síntomas del paciente.

Desde ya estoy muy agradecido por el tiempo concedido y espero su pronta respuesta. Reciba mis sinceros saludos y éxitos en sus funciones.

Atentamente,



Sandro Michael Córdova Carrión  
1150261905

*Figura 1. Oficio solicitud.*

La Figura 2 y Figura 3 contiene la respuesta del **Dr. Victor Hugo Tinoco Montaño** para la implementación del proyecto “**Diseño de un sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente**” al **DIRECTOR DISTRITAL 11D04 CELICA-PINDAL-PUYANGO SALUD.**

 <p>República del Ecuador</p>	<p><b>Ministerio de Salud Pública</b> Coordinación Zonal 7 Salud Dirección Distrital 11D04 Celica-Pindal-Puyango Salud</p>
<p>Oficio Nro. MSP-CZ7-DDS-11D04-C-P-PS-2022-0118-O Alamor, 14 de julio de 2022</p>	
<p><b>Asunto:</b> Dando respuesta a solicitud de implementación de proyecto de titulación "SISTEMA DE INFORMACIÓN SOBRE COVID-19 MEDIANTE LLAMADAS TELEFÓNICAS CON LA AYUDA DE UN AGENTE INTELIGENTE"</p>	
<p>Señor Sandro Michael Córdova Carrión En su Despacho</p>	
<p>De mi consideración:</p>	
<p>Tengo el agrado de dirigirme a usted, con la finalidad de expresarle un cordial y afectuoso saludo, y a la vez desearle éxitos en sus funciones diarias.</p>	
<p>Dando respuesta al oficio de fecha 12 de julio de 2022, suscrito por su persona, y receptado en Ventanilla Única de Atención al Usuario el día 14 de Julio de 2022, en el cual se solicita la implementación de su proyecto de titulación " SISTEMA DE INFORMACIÓN SOBRE COVID-19 MEDIANTE LLAMADAS TELEFÓNICAS CON LA AYUDA DE UN AGENTE INTELIGENTE" en el Distrito 11D04 Celica-Pindal-Puyango-Salud, siento informarle que por el momento no tenemos la posibilidad de realizar la implementación de dicho sistema dentro de la Institución, ya que los recursos son limitados y su implementación incluiría gastos que van fuera del presupuesto asignado.</p>	
<p>Particular que informo para los fines pertinentes.</p>	
<p>Con sentimientos de distinguida consideración.</p>	
<p>Atentamente,</p>	
<p> VICTOR HUGO TINOCO MONTAÑO</p>	
<p>Dr. Victor Hugo Tinoco Montaño <b>DIRECTOR DISTRITAL 11D04 CELICA-PINDAL-PUYANGO SALUD</b></p>	
<p>Referencias: - MSP-CZ7-DDS-11D04-VENT-2022-0083-E</p>	
<p><small>Dirección: Río Amazonas y Guayaquil. Código Postal: 110406 / Loja - Ecuador Teléfono: 593-2- 681-158 / 680-617 www.salud.gob.ec</small></p>	
<p><small>* Documento generado por Quipus</small></p>	
<p> Juntos lo logramos 1/2</p>	

Figura 2. Respuesta oficio.



**Ministerio de Salud Pública**  
Coordinación Zonal 7 Salud  
Dirección Distrital 11D04 Celica-Pindal-Puyango Salud

Oficio Nro. MSP-CZ7-DDS-11D04-C-P-PS-2022-0118-O

Alamor, 14 de julio de 2022

Anexos:  
- solicitud\_implementación\_...\_sandro\_córdova.pdf

Dirección: Río Amazonas y Guayaquil. Código Postal: 110406 / Loja - Ecuador  
Teléfono: 593-2- 681-158 / 680-617 www.salud.gob.ec

\* Documento generado por Quipus



*Figura 3. Respuesta oficio.*

**Anexo 13. Certificado de traducción del resumen**

**CERTIFICADO DE TRADUCCIÓN**

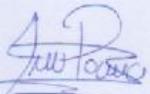
Yo, Fernando Israel Poma Riera, identificada con el número de cédula 1104052210, licenciado en ciencias de la educación mención inglés.

**CERTIFICO:**

Que el texto traducido al idioma inglés que compone el Resumen del trabajo de titulación, **"Sistema de información sobre Covid-19 mediante llamadas telefónicas con la ayuda de un agente inteligente"** de la autoría del estudiante, **Sandro Michael Córdova Carrión** con cédula de identidad **1150261905**, fue realizado y verificado bajo mi supervisión.

Loja, 15 de marzo de 2023

Atentamente,



Lic. Fernando Israel Poma Riera

C.I. 1104052210