

ESCUELA MILITAR DE INGENIERIA  
“Mcal. ANTONIO JOSÉ DE SUCRE”  
BOLIVIA

## TRABAJO DE GRADO



APLICACIÓN WEB PROGRESIVA PARA GESTIONAR LA  
ATENCIÓN MÉDICA DE PACIENTES EN EL CENTRO MÉDICO  
SALUDAME S.R.L.

EST. ARIEL TELLEZ TORRICO

SANTA CRUZ DE LA SIERRA, 2024

**ESCUELA MILITAR DE INGENIERIA**  
“Mcal. ANTONIO JOSÉ DE SUCRE”  
**BOLIVIA**

**TRABAJO DE GRADO**

**APLICACIÓN WEB PROGRESIVA PARA GESTIONAR LA  
ATENCIÓN MÉDICA DE PACIENTES EN EL CENTRO MÉDICO  
SALUDAME S.R.L.**

**EST. ARIEL TELLEZ TORRICO**

Modalidad: Proyecto de grado  
presentado como requisito para  
optar al título de Licenciatura en  
Ingeniería de Sistemas.

**TUTOR: ING. YOELMA YOMAR MELENDRES FLORES**

**SANTA CRUZ DE LA SIERRA**

**CITE :** ING. DE SISTEMAS Nº 021/2024  
**OBJETO :** Aprobación de Temario del  
Trabajo de Grado.  
**ANEXOS :** Temario 3 Hojas

Santa Cruz, 09 de Octubre de 2024

Señor  
Ariel Tellez Torrico  
Presente.

De mi consideración:

En atención a su solicitud y de conformidad a lo dispuesto por el Reglamento RAC-02 "GRADUACIÓN DE GRADO" CAPÍTULO VII "DESARROLLO DEL TRABAJO DE GRADO Y TRABAJO DE GRADO TÉCNICO" Art. 64° REMISIÓN DE DOCUMENTOS A LA DIRECCIÓN DE UNIDAD ACADÉMICA, tengo a bien comunicar a usted que ha sido aprobado el Temario para su Trabajo de Grado, cuyo título es "**APLICACIÓN WEB PROGRESIVA PARA GESTIONAR LA ATENCIÓN MÉDICA DE PACIENTES EN EL CENTRO MÉDICO SALUDAME S.R.L.**".

El plazo para la presentación del mencionado Trabajo de Grado es de (2) dos Semestres Académicos a partir del inicio del 10mo (Décimo) Semestre; vencido dicho plazo el tema quedará caduco y sin efecto.

El trabajo deberá ser presentado, de conformidad a las **Normas para la Presentación de Trabajos de Grado** de la Escuela Militar de Ingeniería sujetándose a las disposiciones vigentes.

Con este motivo, saludo a usted atentamente.



/ACSR/rfc.-  
/Cc.Arch.-

**CITE** : ING. DE SISTEMAS N° 021/2024

**OBJETO** : Aprobación de Temario del  
Trabajo de Grado.

**ANEXOS**: Temario 3 Hojas

**CARRERA**

: INGENIERÍA DE SISTEMAS

**TEMA**

: "APLICACIÓN WEB PROGRESIVA PARA  
GESTIONAR LA ATENCIÓN MÉDICA DE  
PACIENTES EN EL CENTRO MÉDICO SALUDAME  
S.R.L."

**DOCENTE TUTOR**

: Ing. Yoelma Yomar Melendres Flores

**PROPONENTE**

: Est. Ariel Tellez Torrico

**FECHA**

: 09 de octubre de 2024

**TEMARIO**

**RESUMEN EJECUTIVO**

**ABSTRACT**

**FICHA TÉCNICA**

**CAPÍTULO 1. GENERALIDADES**

1.1 INTRODUCCIÓN

1.2 ANTECEDENTES

1.3 PLANTEAMIENTO DEL PROBLEMA

1.3.1 Identificación del Problema

1.3.2 Análisis Causa Efecto

1.3.3 Formulación del problema

1.4 OBJETIVOS Y ACCIONES

1.4.1 Objetivo General

1.4.2 Objetivos Específicos

1.4.3 Acciones

1.5 JUSTIFICACIÓN

1.5.1 Justificación Social

1.5.2 Justificación Económica

1.5.3 Justificación Técnica

**CITE** : ING. DE SISTEMAS N° 021/2024  
**OBJETO** : Aprobación de Temario del  
Trabajo de Grado.  
**ANEXOS**: Temario 3 Hojas

#### 1.6 ALCANCE

- 1.6.1 Alcance Temático
- 1.6.2 Alcance Geográfico
- 1.6.3 Alcance Temporal

#### CAPÍTULO 2. MARCO TEÓRICO

- 2.1 ESQUEMA DEL MARCO TEÓRICO
- 2.2 CONTENIDO DEL MARCO TEÓRICO
- 2.3 DESARROLLO DEL MARCO TEÓRICO
  - 2.3.1 Metodología de la investigación
  - 2.3.2 Arquitectura de software
  - 2.3.3 Análisis y diseño de sistemas
  - 2.3.4 Lenguajes de programación
  - 2.3.5 Programación web
  - 2.3.6 Gestores de base de datos
  - 2.3.7 Visualizadores de datos
  - 2.3.8 Evaluación y preparación de proyectos

#### CAPÍTULO 3. MARCO PRÁCTICO

- 3.1 DISEÑO METODOLÓGICO
  - 3.1.1 Tipo y método de investigación
- 3.2 INGENIERÍA DEL PROYECTO
  - 3.2.1 Diagnóstico de los procesos de pago y atención médica de pacientes en los consultorios.
  - 3.2.2 Selección del modelo de proceso y las herramientas de software para el desarrollo de la aplicación.
  - 3.2.3 Diseño de los requerimientos de la aplicación.
  - 3.2.4 Implementación de los componentes de la aplicación.

**CITE : ING. DE SISTEMAS N° 021/2024**  
**OBJETO : Aprobación de Temario del**  
Trabajo de Grado.  
**ANEXOS: Temario 3 Hojas**

**3.3 ANÁLISIS DE VIABILIDAD**

3.3.1 Viabilidad Técnica

3.3.2 Viabilidad Económica

**CAPITULO 4. CONCLUSIONES Y RECOMENDACIONES**

4.1 CONCLUSIONES

4.2 RECOMENDACIONES

**BIBLIOGRAFÍA**

**GLOSARIO**

**ANEXOS**

Cap. DIM. Aníbal C. Sansusthy Rodríguez Ch. DAEN. Marco Antonio Guisbert Callejas  
JEFE DE CARRERA DIRECTOR DE GRADO  
INGENIERÍA DE SISTEMAS UNIDAD ACADÉMICA SANTA CRUZ

/rfc

## HOJA DE VIDA

### 1. DATOS PERSONALES

**Nombre:** Ariel

**Apellidos:** Tellez Torrico

**Carrera:** Ingeniería de Sistemas

**Lugar de Nacimiento:** Santa Cruz – Bolivia, 27 de junio de 2000.

**Cédula de identidad:** 9667582 SC.

**Correo electrónico:** arieltellez@gmail.com

### 2. ESTUDIOS DE FORMACIÓN

**2024:** Estudiante de 5to Año de la Carrera de Ingeniería de Sistemas.

**Santa Cruz- Bolivia** Escuela Militar De Ingeniería Unidad Académica Santa Cruz.

**2018: Bachiller En Humanidades**

**Santa Cruz- Bolivia** Colegio Marista.

### 3. IDIOMAS

Castellano

Inglés

#### **4. EXPERIENCIA LABORAL**

**Tigo Bolivia**

Cargo: Pasante

Funciones: Analista de datos en el área de BI B2C.

#### **5. CURSO DE EDUCACIÓN CONTINUA**

- **Curso de inglés**

*(Escuela de Militar de Ingeniería, 2021)*

- **Instrucción Militar**

*(Escuela Militar de Ingeniería, 2019)*

## **DEDICATORIA**

Este trabajo está dedicado a:

A mis padres por haberme apoyado en cada fase del desarrollo de este proyecto, dándome su consejo desde el primer día.

A mis sobrinos, por el compromiso que tengo para brindarles un buen ejemplo y ayudarlos en lo que necesiten.

## **AGRADECIMIENTOS**

### **Expreso mis profundos agradecimientos:**

**A:** Mis padres, José Ariel Téllez Tufiño y María del Carmen Torrico Candia, por haberme permitido estudiar la carrera de Ingeniería de Sistemas y siempre brindarme el apoyo necesario cuando más lo necesité.

**A:** Mis hermanos, Mariel, Rodrigo y Natalia, por su apoyo incondicional en el desarrollo del proyecto.

**A:** Mi tutora, Ing. Yoelma Yomar Melendres Flores, por su constante orientación y apoyo en las distintas etapas del desarrollo del proyecto. Su dedicación, paciencia y experiencia fueron fundamentales para alcanzar los objetivos planteados.

**A:** Mi docente revisor, Ing. Gloria Margot Gonzales Fernández, por su disposición a revisar este trabajo con criterio; y por sus aportes significativos, los cuales ayudaron a dar forma y mejorar cada detalle del presente documento.

**A:** Mi docente de trabajo de grado, Ing. Risella Castro Ramos, por su orientación y apoyo durante el tiempo de desarrollo del proyecto. Su compromiso hacia mi persona ha sido una fuente de motivación.

**A:** Todos mis docentes, que, a lo largo de mi formación académica, me brindaron las herramientas y conocimiento necesarios para enfrentar los diversos desafíos profesionales.

**A:** Todas las personas, compañeros de clase y amigos, que de alguna manera han contribuido brindando su apoyo de forma técnica y emocional durante los años de carrera.

**A:** Todo el personal del Centro Médico SaludAME S.R.L., quienes me brindaron su tiempo y constante retroalimentación con los datos provistos en el presente proyecto.

## ÍNDICE



## ÍNDICE

|   | Pág.      |
|---|-----------|
| <b>CAPÍTULO 1. GENERALIDADES .....</b>            | <b>1</b>  |
| 1.1.    INTRODUCCIÓN .....                        | 1         |
| 1.2.    ANTECEDENTES .....                        | 2         |
| 1.2.1.    Antecedentes del Proyecto .....         | 2         |
| 1.2.2.    Antecedentes Institucionales .....      | 4         |
| 1.3.    PLANTEAMIENTO DEL PROBLEMA .....          | 5         |
| 1.3.1.    Identificación del Problema .....       | 5         |
| 1.3.2.    Análisis Causa Efecto .....             | 6         |
| 1.3.3.    Formulación del Problema .....          | 7         |
| 1.4.    OBJETIVOS Y ACCIONES .....                | 7         |
| 1.4.1.    Objetivo General .....                  | 7         |
| 1.4.2.    Objetivos Específicos .....             | 7         |
| 1.4.3.    Acciones .....                          | 7         |
| 1.5.    JUSTIFICACIÓN .....                       | 9         |
| 1.5.1.    Justificación Social .....              | 9         |
| 1.5.2.    Justificación Económica .....           | 9         |
| 1.5.3.    Justificación Técnica .....             | 9         |
| 1.6.    ALCANCE .....                             | 10        |
| 1.6.1.    Alcance Temático .....                  | 10        |
| 1.6.2.    Alcance Geográfico .....                | 11        |
| 1.6.3.    Alcance Temporal .....                  | 12        |
| <b>CAPÍTULO 2. MARCO TEÓRICO .....</b>            | <b>13</b> |
| 2.1.    ESQUEMA DE MARCO TEÓRICO .....            | 13        |
| 2.2.    CONTENIDO DEL MARCO TEÓRICO .....         | 14        |
| 2.3.    DESARROLLO DEL MARCO TEÓRICO .....        | 16        |
| 2.3.1.    Metodología de la Investigación .....   | 16        |
| 2.3.1.1.    Técnica de recolección de datos ..... | 18        |

|   |           |
|---|-----------|
| 2.3.2. Arquitectura de software .....   | 20        |
| 2.3.2.1. Tipos de arquitectura de software .....  | 22        |
| 2.3.3. Análisis y diseño de sistemas .....  | 24        |
| 2.3.3.1. Lenguaje de Modelo Unificado (UML) .....   | 24        |
| 2.3.3.2. Historias de Usuario .....   | 31        |
| 2.3.3.3. Notación de Modelado de Procesos (BPMN).....   | 33        |
| 2.3.3.4. Requisitos funcionales y no funcionales .....  | 36        |
| 2.3.4. Lenguajes de Programación .....  | 38        |
| 2.3.4.1. Programación del lado del servidor .....   | 39        |
| 2.3.4.2. Programación del lado del cliente .....  | 43        |
| 2.3.4.3. Entorno de Desarrollo Integrado (IDE) .....  | 47        |
| 2.3.5. Programación web.....  | 51        |
| 2.3.5.1. Metodologías de desarrollo de software .....   | 52        |
| 2.3.5.2. Marcos de trabajo (Frameworks) del Front End .....   | 59        |
| 2.3.5.3. Marcos de trabajo (Frameworks) del Back End .....  | 64        |
| 2.3.5.4. Calidad del software .....   | 69        |
| 2.3.6. Gestores de base de datos.....   | 71        |
| 2.3.7. Visualizadores de datos.....   | 77        |
| 2.3.8. Evaluación y preparación de proyectos .....  | 82        |
| 2.3.8.1. Viabilidad Técnica .....   | 82        |
| 2.3.8.2. Viabilidad económica.....  | 85        |
| <b>CAPÍTULO 3. MARCO PRÁCTICO.....</b>  | <b>89</b> |
| 3.1. DISEÑO METODOLÓGICO .....  | 89        |
| 3.1.1. Tipo y método de investigación .....   | 89        |
| 3.1.1.1. Técnicas e Instrumentos de Recolección de Datos .....  | 90        |
| 3.2. INGENIERÍA DEL PROYECTO .....  | 90        |
| 3.2.1. Diagnóstico de los procesos de pago y atención médica de pacientes en los consultorios.....              | 90        |
| 3.2.1.1. Requerimientos funcionales .....   | 92        |
| 3.2.1.2. Requerimientos no funcionales .....  | 93        |
| 3.2.2. Selección del modelo de proceso y las herramientas de software para el desarrollo de la aplicación ..... | 94        |

|           |   |     |
|-----------|---|-----|
| 3.2.2.1.  | Selección del Entorno de Desarrollo Integrado (IDE) .....   | 94  |
| 3.2.2.2.  | Selección de la Metodología de Desarrollo.....              | 95  |
| 3.2.2.3.  | Selección del framework del frontend .....                  | 97  |
| 3.2.2.4.  | Selección del framework del backend .....                   | 98  |
| 3.2.2.5.  | Selección del lenguaje de programación .....                | 99  |
| 3.2.2.6.  | Selección del gestor de base de datos .....                 | 100 |
| 3.2.2.7.  | Selección del visualizador de datos .....                   | 101 |
| 3.2.2.8.  | Selección de la arquitectura de software .....              | 102 |
| 3.2.3.    | Diseño de los requerimientos de la aplicación .....         | 104 |
| 3.2.3.1.  | Historias de usuario.....                                   | 104 |
| 3.2.3.2.  | Diagrama de casos de uso del sistema .....                  | 108 |
| 3.2.3.3.  | Módulo de inicio de sesión .....                            | 110 |
| 3.2.3.4.  | Módulo de administración de usuarios .....                  | 111 |
| 3.2.3.5.  | Módulo de administración de pacientes en consultorios ..... | 113 |
| 3.2.3.6.  | Módulo de administración de pagos realizados .....          | 115 |
| 3.2.3.7.  | Módulo del registro de pagos .....                          | 117 |
| 3.2.3.8.  | Diseño conceptual de la base de datos .....                 | 121 |
| 3.2.3.9.  | Diseño lógico de la base de datos .....                     | 123 |
| 3.2.4.    | Implementación de los componentes de la aplicación .....    | 124 |
| 3.2.4.1.  | Estructura del proyecto.....                                | 124 |
| 3.2.4.2.  | Elaboración del diseño físico de la base de datos .....     | 128 |
| 3.2.4.3.  | Codificación del módulo de inicio de sesión .....           | 130 |
| 3.2.4.4.  | Codificación del módulo de usuarios .....                   | 138 |
| 3.2.4.5.  | Codificación del módulo del libro de observaciones .....    | 146 |
| 3.2.4.6.  | Codificación del módulo de pagos .....                      | 149 |
| 3.2.4.7.  | Codificación del módulo de subir pagos del asegurado .....  | 150 |
| 3.2.4.8.  | Realización de pruebas funcionales .....                    | 150 |
| 3.2.4.9.  | Reportes del sistema.....                                   | 158 |
| 3.2.4.10. | Aplicación Web Progresiva .....                             | 160 |
| 3.3.      | ANÁLISIS DE VIABILIDADAD .....                              | 164 |
| 3.3.1.    | Viabilidad Técnica .....                                    | 165 |
| 3.3.1.1.  | Usabilidad .....  | 165 |

|   |     |
|---|-----|
| 3.3.1.2. Seguridad .....                    | 165 |
| 3.3.1.3. Funcionalidad .....                | 165 |
| 3.3.2. Viabilidad Económica .....           | 166 |
| 3.3.2.1. Inversión fija .....               | 166 |
| 3.3.2.2. Inversión diferida .....           | 167 |
| 3.3.2.3. Inversión total del proyecto ..... | 170 |

## **CAPÍTULO 4. CONCLUSIONES Y RECOMENDACIONES ..... 173**

|                            |     |
|----------------------------|-----|
| 4.1. CONCLUSIONES .....    | 173 |
| 4.2. RECOMENDACIONES ..... | 174 |

## **BIBLIOGRAFÍA**

## **GLOSARIO**

## **ANEXOS**

## ÍNDICE DE FIGURAS

|  | Pág. |
|--|------|
| Figura 1: Diagrama ISHIKAWA .....  | 6    |
| Figura 2: Ubicación del Centro Médico SaludAME S.R.L .....   | 12   |
| Figura 3: Esquema de Marco Teórico.....  | 13   |
| Figura 4: Ejemplo de diagrama de clases de un sistema encargado de la gestión de préstamos y reservas de libros y revistas de una biblioteca ..... | 27   |
| Figura 5: Ejemplo de diagrama de casos de uso.....   | 29   |
| Figura 6: Ejemplo de diagrama entidad – relación .....   | 31   |
| Figura 7: Ejemplo de un diagrama BPMN .....  | 36   |
| Figura 8: Ejemplo de tablero Kanban .....  | 56   |
| Figura 9: Pasos de la metodología Scrum.....   | 57   |
| Figura 10: Pasos de la metodología XP .....  | 59   |
| Figura 11: BPMN del proceso de pago anual del seguro médico .....  | 91   |
| Figura 12: BPMN del proceso de atención médica de pacientes en consultorio.....  | 92   |
| Figura 13: Historias de usuario del sistema.....   | 104  |
| Figura 14: Diagrama de casos de uso del sistema .....  | 109  |
| Figura 15: Diagrama de caso de uso del módulo de inicio de sesión .....  | 110  |
| Figura 16: Wireframe de la pantalla de inicio de sesión .....  | 111  |
| Figura 17: Diagrama de casos de uso del módulo de administración de usuarios...112   | 112  |
| Figura 18: Wireframe de la pantalla principal de la gestión de usuarios .....  | 112  |
| Figura 19: Wireframe de la pantalla para agregar usuarios .....  | 113  |
| Figura 20: Diagrama de caso de uso del módulo de administración de pacientes atendidos .....   | 114  |
| Figura 21: Wireframe de la pantalla principal del módulo de administración de pacientes .....  | 114  |
| Figura 22: Wireframe de la pantalla para agregar registro de paciente atendido.....115   | 115  |
| Figura 23: Wireframe de la pantalla principal de la administración de pagos realizados.  | 116  |
| Figura 24: Wireframe de la búsqueda de asegurados .....  | 118  |
| Figura 25: Wireframe de la pantalla para agregar asegurados.....119  | 119  |

|   |     |
|---|-----|
| Figura 26: Wireframe de la pantalla para cancelar el seguro.....                    | 119 |
| Figura 27: Wireframe de la página de registro de datos de contacto .....            | 120 |
| Figura 28: Wireframe de la página de finalización del proceso.....                  | 120 |
| Figura 29: Wireframe de la página para descargar recibos .....                      | 121 |
| Figura 30: Diagrama conceptual de la base de datos.....                             | 122 |
| Figura 31: Diagrama lógico de la base de datos .....                                | 123 |
| Figura 32: Estructura del backend.....  | 124 |
| Figura 33:Estructura del frontend .....   | 125 |
| Figura 34: Estructura de la carpeta src del frontend .....                          | 127 |
| Figura 35: Archivo server.js con la conexión a la base de datos.....                | 129 |
| Figura 36: Estructura lógica del componente Login.jsx .....                         | 130 |
| Figura 37: Estructura visual del componente Login.jsx.....                          | 132 |
| Figura 38: Parte del backend del componente de iniciar sesión.....                  | 134 |
| Figura 39: Inicio de sesión .....   | 135 |
| Figura 40: Estructura lógica de la página inicial .....                             | 136 |
| Figura 41: Mockup de la página inicial .....  | 137 |
| Figura 42: Backend del componente de la página inicial .....                        | 138 |
| Figura 43: Estructura lógica del componente principal de usuarios .....             | 139 |
| Figura 44: Mockup del componente principal de usuarios .....                        | 140 |
| Figura 45: Backend para mostrar y eliminar a los usuarios registrados.....          | 141 |
| Figura 46: Estructura lógica del componente Register.jsx .....                      | 142 |
| Figura 47: Mockup del componente para registrar usuario.....                        | 143 |
| Figura 48: Backend para la inserción de usuarios .....                              | 144 |
| Figura 49: Mockup de la actualización de usuario .....                              | 144 |
| Figura 50: Backend para obtener los datos del usuario seleccionado .....            | 145 |
| Figura 51: Backend para actualizar datos del usuario .....                          | 145 |
| Figura 52: Mockup del componente del libro de observaciones .....                   | 146 |
| Figura 53: Formulario para la inserción de datos en el libro de observaciones ..... | 147 |
| Figura 54: Backend para buscar al cliente en la base de datos.....                  | 148 |
| Figura 55: Backend para insertar datos en libro de observaciones .....              | 148 |
| Figura 56: Mockup del componente que muestra los pagos realizados .....             | 149 |

|   |     |
|---|-----|
| Figura 57: Backend para mostrar la tabla de pagos realizados .....                    | 150 |
| Figura 58: Prueba funcional al backend .....  | 151 |
| Figura 59: Administrador: página inicial.....   | 152 |
| Figura 60: Administrador: usuarios registrados .....                                  | 153 |
| Figura 61: Médico, libro de consultas médicas.....                                    | 153 |
| Figura 62: Secretaria, pagos realizados .....   | 154 |
| Figura 63: Asegurado: Buscarse en la base de datos .....                              | 154 |
| Figura 64: Asegurado, agregar a la base de datos .....                                | 155 |
| Figura 65: Asegurado: Subir imagen de la transferencia .....                          | 156 |
| Figura 66: Asegurado: registrar datos de contacto .....                               | 156 |
| Figura 67: Asegurado, pantalla de finalización del proceso .....                      | 157 |
| Figura 68: Asegurado, pantalla para descargar recibo .....                            | 157 |
| Figura 69: Reporte de pagos realizados.....   | 158 |
| Figura 70: Reporte del monto generado en base a los pagos realizados .....            | 159 |
| Figura 71: Reporte de consultas atendidas .....                                       | 159 |
| Figura 72: Archivo manifest.json .....  | 160 |
| Figura 73: Mensaje que muestra la posibilidad de instalar la aplicación.....          | 161 |
| Figura 74: Archivo index.html, registro del uso del service worker.....               | 161 |
| Figura 75: Consola de desarrollador, service worker funcionando.....                  | 162 |
| Figura 77: Uso del service worker para la funcionalidad offline de la aplicación..... | 164 |

## ÍNDICE DE TABLAS

|  | <b>Pág.</b> |
|--|-------------|
| Tabla 1: Acciones de Investigación.....                              | 8           |
| Tabla 2: Alcance temático.....                                       | 10          |
| Tabla 3: Fundamentación teórica .....                                | 14          |
| Tabla 4: Marco metodológico .....                                    | 89          |
| Tabla 5: Comparación de Entornos de Desarrollo Integrado (IDE) ..... | 95          |
| Tabla 6: Comparación de metodologías de desarrollo.....              | 96          |
| Tabla 7: Comparación de frameworks frontend .....                    | 97          |
| Tabla 8: Comparación de frameworks backend.....                      | 98          |
| Tabla 9: Comparación de gestores de bases de datos .....             | 100         |
| Tabla 10: Comparación de visualizadores de datos .....               | 101         |
| Tabla 11: Historia de usuario 1 .....                                | 105         |
| Tabla 12: Historia de usuario 2.....                                 | 105         |
| Tabla 13: Historia de usuario 3.....                                 | 106         |
| Tabla 14: Historia de usuario 4 .....                                | 106         |
| Tabla 15: Historia de usuario 5 .....                                | 107         |
| Tabla 16: Historia de usuario 6 .....                                | 108         |
| Tabla 17: Costos del hardware .....                                  | 166         |
| Tabla 18: Costo de licencias.....                                    | 167         |
| Tabla 19: Costos del servicio de hosting web .....                   | 168         |
| Tabla 20: Costo de capacitación .....                                | 168         |
| Tabla 21: Cálculo de Story Points.....                               | 169         |
| Tabla 22: Inversión total del proyecto .....                         | 171         |

## ÍNDICE DE ECUACIONES

|  | Pág. |
|--|------|
| ECUACIÓN 1: Ecuación del coste de un Story Point .....                     | 88   |
| ECUACIÓN 2: Ecuación del costo estimado del total de las iteraciones ..... | 170  |
| ECUACIÓN 3: Ecuación de inversión total del proyecto .....                 | 171  |

## **ÍNDICE DE ANEXOS**

ANEXO “A”: INFORMACIÓN REGISTRADA A MANO

ANEXO “B”: INFORMACIÓN REGISTRADA EN HOJAS DE EXCEL

ANEXO “C”: ENTREVISTA PRELIMINAR AL GERENTE GENERAL DEL CENTRO  
MÉDICO SALUDAME S.R.L.

ANEXO “D”: SEGURIDAD Y USABILIDAD IMPLEMENTADA

## RESUMEN EJECUTIVO



## RESUMEN EJECUTIVO

El presente trabajo de grado tiene como propósito el desarrollo de una Aplicación Web Progresiva (PWA) con el objetivo de automatizar la gestión de pagos y la atención médica de los pacientes en los consultorios del Centro Médico SaludAME S.R.L. La problemática identificada radica en la utilización de procesos manuales para registrar y gestionar estos servicios, los cuales presentan una alta propensión a errores, requieren una considerable cantidad de tiempo, y ocasionan pérdidas de información, lo que repercute negativamente en la calidad del servicio prestado, por lo cual se desarrolla un sistema funcional que permite optimizar de manera significativa el proceso de registro de pagos y atenciones médicas de los pacientes, la generación de informes y el seguimiento en tiempo real de los casos atendidos.

La implementación de la PWA facilita un control más eficiente de los procesos, reduce los errores y mejora la toma de decisiones, al contar con datos más precisos y fácilmente accesibles. En cuanto a las tecnologías utilizadas en el desarrollo del proyecto se emplearon herramientas de software libre como JavaScript, HTML, CSS, junto con frameworks como React.js en el frontend y Express.js en el backend, todo ello con una base de datos gestionada en MySQL complementando con reportes realizados en Power BI. El uso de service workers permite la funcionalidad offline de la aplicación, mientras que la metodología Kanban fue adoptada para la gestión ágil del proyecto.

De esta manera se logró desarrollar el sistema, contribuyendo al Centro Médico SaludAME S.R.L. optimizando los procesos de atención médica en los consultorios y la gestión de pagos. Este avance no solo contribuyó a una mayor satisfacción de los asegurados, sino que también generó un menor impacto ambiental al eliminar el uso de papel en los procesos administrativos.

Palabras clave: aplicación web progresiva, sistema web, centralización de datos, Power BI, Kanban, React.

## ABSTRACT

The present degree work aims to develop a Progressive Web Application (PWA) with the goal of automating the management of payments and medical care for patients in the consultancy of Centro Médico SaludAME S.R.L. The identified problem lies in the use of manual processes to record and manage these services, which are highly prone to errors, require a considerable amount of time, and result in information loss, negatively affecting the quality of the service provided. Therefore, a functional system is developed that significantly improves the processes of registering payments and medical care for patients, generating reports, and tracking cases in real-time. The implementation of the PWA facilitates more efficient control of processes, reduces errors, and improves decision-making by providing more precise and easily accessible data.

About the technologies used in the project development, free software tools such as JavaScript, HTML, and CSS were employed, along with frameworks like React.js for the frontend and Express.js for the backend, all complemented by a MySQL-managed database and reports generated in Power BI. The use of service workers enables offline functionality of the application, while the Kanban method was adopted for the agile management of the project. This approach led to the development of the system, contributing to Centro Médico SaludAME S.R.L. by optimizing medical care processes in consultancy and payment management. This advancement not only contributed to greater satisfaction among insured patients but also generated a lower environmental impact by dropping the use of paper in administrative processes.

Keywords: Progressive web app, web system, data centralization, Power BI, Kanban, React.

## FICHA TÉCNICA



**ESCUELA MILITAR DE INGENIERIA**

*“Mcal. Antonio José de Sucre”*

**FICHA DE INVESTIGACIÓN**

1. **Título del proyecto:** APPLICACIÓN WEB PROGRESIVA PARA GESTIONAR LA ATENCIÓN MÉDICA DE PACIENTES EN EL CENTRO MÉDICO SALUDAME S.R.L.

2. **Objetivo:** Desarrollar una Aplicación Web Progresiva para automatizar el registro de pagos y atención médica de pacientes en los consultorios del Centro Médico SaludAME S.R.L.

3. **Tipo de proyecto:**

○ Investigación Aplicada:  Investigación Básica:

4. **Área de investigación:** Gestión del conocimiento y nuevas tecnologías.

5. **Línea de investigación:** Sistemas de información, programación y desarrollo de software.

6. **Carrera:** Ingeniería en Sistemas

7. **Unidad Académica:** Santa Cruz

8. **Periodo Académico:** I - II - 2024

9. **Gestión:** 2024

10. **Nombre del Estudiante:** Ariel Tellez Torrico

11. **Nombre del Tutor:** Ing. Yoelma Yomar Melendres Flores

# CAPÍTULO I

## GENERALIDADES



## **CAPÍTULO 1.**

### **GENERALIDADES**

#### **1.1. INTRODUCCIÓN**

En el contexto actual, donde la tecnología juega un papel cada vez más importante en todos los aspectos de nuestra vida, la atención médica no es una excepción. La gestión eficaz de la atención médica en los consultorios escolares es una necesidad imperante. Los registros manuales y los sistemas de seguimiento tradicionales resultan ineficientes, están propensos a errores y modificaciones no autorizados, lo que puede llevar a una atención médica ineficiente, pérdidas tanto de información como económicas.

El Centro Médico SaludAME S.R.L es una empresa que ofrece un seguro de atención médica contra accidentes, el cual se paga anualmente. Su principal objetivo es brindar servicios de salud a los estudiantes de los colegios afiliados, garantizando una atención oportuna y eficiente en caso de accidentes durante las actividades escolares.

En el presente proyecto, se propone el desarrollo de una Aplicación Web Progresiva (PWA – Progressive Web Apps, por sus siglas en inglés), esta ofrece ventajas significativas sobre las aplicaciones web y tradicionales al ser accesible desde cualquier navegador y no requerir descargar una aplicación. Su diseño responsivo

asegura una experiencia de usuario consistente, más fluida y rápida en cualquier dispositivo, además de funcionar en modo offline y actualizarse automáticamente.

Esta aplicación permitirá llevar un control adecuado de los pacientes atendidos; buscará optimizar los procesos de atención médica, permitiendo un seguimiento en tiempo real de los casos atendidos, la generación de informes y la posibilidad de realizar análisis de datos para mejorar la calidad del servicio.

## **1.2. ANTECEDENTES**

### **1.2.1. Antecedentes del Proyecto**

La necesidad de registrar la información de los pacientes en los centros médicos ha sido un problema antiguo e importante en la Medicina. Con la aparición del papel y el lápiz, se pudieron solventar muchos de los problemas relacionados con el registro y consulta de datos de los pacientes. No obstante, con la aparición de la informática estos problemas se han resuelto en su totalidad evitando así el excesivo consumo de papel en la mayoría de los países del mundo. Sin embargo, en nuestro país se sigue evidenciando el uso del sistema de recogida de información en papel y lápiz en muchos centros médicos y hospitalares.

Como muestra del avance de la informática en la gestión de atención médica a pacientes a continuación, se describe proyectos de investigación realizados tanto a nivel internacional, nacional y local.

A nivel internacional, se realizó el proyecto de investigación titulado: “Sistema de Información Web para el Mejor Control y Acceso a las Historias Clínicas de los Pacientes del Centro de Salud Jequetepeque.” (Pairazaman Esteves & Vigo Escalante, 2017), realizado por estudiantes para optar al título de Licenciatura en Ing. de Sistemas, de la Facultad de Ingeniería de la Universidad Nacional de Trujillo, Perú. El sistema aplica la metodología de desarrollo de software RUP y utiliza las tecnologías

PHP, MySQL, HTML y JavaScript. Cuenta con los módulos para registrar pacientes, generar reportes y administración de usuarios.

A nivel nacional, el proyecto de investigación titulado “Sistema de control y gestión de historiales clínicos apoyado en dispositivos móviles, caso: Centro Médico La Paz” (Centellas Coarite, 2015), desarrollado en la Facultad De Ciencias Puras Y Naturales, Carrera De Informática de la Universidad Mayor De San Andrés. Describe el diseño e implementación del sistema, que permite mejorar las tareas de registro, consulta y recordatorio de la información clínica de los pacientes. El sistema se basa en el estándar HL7 CDA para la estructura de los documentos clínicos y utiliza la tecnología web y móvil para facilitar el acceso y la comunicación. El modelo de proceso utilizado fue la metodología UML (Lenguaje Unificado de Modelo, por su abreviatura) y utiliza una página web para el registro de usuarios del sistema y una aplicación móvil nativa para el manejo de las historias clínicas.

A nivel local, el proyecto de investigación titulado: “Desarrollo de un sistema para el control y seguimiento del historial clínico de pacientes asegurados a la Corporación de Seguro Social Militar (COSSMIL) Regional Santa Cruz” (Ramos Duran, 2022), desarrollada en la Escuela Militar de Ingeniería Unidad Académica Santa Cruz. El modelo de proceso fue un modelo vista controlador, y utilizando una página web desarrollada con PHP y MySQL, utilizando Laravel como framework para el registro de pacientes asegurados al COSSMIL.

Cada una de las tecnologías usadas en los proyectos mencionados tiene sus ventajas y desventajas: las aplicaciones móviles nativas tienen la ventaja de acceder al hardware del dispositivo en su totalidad, son rápidas y funcionan sin conexión a Internet; sus desventajas son que se requiere más trabajo dado a que es necesario reescribir el código para dos plataformas diferentes y es muy costoso. Por otro lado, las aplicaciones web son almacenadas en un servidor y son multiplataforma, son fáciles de programar ya que un mismo código sirve para todas las plataformas y son mucho más baratas de desarrollar comparado a una aplicación nativa; sin embargo, tiene como desventaja el hecho de requerir siempre una conexión a Internet estable,

no tiene acceso al hardware del dispositivo y tiene una falta de visibilidad dado a que no aparece en las tiendas de aplicaciones de los teléfonos inteligentes.

Un enfoque que se ha visto recientemente son las aplicaciones web progresivas (Progressive Web Apps, por su traducción del inglés). Según (SOSSA, 2019), “Las aplicaciones web progresivas, es un término de un nuevo tipo de aplicaciones que acerca a la unificación de aplicaciones web-nativas, incrementando su funcionalidad, conforme las capacidades del dispositivo en el que se ejecuta, de ahí la palabra progresiva, web porque se hace referencia a su desarrollo basado en tecnologías web.”. De acuerdo con (Tandel & Jamadar, 2018), “Las aplicaciones web progresivas funcionan para todos los usuarios, independientemente del navegador o el sistema operativo que utilicen, y son responsivas, adaptándose a cualquier tipo de dispositivo, ya sea de escritorio, móvil o tableta. Gracias al uso de los Service Workers y la caché, las aplicaciones web progresivas pueden trabajar sin conexión o con redes de baja calidad. Además, se pueden añadir a la pantalla de inicio del dispositivo sin necesidad de pasar por una tienda de aplicaciones, son descubribles por los motores de búsqueda ya que están construidas con tecnologías web estándar, pueden enviar notificaciones push para mantener a los usuarios involucrados con el contenido, y se sirven a través de HTTPS, garantizando que no puedan ser manipuladas por terceros. Todo esto hace que las aplicaciones web progresivas sean una opción atractiva para muchos desarrolladores y empresas”. Así se ve la importancia de las aplicaciones web progresivas en la unificación del desarrollo web y el desarrollo de aplicaciones móviles nativas, combinando sus ventajas y permitiendo una mejor interacción entre el usuario y el sistema.

### **1.2.2. Antecedentes Institucionales**

SaludAME S.R.L es una empresa de servicios médicos que brinda asistencia médica de emergencia contra accidentes en su Centro Médico. El doctor José Gary Vásquez Ribera la fundó el 06 de junio del 2008, para ofrecer un servicio integral en salud a los pacientes. Actualmente, la empresa cuenta con consultorios habilitados en cada una

de las 21 instituciones afiliadas de Santa Cruz de la Sierra, bajo modalidad de seguro anual cancelado a principios de cada gestión, con una base anual variable de 50 000 asegurados.

Actualmente el proceso para recibir atención médica en los consultorios es el siguiente: en las instituciones los asegurados tienen que realizar el pago por concepto de seguro anual al número de cuenta de SaludAME y enviar su comprobante de pago por WhatsApp para su registro, que se realiza de manera manual. Una vez realizado el pago, cuando el asegurado se accidenta en la institución, es llevado al consultorio habilitado dentro de la misma, donde el médico lo asiste, anotando las observaciones con respecto al paciente en un libro de forma manual, que se entrega al centro médico a final de mes. Ver figura 11 en el anexo. Si el paciente requiere asistencia médica especializada, se le brinda los primeros auxilios y se lo deriva a otro centro médico con la especialidad requerida.

Mensualmente, en su casa matriz, SaludAME realiza el registro en planillas de Excel del total de pacientes atendidos en cada consultorio médico, el total de pacientes atendidos por diagnóstico, y el número de pacientes derivados a otros centros médicos, en base al libro de observaciones. Ver figura 12, 13 y 14 en el anexo. Al no estar automatizado este proceso, se genera un costo en tiempo, esfuerzo y materiales por parte del personal de SaludAME.

### **1.3. PLANTEAMIENTO DEL PROBLEMA**

#### **1.3.1. Identificación del Problema**

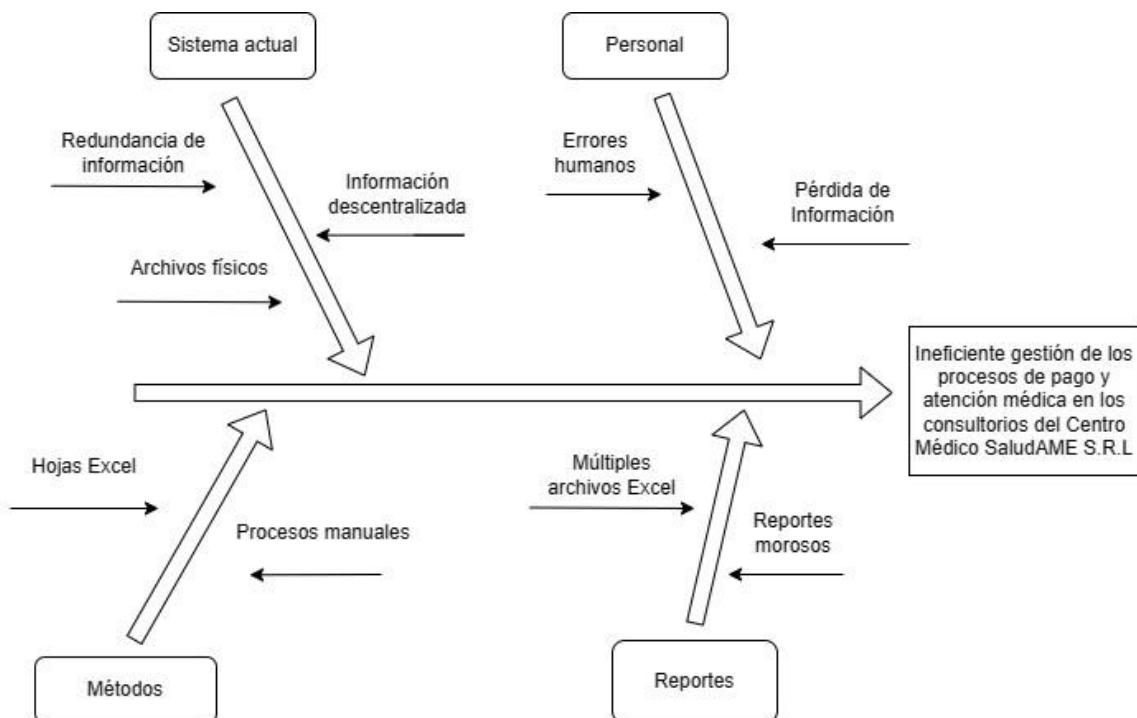
Para el registro de los pacientes que asisten a los consultorios, los correspondientes procedimientos médicos (diagnóstico y tratamiento) y las cancelaciones por concepto de pago anual del seguro de las instituciones afiliadas, se utilizan registros manuales, que luego son digitalizados a hojas de Excel. Los reportes se generan cada mes, por lo tanto, el personal requiere varias horas de trabajo para consolidar la información para generar las estadísticas necesarias.

Al ser la consolidación de la información un proceso manual, esta genera errores involuntarios por parte del personal de SaludAME tales como la pérdida de hojas, la falta de legibilidad en la escritura de médicos en el caso de registros manuales; modificaciones no autorizadas y errores de escritura en la computadora en el caso de las hojas de Excel; lo cual produce en el peor de los casos pérdida de información y consecuentemente no refleja datos reales, creando errores en el control de los servicios ofertados tanto en la recaudación como en el registro de pacientes y medicamentos utilizados.

### 1.3.2. Análisis Causa Efecto

A continuación, en la Figura 1, se presenta el Diagrama Causa y Efecto.

**Figura 1: Diagrama ISHIKAWA**



*Fuente: Elaboración propia*

### **1.3.3. Formulación del Problema**

¿Cómo automatizar la gestión de pagos y atención médica de pacientes en los consultorios del Centro Médico SaludAME S.R.L.?

## **1.4. OBJETIVOS Y ACCIONES**

### **1.4.1. Objetivo General**

Desarrollar una Aplicación Web Progresiva para automatizar el registro de pagos y atención médica de pacientes en los consultorios del Centro Médico SaludAME S.R.L.

### **1.4.2. Objetivos Específicos**

- Diagnosticar los procesos de pago y atención médica de pacientes en los consultorios.
- Seleccionar el modelo de proceso y las herramientas de software para el desarrollo de la aplicación.
- Diseñar los requerimientos de la aplicación.
- Implementar los componentes de la aplicación.

### **1.4.3. Acciones**

En la presente tabla, se evidencian las acciones de investigación para cada objetivo específico correspondiente.

**Tabla 1: Acciones de Investigación**

| OBJETIVOS ESPECIFICOS  | ACCIONES DE INVESTIGACIÓN  |
|--|--|
| Diagnosticar los procesos de pago y atención médica de pacientes en los consultorios.                | Recolectar los requerimientos de la aplicación.  |
|  | Modelar los procesos de pago y atención médica de pacientes.                             |
|  | Describir los requerimientos funcionales y no funcionales de la aplicación.              |
| Seleccionar el modelo de proceso y las herramientas de software para el desarrollo de la aplicación. | Elegir la metodología de desarrollo de software a utilizar a lo largo del proyecto.      |
|  | Escoger el lenguaje de programación y el entorno de desarrollo.                          |
|  | Determinar el sistema gestor de base de datos a utilizar.                                |
|  | Escoger la arquitectura de software a ser aplicada.                                      |
| Diseñar los requerimientos de la aplicación.   | Analizar los requerimientos de la aplicación.  |
|  | Modelar los componentes de la aplicación.  |
|  | Realizar el diseño conceptual y lógico de la base de datos.                              |
| Implementar los componentes de la aplicación.  | Elaborar el diseño físico de la base de datos.   |
|  | Codificar el módulo de atención médica en consultorio.                                   |
|  | Codificar el módulo de pago del seguro médico.   |
|  | Realizar pruebas funcionales.  |
|  | Generar los reportes de la aplicación utilizando herramientas de visualización de datos. |

*Fuente: Elaboración propia*

## **1.5. JUSTIFICACIÓN**

### **1.5.1. Justificación Social**

La administración de SaludAME se beneficiará con la implementación del proyecto para poder visualizar rápidamente el número de estudiantes atendidos en los consultorios, así como el monto generado y gastado de cada institución y paciente, además de evitar el impacto ambiental que produce el excesivo consumo de papel. El personal médico de los consultorios se verá beneficiado con el proyecto evitando la fatiga al escribir, y así contar con más tiempo para poder atender a los pacientes; y los pacientes se verán beneficiados por una atención médica más eficiente dado a lo mencionado previamente.

### **1.5.2. Justificación Económica**

El sistema se realizará utilizando software libre, permitiendo el ahorro de costes en licencias y mantenimiento.

Por otro lado, desde el punto de vista económico, se ahorrarán costos de operación y de recursos humanos para la empresa a la hora de generar reportes e introducir datos de manera manual de forma repetitiva, evitando la contratación de personal temporal durante el periodo del registro a inicios de gestión.

### **1.5.3. Justificación Técnica**

Desde el punto de vista técnico, la empresa dispondrá de un sistema flexible y escalable, utilizando procesos innovadores y aplicando nuevas tecnologías para resolver los problemas existentes.

Aprovechando las ventajas de las Aplicaciones Web Progresivas, en los consultorios se podrá acceder al sistema sin importar la plataforma desde la cual se ingrese.

Hoy, cada médico tiene su smartphone por el que puede ingresar a la aplicación para el registro de consultas médicas. Para el registro de pagos en la oficina central, se cuenta con 3 computadoras de escritorio.

## 1.6. ALCANCE

### 1.6.1. Alcance Temático

Las áreas de investigación que abordara el siguiente proyecto son: Metodología de Investigación, Ingeniería de software, Programación orientada a objetos, Análisis y Diseño de Sistemas II, Base de datos, Ingeniería web y Programación Avanzada.

A continuación, en la Tabla 2 se presentan los temas a utilizar para la Elaboración del Proyecto de Grado:

**Tabla 2: Alcance temático**

| ÁREAS DE INVESTIGACIÓN           | TEMA  |
|----------------------------------|---|
| Metodología de Investigación     | <ul style="list-style-type: none"><li>• Técnicas de relevamiento de requisitos.</li></ul>   |
| Ingeniería de Software           | <ul style="list-style-type: none"><li>• Requerimientos del software.</li><li>• Requisitos del software.</li><li>• Metodologías de desarrollo de software.</li><li>• Arquitecturas del software.</li></ul> |
| Análisis y Diseño de Sistemas II | <ul style="list-style-type: none"><li>• Lenguaje Unificado de Modelado (UML).</li></ul>   |
| Base de datos                    | <ul style="list-style-type: none"><li>• Entidad, relaciones y tipo de relaciones.</li><li>• Modelos de datos.</li><li>• Gestores de base de datos.</li></ul>  |

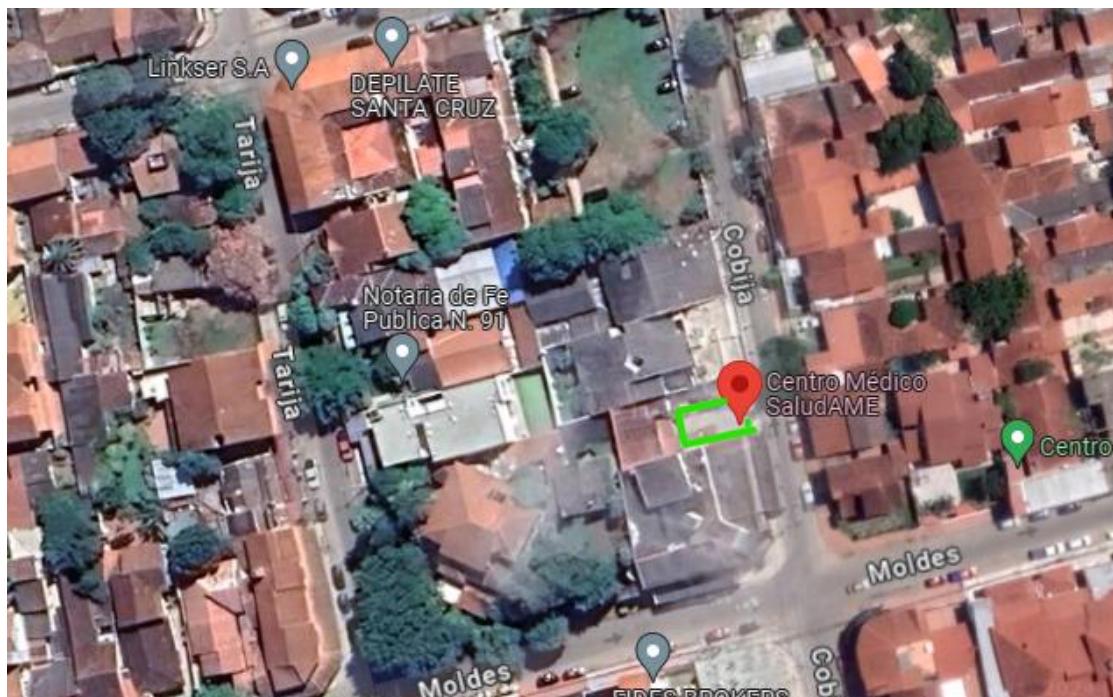
|                                  |   |
|----------------------------------|---|
|                                  | <ul style="list-style-type: none"> <li>• Lenguaje de consulta estructurada.</li> <li>• Procedimientos almacenados.</li> <li>• Transacciones.</li> </ul>   |
| Ingeniería web                   | <ul style="list-style-type: none"> <li>• Modelo del procesado web</li> </ul>  |
| Programación Avanzada            | <ul style="list-style-type: none"> <li>• Desarrollo de aplicaciones web</li> </ul>  |
| Programación orientada a objetos | <ul style="list-style-type: none"> <li>• Entorno integrado de desarrollo (IDE).</li> <li>• Lenguajes de programación híbridos y puros.</li> <li>• Lenguajes de programación, clases.</li> </ul> |

*Fuente: Elaboración propia.*

#### **1.6.2. Alcance Geográfico**

El presente trabajo se realizará en las instalaciones del Centro Médico SaludAME. Ver figura 2.

Figura 2: Ubicación del Centro Médico SaludAME S.R.L.



Fuente: Google Maps, 2024

### 1.6.3. Alcance Temporal

El trabajo se realizará en diez meses, según el calendario académico de la Escuela Militar de Ingeniería Unidad Académica Santa Cruz, gestión I-2024 / II-20

## CAPÍTULO II

### MARCO TEÓRICO



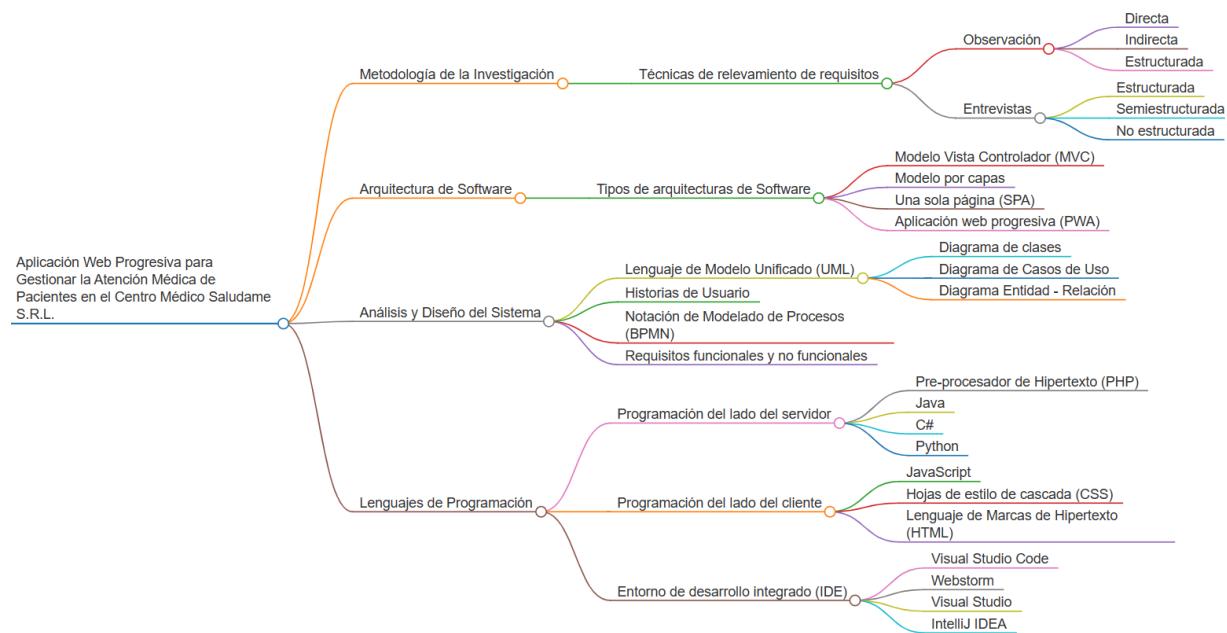
## CAPÍTULO 2.

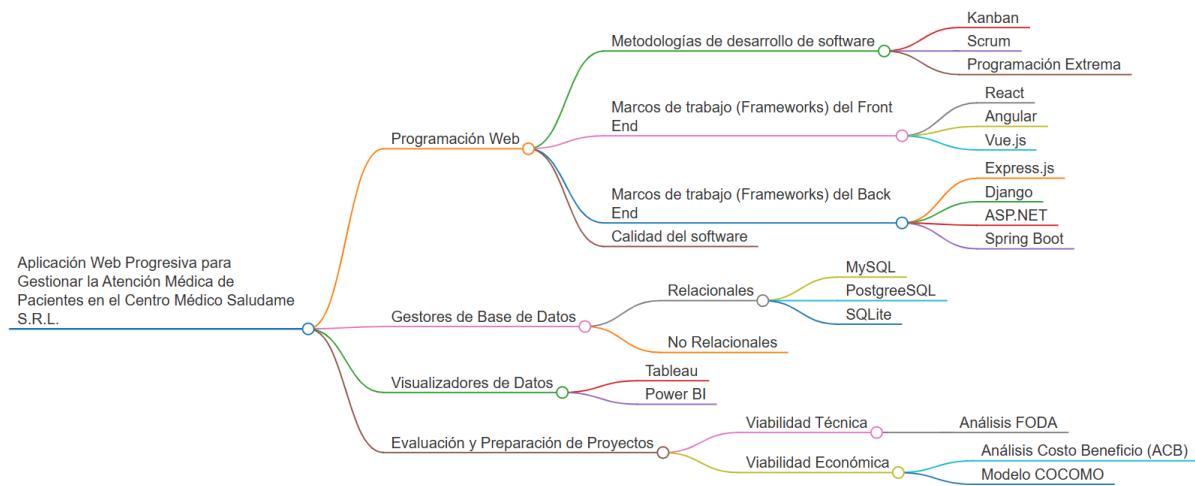
### MARCO TEÓRICO

#### 2.1. ESQUEMA DE MARCO TEÓRICO

En este capítulo se muestra la sustentación teórica para elaborar el marco práctico, a continuación, se presentan los siguientes puntos.

**Figura 3: Esquema de Marco Teórico**





Fuente: Elaboración propia

## 2.2. CONTENIDO DEL MARCO TEÓRICO

Para la elaboración del presente marco teórico se considera la siguiente fundamentación teórica expuesta en la Tabla 3, permitiendo elaborar las acciones de acuerdo con cada objetivo específico.

Tabla 3: Fundamentación teórica

| OBJETIVOS ESPECÍFICOS   | ACCIONES DE INVESTIGACIÓN   | FUNDAMENTACIÓN TEÓRICA  |
|---|---|---|
| Diagnosticar los procesos de pago y atención médica de pacientes en los consultorios. | Recolectar los requerimientos de la aplicación.                             | <ul style="list-style-type: none"> <li>• Metodología de la Investigación (Técnica de recolección de datos)</li> </ul> |
|   | Modelar los procesos de pago y atención médica de pacientes.                | <ul style="list-style-type: none"> <li>• Ingeniería de software (Requerimientos del software)</li> </ul>              |
|   | Describir los requerimientos funcionales y no funcionales de la aplicación. | <ul style="list-style-type: none"> <li>• Análisis y diseño de Sistemas II (Modelado de</li> </ul>                     |

|  |  |  |
|--|--|--|
|  |  | proceso de negocio).   |
| Seleccionar el modelo de proceso y las herramientas de software para el desarrollo de la aplicación. | <p>Elegir la metodología de desarrollo de software a utilizar a lo largo del proyecto.</p> <p>Escoger el lenguaje de programación y el entorno de desarrollo.</p> <p>Determinar el sistema gestor de base de datos a utilizar.</p> <p>Escoger la arquitectura de software a ser aplicada</p> | <ul style="list-style-type: none"> <li>• Ingeniería de software (Metodologías de desarrollo de software, arquitectura de software)</li> <li>• Programación orientada a objetos (POO) (Entorno de desarrollo integrado, lenguajes de programación híbridos y puros)</li> <li>• Base de datos (Gestores de base de datos)</li> </ul> |
| Diseñar requerimientos de la aplicación.   | <p>Analizar los requerimientos de la aplicación.</p> <p>Modelar los componentes de la aplicación.</p> <p>Realizar el diseño conceptual y lógico de la base de datos.</p>   | <ul style="list-style-type: none"> <li>• Análisis y diseño de sistemas II (Lenguaje de modelo unificado, diagramas de lenguaje de modelo unificado)</li> <li>• Base de datos (Entidad, relaciones y tipo de relaciones; modelos de datos)</li> <li>• Ingeniería de software (Requisitos del software)</li> </ul>                   |
| Implementar componentes de la aplicación.  | <p>Elaborar el diseño físico de la base de datos.</p> <p>Codificar el módulo de pago del seguro médico.</p>  | <ul style="list-style-type: none"> <li>• Base de datos (Lenguaje de consulta estructurada, procedimientos)</li> </ul>  |

|  |   |   |
|--|---|---|
|  | <p>Codificar el módulo de atención médica en consultorio.</p> <p>Generar los reportes de la aplicación utilizando herramientas de visualización de datos.</p> | <p>almacenados, transacciones, herramientas de visualización de datos)</p> <ul style="list-style-type: none"> <li>• POO (Lenguaje de programación, clases)</li> <li>• Programación avanzada (Desarrollo de aplicaciones web)</li> </ul> |
|--|---|---|

*Fuente: Elaboración propia*

## 2.3. DESARROLLO DEL MARCO TEÓRICO

### 2.3.1. Metodología de la Investigación

La metodología de la investigación se considera una parte fundamental en cualquier estudio científico o académico, utilizándose en múltiples ámbitos para diversos fines, tales como crear nuevos sistemas y resolver problemas existentes. A partir de este punto se definen los siguientes puntos:

“El proyecto de investigación es una actividad que consta de varios pasos concomitantes para la resolución de un problema científico, dentro de un periodo de tiempo y un presupuesto definidos.” (Moguel, 2005)

“La investigación científica se concibe como un conjunto de procesos sistemáticos y empíricos que se aplican al estudio de un fenómeno; es dinámica, cambiante y evolutiva. Se puede manifestar de tres formas: cuantitativa, cualitativa y mixta. Esta última implica combinar las dos primeras. Cada una es importante, valiosa y respetable por igual” (Hernández-Sampieri et al., 2014)

Dicho de otra manera, se lleva a cabo un estudio científico con el objetivo de obtener una respuesta coherente y relevante con relación al tema en cuestión. Para lograr una

respuesta fiable, es esencial seguir un proceso que consta de etapas lógicamente conectadas basadas en conocimientos verificados.

Existen 3 tipos de enfoques para la investigación científica, los cuales son:

- Enfoque cualitativo: Este enfoque permite alcanzar un análisis sistemático de información más subjetiva. A partir de ideas y opiniones sobre un determinado asunto, se abre el análisis no estadístico de los datos, que luego son interpretados de una forma subjetiva pero lógica y fundamentada.
- Enfoque cuantitativo: En este enfoque, el análisis de la información se basa en cantidades y/o dimensiones. Es decir, el elemento numérico tiene protagonismo. Cuando en una investigación se usa un enfoque cuantitativo, las hipótesis del investigador se someten a mediciones numéricas y sus resultados se analizan de forma estadística.
- Enfoque mixto: Se trata de un paradigma relativamente reciente que combina los enfoques cuantitativo y cualitativo en un mismo estudio. Aunque no es muy popular entre los científicos, ha encontrado acogida en algunos estudios relacionados con las ciencias sociales. La recolección y el análisis de los datos combinan los métodos estandarizados e interpretativos. Se cruzan resultados de uno u otro enfoque. (Lifeder, 2020)

Los elementos fundamentales a la hora de realizar una investigación son los siguientes:

- Identificación del tema de investigación
- Lugar de desarrollo del estudio
- Esquematización de los objetos principales (Tema, problema, objetivos, entre otros)
- Recopilación de datos
- Análisis de datos
- Elaboración de los resultados de la investigación
- Redacción del informe final

### 2.3.1.1. Técnica de recolección de datos

La técnica de recolección de datos es una técnica utilizada para realizar un plan de procedimientos específico para cada proyecto de investigación. Según (Parra, 2020) “La recolección de datos se refiere al enfoque sistemático de reunir y medir información de diversas fuentes a fin de obtener un panorama completo y preciso de una zona de interés y permite a un individuo o empresa responder a preguntas relevantes, evaluar los resultados y anticipar mejor las probabilidades y tendencias futuras.”

Recolectar los datos implica elaborar un plan detallado de procedimientos que nos conduzcan a reunir datos con un propósito específico. Este plan incluye determinar:

- Las fuentes de donde se obtendrán los datos.
- En dónde se localizan tales fuentes.
- El medio o método por el cual se recolectarán los datos.
- De qué forma se prepararán esos datos para analizarse y responder al planteamiento del problema. (Hernández-Sampieri et al., 2014)

#### a) **Origen de la fuente**

El origen de la fuente tiene una fuerte relación con el tipo de información que se maneja en la investigación, existiendo dos tipos de orígenes de fuentes de información, los cuales son:

- Fuentes primarias: Son todos aquellos usuarios y acompañantes a quienes se les aplicó un instrumento de investigación. Estas fuentes contienen información original, que ha sido publicada por primera vez y que no ha sido filtrada, interpretada o evaluada por nadie más.
- Fuentes secundarias: Son las que contienen información primaria, sintetizada y reorganizada. Están especialmente diseñadas para facilitar y maximizar el acceso a las fuentes primarias o a sus contenidos. Parten de datos preelaborados, como pueden ser datos obtenidos de anuarios estadísticos, de Internet, de medios de comunicación, de bases de datos procesadas con otros

fines, artículos y documentos relacionados, libros, tesis, informes oficiales, etc. (Soberón & Acosta, 2009)

**b) Medio o método para la recolección de datos**

Los medios para llevar a cabo una recolección de datos adecuada son:

- Observación. – Según (Hernández-Sampieri et al., 2014), “Este método de recolección de datos consiste en el registro sistemático, válido y confiable de comportamientos y situaciones observables, a través de un conjunto de categorías y subcategorías. Útil, por ejemplo, para analizar conflictos familiares, eventos masivos (como la violencia en los estadios de fútbol), la aceptación-rechazo de un producto en un supermercado, el comportamiento de personas con capacidades mentales distintas, la adaptación de operarios a una nueva maquinaria, etc.”. Dentro de la observación se pueden clasificar distintos tipos de observación, entre los cuales están (Díaz Sanjuán, 2010): la observación directa que ocurre cuando el investigador se pone en contacto personalmente con el hecho o fenómeno que trata de investiga, la observación indirecta que es cuando el investigador entra en conocimiento del hecho o fenómeno observado a través de las observaciones realizadas anteriormente por otra persona; la observación no estructurada llamada también simple o libre, es la que se realiza sin la ayuda de elementos técnicos especiales, en cambio la observación estructurada es la que se realiza con la ayuda de elementos técnicos apropiados, tales como: fichas, cuadros, tablas, etc., por lo cual se le denomina observación sistemática.
- Entrevistas. – La entrevista es un proceso de comunicación que se realiza normalmente entre dos personas; en este proceso el entrevistado obtiene información del entrevistado de forma directa. Si se generalizara una entrevista sería una conversación entre dos personas por el mero hecho de comunicarse, en cuya acción la una obtendría información de la otra y viceversa. En tal caso los roles de entrevistador / entrevistado irían cambiando a lo largo de la

conversación. La entrevista no se considera una conversación normal, si no una conversación formal, con una intencionalidad, que lleva implícitos unos objetivos englobados en una Investigación.” Las entrevistas se pueden clasificar en 3 tipos: semiestructurada, en donde se determina de antemano cual es la información relevante que se quiere conseguir. Se hacen preguntas abiertas dando oportunidad a recibir más matices de la respuesta, permite ir entrelazando temas, pero requiere de una gran atención por parte del investigador para poder encauzar y estirar los temas. (Actitud de escucha); no estructurada, en donde no existe un guión previo. El investigador tiene como referentes la información sobre el tema. La entrevista se va construyendo a medida que avanza la entrevista con las respuestas que se dan y requiere gran preparación por parte de investigador, documentándose previamente sobre todo lo que concierne a los temas que se tratan; y la estructurada, en donde el investigador planifica previamente las preguntas mediante un guion preestablecido, secuenciado y dirigido, por lo que dejan poca o ninguna posibilidad al entrevistado de réplica o de salirse del guión. Son preguntas cerradas (si, no o una respuesta predeterminada). (Peláez et al., 2013).

### **2.3.2. Arquitectura de software**

De acuerdo con (Cervantes, 2014), que cita a (Bass et al., 2012), “la Arquitectura de Software se refiere a “las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos”. El término “elementos” dentro de la definición del SEI es vago a propósito, pues puede referirse a distintas entidades relacionadas con el sistema. Los elementos pueden ser entidades que existen en tiempo de ejecución (objetos, hilos), entidades lógicas que existen en tiempo de desarrollo (clases, componentes) y entidades físicas (nodos, directorios). Por otro lado, las relaciones entre elementos dependen de propiedades visibles (o públicas) de los elementos, quedando ocultos los detalles de implementación. Finalmente, cada conjunto de elementos relacionados de un tipo

particular corresponde a una estructura distinta, de ahí que la arquitectura está compuesta por distintas estructuras.

“Dicho en pocas palabras, es un modelo repetible que posibilita el desarrollo de un sistema. Es importante tener en mente que la selección del tipo de arquitectura de software influye en la calidad, rendimiento, escalabilidad y facilidad de mantenimiento de un proyecto y podemos decir que estos factores se relacionan estrechamente con su éxito. Es así como la arquitectura de software involucra decisiones acerca de:

- Estructuras y componentes que integran un sistema
- Protocolos de comunicación
- Distribución física de los elementos
- Atribución de funciones a las diferentes estructuras
- Desempeño
- Control
- Sincronización y acceso a los datos

También es imprescindible tener en cuenta que a medida que se desarrolla un software, crece tanto su tamaño, como su nivel de complejidad, razón por la cual aumenta el “alcance del proyecto” y tiende a rebasar las estructuras de datos y algoritmos.

Por consiguiente, estructurar una arquitectura que optimice el entendimiento de dichos componentes se convierte en un camino elemental para presentar un proyecto exitoso.” (Palmero, 2021)

Como menciona (García, 2024), una buena arquitectura de software puede proporcionar varios beneficios a las empresas, entre ellos:

- Mayor eficiencia: Una arquitectura bien estructurada permite un desarrollo más rápido y eficiente, evitando la duplicación de esfuerzos y minimizando los errores.
- Mayor calidad: Una arquitectura sólida proporciona una base estable para el software, lo que conduce a una mayor calidad y confiabilidad del producto final.

- Facilita la integración: Una arquitectura de software bien diseñada permite la integración de diferentes sistemas y componentes, lo que facilita la comunicación entre ellos y mejora la interoperabilidad.
- Sistemas más seguros: A través de la aplicación de patrones de diseño de software, así como la integración en ecosistemas de integración que permitan probar el software desarrollado se obtendrán sistemas más seguros y resilientes contra ataques de seguridad, aparición de vulnerabilidades, etc.
- Reutilización: Si se trata de una arquitectura modular y mediante el uso de patrones, se puede reutilizar y amortizar la inversión en el desarrollo para la creación de otros sistemas de software.

#### 2.3.2.1. Tipos de arquitectura de software

Existen distintos tipos de arquitectura de software, entre los cuales se mencionan:

- Arquitectura Modelo Vista Controlador (MVC): El patrón MVC se utiliza porque permite separar componentes de un programa basándose en la responsabilidad de cada uno, por ende, si se requiere hacer una modificación en una parte específica del código, el resto permanece intacto. (Burin, 2023)
- Arquitectura en capas: En este enfoque, una aplicación se divide en diferentes capas lógicas que cumplen con ciertas características. Si hablamos en particular de una implementación de tres niveles, existirán capas como la presentación, la lógica de negocio y el acceso a datos. Cada capa tiene un propósito específico y se comunica con otras capas a través de interfaces bien definidas. (Zottola, 2023)
- Arquitectura de una aplicación de una sola página (SPA): Esta arquitectura de aplicación web está diseñada para mostrar sólo el contenido relevante. Para que esto suceda, primero carga la página web relevante y luego actualiza dinámicamente la representación de su contenido con la información solicitada solamente. En otras palabras, no se remite al servidor para cargar nuevas páginas, sino que envía peticiones sólo para las partes necesarias de la página

web. Las aplicaciones de una sola página contribuyen a un rendimiento más fluido y a una experiencia de usuario más intuitiva. (Juice Studio, 2022)

- Arquitectura de una aplicación web progresiva (PWA): La arquitectura de aplicaciones web progresivas (PWA) es un enfoque del desarrollo de aplicaciones web que combina lo mejor de las aplicaciones nativas y de las aplicaciones web tradicionales. Proporciona una experiencia de usuario similar a la de una aplicación nativa, pero con toda la flexibilidad de una aplicación web. Las PWA utilizan service workers, que se trata de código escrito en JavaScript para habilitar la funcionalidad offline y las notificaciones de audio. (Hernández, 2023)
- Arquitectura cliente-servidor: Dentro del esfuerzo humano a través de la tecnología que día a día perfecciona, está el persistente intento de maximizar una comunicación. Es aquí, donde consideramos como una de las ramas por excelencia preocupadas del asunto, Redes de computadores. En particular, este informe trata sobre arquitectura Cliente - Servidor, el cual es un modelo de una aplicación distribuida en el cual se basa en dos actores: Uno con rol de proveedor de recursos y otro con rol consultor sobre los recursos. El cliente es un programa ejecutable que participa activamente en el establecimiento de las conexiones. Envía una petición al servidor y se queda esperando por una respuesta. Su tiempo de vida es finito una vez que son servidas sus solicitudes, termina el trabajo. Por otro lado, el servidor es un programa que ofrece un servicio que se puede obtener en una red. Acepta la petición desde la red, realiza el servicio y devuelve el resultado al solicitante. Al ser posible implantarlo como aplicaciones de programas, puede ejecutarse en cualquier sistema donde exista TCP/IP y junto con otros programas de aplicación. El servidor comienza su ejecución antes de comenzar la interacción con el cliente. Las ventajas de este modelo respecto de otras posibles arquitecturas de red serían: es centralizado, es de fácil mantenimiento, y se puede aumentar la capacidad de clientes y servidores por separado. (Lizama et al., 2016)

### **2.3.3. Análisis y diseño de sistemas**

Según (Valenzuela, 2010), que cita a (Senn, 1992) “El análisis y diseño de sistemas se refiere al proceso de examinar la situación de una empresa con el propósito de manejarla con métodos y procedimientos más adecuados”. Se puede dividir en dos: el análisis de sistemas que comprende la planificación, el levantamiento inicial de información y el estudio en detalle del sistema actual para luego recomendar o estructurar las especificaciones necesarias para el nuevo sistema; y el diseño que consiste en llevar a cabo el sistema por medio de la clasificación y empleo de la información de manera que se pueda ofrecer una alternativa mucho más viable.

#### **2.3.3.1. Lenguaje de Modelo Unificado (UML)**

“El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual de propósito general que se utiliza para especificar, visualizar, construir y documentar los artefactos de un sistema software. Captura decisiones y conocimiento sobre sistemas que deben ser construidos. Se usa para comprender, diseñar, ojear, configurar, mantener y controlar la información sobre tales sistemas. Está pensado para ser utilizado con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre las técnicas de modelado e incorporar las mejores prácticas de software actuales en una aproximación estándar. UML también contiene construcciones organizativas para agrupar los modelos en paquetes, lo que permite a los equipos de software dividir grandes sistemas en piezas con las que se pueda trabajar, comprender y controlar las dependencias entre paquetes y gestionar las versiones de las unidades del modelo, en un entorno de desarrollo complejo. Contiene construcciones para representar las decisiones de implementación y para organizar elementos de tiempo de ejecución en componentes” (Rumbaugh et al., 2007).

El lenguaje de modelo unificado cuenta con diferentes tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

### a) Diagrama de clases

Los diagramas de clases son uno de varios tipos de diagramas estructurales de UML. Los diagramas estructurales muestran la estructura estática de un sistema, en lugar de mostrar cómo cambia un objeto con el tiempo. Los diagramas de clases visualizan las clases de un sistema y las relaciones entre ellas. En diseño orientado a objetos, las clases crean y operan objetos. Los objetos son instancias de clases. Por lo tanto, las clases son elementos de alto nivel esenciales de un sistema. Se derivan durante el diseño y se utilizan para comunicarse sobre el diseño o los cambios en el diseño. En un diagrama de clases, los nombres de las clases son los mismos que los nombres de los objetos porque el propósito de una clase es definir los atributos y operaciones para cada instancia de objeto en el sistema. Una clase es un modelo para un objeto, y un diagrama de clases es el modelo estático del sistema.

Un diagrama de clases UML tiene dos propósitos principales como modelo estático de un sistema orientado a objetos:

- Visualizar las clases de un sistema y sus propiedades.
- Mostrar y analizar las relaciones entre las clases.

Además de estos, los diagramas de clases UML también son la base para los diagramas de componentes y despliegue que muestran los aspectos de hardware y software de un sistema.

La notación del diagrama de clases UML se compone por los siguientes elementos:

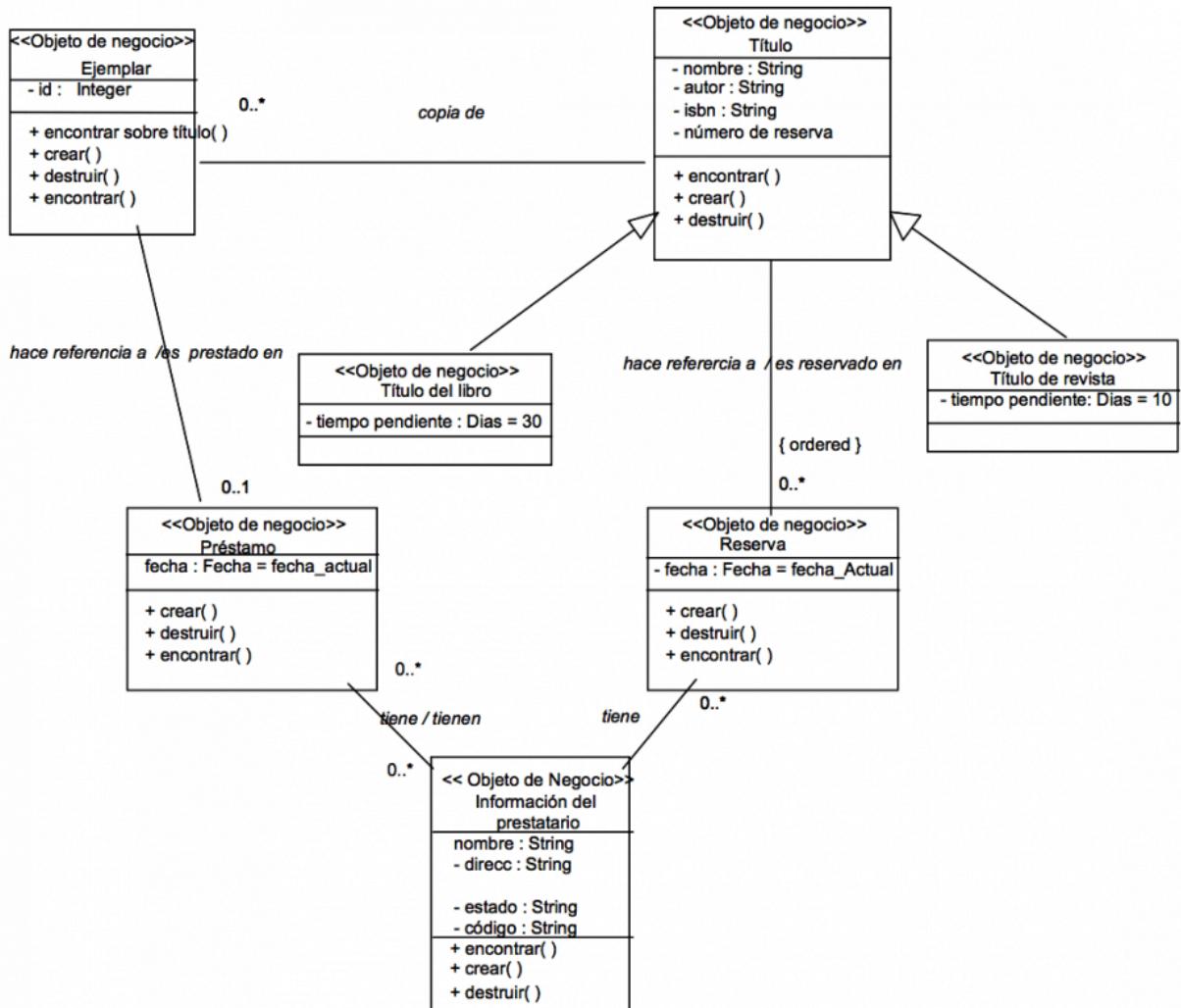
- Clases: Se representan con rectángulos que contienen el nombre de la clase, atributos y métodos. El nombre de la clase es obligatorio y se muestra en negrita.
- Relaciones: Se visualizan con líneas y puntas de flecha. Las relaciones incluyen herencia, asociaciones (bilaterales y unilaterales), agregación, composición y dependencia.

- Visibilidad: Los signos más (+) y menos (-) indican si los atributos y métodos son públicos o privados. El signo de almohadilla (#) indica visibilidad protegida.
- Multiplicidad: Se denota con números en la intersección de una línea de relación y un modelo de clase, definiendo cuántas instancias de un objeto participan en una relación.

Los diagramas de clases UML son herramientas prácticas de modelado para construir una arquitectura de software. Junto con otros diagramas UML, los desarrolladores y las partes interesadas visualizan diferentes vistas de un sistema. Estos diagramas nos ayudan a entender cómo funciona el sistema, cómo se comporta y cómo se relacionan sus partes. Crear diagramas de clases durante el diseño facilita el proceso de desarrollo al mostrar claramente las clases, sus atributos y sus métodos. También muestran cómo se relacionan las clases entre sí. Ver una construcción conceptual de un sistema antes de que se escriba cualquier código ayuda a los desarrolladores a comunicarse entre sí y con otras partes interesadas. (Miro, 2023)

Un ejemplo de diagrama de clases se puede apreciar a continuación, representa el estudio del sistema encargado de la gestión de préstamos y reservas de libros y revistas de una biblioteca.

**Figura 4: Ejemplo de diagrama de clases de un sistema encargado de la gestión de préstamos y reservas de libros y revistas de una biblioteca.**



Fuente: (Cillero, 2023)

### b) Diagrama de Casos de Uso

Un diagrama de casos de uso es una representación gráfica que muestra cómo los usuarios interactúan con un sistema. Este tipo de diagramas se utiliza comúnmente en el desarrollo de software y la ingeniería de sistemas para visualizar las diferentes formas en que un usuario puede interactuar con un sistema, detallando todas las posibles acciones y respuestas. El propósito principal de un diagrama de casos de uso

es identificar y clarificar los requisitos funcionales del sistema. En otras palabras, se centra en "qué" hará el sistema desde la perspectiva del usuario, en lugar de "cómo" lo hará. Los actores en el diagrama representan a los usuarios o entidades externas que interactúan con el sistema, mientras que los casos de uso son las acciones o funciones específicas que el sistema puede realizar.

En su forma más básica, un diagrama de casos de uso consta de una serie de cajas que representan los casos de uso, y líneas o flechas que representan las interacciones entre los actores y los casos de uso. Sin embargo, estos diagramas pueden volverse bastante complejos a medida que se agregan más actores y casos de uso, lo que refleja la complejidad inherente del sistema que se está diseñando.

Un diagrama de casos de uso es una herramienta crucial para definir y comunicar efectivamente los requisitos funcionales del sistema a todas las partes interesadas en un proyecto. Es una forma eficaz y visualmente intuitiva para analizar y presentar las interacciones entre los usuarios y el sistema.

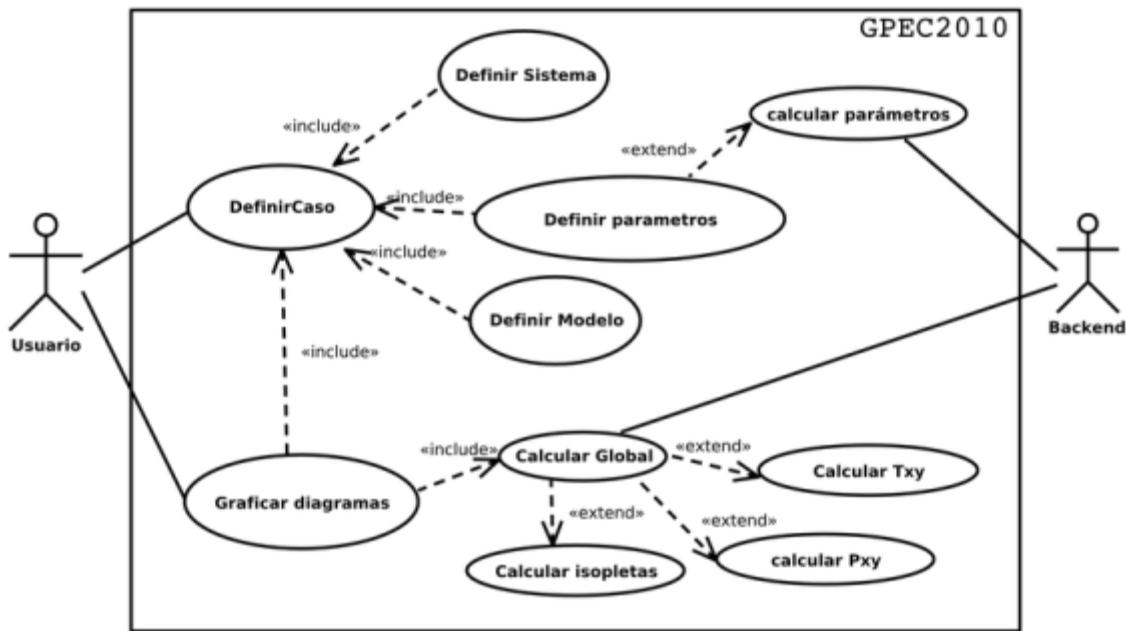
Los diagramas de casos de uso suelen tener cuatro componentes principales:

- **Actores:** Representan a los usuarios o cualquier otra entidad que interactúa con el sistema. Pueden ser personas, pero también pueden ser otros sistemas o procesos. En un diagrama, los actores suelen representarse como figuras de palo o iconos fuera del sistema.
- **Casos de uso:** Son las acciones o funciones específicas que el sistema puede realizar en respuesta a una interacción de un actor. Se representan como óvalos dentro del sistema.
- **Sistema:** El sistema es lo que se está diseñando o analizando. Se representa como un rectángulo grande que contiene todos los casos de uso.
- **Relaciones:** Describen cómo los actores y los casos de uso interactúan entre sí. Se representan como líneas entre los actores y los casos de uso. Las relaciones pueden ser simples (un actor realiza un caso de uso) o más complejas (un caso de uso incluye otros casos de uso, un caso de uso extiende otro caso de uso, etc.).

Los diagramas de casos de uso cuentan con múltiples beneficios, tales como una comunicación eficaz entre los usuarios técnicos y no técnicos, la identificación temprana de problemas, una planificación precisa, ayuda a facilitar la creación de pruebas y es una excelente forma de documentación, proporcionando una visión general rápida del sistema para futuras referencias. (Boardmix, 2024)

En la siguiente figura, se muestra un ejemplo de diagramas de caso de uso.

**Figura 5: Ejemplo de diagrama de casos de uso**



Fuente: (Alava, 2015)

### c) Diagrama Entidad – Relación

Un diagrama entidad-relación, también conocido como modelo entidad relación o ERD, es un tipo de diagrama de flujo que ilustra cómo las "entidades", como personas, objetos o conceptos, se relacionan entre sí dentro de un sistema. Los diagramas entidad-relación se usan a menudo para diseñar o depurar bases de datos relacionales en los campos de ingeniería de software, sistemas de información empresarial, educación e investigación. Estos diagramas emplean un conjunto definido de

símbolos, tales como rectángulos, diamantes, óvalos y líneas de conexión para representar la interconexión de entidades, relaciones y sus atributos. Son un reflejo de la estructura gramatical y emplean entidades como sustantivos y relaciones como verbos.

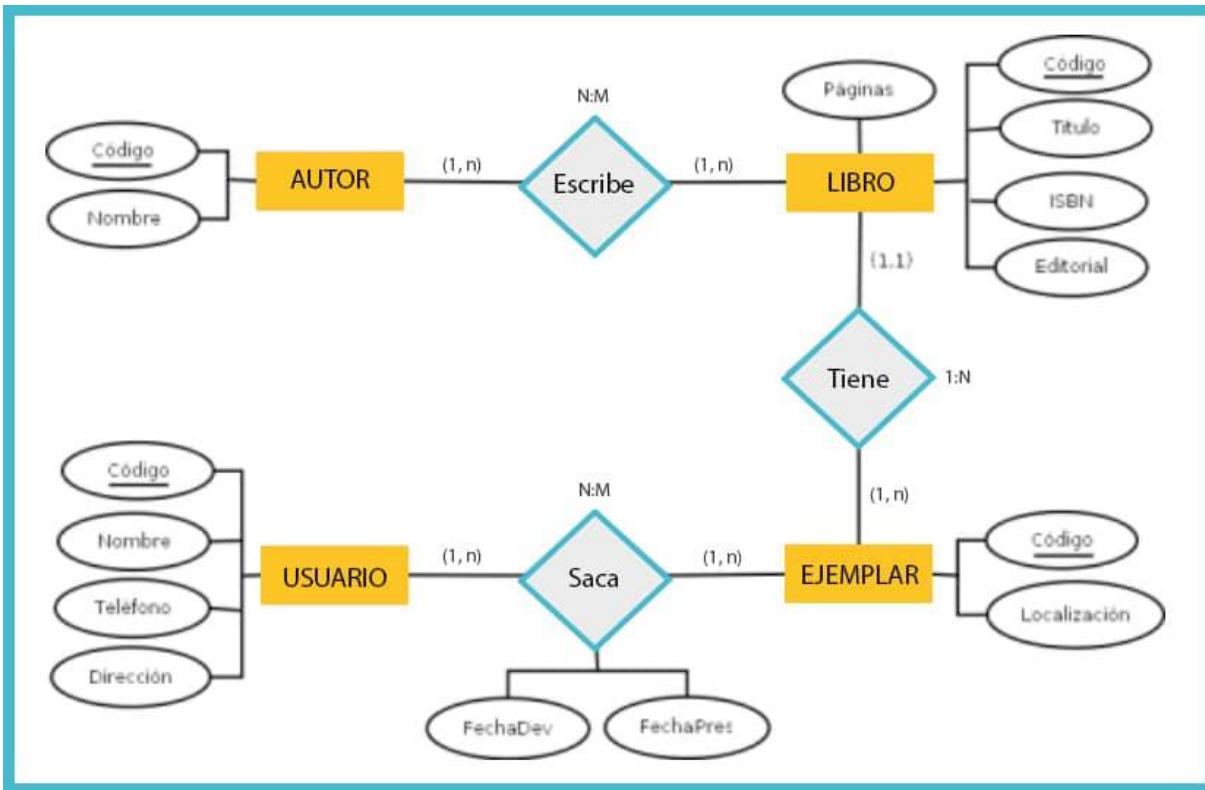
Los diagramas entidad-relación tienen múltiples usos, tales como el modelado y diseño de bases de datos, la solución de problemas de bases de datos, para sistemas de información empresarial y la reingeniería de procesos de negocio.

Los diagramas entidad-relación se componen de los siguientes elementos:

- Entidades: Representadas como rectángulos, son objetos o conceptos (como personas o eventos) que pueden tener datos asociados. Por ejemplo: un cliente, estudiante, auto o producto.
  - Relaciones: Ilustradas con diamantes o líneas, describen cómo las entidades interactúan entre sí, como asociaciones o acciones. Por ejemplo, el estudiante mencionado podría inscribirse en un curso. Las dos entidades serían el estudiante y el curso, y la relación representada es el acto de inscribirse, que conecta ambas entidades de ese modo.
  - Atributos: Mostrados como óvalos, son propiedades o características de las entidades o relaciones.
  - Cardinalidad: Define los atributos numéricos de la relación entre dos entidades o conjuntos de entidades. Las tres relaciones cardinales principales son uno a uno, uno a muchos y muchos a muchos. Un ejemplo de uno a uno sería un estudiante asociado a una dirección de correo electrónico. Un ejemplo de uno a muchos (o muchos a uno, en función de la dirección de la relación) sería un estudiante que se inscribe en muchos cursos, y todos esos cursos se asocian a ese estudiante en particular. Un ejemplo de muchos a muchos sería los estudiantes en grupo están asociados a múltiples miembros de la facultad y a su vez los miembros de la facultad están asociados a múltiples estudiantes.
- (Lucidchart, 2017)

Un ejemplo de diagrama entidad-relación se describe a continuación.

Figura 6: Ejemplo de diagrama entidad – relación



Fuente: (Ilerna, 2019)

### 2.3.3.2. Historias de Usuario

Una historia de usuario es una técnica utilizada en el desarrollo de software y en la gestión de proyectos ágiles para describir los requisitos y funcionalidades que un usuario desea o necesita en un sistema. Se utiliza para representar las necesidades del usuario desde su perspectiva y como una manera de comunicar claramente los requisitos del sistema a los desarrolladores y otros miembros del equipo.

Una historia de usuario generalmente sigue una estructura simple que incluye tres componentes principales:

- Título: Es una breve descripción que resume el objetivo o la necesidad del usuario. Suele ser una oración corta que captura el propósito principal de la historia.
- Descripción: Aquí se proporciona una explicación más detallada de la historia de usuario. Puede incluir información sobre el contexto del usuario, su objetivo y cualquier restricción o requerimiento específico.
- Criterios de aceptación: Son los criterios que se deben cumplir para que la historia de usuario se considere completada correctamente. Estos criterios son acordados entre el equipo de desarrollo y el cliente o usuario y suelen ser expresados en forma de lista de verificación o puntos que deben cumplirse.

Las historias de usuario son escritas en lenguaje natural y se centran en el usuario final y sus necesidades, en lugar de enfocarse en los aspectos técnicos de implementación y se utilizan como base para planificar y priorizar el trabajo del equipo de desarrollo, y suelen agruparse en conjuntos llamados “epic” o “temas” relacionados.

Las historias de usuario son una herramienta valiosa para fomentar la colaboración y la comunicación efectiva entre los miembros del equipo y los interesados en el proyecto. Además, ayudan a mantener el enfoque en las necesidades reales de los usuarios durante todo el ciclo de desarrollo del software. (Hernández Sola, 2020)

Las historias de los usuarios describen el por qué y el qué hay detrás del trabajo diario de los miembros del equipo de desarrollo, mediante un enfoque visual que permite ver, priorizar y evaluar los requisitos.

Existen 3 estructuras fundamentales para la redacción de historias de usuario:

- Regla de las 3'Cs: Como primer elemento está la Tarjeta (card), nos dice que la historia debe ser lo suficientemente breve como para que su definición entre en una tarjeta, a lo mucho en una o dos frases en un lenguaje común y entendible por el usuario. La Conversación (conversation) describe que la historia debe permitir un diálogo entre quien expresa la necesidad y quien la satisface. Y por último la Confirmación (confirmation) que significa que la historia debe contener

los elementos necesarios para determinar que se ha entregado el valor buscado por la necesidad.

- Como [perfil], [quiero] [para]: La estructura más básica consiste en estos tres elementos, que en inglés serían Role–Feature–Reason. No hay que olvidar que uno de los beneficios de las historias de usuario es que coloca al cliente en el centro de todos los procesos, por lo mismo se tienen distintos frentes de los cuales se rescatan soluciones ágiles e innovadoras. Por ejemplo: Como director de ventas, quiero registrar los ingresos y cantidades que me solicitan mis clientes para trazar una estrategia comercial con mis proveedores.
- Pruebas de Aceptación: Utiliza el formato Given–When–Then para especificar escenario, condiciones de acción y resultado esperado. Es derivado del BDD (Behavior Driven Development) y ayuda a definir pruebas para verificar la aceptación de la historia. (Salazar, 2021)

#### 2.3.3.3. Notación de Modelado de Procesos (BPMN)

Si los procesos tuvieran un lenguaje oficial, ese lenguaje sería BPMN (Business Process Modeling Notation en inglés). Notación de Modelado de Procesos (BPMN) es una representación gráfica estándar de los procesos y flujos de trabajo de una empresa. Es un diagrama de flujo que representa a los participantes, las opciones y la dirección de dichos procesos mediante gráficos e imágenes estandarizados. El BPMN es un estándar propagado y publicado por el Grupo de Gestión de Objetos (OMG).

Los gráficos utilizados en BPMN están creados para ser intrincados pero sencillos de leer. Tan sencillos que no se necesitan conocimientos de informática para entenderlos. Su sencillez hace posible que ejecutivos, analistas y empleados de implementación tecnológica utilicen el mismo gráfico para promover la optimización y la digitalización. Atrás quedaron los días en que los procesos de negocio necesitaban miles de documentos y archivos separados.

Cuando se describe un proceso de negocio, se puede decidir hacerlo a través de varios métodos. Por lo general, el más útil es un diagrama pictórico, es decir, utilizar

imágenes para describir los distintos elementos del proceso. Aunque existen diferentes opciones de diagramas pictóricos, el estándar BPMN es el más recomendado.

Al igual que ocurre con los diagramas de flujo, se puede utilizar una herramienta BPM en numerosas etapas del ciclo de vida de un proyecto. Por ejemplo, puede emplearse al principio del proyecto para comprender su estado actual. Además, a medida que el proyecto avanza y empieza a tomar forma, puede desarrollarse un modelo «To-Be» (a realizar). El BPMN se utiliza cuando es crucial visualizar los pasos necesarios para completar un proyecto concreto, el orden en que deben ejecutarse y quién está a cargo de llevar a cabo cada tarea.

Los tres objetivos principales de BPMN son:

- El objetivo principal de BPMN es proporcionar un conjunto de artefactos que se pueden utilizar para describir de forma adecuada y completa un proceso de negocio. Cualquier usuario empresarial puede utilizarlo. No necesita ayuda de desarrolladores informáticos, programación ni ningún conocimiento de TI. Los compañeros de RRHH, de atención al cliente y de cualquier departamento pueden utilizar esta tecnología.
- El segundo objetivo relevante en BPMN es lo completo que es. Los usuarios de negocio pueden modelar cada proceso de negocio, sin importar lo complejo que sea. Disponen de un conjunto completo de artefactos que le permiten describir el comportamiento de su proceso de negocio por completo.
- Por último, ser un estándar mundial. La mayoría de las personas que trabajan con procesos en todo el mundo conocen y utilizan este estándar.

El lenguaje BPMN se basa en diagramas de flujo y notaciones gráficas. Las notaciones pueden separarse en cuatro categorías para la diagramación:

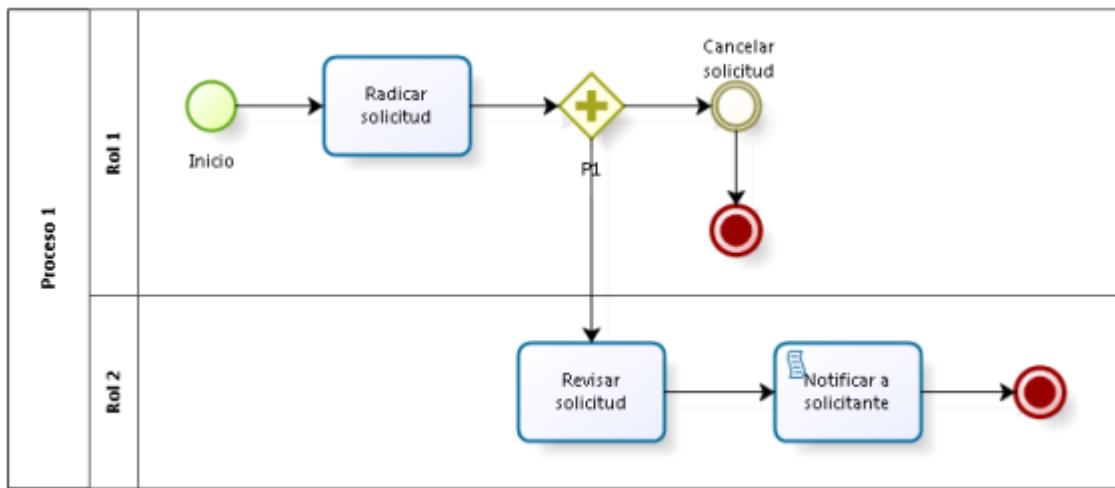
- Objetos de flujo: Elementos descriptivos utilizados para describir un proceso, como eventos, actividades y pasarelas. Los procesos suelen iniciarse con un evento de inicio, seguido de actividades/tareas y pasarelas (puntos de decisión), y terminan con un evento final. Los procesos complejos también

incorporan subprocesos y eventos intermedios, así como varios tipos de pasarelas para demostrar cómo avanza el flujo de trabajo a través del diagrama. Por ejemplo, una pasarela exclusiva solo tiene una opción de movimiento. Sin embargo, una pasarela inclusiva ofrece posibilidades basadas en la selección realizada en la pasarela.

- **Conexión de objetos:** Símbolos utilizados para vincular objetos de flujo como flujos de mensajes, flujos de secuencias y asociaciones. Los flujos se ilustran mediante líneas discontinuas o rectas con flechas, mientras que las asociaciones se representan mediante una línea de puntos para indicar que determinados documentos o artefactos están asociados a un evento o pasarela específicos.
- **Carriles de nado (swimlanes, por su traducción del inglés):** Son contenedores que separan un conjunto de actividades de otro, como pools y carriles. Los pools (piscinas, por su traducción del inglés) son los actores clave de un proceso ilustrado a través de un diagrama BPMN. Un pool distinto puede representar una organización, departamento o consumidor diferente implicado en el proceso. Los carriles dentro del pool representan las actividades y el flujo para un determinado rol de trabajo o participante, identificando quién es responsable de varios pasos de un proceso.
- **Artefactos:** Información adicional sobre el flujo de trabajo, como objetos de datos, grupos y anotaciones. Un objeto de datos representa los datos necesarios para una actividad, un grupo representa una agrupación lógica de actividades y una anotación describe lo que ocurre en un área específica del diagrama. Se trata de la información necesaria o generada para llevar a cabo un proceso de negocio. Incluye entradas de datos, salidas de datos, objetos de datos y repositorios de datos. (floksu, 2022)

A continuación, en la figura 7 se presenta un ejemplo de diagrama BPMN:

Figura 7: Ejemplo de un diagrama BPMN



Fuente: (Poveda Caputo, 2016)

#### 2.3.3.4. Requisitos funcionales y no funcionales

Un requisito funcional es una declaración de cómo debe comportarse un sistema. Define lo que el sistema debe hacer para satisfacer las necesidades o expectativas del usuario. Los requisitos funcionales se pueden considerar como características que el usuario detecta. Son diferentes de los requisitos no funcionales, que definen cómo debe funcionar internamente el sistema (p. ej., rendimiento, seguridad, etc.).

Los requisitos funcionales se componen de dos partes: función y comportamiento. La función es lo que hace el sistema (por ejemplo, "calcular el impuesto sobre las ventas"). El comportamiento es cómo lo hace el sistema (p. ej., "El sistema calculará el impuesto sobre las ventas multiplicando el precio de compra por la tasa impositiva"). Al crear requisitos funcionales, es importante tener en cuenta que deben ser específicos, medibles, alcanzables, relevantes y limitados en el tiempo (SMART). En otras palabras, sus requisitos funcionales deben:

- Ser específico sobre lo que debe hacer el sistema
- Ser medible para que pueda saber si el sistema lo está haciendo.
- Ser alcanzable dentro del marco de tiempo que ha establecido
- Sea relevante para sus objetivos comerciales
- Tener un límite de tiempo para que pueda seguir el progreso

Al seguir estas pautas, puede estar seguro de que sus requisitos funcionales son claros y ayudarán a su equipo de desarrollo a crear el producto adecuado.

Para una mejor comprensión de los requisitos funcionales, se presentan algunos ejemplos.

- Ejemplo 1: Un usuario podrá iniciar sesión en el sistema utilizando su nombre de usuario y contraseña.

En este ejemplo, la función es "iniciar sesión" y el comportamiento es "El sistema permitirá que un usuario inicie sesión con su nombre de usuario y contraseña".

- Ejemplo 2: El sistema calculará el impuesto a las ventas por la compra del usuario

En este ejemplo, la función es "calcular el impuesto sobre las ventas" y el comportamiento es "El sistema calculará el impuesto sobre las ventas multiplicando el precio de compra por la tasa impositiva". (Visure Solutions, 2022)

Por otro lado, un requisito no funcional define el atributo de calidad de un sistema de software. Representan un conjunto de estándares utilizados para juzgar la situación específica. operación de un sistema. Ejemplo, ¿qué tan rápido se carga el sitio web?

Un requisito no funcional es esencial para garantizar la usabilidad y eficacia de todo el sistema de software. No cumplir con los requisitos no funcionales puede dar como resultado que los sistemas no satisfagan las necesidades de los usuarios.

Los requisitos no funcionales permiten imponer restricciones o restricciones en el diseño del sistema en los distintos trabajos pendientes ágiles. Por ejemplo, el sitio debería cargarse en 3 segundos cuando el número de simultáneas. Los usuarios

estadounidenses son > 10000. La descripción de los requisitos no funcionales es tan crítica como un requisito funcional. Algunos beneficios de realizar requisitos no funcionales son:

- Garantizan que el sistema de software siga las reglas legales y de cumplimiento.
- Garantizan la confiabilidad, disponibilidad y rendimiento del sistema.
- Garantizan una buena experiencia de usuario y facilidad de uso del software.
- Ayudan a formular la política de seguridad del sistema de software.

A continuación, se muestran algunos ejemplos de requisitos no funcionales en ingeniería de software:

- Los usuarios deben cambiar la contraseña de inicio de sesión asignada inicialmente inmediatamente después del primer inicio de sesión exitoso. Además, la inicial nunca debe reutilizarse.
- A los empleados nunca se les permitió actualizar su información salarial. Dicho intento debe informarse al administrador de seguridad.
- Todo intento fallido de un usuario de acceder a un dato se registrará en un registro de auditoría.
- Un sitio web debe ser lo suficientemente capaz de manejar 20 millones de usuarios sin afectar su rendimiento.
- El software debe ser portátil. Por lo tanto, pasar de un sistema operativo a otro no crea ningún problema.
- Deberían auditarse la privacidad de la información, la exportación de tecnologías restringidas, los derechos de propiedad intelectual, etc. (Martín, 2023)

#### **2.3.4. Lenguajes de Programación**

Un lenguaje de programación, en palabras simples, es el conjunto de instrucciones a través del cual los humanos interactúan con las computadoras. Un lenguaje de

programación nos permite comunicarnos con las computadoras a través de algoritmos e instrucciones escritas en una sintaxis que la computadora entiende e interpreta en lenguaje de máquina. Los lenguajes de programación permiten a las computadoras procesar de forma rápida y eficientemente grandes y complejas cantidades de información. Por ejemplo, si a una persona se le da una lista de números aleatorios que van de uno a diez mil y se le pide que los coloque en orden ascendente, es probable que tome una cantidad considerable de tiempo e incluya algunos errores, mientras que, si le asigna la misma instrucción a una computadora utilizando un lenguaje de programación, podrás obtener la respuesta en unos cuantos segundos y sin errores.

Para utilizar un lenguaje de programación, de manera efectiva, debemos estudiarlo y comprenderlo desde tres perspectivas:

- Sintaxis: el conjunto de símbolos y reglas para formar sentencias.
- Semántica: las reglas para transformar sentencias en instrucciones lógicas.
- Pragmática: utilizando las construcciones particulares del lenguaje.

En español, las letras forman palabras que forman oraciones. En los lenguajes de programación, los caracteres forman sentencias que en conjunto forman instrucciones.  
(López Mendoza, 2020)

#### **2.3.4.1. Programación del lado del servidor**

Los Lenguajes de programación del lado del servidor son especialmente útiles en trabajos que se tiene que acceder a información centralizada, situada en una base de datos en el servidor, y cuando por razones de seguridad los cálculos no se pueden realizar en la computadora del usuario.

Es importante destacar que los lenguajes de programación del lado del servidor son necesarios porque para hacer la mayoría de las aplicaciones web se debe tener acceso a muchos recursos externos a la computadora del cliente, principalmente bases de datos alojadas en servidores de Internet. Un caso claro es un banco: no tiene

ningún sentido que el cliente tenga acceso a toda la base de datos, sólo a la información que le concierne.

Las ventajas de este tipo de programación son que el cliente no puede ver los scripts, ya que se ejecutan y transforman en HTML antes de enviarlos. Además, son independientes del navegador del usuario, ya que el código que reciben es HTML fácilmente interpretable. Como desventajas se puede señalar que será necesario un servidor más potente y con más capacidades que el necesario para las páginas de cliente. Además, estos servidores podrán soportar menos usuarios concurrentes, porque se requerirá más tiempo de procesamiento para cada uno. (Instituto Tecnológico de Matehuala, 2013)

#### **a) Preprocesador de Hipertexto (PHP)**

PHP es el acrónimo recursivo de Preprocesador de Hipertexto (del inglés, Hypertext Preprocessor), un lenguaje de programación de código abierto muy utilizado especialmente en el desarrollo de aplicaciones y web, este lenguaje de código puede ser incrustado en HTML5 puesto que es un código muy variable favorece el enlace entre la interfaz de usuario y los servidores.

La principal función de PHP, pero no la única, es la agilidad y velocidad con la que conecta a los usuarios con los servidores gracias a que esta se ejecuta en las bases de datos de los servidores como se ha mencionado previamente.

Se puede encontrar PHP en todos los plugins y plantillas de WordPress, es uno de los usos más extendidos de este lenguaje. Al ser un código abierto y simple es muy sencillo gestionar todas las extensiones que sirven para conservar todos los sitios web de la plataforma actualizados, según se vaya creando diferentes funciones.

Algunos beneficios de desarrollar en PHP son los siguientes:

- Es uno de los lenguajes más sencillos que puedan encontrar en el mundo de la programación.

- Puede ser usado en cualquier tipo de servidor y distintos sistemas operativos virtuales.
- PHP puede combinarse otros lenguajes de programación como HTML, CSS o Java.
- Soporta gran cantidad de bases de datos diferentes, por esta razón es ideal para la creación de web y aplicaciones basadas en bases de datos.(Arimetrics, 2022)

Sin embargo, como menciona (Tapia, 2018), las desventajas de PHP son que el lenguaje sólo se ejecuta en un servidor, por lo tanto, se necesita un servidor web para que funcione, el código no puede ser fácilmente ocultado y si no se configura correctamente se dejan abiertas muchas brechas de seguridad que a la larga traerán problemas.

**b) Java**

Desde su lanzamiento en 1995, el lenguaje de programación de Java ha sido uno de los más populares y utilizados por diseñadores, programadores e informáticos. Su crecimiento ha sido tan grande que hoy en día son pocas las industrias que no utilizan alguna de sus aplicaciones para la creación de sitios web, sistemas informáticos y software.

Java es un lenguaje de programación utilizado para crear software compatible con una gran diversidad de sistemas operativos. Este lenguaje tiene la particularidad de ser compilado e interpretado al mismo tiempo; esto significa que es un lenguaje simplificado que convierte automáticamente el código en instrucciones de máquina. (Coppola, 2023)

De acuerdo con (Nazarevich, 2021), algunas de las ventajas de usar Java para el desarrollo web son: es un lenguaje de alto nivel, su estabilidad, está orientado a objetos, su robustez, su seguridad, su portabilidad y el mantenimiento es bastante barato.

No obstante, (Jesús, 2024) menciona que algunas de las desventajas de Java son: el rendimiento y uso de recursos, la curva de aprendizaje puede ser compleja y la dependencia de la máquina virtual de Java (JVM, por sus siglas en inglés).

**c) C#**

Es un lenguaje de programación multiparadigma desarrollado por Microsoft, que evoluciona de la familia de lenguajes C -como su nombre indica- tomando lo mejor de los lenguajes C y C++ y que se asemeja mucho a lenguajes de alto nivel de abstracción como Java y JavaScript. Form parte de la plataforma .NET de Microsoft, una API que se ha convertido en una de las principales plataformas de desarrollo debido a la facilidad que ofrece para la construcción de todo tipo de aplicaciones multiplataforma sólidas y duraderas.

Se caracteriza principalmente por ser un lenguaje de programación orientado a objetos (POO) polivalente y fácil de aprender. Sin perder potencia original de C, da la posibilidad de acceder a bajo nivel al núcleo de los sistemas operativos, trabajar con punteros a memoria e interactuar con elementos físicos de los dispositivos.

Las ventajas de desarrollar en C# incluyen su naturaleza multiparadigma y la facilidad para construir aplicaciones multiplataforma robustas y duraderas. (TRBL Services, 2021).

Como desventajas de desarrollar en C#, (Proscont, 2023) menciona que tiene una mayor curva de aprendizaje, necesita un entorno de desarrollo específico, tiene un mayor tiempo de compilación y no es tan adecuado para sistemas de baja potencia.

**d) Python**

Python es un lenguaje de programación de código abierto que posee un enfoque imperativo, orientado a objetos y de alto nivel. Además, se caracteriza por tener una sintaxis clara y concisa, lo que lo convierte en un lenguaje relativamente fácil de utilizar.

Por otra parte, es un lenguaje de programación que no posee un objetivo específico, sino que se utiliza en diferentes áreas, tales como la ciencia de datos, la inteligencia artificial o el desarrollo web, entre otras implementaciones.

Una de las ventajas de Python respecto a otros lenguajes de programación es que posee una sintaxis clara y concisa. Por otro lado, una de las características principales de Python es su importancia dentro del ecosistema tecnológico en general. Sus innumerables librerías resuelven complejos problemas de casi cualquier sector. Cabe destacar que Python puede ejecutarse en diferentes plataformas o software operativos, tales como Windows, Mac y Linux. Esto hace que también se pueda denominar un lenguaje multiplataforma. (Llamas, 2023)

No obstante, (Cristancho, 2022) menciona que las desventajas de programar en Python son el consumo excesivo de memoria, un procesamiento lento y que no es perfecto para todos los tipos de desarrollo.

#### **2.3.4.2. Programación del lado del cliente**

En el desarrollo web, el "lado del cliente" hace referencia a todo lo que en una aplicación web se muestra o tiene lugar en el cliente (dispositivo del usuario final). Esto incluye lo que ve el usuario, como texto, imágenes y el resto de la interfaz de usuario, junto con cualquier acción que una aplicación lleve a cabo en el navegador del usuario.

Los lenguajes de marcado como HTML y CSS son interpretados por el navegador en el lado del cliente. Además, muchos desarrolladores actuales están incluyendo procesos del lado del cliente en la arquitectura de sus aplicaciones y dejando de hacer todo en el lado del servidor; la lógica de negocio para las páginas web dinámicas, por ejemplo, suele ejecutarse en el lado del cliente en una aplicación web moderna. Los procesos del lado del cliente se escriben casi siempre en JavaScript.

El lado del cliente también se conoce como frontend, aunque estos dos términos no significan exactamente lo mismo. El lado del cliente hace referencia únicamente al

lugar en el que se ejecutan los procesos, mientras que el frontend hace referencia a los tipos de procesos que se ejecutan en el lado del cliente. (Cloudflare, 2022)

### a) **JavaScript**

JavaScript es un lenguaje de «scripting» (una programación ligera) interpretado por casi todos los navegadores, que permite añadir a las páginas web efectos y funciones adicionales a los contemplados en el estándar HTML. JavaScript fue desarrollado por Netscape Corporation para su Navigator 2.0, y por su sencillez sigue siendo una herramienta muy útil en la elaboración de páginas web que tengan algo más que texto. Microsoft desarrolló el JScript para su Explorer que, en términos generales, es compatible con Netscape.

Conviene aclarar que JavaScript no es un lenguaje de programación propiamente dicho. Es un lenguaje de scripts (guiones o rutinas). Se parece más, por lo tanto, a las macros de los procesadores de texto u hojas de cálculo. No es posible hacer un programa completo con JavaScript. Ni se desarrolló para eso, ni las posteriores versiones le han conferido los elementos necesarios para ello.

En términos generales, JavaScript nos permite mejorar la gestión cliente/servidor. Un guión de JavaScript puede tratar y gestionar localmente, en el cliente (navegador del usuario), eventos tales como:

- Comprobar la validez de los campos cumplimentados en un formulario
- Abrir y cerrar ventanas
- Cambios dinámicos en una página (aspecto y contenidos)
- Tratamiento de cadenas de texto
- Operaciones aritméticas
- También puede utilizarse para realizar varias tareas a la vez, como abrir una página web y simultáneamente visualizar un video, reproducir sonido o ejecutar un applet de Java.

Dado que su cometido es ampliar el HTML, JavaScript es un lenguaje con algunas limitaciones que, indirectamente, confieren seguridad para el navegante.

Así, por ejemplo, JavaScript no puede (no sabe) crear, leer, modificar o borrar ficheros del usuario, con una excepción: las cookies. Carece también de mecanismos para establecer conexiones de red, llegando como mucho a ser capaz de enviar correos o formularios. (Rodríguez, 2005)

### **b) Hojas de estilo de cascada (CSS)**

Cascading Style Sheets (CSS) es un lenguaje de programación que sirve para determinar el diseño de los documentos electrónicos. Con la ayuda de unas sencillas instrucciones -presentadas en forma de código fuente claro-, los elementos del sitio web, como el diseño, el color y la tipografía, pueden adaptarse como se deseé. Gracias a las hojas de estilo en cascada, la estructura semántica y el contenido del documento no se ven afectados. CSS surgió a mediados de la década de 1990 y ahora se considera el lenguaje de hojas de estilo estándar en Internet.

CSS, al igual que HTML, es uno de los lenguajes centrales de Internet. Mientras que para añadir texto a un sitio web se utiliza HTML y se estructura semánticamente, para definir el diseño del contenido se utiliza CSS. Aunque HTML y CSS se utilizan en combinación, las instrucciones de diseño de CSS y los elementos de HTML existen por separado. Esto significa que una máquina puede leer un documento electrónico incluso sin CSS. Con la ayuda de CSS, el contenido del navegador se prepara visualmente y se presenta de forma atractiva.

Por ejemplo, CSS permite controlar algunas especificaciones de forma centralizada. Esto significa que elementos similares (como todos los hipervínculos o imágenes) dentro de un mismo documento pueden ser reconocidos y formateados mediante un único comando. Las instrucciones de diseño no tienen que estar en forma de hoja de estilo interna en el propio documento HTML. Si se guardan las instrucciones CSS en una hoja de estilo externa, es decir, en un archivo separado, ésta puede utilizarse también para otros documentos.

Además de las instrucciones básicas de visualización relativas a los colores, las formas y la tipografía de los elementos HTML, ahora existen módulos más sofisticados en CSS. Con ellos se pueden, por ejemplo, definir animaciones o representaciones diferentes según el medio de salida. De este modo, el mismo documento HTML puede prepararse de forma idéntica para todos los medios posibles. Como el contenido y el diseño están separados en este documento, el código del sitio web es más claro. El llamado lenguaje de estilo SASS ofrece aún más posibilidades, pero no sustituye por completo al CSS. (IONOS, 2021)

### c) HTML

HTML es el lenguaje con el que se define el contenido de las páginas web. Corresponde a las siglas en inglés de Lenguaje de Marcado de Hipertexto, básicamente son un conjunto de etiquetas que el navegador interpreta y se emplean para definir el texto y otros elementos que compondrán una página web, como imágenes, listas, tablas, vídeos, etc.

El lenguaje HTML sirve para describir la estructura básica de una página y organizar la forma en que se mostrará su contenido, además de que HTML permite incluir enlaces hacia otras páginas o documentos.

Hay que mencionar que el HTML no es un lenguaje de programación, ya que no cuenta con funciones aritméticas, variables o estructuras de control propias de estos lenguajes, por lo que el HTML únicamente sirve para crear páginas web estáticas. Sin embargo, este lenguaje es muy útil ya que combinado con otros lenguajes de programación obtenemos páginas web dinámicas como las que conocemos hoy en día. (Vadavo, 2023)

HTML funciona mediante el uso de etiquetas. Según (Nużyńska, 2024), una etiqueta/marca HTML es un comando que indica qué hacer y cómo mostrar un elemento HTML en la página web. Toma la forma de texto, encerrado entre corchetes. Las etiquetas HTML vienen en pares. La etiqueta inicial se ve así: <etiqueta>. La

etiqueta final se crea agregando / a la etiqueta inicial: </tag>. Algunos de las etiquetas más comunes de HTML son:

- <head> para información sobre el documento
- <body> para el contenido
- <div> división o bloque dentro del contenido, el más común para componer elementos complejos
- <a> para enlaces
- <strong> para poner el texto en negrita
- <br> para saltos de línea
- <p> para párrafos
- <h1>...<h6> para títulos dentro del contenido
- <img> para añadir imágenes al documento (Henry Tecnología S.A.S, 2021)

#### 2.3.4.3. Entorno de Desarrollo Integrado (IDE)

Un IDE o entorno de desarrollo integrado es una aplicación de tipo software que combina en un solo lugar todas las herramientas necesarias para realizar un proyecto de desarrollo de software. Ofrece una interfaz que permite escribir código, organizar grupos de texto y automatizar las tareas redundantes de programación.

Más que un editor de código, los IDE combinan las funcionalidades de varios procesos de programación en un mismo lugar. Poseen al menos un editor, un compilador, un depurador, y las funcionalidades de compleción de código o de gestión de código genérico. Ciertos IDE se focalizan en un lenguaje de programación específico como Python o Java. Sin embargo, la gran mayoría son también compatibles con múltiples lenguajes de programación. (DataScientest, 2022)

Un IDE proporciona a los programadores una interfaz unificada y eficiente para escribir código, así como herramientas de construcción, compilación, depuración y gestión de proyectos. Además, puede incluir características adicionales, como sugerencias de código, resaltado de sintaxis, integración con sistemas de control de versiones y

pruebas automatizadas. Al integrar todas estas funcionalidades en una sola plataforma, un entorno de desarrollo integrado ayuda a los desarrolladores a aumentar su productividad y eficiencia en el ciclo de desarrollo de software. (M. Lopez, 2023)

### a) **Visual Studio Code**

Visual Studio Code (VS Code) es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación.

Para tener una idea de la popularidad de Visual Studio Code y la aceptación que ha tenido en el mundo de desarrollo, podemos consultar datos. Según una encuesta realizada por Stack Overflow a más de 80,000 desarrolladores en mayo del 2021, Visual Studio Code es el entorno de desarrollo más usado y con mucha diferencia, un 71.06%.

VS Code tiene una gran variedad de características útiles para agilizar el trabajo, que lo hacen el editor preferido por muchos para trabajar los proyectos, las cuales son:

- **Multiplataforma:** Es una característica importante en cualquier aplicación y más si trata de desarrollo. Visual Studio Code está disponible para Windows, GNU/Linux y macOS.
- **IntelliSense:** Esta característica está relacionada con la edición de código, autocompletado y resaltado de sintaxis, lo que permite ser más ágil a la hora de escribir código. Como su nombre lo indica, proporciona sugerencias de código y terminaciones inteligentes en base a los tipos de variables, funciones, etc. Con la ayuda de extensiones se puede personalizar y conseguir un IntelliSense más completo para cualquier lenguaje.
- **Depuración:** Visual Studio Code incluye la función de depuración que ayuda a detectar errores en el código. De esta manera, nos evitamos tener que revisar

línea por línea a puro ojo humano para encontrar errores. VS Code también es capaz de detectar pequeños errores de forma automática antes de ejecutar el código o la depuración como tal.

- Uso del control de versiones: Visual Studio Code tiene compatibilidad con Git, por lo que puedes revisar diferencias o lo que conocemos con git diff, organizar archivos, realizar commits desde el editor, y hacer push y pull desde cualquier servicio de gestión de código fuente (SMC). Los demás SMC están disponible por medio de extensiones.
- Extensiones: Las extensiones nos permiten personalizar y agregar funcionalidad adicional de forma modular y aislada. Por ejemplo, para programar en diferentes lenguajes, agregar nuevos temas al editor, y conectar con otros servicios. Realmente las extensiones nos permiten tener una mejor experiencia, y lo más importante, no afectan en el rendimiento del editor, ya que se ejecutan en procesos independientes. (Flores, 2022)

Sin embargo, (WebDesing, 2022) menciona algunas desventajas de Visual Studio Code, tales como: la lentitud en proyectos grandes, la interfaz de usuario puede ser confusa y puede tener problemas de estabilidad.

### b) **Webstorm**

WebStorm es una herramienta de desarrollo web que ofrece numerosas ventajas a los programadores. Una de las características más notables de esta aplicación es su funcionalidad completa, que incluye soporte para múltiples lenguajes de programación web como HTML, CSS, JavaScript y TypeScript. Esto significa que los desarrolladores pueden trabajar en proyectos web completos sin necesidad de cambiar de herramienta.

Además de su completa funcionalidad, WebStorm también ofrece una notable eficiencia en el proceso de desarrollo web. Tiene una amplia gama de herramientas inteligentes que hacen que escribir código sea más fácil y rápido. Por ejemplo, el editor de código inteligente de WebStorm ofrece sugerencias y correcciones de código

automáticas, lo que ahorra tiempo y reduce los errores. También proporciona una navegación rápida y sencilla a través del código, gracias a funciones como la navegación estructural y la búsqueda contextual. (Vidal, 2023)

Las desventajas de WebStorm, según (Bolufer, 2024a) son el costo de la licencia, tiene una curva de aprendizaje inicial para nuevos desarrolladores y tiene altos requisitos de hardware para funcionar correctamente.

### c) Visual Studio

Visual Studio es una herramienta de desarrollo eficaz que permite completar todo el ciclo de desarrollo en un solo lugar. Es un entorno de desarrollo integrado (IDE) completo que se puede usar para escribir, editar, depurar y compilar el código y, luego, implementar la aplicación. Aparte de la edición y depuración del código, Visual Studio incluye compiladores, herramientas de finalización de código, control de código fuente, extensiones y muchas más características para mejorar cada fase del proceso de desarrollo de software.

Visual Studio proporciona a los desarrolladores un entorno de desarrollo enriquecido para desarrollar código de alta calidad de forma eficaz y colaborativa.

- Instalador basado en cargas de trabajo: instale solo lo que necesita.
- Herramientas y características de codificación eficaces: todo lo que necesita para compilar sus aplicaciones en un solo lugar.
- Compatibilidad con varios lenguajes: código en C++, C#, JavaScript, TypeScript, Python, etc.
- Desarrollo multiplataforma: compilación de aplicaciones para cualquier plataforma.
- Integración del control de versiones: colaboración en el código con compañeros de equipo.
- Desarrollo asistido por inteligencia artificial: escritura de código de forma más eficaz con ayuda de la inteligencia artificial. (Microsoft, 2023)

No obstante, las desventajas de usar Visual Studio, como menciona (Bolufer, 2024b) son el excesivo consumo de recursos de hardware, el costo de la licencia que puede resultar costosa y no accesible para todos los usuarios y que está principalmente orientado al desarrollo de aplicaciones y programas del ecosistema de Microsoft.

#### d) **IntelliJ IDEA**

Desarrollado por la empresa JetBrains, IntelliJ IDEA es un entorno de desarrollo integrado (IDE) para Java y Kotlin diseñado para maximizar la productividad de los desarrolladores. Realiza las tareas rutinarias y repetitivas por usted al proporcionar finalización inteligente de código, análisis de código estático y refactorizaciones, y le permite concentrarse en el lado positivo del desarrollo de software, lo que lo convierte no solo en una experiencia productiva sino también agradable. (JetBrains, 2024)

Algunas ventajas que destacan es su interfaz de usuario intuitiva, soporta diferentes herramientas tales como Git, Maven y Gradle.

Sin embargo, sus desventajas son: una curva de aprendizaje pronunciada para los desarrolladores, el costo de la licencia y los requisitos de hardware son elevados. (Bolufer, 2023)

#### **2.3.5. Programación web**

La programación web es el proceso de crear aplicaciones y sitios web utilizando lenguajes de programación y herramientas específicas. Los programadores web utilizan una variedad de lenguajes de programación como HTML, CSS, JavaScript, PHP y Ruby para crear sitios web y aplicaciones que se ejecutan en línea a través de navegadores web.

El objetivo principal de la programación web es crear sitios web dinámicos e interactivos que brinden una experiencia de usuario atractiva y eficiente. Los programadores web utilizan lenguajes de programación específicos para crear

contenido dinámico, como formularios web, botones, menús desplegables, animaciones y efectos visuales.

La programación web es esencial en la era digital actual, ya que la mayoría de las empresas y organizaciones tienen presencia en línea. La programación web permite a las empresas crear y personalizar sitios web que se adapten a sus necesidades específicas. Además, la programación web permite a las empresas interactuar con sus clientes en línea a través de formularios web, chat en vivo y otras herramientas interactivas. (Moreira, 2023)

#### **2.3.5.1. Metodologías de desarrollo de software**

La metodología para el desarrollo de software es un modo sistemático de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito. Una metodología para el desarrollo de software comprende los procesos a seguir sistemáticamente para idear, implementar y mantener un producto software desde que surge la necesidad del producto hasta que cumplimos el objetivo por el cual fue creado. Una metodología de desarrollo de sistemas no tiene que ser necesariamente adecuada para usarla en todos los proyectos. Cada una de las metodologías disponibles es más adecuada para tipos específicos de proyectos, basados en consideraciones técnicas, organizacionales, de proyecto y de equipo. (Maida & Pacienza, 2015)

En la actualidad se pueden diferenciar dos grandes grupos de metodologías de desarrollo de software: las ágiles y las tradicionales. A continuación, se explican las características de cada una de ellas.

- **Metodologías de desarrollo de software tradicionales:** Las metodologías de desarrollo de software tradicionales se caracterizan por definir total y rígidamente los requisitos al inicio de los proyectos de ingeniería de software. Los ciclos de desarrollo son poco flexibles y no permiten realizar cambios, al contrario que las metodologías ágiles; lo que ha propiciado el incremento en el uso de las segundas. La organización del trabajo de las metodologías

tradicionales es lineal, es decir, las etapas se suceden una tras otra y no se puede empezar la siguiente sin terminar la anterior. Tampoco se puede volver hacia atrás una vez se ha cambiado de etapa. Estas metodologías, no se adaptan nada bien a los cambios, y el mundo actual cambia constantemente. Las principales metodologías tradicionales o clásicas son: cascada, prototipado, espiral, incremental y el diseño rápido de aplicaciones (RAD, por sus siglas en inglés).

- Metodologías de desarrollo de software ágiles: las metodologías ágiles de desarrollo de software son las más utilizadas hoy en día debido a su alta flexibilidad y agilidad. Los equipos de trabajo que las utilizan son mucho más productivos y eficientes, ya que saben lo que tienen que hacer en cada momento. Además, la metodología permite adaptar el software a las necesidades que van surgiendo por el camino, lo que facilita construir aplicaciones más funcionales. Se basan en la metodología incremental, en la que en cada ciclo de desarrollo se van agregando nuevas funcionalidades a la aplicación final. Sin embargo, los ciclos son mucho más cortos y rápidos, por lo que se van agregando pequeñas funcionalidades en lugar de grandes cambios. Este tipo de metodologías permite construir equipos de trabajo autosuficientes e independientes que se reúnen cada poco tiempo para poner en común las novedades. Poco a poco, se va construyendo y puliendo el producto final, a la vez que el cliente puede ir aportando nuevos requerimientos o correcciones, ya que puede comprobar cómo avanza el proyecto en tiempo real. Las principales metodologías ágiles son: Kanban, Scrum, Programación Extrema (XP, por sus siglas del inglés) y Lean. (Santander Universidades, 2020)

a) **Kanban**

El método Kanban es un tipo de sistema basado en la metodología ágil en el que se busca conseguir un proceso productivo, organizado y eficiente a la hora de poder llevar a cabo las diferentes tareas en un departamento. Puede tratarse de un proyecto en

concreto o bien de un departamento interno en sí. Fomenta el trabajo en equipo, funcional y con un flujo permanente de tareas.

Kanban es una palabra japonesa, que significa panel / cartel. La idea de este sistema de organización es clasificar las tareas en 4 apartados; Por Hacer, En proceso, Pendiente de revisión, Hecho. Dentro de estos apartados, cada miembro clasifica sus tarjetas en función de las prioridades acordadas. Las herramientas, como por ejemplo Trello, facilitan la comunicación entre los miembros del equipo, dan la posibilidad de mover tarjetas entre los mismos o de crear tarjetas no solo propias, sino que pueden ser asignadas a terceros. Por otro lado, Kanban garantiza un resultado fiable y de calidad, debido a las diferentes fases con las que cuenta este tipo de sistemas. Permite reducir el caos al organizar las tareas, evitar saturaciones mediante la priorización de las tarjetas y no saturar el cuello de botella ya que en todo momento eres consciente del trabajo pendiente y el realizado.

Para poder agilizar las tareas, Kanban cuenta con cuatro principio o características básicas que lo fundamentan:

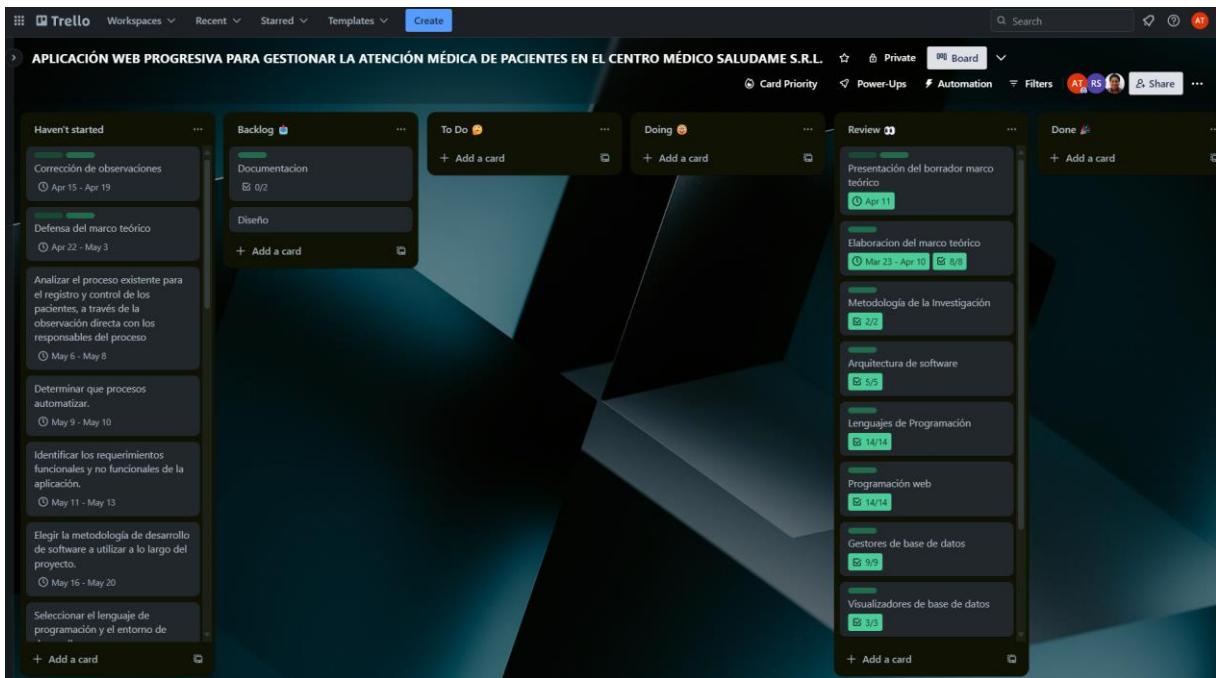
- Visualización en el Modelo Kanban: Para poder comprender en qué momento del desarrollo se encuentra el proyecto o revisar los temas tratados, Kanban es totalmente transparente, de forma que se cuenta con acceso a todas las tareas en cualquier momento. Lo cual permite organizarse en los diferentes bloques (Por Hacer, En proceso, Pendiente de revisión, Hecho) y hacer modificaciones para el buen funcionamiento del equipo. Los principales elementos visuales son la tarjeta Kanban, que contiene información como el nombre de la tarea, la descripción, la persona asignada, la prioridad, comentarios, enlaces externos, así como las subtareas asociadas; y el tablero Kanban, que es una herramienta para plasmar todo el flujo de los componentes. La forma en la que se hacía (y hoy en día se hace aún en muchos entornos) es con una pizarra blanca o tablero de corcho dividida en filas y columnas y notas adhesivas (post-its, por su traducción del inglés). Los Post-Its representaban las tarjetas Kanban y estas se podían mover por las principales columnas antes mencionadas: Por Hacer,

En proceso, Pendiente revisión y Hecho. Ese suele ser el flujo habitual, aunque cada departamento puede adaptarlo a sus necesidades e incluso dividir en subsecciones cada columna.

- Tareas en proceso: El principio 2 de la metodología Kanban dice que no se debería de trabajar con más de una tarjeta a la vez. Y esta tarjeta, se clasifica en uno de los tres bloques del tablero Kanban al finalizar la jornada: Por hacer, En Proceso, Pendiente revisión, Hecho. Durante la realización de las tareas, surgen dudas, o se ven fallos / mejoras. Kanban fomenta la continua modificación de las tareas, ya que se trata de un sistema de trabajo inmediato, compuesto por pequeñas tareas de corta duración. Centrarse en tareas de corta duración y etiquetarlas según su estado no solo permite ser más productivo, sino que permite afrontar el problema de una forma más eficaz que con una tarea más grande en la que se tiene la sensación de que no hay avance.
- Priorización de tareas en método Kanban: Gracias a este concepto, cuando se va al bloque de las tareas pendientes, ya se tiene claro cuál es el siguiente tema que tratar. La transparencia que permite ver todos los ejercicios a realizar hace que sea posible una gestión mejor del tiempo y colocar las tareas con un orden coherente para facilitar el trabajo propio y el del equipo. La mayor parte de herramientas para implementar la metodología Kanban permiten o bien ordenar arrastrando y soltando las tareas de acuerdo con la prioridad para el equipo o dependencias o incluso asignarles un valor. Además, una buena práctica, suele ser la incorporación de filas de acuerdo con diferentes prioridades/conceptos dentro de una misma columna. Otra opción es utilizar una escala de colores para priorizar.
- Medir el tiempo: Gracias al sistema de situar las tareas en “Haciendo”, durante el tiempo que se trabaja en ellas y etiquetar las tarjetas según el tema tratado, se puede hacer un seguimiento del tiempo invertido en cada función, departamento o campo. (Siderova, 2021)

A continuación, se presenta un ejemplo de un tablero Kanban enfocado en el presente proyecto.

Figura 8: Ejemplo de tablero Kanban



Fuente: Elaboración propia

## b) Scrum

Scrum es una metodología que ayuda a los equipos a colaborar y realizar un trabajo de alto impacto. La metodología Scrum proporciona un plan de valores, roles y pautas para ayudar al equipo a concentrarse en la iteración y la mejora continua en proyectos complejos. Por otra parte, en Scrum se trabaja con equipos pequeños multidisciplinares en ciclos iterativos centrados en el cliente y se crea un producto de forma incremental.

Los tres pilares básicos sobre los que se basa la metodología Scrum son: roles, eventos y artefactos. Asimismo, un proceso Scrum se ejecuta durante un sprint que, generalmente, son sesiones de trabajo de dos semanas con entregas específicas que

deben realizarse al final. Hay dos eventos adicionales que son también claves para entender qué es Scrum. Las reuniones diarias de actualización, como su nombre indica, se dan una vez al día. Estas representan una oportunidad para que el equipo de Scrum se conecte durante 15 minutos y coordine las actividades diarias. El segundo evento, el análisis retrospectivo del sprint, tiene lugar al final de cada sprint. Durante el análisis retrospectivo del sprint, que estará a cargo del Scrum Máster, el equipo de trabajo tiene la oportunidad de reflexionar con respecto al sprint y hacer ajustes para futuros sprints. Al final de cada sprint se genera un incremento que se suma al resto de producto desarrollado. (Martins, 2024)

A continuación, en la figura 9 se describen los pasos de la metodología Scrum.

**Figura 9: Pasos de la metodología Scrum**



*Fuente: (Sinnaps, 2017)*

### c) Programación Extrema (XP)

La Programación Extrema (XP, por sus siglas en inglés) es una metodología ágil de gestión de proyectos que se centra en la velocidad y la simplicidad con ciclos de desarrollo cortos. Fue desarrollada por Kent Beck a fines de la década de 1990 para

gestionar el desarrollo de un sistema de software de nómina para Chrysler llamado Proyecto C3.

A diferencia de otras metodologías ágiles, XP tiene reglas y valores estrictos que rigen cómo se realiza el trabajo; además, es muy creativa y colaborativa, ya que promueve el trabajo en equipo durante todas las etapas de desarrollo. (Raeburn, 2024).

Algunas características clave de la metodología XP son:

- Comunicación constante entre el cliente y el equipo de desarrollo.
- Respuesta rápida a los cambios constantes.
- La planificación es abierta con un cronograma de actividades flexible.
- El software que funciona está por encima de cualquier otra documentación.
- Los requisitos del cliente y el trabajo del equipo del proyecto son los principales factores de éxito de este.

Esta metodología XP está dentro de las denominadas metodologías ágiles. Sin embargo, tiene sus peculiaridades. En ella, hay unos roles de equipo definidos y unas iteraciones que se van repitiendo cada semana o 3-5 semanas.

La metodología XP se centra en la comunicación con todos los involucrados en el proyecto, así como la reutilización del código ya desarrollado y la realimentación. (Sinnaps, 2020)

A continuación, en la figura se presenta un ejemplo del flujo de trabajo de la metodología de desarrollo de software XP.

**Figura 10: Pasos de la metodología XP**



*Fuente:* (Calvo, 2018)

### 2.3.5.2. Marcos de trabajo (Frameworks) del Front End

Un marco de trabajo (framework, por su traducción del inglés) es un producto de software que simplifica el desarrollo y mantenimiento de proyectos complejos. Estos framework contienen módulos de software listos para usar que los desarrolladores pueden emplear para resolver tareas comunes de programación, como gestionar solicitudes AJAX (JavaScript asíncrono y XML, por sus siglas en inglés) o establecer una estructura de archivos. Además, establecen las reglas para construir la arquitectura de la aplicación, brindando una estructura base que puede extenderse y modificarse según los requisitos específicos.

Los frameworks pueden incluir utilidades, bibliotecas de código, lenguajes de secuencias de comandos y otros componentes de software que simplifican la integración y desarrollo de diversos elementos en un gran producto de software.

Gracias a estos, los desarrolladores ya no comienzan desde cero en sus proyectos, sino que disponen de una base sólida para implementar características específicas. Esto acelera el proceso de desarrollo, aumenta la productividad y mejora la fiabilidad de las soluciones.

En el contexto del desarrollo web, el frontend se refiere a la interfaz gráfica de usuario de un sitio web o una aplicación, es decir, la parte visible y con la que los usuarios interactúan.

Un framework frontend, por lo tanto, es un producto o herramienta de software que proporciona una base para el desarrollo de la interfaz de usuario de soluciones web. (BitDegree, 2024)

**a) React**

React es una biblioteca de JavaScript desarrollada por Facebook que se utiliza para construir interfaces de usuario interactivas y eficientes. Se centra en la creación de interfaces de usuario declarativas, lo que significa que se describe cómo debería lucir la interfaz en un estado determinado y React se encarga de actualizar automáticamente la interfaz cuando los datos cambian.

React se ha vuelto muy popular en el desarrollo web debido a su enfoque en la construcción de interfaces de usuario eficientes y su flexibilidad para trabajar con otras tecnologías y bibliotecas. Algunas de las características clave de React son:

- **Componentes:** React organiza la interfaz de usuario en componentes reutilizables, que pueden ser de distinto tipo y tamaño. De esta forma, se facilita modularidad en el código, simplificando la construcción y el mantenimiento de las aplicaciones, ya que cada componente se puede desarrollar y probar de forma aislada antes de integrarlo en el desarrollo final.
- **Estados y Desarrollo Declarativo:** React permite al desarrollador describir cómo debe verse la interfaz dependiendo del estado de la aplicación. De este modo, cuando un componente cambia de estado, React hace un seguimiento de esos

cambios y actualiza la interfaz de usuario para que aparezca acorde al nuevo estado. Esto, además de simplificar la gestión del estado, implica que la interfaz siempre estará sincronizada y actualizada con los datos de la aplicación.

- Virtual DOM (Modelo de objetos del documento, por sus siglas del inglés): React aplica las modificaciones primero en una representación virtual del DOM, de forma que posteriormente solo se apliquen al DOM real los cambios necesarios. Así, se reduce la carga en el navegador y se optimiza el rendimiento.
- JSX: React utiliza JSX (JavaScript XML), que permite escribir código que se asemeja al marcado HTML. Gracias a ello, la creación de componentes se simplifica y mejora su legibilidad.
- Unidireccionalidad de datos: esto significa que los datos solo fluyen en una dirección a través de la aplicación. Esto facilita el seguimiento y el mantenimiento, con lo que se consigue un código más predecible y fácil de depurar.
- Herramientas complementarias: React cuenta con numerosas opciones de herramientas y bibliotecas, de entre las que destacamos Redux, que facilitan y agilizan el desarrollo de plataformas en React. (López Mora, 2023)

**b) Angular**

Angular es un framework de desarrollo de aplicaciones web de código abierto y de alto rendimiento creado por Google. Proporciona una estructura y conjunto de herramientas para construir aplicaciones web dinámicas y escalables. Angular sigue el patrón de diseño Modelo-Vista-Controlador (MVC) y utiliza HTML como lenguaje de plantillas para la construcción de interfaces de usuario. Angular se utiliza para construir aplicaciones web de una sola página (Single-Page Applications, SPA) y aplicaciones web escalables. Proporciona una estructura sólida para el desarrollo de aplicaciones complejas, facilitando la separación de responsabilidades y la reutilización de código. Angular ofrece una amplia gama de características, como enlace de datos bidireccional, inyección de dependencias, enrutamiento y manejo de eventos, lo que

facilita la construcción de interfaces de usuario interactivas y dinámicas. (Imagina, 2024)

Angular ofrece una amplia gama de características y herramientas que hacen que el desarrollo de aplicaciones web sea más eficiente y productivo. A continuación, se presentan algunas de las características más destacadas de Angular:

- **TypeScript:** Angular utiliza TypeScript, un lenguaje de programación desarrollado por Microsoft. TypeScript es una mejora de JavaScript que agrega características de programación orientada a objetos y un sistema de tipos estático. Esto brinda beneficios como una mejor detección de errores, autocompletado de código y mayor legibilidad del código fuente.
- **Arquitectura de componentes:** El enfoque de Angular en la arquitectura de componentes facilita la reutilización de código y la separación de responsabilidades. Cada componente encapsula su propia lógica y vista, lo que permite desarrollar y mantener aplicaciones complejas de manera más sencilla. Además, los componentes se pueden combinar para crear interfaces de usuario más complejas.
- **Enlace de datos bidireccional:** El enlace de datos bidireccional de Angular es una característica poderosa que sincroniza automáticamente los datos entre el modelo y la vista. Esto elimina la necesidad de escribir código adicional para mantener la coherencia entre los datos y la interfaz de usuario. Los cambios en los datos se reflejan instantáneamente en la interfaz de usuario y los eventos del usuario se actualizan en el modelo de datos subyacente.
- **Inyección de dependencias:** Angular tiene un sistema de inyección de dependencias integrado que facilita la gestión de las dependencias y la creación de pruebas unitarias. La inyección de dependencias permite agregar o reemplazar fácilmente componentes y servicios en tiempo de ejecución, lo que hace que las aplicaciones sean más flexibles y mantenibles.
- **Enrutamiento:** El enrutamiento en Angular permite navegar entre diferentes vistas y componentes de manera sencilla. Esto es especialmente útil en

aplicaciones de una sola página (SPA), donde la interfaz de usuario puede cambiar dinámicamente sin necesidad de recargar la página. El enrutamiento en Angular se basa en URL amigables, lo que permite compartir enlaces directos a secciones específicas de la aplicación.

### c) **Vue.js**

Vue.js es un framework open source de JavaScript, que permite la creación de interfaces de usuario y aplicaciones de una sola página (single-page application o SPA, en inglés), de una forma muy sencilla. Fue creado o desarrollado por un ex empleado de Google, Evan You, en el año 2014. Con respecto a otros frameworks, la curva de aprendizaje es baja, si conoces los fundamentos de JavaScript.

Vue.js se ha vuelto tan popular porque ha sabido implementar lo mejor de otros frameworks, eliminando todo aquello que no aporta valor. Dejando libertad a los desarrolladores para personalizar totalmente los proyectos, generando un mayor control. Como beneficio directo, se dispone de aplicaciones muy livianas con una velocidad de carga mucho superior a otros proyectos realizados en otros frameworks. (Barragán, 2021)

Con Vue.js se implementa lo que se conoce como arquitectura de componentes. Permite dividir las aplicaciones en bloques con funcionalidades independientes, llamados componentes. Esos bloques podrían ser una cabecera, un menú, un listado, una ficha de producto, etc. En realidad, cualquier cosa que podamos necesitar puede ser un componente. Además, unos componentes se pueden apoyar en otros, de modo que en un listado de productos podemos tener fichas de productos, que a su vez pueden estar compuestas por datos, botones, desplegables con información, etc.

Gracias a la arquitectura de componentes podemos construir aplicaciones a base de piezas reutilizables, de modo que, al final, las aplicaciones no son más que árboles de componentes que trabajan entre sí para implementar funcionalidades tan complejas como sea necesario.

Otra de las ventajas de Vue.js es la posibilidad de creación de vistas reactivas. Esto quiere decir que Vue permite actualizar el HTML y CSS (la parte visual de la página) cuando cambian los datos de la aplicación, sin que el desarrollador tenga que invertir tiempo en propagar esos cambios de manera manual en cada lugar de la página donde se visualizan los datos que se alteraron. (García de Zúñiga, 2020)

#### **2.3.5.3. Marcos de trabajo (Frameworks) del Back End**

Backend es un término desarrollo web que hace referencia a un tipo de programación particular, en el que se configuran todos los aspectos lógicos de una página web o aplicación. Para algunos, el backend es la programación de todo lo que el usuario final no ve, es decir, el acceso a las bases de datos, el procesamiento de los datos ingresados por los usuarios, y la ejecución de un script, por ejemplo. (Machuca, 2022)

Un framework back-end, por otro lado, se enfoca en el desarrollo de la lógica del servidor y la gestión de la base de datos. Estos frameworks permiten a los desarrolladores crear la parte "invisible" de un sitio web o una aplicación, es decir, la parte que se ejecuta en el servidor y maneja las solicitudes de los usuarios. (codigocreativo, 2023)

##### **a) Express.js**

Express.js, a veces también llamado «Express», es un framework de backend Node.js minimalista, rápido y similar a Sinatra, que proporciona características y herramientas robustas para desarrollar aplicaciones de backend escalables. Ofrece el sistema de enrutamiento y características simplificadas para ampliar el framework con componentes y partes más potentes en función de los casos de uso de la aplicación.

El framework proporciona un conjunto de herramientas para aplicaciones web, peticiones y respuestas HTTP, enrutamiento y middleware para construir y desplegar aplicaciones a gran escala y preparadas para la empresa. También proporciona una herramienta de interfaz de línea de comandos (CLI) llamada Node Package Manager (NPM), donde los desarrolladores pueden obtener paquetes desarrollados. También

obliga a los desarrolladores a seguir el principio de No te repitas (Don't Repeat Yourself, por sus siglas en inglés).

El principio DRY pretende reducir la repetición de patrones de software, sustituyéndolos por abstracciones, o utilizando normalizaciones de datos para evitar la redundancia. (Kinsta, 2019)

Algunas características de Express.js son:

- Simple y fácil de usar: Express.js es ideal incluso para los programadores noveles.
- Un solo lenguaje utilizado: Aunque está categorizado como backend, se puede considerar un desarrollo full-stack porque utiliza un único lenguaje tanto para el frontend como para el backend. Esto permite a los desarrolladores construir y terminar aplicaciones en el menor tiempo posible. También simplifica el proceso.
- Utiliza la plataforma Node.js: Al estar basado en Node.js, Express es más adaptable.
- Funcionalidad del backend: Express.js se encarga del enrutamiento, las sesiones, el manejo de errores y otros detalles del backend.
- Framework no dogmático: El marco Express facilita la integración. Permite a los desarrolladores elegir libremente qué servicios o middleware de terceros utilizar. (Leroux, 2021)

**b) Django**

Django es un framework de desarrollo para Python que se emplea para la creación de páginas web. Se trata de una herramienta de código abierto y gratuita que cuenta con una comunidad amplia y que comparte recursos constantemente. Además, Django también cuenta con funciones de pago que pueden facilitar más el trabajo de los desarrolladores.

Django es una herramienta que se puede usar para el desarrollo full-stack de aplicaciones y páginas web, así como para el desarrollo de servidores. Está considerado como el mejor framework para el desarrollo de aplicaciones web con Python y es uno de los marcos de desarrollo más demandados por los programadores que trabajan con este lenguaje en el desarrollo web. (Canle Fernández, 2022a)

Algunas de sus características principales son:

- Arquitectura MVC y ORM: Django sigue el patrón de diseño Modelo-Vista-Controlador (MVC), que organiza el código en modelos (representación de datos), vistas (presentación y lógica de usuario) y controladores (manejo de la lógica de negocios). Además, Django utiliza un mapeo objeto-relacional (ORM) que facilita la interacción con la base de datos utilizando objetos Python, eliminando la necesidad de escribir SQL directamente.
- Sistema de Plantillas: Django ofrece un sistema de plantillas que simplifica la presentación de datos en las páginas web. Las plantillas permiten la mezcla de HTML con código Python, proporcionando una forma eficiente y legible de construir interfaces de usuario dinámicas y atractivas.
- Administrador de Django: El administrador de Django es una interfaz de administración preconstruida que permite a los desarrolladores y administradores gestionar fácilmente los datos de la aplicación. Esta herramienta automatiza tareas comunes relacionadas con la administración de la base de datos, proporcionando una interfaz intuitiva para la gestión de contenido.
- Autenticación y Autorización de Usuarios: Django simplifica la implementación de sistemas de autenticación y autorización de usuarios. Ofrece un conjunto de herramientas que permiten gestionar registros de usuarios, recuperación de contraseñas y control de acceso basado en roles, brindando seguridad y flexibilidad a las aplicaciones.
- Enrutamiento y Control de URL: El enrutamiento en Django se gestiona a través de un sistema de control de URL. Esto permite mapear las URL a funciones

específicas en las vistas, facilitando la navegación y la interacción del usuario con la aplicación. (codigovanguardia, 2023)

### c) **ASP.NET**

ASP.NET es un framework open source multiplataforma creado por Microsoft. Es decir, estamos ante un entorno de trabajo basado en código abierto que está pensado para el desarrollo y la ejecución de aplicaciones y servicios web modernos. Al ser multiplataforma es posible ejecutar aplicaciones ASP.NET tanto en Windows, Linux y macOS como en contenedores. (García Carmona, 2022)

Las características clave de ASP.NET son:

- **Procesamiento en el servidor:** En ASP.NET, gran parte del procesamiento de la aplicación ocurre en el servidor. Esto significa que las solicitudes de los clientes son manejadas por el servidor, que procesa la lógica de la aplicación y genera la respuesta enviada de vuelta al cliente. Este enfoque minimiza la carga en los dispositivos del usuario y proporciona un rendimiento más consistente.
- **Interacción con el cliente:** A pesar de que gran parte del procesamiento se realiza en el servidor, ASP.NET permite una interacción dinámica con el cliente mediante el uso de tecnologías como AJAX. Esto posibilita actualizaciones de página parciales sin tener que recargar toda la página, mejorando la experiencia del usuario.
- **Librerías y herramientas disponibles:** El marco incluye una amplia gama de librerías y herramientas que facilitan tareas comunes en el desarrollo web, como la manipulación de URL, el manejo de sesiones y la gestión de formularios. Estas herramientas permiten a los desarrolladores concentrarse en la lógica de la aplicación en lugar de preocuparse por detalles de bajo nivel.
- **Soporte para múltiples lenguajes de programación:** ASP.NET es compatible con lenguajes como C#, Visual Basic, F#, y más. Esto permite a los equipos de desarrollo elegir el lenguaje que mejor se adapte a sus habilidades y

preferencias, fomentando la colaboración y la eficiencia en el desarrollo. (A. M. Lopez, 2024)

**d) Spring Boot**

Spring Boot es una herramienta de desarrollo que se basa en Spring Framework, un marco de trabajo empresarial de código abierto que se utiliza para crear aplicaciones autónomas de producción en Java. Se enfoca en simplificar y acelerar el desarrollo de aplicaciones web y microservicios, ofreciendo una configuración automática y la capacidad de crear aplicaciones que se ejecutan de forma independiente sin necesidad de un servidor web externo.

Algunas de las características de Spring Boot más destacadas son:

- **Configuración automática:** Spring Boot utiliza la configuración automática para inicializar las aplicaciones con dependencias predefinidas, eliminando la necesidad de configuración manual y acelerando el proceso de desarrollo.
- **Enfoque obstinado:** Spring Boot elige y configura automáticamente las dependencias de iniciador según las necesidades del proyecto, evitando que el usuario tenga que tomar todas las decisiones y configurar manualmente.
- **Aplicaciones autónomas:** Spring Boot permite crear aplicaciones que se ejecutan de forma independiente sin depender de un servidor web externo. Durante el proceso de inicialización, se integra un servidor web en la aplicación, lo que le permite funcionar de manera autónoma.
- **Flexibilidad:** A pesar de simplificar el proceso de desarrollo, Spring Boot no sacrifica la flexibilidad que ofrece Spring Framework. Los desarrolladores pueden utilizar todas las funciones de Spring Framework y agregar fácilmente dependencias adicionales en la aplicación. (Bustos, 2023)

#### 2.3.5.4. Calidad del software

El Aseguramiento de la Calidad del Software (QA, Quality Assurance por sus siglas en inglés) es un conjunto de prácticas, metodologías y actividades adoptadas en el proceso de desarrollo con el objetivo de garantizar que el software producido cumpla con altos estándares de calidad. El QA en el contexto del desarrollo de software es esencial para reducir defectos, mejorar la confiabilidad del software, garantizar la seguridad y satisfacer al cliente. Algunas de las principales actividades involucradas en el Aseguramiento de la Calidad del Software en el proceso de desarrollo son:

- Definición de estándares de calidad: antes de comenzar el desarrollo, es necesario establecer estándares de calidad claros que el software debe cumplir. Estos estándares pueden incluir aspectos como funcionalidad, confiabilidad, rendimiento y seguridad.
- Planificación de QA: se debe planificar cómo se incorporará el QA en el ciclo de desarrollo. Esto incluye la planificación de actividades de prueba, la definición de criterios de aceptación y la estimación de recursos necesarios.
- Revisión y control de requisitos: un paso crucial es revisar los requisitos del software para asegurarse de que sean claros, completos, correctos y coherentes. Esto evita cambios costosos en etapas posteriores del desarrollo.
- Pruebas: las pruebas son una parte fundamental del Aseguramiento de la Calidad. Incluyen pruebas unitarias, pruebas de integración, pruebas de sistema y pruebas de aceptación para verificar que el software funcione correctamente y cumpla con los requisitos.
- Automatización de pruebas: la automatización de pruebas permite realizar pruebas repetitivas de manera eficiente, mejorando la cobertura de las pruebas y la velocidad de entrega.
- Gestión de defectos: cuando se identifican defectos, el Aseguramiento de la Calidad ayuda a documentarlos adecuadamente y coordinar las correcciones. La gestión de defectos puede incluir la priorización según la gravedad y su resolución.

- Seguridad: el Aseguramiento de la Calidad del Software debe tener en cuenta la seguridad del software, identificando y mitigando posibles vulnerabilidades y riesgos de seguridad.
- Rendimiento: asegurarse de que el software sea eficiente y cumpla con los requisitos de rendimiento es otro aspecto importante del QA. Esto puede incluir pruebas de carga y rendimiento.
- Documentación: el QA requiere la creación de documentación detallada de pruebas, resultados y procesos para rastrear las actividades realizadas y los resultados obtenidos.
- Mejora continua: el QA es un proceso continuo de mejora. Las lecciones aprendidas de lanzamientos anteriores y los comentarios de los clientes se utilizan para mejorar constantemente el proceso de desarrollo.

Los objetivos clave para el QA son:

- Cumplimiento de Estándares: Asegurar que el software se adhiera a los estándares de calidad establecidos.
- Satisfacción del Usuario: Satisfacer las necesidades y expectativas de los usuarios finales.
- Prevención de Defectos: Implementar prácticas para reducir la incidencia de defectos en el software.
- Mejora Continua: Fomentar la mejora continua en los procesos de desarrollo de software. (W&B Asset Studio, 2023)

Los siguientes son estándares ampliamente reconocidos y utilizados en la industria con relación a la calidad del software:

- ISO 9001. Norma para la implementación de un método o Sistema de Gestión de la Calidad (SGC), supone la acreditación de la capacidad para satisfacer los requisitos de calidad. Aporta así una serie de requisitos genéricos (no circunscritos al software) y aplicables a cualquier organización.

- ISO 10005:2018. Ofrece una guía para gestionar un plan de calidad todo el ciclo de vida.
- ISO IEC 25000. Supone una familia de normas también conocida como SQuaRE (Software product Quality Requirements and Evaluation), define un marco de referencia para la calidad del producto de software. Evalúa un software en 8 áreas, incluyendo Adecuación funcional, fiabilidad, usabilidad, eficiencia, compatibilidad, seguridad, mantenibilidad y portabilidad.
- ISO 33000 Calidad de los procesos de desarrollo de software. Se enfoca en la evaluación de la calidad de los procesos de desarrollo de software, y también se lo conoce como SPICE (Software Process Improvement and Capability Determination). Así, busca conocer la evolución en el tiempo sobre los procesos de desarrollo, hacer un seguimiento respecto a la competencia y determinar posibles estrategias de mejora.
- CMMI (Capability Maturity Model Integration). Proporciona un marco de referencia para evaluar y mejorar la madurez de los procesos en el desarrollo de software. Se centra en áreas como la gestión de proyectos, de la calidad, la configuración, o riesgos, entre otros. (Icaria Technology, 2023)

### 2.3.6. Gestores de base de datos

Una base de datos es una colección de información (datos), generalmente relacionada, que se almacena electrónicamente y en la que se puede buscar. La función de una base de datos no es únicamente la de almacenar datos, sino que también debe hacerlo en un formato que permita una búsqueda eficiente y una recuperación rápida de la información, además de garantizar su seguridad.

Las funciones en sí de la base de datos son procedimientos que realizan operaciones particulares sobre los datos contenidos. Las funciones de la base de datos incluyen operaciones básicas de CRUD, que es un acrónimo para las siglas en inglés de crear, leer, actualizar y eliminar datos. En diferentes lenguajes de consulta, estas

operaciones básicas pueden tener nombres diferentes; por ejemplo, insert en lugar de create para introducir un registro en el sistema.

Una base de datos se gestiona por medio de un sistema de administración de bases de datos que puede ser normal (DBMS) o relacional (RDBMS). Los sistemas de bases de datos son una combinación de hardware físico que almacena datos, un software complejo DBMS o RDBMS y diferentes lenguajes informáticos para acceder y manipular la información. (Paessler, 2022)

#### **a) Relacionales**

Una base de datos relacional es una recopilación de información que se organiza de tal manera que establece relaciones definidas para facilitar el acceso a los datos. En el modelo de una base de datos relacional, las estructuras de datos, incluidas las tablas de datos, los índices y las vistas, permanecen separadas de las estructuras de almacenamiento físico. Esto permite que los administradores de estas puedan editar los datos sin afectar directamente a cómo se organiza la estructura de la información.

Uno de los puntos clave de este tipo de bases de datos es que se trata de una forma de almacenar información que funciona bien con datos estructurados. Es decir, con datos que tienen un formato estandarizado y una estructura bien definida.

Una base de datos relacional funciona almacenando y recopilando información que relaciona unos datos con otros. Cada fila de una tabla en estas bases de datos contiene un registro con un identificador único y cada columna de la tabla tiene los atributos asociados a esos datos. De esta manera, cada registro es capaz de asignar un valor a cada una de las funciones, lo cual facilita la identificación de las relaciones entre los datos. Para consultar una base de datos relacional es necesario emplear un lenguaje de consulta estructurado. Además, para acceder a la información que se ha recopilado en este tipo de base de datos es necesario seguir unas reglas determinadas. Estas reglas definen y garantizan que la base de datos sea precisa y accesible. (Canle Fernández, 2022b)

## 1) MySQL

MySQL es un sistema de gestión de bases de datos relacional que, en los últimos años, ha logrado posicionarse como una de las mejores opciones para el desarrollo web gracias a su naturaleza de código abierto, a su solidez y su gran flexibilidad. La elección del nombre, derivado de "My" en honor a la hija del cofundador, Michael Widenius, y SQL (Structured Query Language, lenguaje de consulta estructurado por sus siglas del inglés), refleja su enfoque estructurado y centrado en consultas.

Además, destaca por su excepcional rendimiento, ofreciendo un alto nivel de eficiencia en el procesamiento de datos. Pero también por su fiabilidad y su capacidad para adaptarse a distintas necesidades, lo que la convierte en una herramienta imprescindible para construir aplicaciones sólidas y eficientes. (M. Lopez, 2024)

MySQL presenta algunas ventajas que lo hacen muy interesante para los desarrolladores. La más evidente es que trabaja con bases de datos relacionales, es decir, utiliza tablas múltiples que se interconectan entre sí para almacenar la información y organizarla correctamente.

Al ser basada en código abierto es fácilmente accesible y la inmensa mayoría de programadores que trabajan en desarrollo web han pasado usar MySQL en alguno de sus proyectos porque al estar ampliamente extendido cuenta además con una ingente comunidad que ofrece soporte a otros usuarios. Pero estas no son las únicas características como veremos a continuación:

- **Arquitectura Cliente y Servidor:** MySQL basa su funcionamiento en un modelo cliente y servidor. Es decir, clientes y servidores se comunican entre sí de manera diferenciada para un mejor rendimiento. Cada cliente puede hacer consultas a través del sistema de registro para obtener datos, modificarlos, guardar estos cambios o establecer nuevas tablas de registros, por ejemplo.
- **Compatibilidad con SQL:** SQL es un lenguaje generalizado dentro de la industria. Al ser un estándar MySQL ofrece plena compatibilidad por lo que si

se ha trabajado en otro motor de bases de datos no habrá problemas en migrar a MySQL.

- **Vistas:** Desde la versión 5.0 de MySQL se ofrece compatibilidad para poder configurar vistas personalizadas del mismo modo que podemos hacerlo en otras bases de datos SQL. En bases de datos de gran tamaño las vistas se hacen un recurso imprescindible.
- **Procedimientos almacenados.** MySQL posee la característica de no procesar las tablas directamente, sino que a través de procedimientos almacenados es posible incrementar la eficacia de nuestra implementación.
- **Desencadenantes.** MySQL permite además poder automatizar ciertas tareas dentro de nuestra base de datos. En el momento que se produce un evento otro es lanzado para actualizar registros u optimizar su funcionalidad.
- **Transacciones.** Una transacción representa la actuación de diversas operaciones en la base de datos como un dispositivo. El sistema de base de registros avala que todos los procedimientos se establezcan correctamente o ninguna de ellas. En caso por ejemplo de una falla de energía, cuando el monitor falla u ocurre algún otro inconveniente, el sistema opta por preservar la integridad de la base de datos resguardando la información. (Robledano, 2019)

## 2) **PostgreSQL**

PostgreSQL es un sistema para gestionar bases de datos de muy alto nivel, completamente de software libre y con una licencia BSD (Berkeley Software Distribution, Distribución de software de Berkeley por sus siglas en inglés), compatible con cualquier uso, ya sea personal o comercial.

PostgreSQL es un sistema considerado como empresarial. Tratándose de la aplicación de bases de datos más avanzado de código abierto, podría utilizarse por su funcionalidad y potencia como reemplazo de otras bases de datos comerciales, incluso del poderoso Oracle. Hasta cierto punto, resulta extraño que no sea el más popular, condición que ostenta MySQL. PostgreSQL tiene dos ventajas fundamentales, primero

en lo que respecta a su funcionalidad y capacidad de trabajar con mayores cantidades de datos, pero también en lo que respecta a su licencia. MySQL tiene una licencia dual, lo que significa que para proyectos comerciales habría que pagar por su uso. Sin embargo, PostgreSQL tiene una única licencia totalmente abierta para cualquier uso. (Arsys, 2018)

Las características más importantes de PostgreSQL son:

- El lenguaje SQL que usa es muy próximo al estándar ISO/IEC, gracias a lo que resulta relativamente sencillo portar consultas y scripts de otros sistemas de bases de datos, y así aprender fácilmente las variantes de este lenguaje.
- Cumple con ACID, es decir provee atomicidad, consistencia, aislamiento y durabilidad para sus operaciones.
- Permite crear esquemas, tablas heredadas y triggers orientados a eventos que no poseen otros motores.
- Permite definir procedimientos, no solo en PostgreSQL, sino también en otros muchos lenguajes como Pearl, TCL o Python. Incluso si lenguaje a usar no está soportado, se puede definir con nuevas extensiones.
- Permite extender la funcionalidad con extensiones, provistas por la propia PostgreSQL, por terceros o incluso programando por cuenta propia.
- Tiene un soporte nativo de replicación maestro-esclavo, pero también es posible añadir otros tipos a través de productos de terceros, libres o de pago.
- También provee una excelente escalabilidad vertical. (Gonzalez Gil, 2018)

### 3) **SQLite**

SQLite es una herramienta de software libre, que permite almacenar información en dispositivos empotrados de una forma sencilla, eficaz, potente, rápida y en equipos con pocas capacidades de hardware, como puede ser una PDA o un teléfono celular. SQLite implementa el estándar SQL92 y también agrega extensiones que facilitan su uso en cualquier ambiente de desarrollo. Esto permite que SQLite soporte desde las consultas más básicas hasta las más complejas del lenguaje SQL, y lo más importante

es que se puede usar tanto en dispositivos móviles como en sistemas de escritorio, sin necesidad de realizar procesos complejos de importación y exportación de datos, ya que existe compatibilidad al 100% entre las diversas plataformas disponibles, haciendo que la portabilidad entre dispositivos y plataformas sea transparente. (Rómmel, 2007)

SQLite presenta varias características específicas que la convierten en una opción interesante para el desarrollo de pequeñas aplicaciones que manejen bases de datos.

- Multisistema. La biblioteca SQLite se incluye en los sistemas operativos más utilizados, como Windows, Linux, Android y los sistemas de Apple (iOS y macOS).
- Código abierto. El uso de SQLite no requiere del pago de una licencia.
- Multilenguaje. Dispone de diferentes API (Application Programming Interface, Interfaz de Aplicación Programable por sus siglas en inglés) que le permiten trabajar con lenguajes de programación como C++, Python o PHP, entre otros.
- Soporta múltiples tablas, índices y vistas.
- No necesita configuración ni administración.
- Sencillez. SQLite dispone de una API que es muy simple, por lo que su uso es muy fácil y no requiere de grandes conocimientos técnicos.
- Autonomía. No tiene dependencias externas. (hostingplus, 2021)

## b) No Relacionales

Una base de datos no relacional, también conocida como base de datos NoSQL (Not Only SQL), es un tipo de sistema de gestión de bases de datos diseñado para manejar y almacenar datos de manera flexible y escalable, sin seguir el modelo de tablas y relaciones de las bases de datos relacionales tradicionales. A diferencia de las bases de datos relacionales, las bases de datos no relacionales permiten almacenar y recuperar datos de forma más libre y menos estructurada. Están diseñadas para manejar grandes volúmenes de datos no estructurados o semi estructurados, como documentos, gráficos, clave-valor o columnas.

Las bases de datos no relacionales se utilizan en escenarios donde la escalabilidad, la velocidad y la flexibilidad son prioritarias. Son especialmente adecuadas para aplicaciones web y móviles con grandes cantidades de datos y altos requerimientos de rendimiento, como redes sociales, sistemas de gestión de contenido y aplicaciones en tiempo real. (Universidad Francisco de Vitoria, 2023).

Según (Rodriguez, 2021), con respecto a los formatos que se utilizan en las bases de datos no relacionales, podríamos decir que el formato más popular es el del documento. En muchos casos, lo que se utiliza es un objeto con una clave y un valor para que el acceso a la información sea pueda realizar de una forma sencilla. Los principales sistemas gestores de bases de datos no relacionales son: MongoDB, Redis y Cassandra.

### **2.3.7. Visualizadores de datos**

La visualización de datos es la representación de datos mediante el uso de gráficos comunes, como cuadros, diagramas, infografías e incluso animaciones. Estas representaciones visuales de información comunican relaciones complejas de datos y perspectivas (insights) en datos de una manera fácil de entender.

La visualización de datos se puede utilizar para diversos propósitos, y es importante tener en cuenta que su uso no está reservado únicamente para los equipos de datos. La gerencia también la aprovecha para transmitir la estructura y jerarquía organizativa, mientras que los analistas de datos y los científicos de datos la utilizan para descubrir y explicar patrones y tendencias. (Berinato, 2016) clasifica la visualización de datos en cuatro propósitos clave: generación de ideas, ilustración de ideas, descubrimiento visual y visualización de datos cotidiana. (IBM, 2023)

Estas representaciones pueden ser herramientas eficaces para la comunicación y la colaboración, y aportan más valor a los informes, el periodismo, las aplicaciones o cualquier contexto en el que se requiera compartir la información. (Microsoft, 2018)

## a) Tableau

Tableau es una herramienta de visualización de datos potente utilizada en el área de la Inteligencia de negocios (más conocida como Business Intelligence). Simplifica los datos en bruto en un formato muy fácil de entender.

La esencia de Tableau es simple y a la vez muy relevante: ayudar a las personas y empresas a ver y comprender todos sus datos. Y esto lo consigue ofreciendo a los usuarios toda una selección de herramientas útiles e intuitivas de inteligencia de negocios. A través de funciones simples como la de arrastrar y soltar, cualquier persona puede acceder y analizar de forma sencilla datos, e incluso, crear informes y compartir esta información con otros usuarios. (SPNet, 2023)

Tableau ofrece cuatro herramientas principales, las cuales son:

- Tableau Desktop: Es la versión principal de la plataforma. Es considerado un modelo de excelencia en análisis visual, es donde se realiza el análisis de datos. Posee una interfaz sencilla de utilizar. Permite importar datos de distintos servidores, de la nube o del big data y compaginarlas sin necesidad de escribir un código. Fomenta la realización de análisis más profundos a partir de los datos.
- Tableau Cloud: Es una plataforma basada en la nube de Tableau. Es una plataforma de arquitectura simple, rápida y que permite autogestionarse. Es adaptable a las necesidades de cada empresa. Permite tomar decisiones con base en los datos y a la manera eficaz de ordenarlos para interpretarlos.
- Tableau Prep: Su principal propósito es facilitar la limpieza de datos y su organización, facilitando a los usuarios y analistas comenzar más rápido sus análisis. Ofrece una visión completa de todos los datos.
- Tableau Server: Esta herramienta permite alojar todos los datos en la nube de Tableau, ofreciendo un entorno seguro para almacenar, administrar y realizar los análisis de los datos de las empresas. Hace hincapié en la seguridad del sitio y en la tranquilidad de tener los datos protegidos.

Las principales funcionalidades de Tableau son:

- Importación de datos: Tableau permite ingresar a la plataforma datos de distintas fuentes, ya sean archivos en distintos formatos, como PDF, o conectarse a distintas plataformas de almacenamiento de datos.
- Almacenamiento de datos: Permite ir controlando que funcione correctamente la automatización de recolección de datos, monitoreando los datos en vivo y programar alertas que avisen cuando falla alguna conexión.
- Colaboración entre miembros: Su interfaz permite y fomenta la colaboración entre los distintos miembros de una empresa de manera segura, gracias a sus distintas herramientas.
- Optimización para uso en móviles: Permite la creación de cuadros que se optimizan para cada dispositivo, sin importar si es móvil o un computador.

Con respecto al tema de precios y versiones, Tableau ofrece distintos planes, dependiendo el tipo de usuario, ya que diferencia entre usuario Creator, Explorer o Viewer. Esto cuesta cada plan:

- Plan Creator: Su valor es de 70U\$S por usuario y por mes. En este plan se incluye una licencia creator y acceso a las herramientas Desktop, Prep y Server.
- Plan Explorer: Su valor es de 42U\$S por usuario y por mes. Solamente incluye una licencia Explorer para la herramienta Server. Lo que sí, exige un mínimo de 5 usuarios.
- Plan Viewer: Su valor es de 15U\$S por usuario y por mes. La particularidad es que requiere un mínimo de 100 usuarios. Incluye licencia Viewer para Server. (Mancuzo, 2023)

**b) Power BI**

Power BI es una solución de análisis empresarial basado en la nube, que permite unir diferentes fuentes de datos, analizarlos y presentar un análisis de estos a través de informes y paneles. Con Power BI se tiene de manera fácil acceso a datos dentro y

fueras de la organización casi en cualquier dispositivo. Estos análisis pueden ser compartidos por diferentes usuarios de la misma organización; por lo que directivos, financieros, comerciales, etc., pueden disponer de la información del negocio en tiempo real.

Se conforma fundamentalmente de estos componentes:

- Power BI Desktop: aplicación gratuita de escritorio para transformar, visualizar datos y crear informes de estos.
- Power BI Service: servicio online con funcionalidad similar a la aplicación desktop y permite publicar informes y configurar la actualización de datos automáticamente para que el personal de la organización tenga los datos actualizados.
- Power BI Mobile: aplicación móvil disponible para Windows, iOS y Android para visualizar informes y que se actualiza automáticamente con los cambios de los datos. (Closed Menendez, 2020)

Algunas ventajas de Power BI son:

- La información se actualiza en tiempo real.
- Permite visualizar datos de forma sencilla sin necesitar conocimientos específicos de programación por lo que la experiencia del usuario es muy buena.
- Es gratuito, con versión en la nube y descargable en escritorio, aunque también hay opción de pago.
- Colaboración: permite el trabajo colaborativo entre diferentes departamentos de una empresa.
- Acceso desde diferentes dispositivos: tanto Android como iOS.
- Buena integración con multitud de plataformas y herramientas de otros fabricantes.

- Polivalente: sus amplias posibilidades la convierten en una herramienta útil para diferentes departamentos de una empresa (Recursos Humanos, Contabilidad, Logística, Ventas, etc.). (Universidad Internacional de La Rioja, 2021)

Power BI ofrece una variedad de opciones de licencia para satisfacer las necesidades de todos los usuarios, las cuales son las siguientes:

- Power BI Gratis: Power BI Desktop y Mobile son versiones gratuitas de la herramienta que brindan a los usuarios acceso básico a funcionalidades de análisis de datos. Aunque son gratuitas, estas versiones ofrecen prestaciones significativas, incluyendo la capacidad de recopilar y analizar datos de múltiples fuentes, así como la exportación de informes en varios formatos como CSV, Excel y PDF. Sin embargo, algunas limitaciones importantes incluyen la incapacidad de compartir informes con otros usuarios y la ausencia de funciones avanzadas como el análisis en Excel integrado y las suscripciones por correo electrónico.
- Power BI Pro: Es la opción ideal para aquellos que requieren funcionalidades avanzadas y un entorno de análisis de datos más robusto. Con un precio mensual de 9,40€ por usuario, esta versión ofrece características adicionales como la capacidad de crear y compartir espacios de trabajo, embeder paneles visuales en otras aplicaciones y agregar capas de seguridad a los datos. Además, Power BI Pro permite la actualización de datos incrementales, la publicación de informes para compartir y el almacenamiento de hasta 10 GB por usuario. Esta versión está incluida en el paquete de Microsoft 365 E5 y está diseñada para satisfacer las necesidades de uso profesional.
- Power BI Premium: Para organizaciones que requieren capacidades avanzadas de análisis de datos y administración, Power BI Premium es la opción ideal. Esta versión incluye todas las funcionalidades de Power BI Pro, además de características adicionales como el uso de IA avanzada para la preparación de datos y la simplificación de la administración de datos a cualquier escala empresarial. Con opciones de licencia por usuario o por capacidad, Power BI

Premium ofrece un rendimiento superior, con un tamaño máximo de conjunto de datos de hasta 100 GB por usuario y hasta 400 GB en la opción de capacidad. Otras características exclusivas de Power BI Premium incluyen distribución geográfica, aislamiento y réplicas de solo lectura para una mayor seguridad y rendimiento. (Evotic, 2023)

### **2.3.8. Evaluación y preparación de proyectos**

La evaluación de los proyectos es un proceso de valoración en el cual se analizan todos los elementos que intervienen en el proyecto con el fin de determinar su viabilidad y eficacia, calcular los posibles riesgos y determinar las respuestas. Se trata de una fase fundamental, con independencia de sus características y tamaño del proyecto en cuestión. No obstante, el término evaluar implica mucho más que valorar. Este concepto supone la recogida y análisis de datos de manera continua.

La importancia de la evaluación de proyectos radica en que supone implementar un seguimiento y control que permita establecer una comparación para poder determinar y medir la evolución del proyecto, detectar desviaciones y necesidades y establecer las medidas de mejora necesarias a lo largo del proceso. Además, todo proceso de evaluación precisa de un monitoreo, un seguimiento y control continuo que permita:

- Comprobar que el proyecto evoluciona conforme al plan diseñado.
- Detectar amenazas y oportunidades, anticiparse a ellas y tomar las decisiones oportunas en cada momento.

Una evaluación continua y bien definida puede prevenir algunas de las causas por las que fracasan los proyectos y contribuye en la consecución de los objetivos propuestos. (Pérez, 2021)

#### **2.3.8.1. Viabilidad Técnica**

La viabilidad técnica se refiere a la posibilidad de realizar un proyecto o implementar una idea con éxito desde el punto de vista tecnológico. En otras palabras, se trata de

evaluar si las herramientas, los conocimientos técnicos y la energía necesarios para hacer realidad la idea están disponibles y son factibles.

Analizar la viabilidad técnica ayuda a identificar posibles obstáculos y a determinar si el proyecto vale la pena. Es un ejercicio que implica examinar las leyes naturales que rigen el proyecto, evaluar las necesidades energéticas y los recursos disponibles, y analizar las características tecnológicas y las herramientas requeridas para el éxito del proyecto. (Nopal Consulting, 2023)

### **a) Análisis FODA**

Un análisis FODA (Fortalezas, Oportunidades, Debilidades y Amenazas) es una herramienta diseñada para comprender la situación de un negocio a través del análisis de sus fortalezas, oportunidades, debilidades y amenazas.

El análisis FODA es una herramienta muy valiosa para cualquier y resulta fundamental para la toma de decisiones actuales y futuras pues da la pauta para conocer lo que se está haciendo bien y todo aquello que representa un reto actual o potencial.

El análisis FODA permite tener un panorama más amplio, crea un diagnóstico certero y útil para detectar ventajas competitivas, problemas internos y externos, determinar el curso que deberá seguir la compañía y difundir mejor las características de valor del negocio, tanto a los miembros del equipo como a los clientes.

El análisis FODA utiliza las fortalezas, oportunidades, debilidades y amenazas, como pilares que dan contexto a la situación de la empresa. A nivel interno se valoran las fortalezas y debilidades; a nivel externo, se considera el posible impacto de las amenazas y oportunidades.

Además de esto, existen tres características secundarias útiles para el análisis FODA.

- **Subjetividad:** El análisis FODA no es una herramienta objetiva, ya no depende simplemente de informes financieros, cálculos o fórmulas matemáticas, sino

que es realizada por personas, las cuales determinan qué elementos son clave para el futuro de la organización.

- Iniciativa: Una matriz FODA no aporta resultados decisivos, sino que brinda información para el desarrollo de estrategias. Estas deben ser recopiladas y jerarquizadas para su posterior ejecución, con base en criterios de decisión complementarios.
- Temporalidad: El análisis FODA hace referencia a un momento concreto de la organización o de un proyecto en específico. Debe hacerse con regularidad ya que la empresa y el mercado son entes que cambian constantemente, dependiendo de la tecnología, de los clientes y de circunstancias sociales.

Existen dos tipos de matriz FODA:

- Análisis FODA de empresa o proyecto: Es el más común y se centra en las fortalezas, oportunidades, debilidades y amenazas relacionadas con una empresa, negocio, institución o proyecto. Para realizarlo debe tenerse en cuenta una perspectiva general que incluya empleados, proveedores, clientes y competidores. El objetivo de realizar este análisis FODA es hallar los puntos más relevantes de cada área y definir las estrategias que se llevarán a cabo para optimizarlas.
- Análisis FODA personal: Esta herramienta también se utiliza a nivel personal, pues es un recurso fácil de implementar y muy eficaz. Con un FODA personal se pueden revisar los objetivos personales y profesionales, y definir las circunstancias particulares de cada individuo. Para que este análisis sea verdaderamente útil es recomendable realizar esta introspección de manera realista y autocrítica.

Un análisis FODA se compone de cuatro cuadrantes, las cuales son:

- Fortalezas: Las fortalezas representan los puntos fuertes, ventajas, méritos, atributos y diferenciadores de una empresa. Detectar los aspectos positivos sirve para sentar las bases del futuro de la organización.

- **Oportunidades:** Las oportunidades representan todas las posibilidades, aptitudes, eventualidades de las cuales la empresa puede beneficiarse. Son todos aquellos aspectos que pueden ayudar a una organización a alcanzar sus metas.
- **Debilidades:** Las debilidades de una empresa en el análisis FODA representan los defectos, carencias o vicios que ocurren dentro de la organización y la colocan en un punto desfavorable en comparación con sus competidores. La falta de capacitación, equipo o tecnología son factores que ponen en desventaja a cualquier empresa.
- **Amenazas:** Las amenazas son problemas, desafíos, obstáculos o dificultades por los que puede atravesar una empresa. Las amenazas pueden provenir por cualquier tipo de situación; por ejemplo, cambios en el mercado o hasta crisis globales, como la que se experimentó con la pandemia del COVID-19. Estas situaciones pueden provocar conflictos o hasta poner en riesgo la permanencia de la organización. (Pursell, 2024)

#### **2.3.8.2. Viabilidad económica**

La viabilidad económica determina el potencial que tiene un proyecto empresarial y es la base sobre la que se debe edificar cualquier negocio. Para determinar la viabilidad de cualquier organización es preciso analizar aspectos técnicos, económicos y comerciales con la finalidad de valorar el retorno de la inversión. Este estudio completo permite tomar decisiones objetivas y elaborar una planificación estratégica de una empresa en base a las fortalezas, debilidades, oportunidades y amenazas asociadas al negocio y su mercado.

Analizar la viabilidad de un proyecto empresarial es un proceso exhaustivo que debe tener en cuenta todos los desafíos y riesgos que se presentan. Al concluir el diagnóstico se tendrá claro si el proyecto es técnicamente posible, si se adapta a las necesidades del mercado y si producirá beneficios futuros. En definitiva, se determina

si es un proyecto factible y con perspectivas de éxito o no. (Cámara de Comercio de Oviedo, 2020)

**b) Análisis Costo Beneficio (ACB)**

El análisis de costo-beneficio es un proceso que se realiza para medir la relación que existe entre los costes de un proyecto y los beneficios que otorga. Su objetivo es determinar si una próxima inversión es rentable o no para una empresa.

El costo-beneficio (B/C) también es conocido como índice neto de rentabilidad. Esta herramienta es muy utilizada por las empresas, ya que les permite llevar la administración financiera en hojas de cálculo, sustentada en bases de datos. Esto ayuda a los dirigentes a tomar decisiones más acertadas acerca de la inversión y manejo de recursos. (Rodrigues, 2023)

La técnica de análisis coste/beneficio tiene como objetivo fundamental proporcionar una medida de los costes en que se incurre en la realización de un proyecto y comparar dichos costes previstos con los beneficios esperados de la realización de dicho proyecto. Esta medida o estimación servirá para:

- Valorar la necesidad y oportunidad de acometer la realización del proyecto.
- Seleccionar la alternativa más beneficiosa para la realización del proyecto.
- Estimar adecuadamente los recursos económicos necesarios en el plazo de realización del proyecto.

Es de destacar la necesidad cada vez mayor de guiarse por criterios económicos y no solo técnicos para la planificación de trabajos y proyectos. Por ello se hace una primera introducción sobre las técnicas y métodos de evaluación de conceptos económicos, con el fin de proporcionar a los profesionales criterios que les ayuden en la planificación de proyectos y evaluación de alternativas.

### c) **Modelo COCOMO**

El Modelo Constructivo de Costes (o COCOMO, por su acrónimo del inglés COnstructive COst MOdel) es un modelo utilizado para la estimación de costes de software. Este modelo fue desarrollado por Barry W. Boehm a finales de los años 70 y comienzos de los 80. Incluye tres submodelos, cada uno ofrece un nivel de detalle y aproximación cada vez mayor, a medida que avanza el proceso de desarrollo del software: básico, intermedio y detallado.

Algunas de las características e inconvenientes del modelo COCOMO son:

- Perteneciente a la categoría de modelos de subestimaciones basados en estimaciones matemáticas. Está orientado a la magnitud del producto final, midiendo el “tamaño” del proyecto, en líneas de código principalmente.
- Los resultados no son proporcionales a las tareas de gestión ya que no tiene en cuenta los recursos necesarios para realizarlas.
- Se puede desviar de la realidad si se indica mal el porcentaje de líneas de comentarios en el código fuente.
- Es un tanto subjetivo, puesto que está basado en estimaciones y parámetros que pueden ser “vistos” de distinta manera por distintos analistas que usen el método.
- Se miden los costes del producto de acuerdo con su tamaño y otras características, pero no la productividad.
- La medición por líneas de código no es válida para orientación a objetos.
- Utilizar este modelo puede resultar un poco complicado, en comparación con otros métodos (Fuentes Arce et al., s/f)

### d) **Story Points**

Se podría afirmar que: “Los Story points son una medida de complejidad y esfuerzo, algunos equipos optan por tomarlos como medida complejidad; pero estos también se pueden orientar para estimar el esfuerzo requerido para desarrollar un producto,

incluyendo los riesgos y la incertidumbre. Definamos entonces un Story Point a partir de la siguiente ecuación (María et al., 2011):

**Ecuación 1: Ecuación del coste de un Story Point**

$$\text{Story Point} = \text{Esfuerzo} + \text{Complejidad} + \text{Riesgo} \quad (1)$$

*Fuente: (María et al., 2011)*

Los Story deben ser:

- Comprensibles para los clientes y desarrolladores.
- Comprobables
- Del tamaño justo.
- Se recomienda que sean pequeños como para que el programador pueda crear varios en una iteración

Los Story Points indican el tamaño y la complejidad dado un User Story con relación a otro story que son parte del proyecto. Determinar el número de Story points en cada Story es subjetivo. Los Story points permiten estimar esfuerzo sin tratar de estimar cuánto tiempo tomará.

Algunos desarrolladores estiman los Story points basados en la experiencia y conocimientos, a partir de ahí, hacen un estimativo del desgaste que empleará la realización de esa tarea; es así como son asignados los niveles de los Story Points. Esta es la forma más sencilla para estimar Story referenciándose a un Story similar producto de la experiencia, en este caso, el Story probablemente tendrá el mismo valor que un Story similar.

Otros lo estiman, leyendo la lista de tareas a realizar, toman lo que parece ser lo más sencillo, basados en la experiencia, y le asignan el valor de 2. Y a partir de allí, todo lo demás se estima en relación con esa tarea.

## CAPÍTULO III

## MARCO PRÁCTICO



## **CAPÍTULO 3.**

### **MARCO PRÁCTICO**

#### **3.1. DISEÑO METODOLÓGICO**

##### **3.1.1. Tipo y método de investigación**

El presente Trabajo de Grado se realizó de acuerdo con la metodología de investigación aplicada orientada a la investigación de campo, que seguirá las siguientes etapas que se muestran en la siguiente tabla:

**Tabla 4: Marco metodológico**

| <b>PROCESO METODOLÓGICO</b>                            | <b>APLICACIÓN</b>   |
|--|---|
| Investigación y recolección de datos y requerimientos. | Ánalysis de los procesos de pagos y registro de pacientes en los consultorios del Centro Médico SaludAME S.R.L. |
|  | Recolección de la información mediante entrevistas.   |
| Diseño del sistema.                                    | Modelar los componentes.  |
| Implementación del sistema.                            | Programación de los componentes, utilizando tecnologías, herramientas y técnicas actuales.                      |
| Evaluación de resultados                               | Evaluar los resultados de los reportes.   |

*Fuente: Elaboración propia*

### **3.1.1.1. Técnicas e Instrumentos de Recolección de Datos**

Las técnicas e instrumentos de recolección de datos utilizados en el presente Trabajo de Grado estuvieron dadas con base al origen de la información y sus medios de recolección:

- Se utilizó información primaria para la recolección de datos, debido a que se cuenta con un contacto directo con el objetivo de análisis.
- Para los métodos de recolección, se utilizó una combinación de una entrevista complementada con el uso de la observación directa del proceso a automatizar para lograr un buen resultado.

Como resultado de la entrevista (ver Anexo D), se pudo obtener información relevante acerca del proceso de pagos de asegurados y la atención médica en los consultorios ubicados en las instituciones aseguradas.

## **3.2. INGENIERÍA DEL PROYECTO**

### **3.2.1. Diagnóstico de los procesos de pago y atención médica de pacientes en los consultorios**

Mediante la observación, y una entrevista al gerente general de SaludAME, se da a conocer el proceso para recibir atención médica en los consultorios, que se describe de la siguiente manera: Cuando el asegurado se accidenta en la institución, es llevado al consultorio habilitado dentro de la misma, donde el médico lo asiste, anotando las observaciones con respecto al paciente en un libro de forma manual, que se entrega al centro médico a final de mes. Si el paciente requiere asistencia médica especializada, se le brinda los primeros auxilios y se lo deriva a otro centro médico con la especialidad requerida.

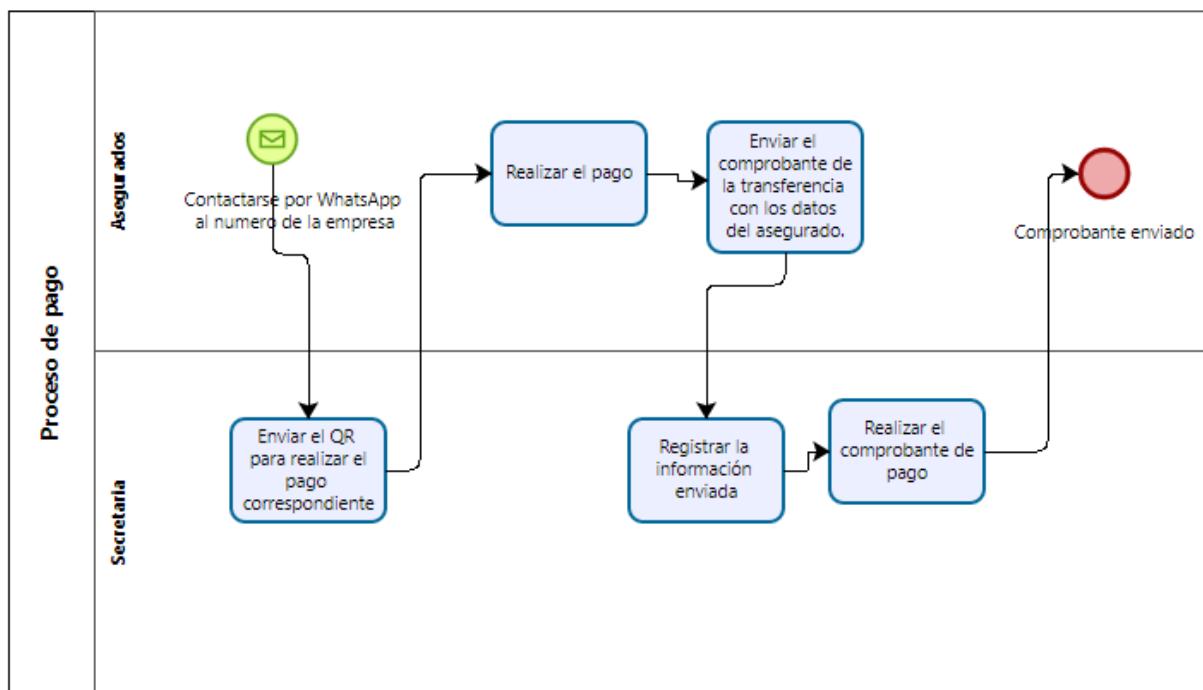
Mensualmente, en su casa matriz se realiza el registro en planillas de Excel del total de pacientes atendidos en cada consultorio médico, el total de pacientes atendidos

por diagnóstico, y el número de pacientes derivados a otros centros médicos, en base al libro de observaciones. (ver Anexo A y B)

El proceso de pago del seguro médico se realiza de la siguiente manera: en las instituciones afiliadas, los asegurados tienen que realizar el pago por concepto de seguro anual al número de cuenta de SaludAME y enviar su comprobante de pago por WhatsApp para su registro, que se realiza de manera manual.

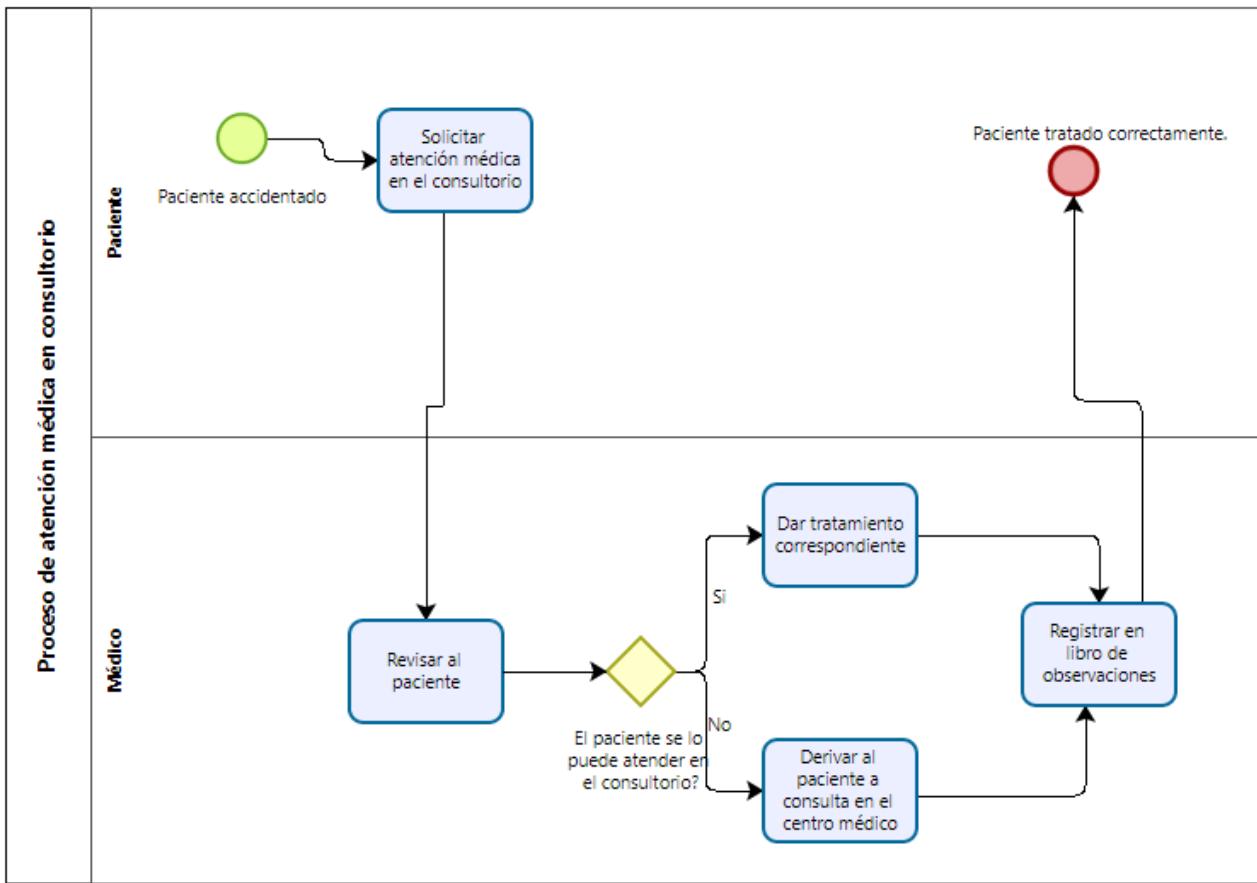
A continuación, se presentan los diagramas de modelado de proceso de negocio (BPMN, por sus siglas del inglés) tanto del proceso de pago como del proceso de atención médica de los pacientes en los consultorios de las instituciones afiliadas al seguro.

**Figura 11: BPMN del proceso de pago anual del seguro médico**



*Fuente: Elaboración propia*

Figura 12: BPMN del proceso de atención médica de pacientes en consultorio



Fuente: Elaboración propia

### 3.2.1.1. Requerimientos funcionales

En los siguientes puntos se describen los requerimientos funcionales del sistema:

#### a) Registro de atención médica en los consultorios

- Capturar la información del asegurado al momento de ingresar al consultorio.
- Registrar las observaciones del médico sobre el paciente de forma digital en una base de datos centralizada.
- Permitir que el médico anote si el paciente requiere asistencia médica especializada y registrar la derivación al centro médico.

**b) Gestión de pagos del seguro médico**

- Implementar un sistema para registrar pagos realizados por los asegurados, incluyendo la capacidad de subir comprobantes de pago.

**c) Reportes y análisis**

- Generar reportes sobre el total de pacientes atendidos por colegio, el total por diagnóstico y el número de pacientes derivados.
- Proveer visualización de datos acerca de la cantidad de estudiantes cancelados por colegio, por fecha, por gestión, así como ver el monto recaudado por los parámetros anteriormente mencionados.

**3.2.1.2. Requerimientos no funcionales**

**a) Seguridad**

- Garantizar que todos los datos médicos y personales sean almacenados de manera segura y en cumplimiento con las regulaciones de privacidad y protección de datos.
- Implementar autenticación y autorización para acceder a los diferentes módulos del sistema, asegurando que solo personal autorizado pueda acceder a datos sensibles.

**b) Usabilidad**

- Diseñar una interfaz de usuario intuitiva y fácil de usar para el personal médico y administrativo.

**c) Compatibilidad**

- Garantizar que el sistema sea compatible con diferentes dispositivos y navegadores web.

**d) Rendimiento**

- Optimizar el sistema para manejar grandes volúmenes de datos sin degradación del rendimiento
- Garantizar tiempos de respuesta rápidos para la entrada y consulta de datos médicos

**e) Escalabilidad**

- Diseñar el sistema para escalar fácilmente con el crecimiento del volumen de datos
- Permitir la adición de nuevas funcionalidades sin afectar el rendimiento del sistema existente.

**3.2.2. Selección del modelo de proceso y las herramientas de software para el desarrollo de la aplicación**

De acuerdo con la información obtenida, se seleccionarán el software y las herramientas de software a utilizar.

**3.2.2.1. Selección del Entorno de Desarrollo Integrado (IDE)**

A continuación, se muestra una tabla comparativa entre los entornos de desarrollo integrados considerados para el proyecto.

Tabla 5: Comparación de Entornos de Desarrollo Integrado (IDE)

| Criterios de selección              | Visual Studio Code  | Visual Studio  |
|-------------------------------------|---|--|
| <b>Peso y rendimiento</b>           | Ligero y rápido, ideal para equipos con recursos limitados.   | Más pesado, requiere más recursos del sistema.                   |
| <b>Lenguajes soportados</b>         | HTML, CSS, JavaScript, TypeScript con soporte de extensiones. | HTML, CSS, JavaScript, TypeScript, y C#                          |
| <b>Soporte de proyectos grandes</b> | Adequado para proyectos pequeños y medianos.                  | Ideal para grandes soluciones y proyectos empresariales.         |
| <b>Extensibilidad</b>               | Altamente extensible con un vasto Marketplace de extensiones  | Extensible, pero con un enfoque en herramientas empresariales.   |
| <b>Soporte y comunidad</b>          | Amplia comunidad y soporte continuo de Microsoft.             | Soporte empresarial de Microsoft y amplia comunidad profesional. |
| <b>Costo</b>                        | Gratis.   | Versión gratuita y versiones de pago.                            |

*Fuente: Elaboración propia*

Dada la comparación previa, para el entorno de desarrollo integrado, se optó por utilizar Visual Studio Code, dado al rendimiento en equipos con recursos limitados, una gran extensibilidad, un buen soporte de la comunidad y el costo.

### 3.2.2.2. Selección de la Metodología de Desarrollo

A continuación, se presenta una tabla comparativa entre las metodologías de desarrollo a escoger para el presente proyecto.

Tabla 6: Comparación de metodologías de desarrollo

| Criterios de selección      | Kanban   | Scrum  | XP   |
|-----------------------------|--|--|--|
| <b>Propósito</b>            | Proceso productivo, organizado y eficiente.  | Colaboración y trabajo de alto impacto.                  | Velocidad y simplicidad en ciclos de desarrollo. |
| <b>Flujo de trabajo</b>     | Continuo, tareas movidas entre estados (Por Hacer, En Proceso, Pendiente de revisión, Hecho) | Iterativo e incremental, basado en sprints de 2 semanas. | Iteraciones semanales de 3 a 5 semanas.          |
| <b>Tamaño del equipo</b>    | Flexible.  | Equipos pequeños y multidisciplinares.                   | Equipos colaborativos en todas las etapas.       |
| <b>Iteración</b>            | No hay iteraciones fijas, flujo continuo entre tareas.                                       | Sprints de 2 semanas.                                    | Iteraciones cortas y repetitivas.                |
| <b>Roles definidos</b>      | No estrictamente definidos.  | Scrum Master, Product Owner, Equipo de Desarrollo.       | Roles de equipo definidos.                       |
| <b>Entrega del producto</b> | Continuo, a medida que se completan las tareas.  | Incremento al final de cada sprint.                      | Software funcional entregado regularmente.       |

Fuente: Elaboración propia

Dadas las características del proyecto, se optó por utilizar la metodología de desarrollo Kanban, dado al tamaño del equipo necesario para la metodología, el flujo de trabajo, la entrega continua del producto y la iteración.

### 3.2.2.3. Selección del framework del frontend

A continuación, se presenta una tabla comparativa con los frameworks del frontend para desarrollar el presente proyecto.

Tabla 7: Comparación de frameworks frontend

| Criterios de selección         | React  | Angular  | Vue.js   |
|--------------------------------|--|--|--|
| <b>Tamaño y rendimiento</b>    | Ligero y rápido, solo incluye lo esencial.             | Más pesado, incluye muchas características integradas.                           | Ligero y rápido, con un enfoque modular                |
| <b>Flexibilidad y libertad</b> | Alta, permite estructurar proyectos de varias maneras. | Estructurado, sigue una arquitectura rígida.                                     | Alta, enfoque flexible y progresivo.                   |
| <b>Curva de aprendizaje</b>    | Moderada a alta, requiere aprendizaje de JSX.          | Alta, aprendizaje de TypeScript.   | Baja a moderada, diseño intuitivo y enfoque progresivo |
| <b>Desempeño</b>               | Rápido, especialmente con componentes pequeños.        | Rápido, pero puede ser más lento en aplicaciones muy grandes debido a su tamaño. | Rápido, eficiente con el DOM virtual.                  |

|  |  |   |  |
|--|--|---|--|
| <b>Documentación y comunidad</b>       | Excelente documentación y una comunidad activa.              | Excelente documentación y soporte de Google.      | Excelente documentación y comunidad creciente. |
| <b>Mantenimiento y actualizaciones</b> | Alta frecuencia de actualizaciones, respaldado por Facebook. | Actualizaciones regulares, respaldado por Google. | Actualizaciones regulares, comunidad activa.   |

*Fuente: Elaboración propia*

Conforme a las características presentadas entre las tres tecnologías, se optó por el uso de React, dado al desempeño, tamaño y rendimiento; la buena curva de aprendizaje y una excelente documentación y comunidad.

#### 3.2.2.4. Selección del framework del backend

A continuación, se presenta una tabla comparativa con los frameworks del backend para el desarrollo del presente proyecto.

**Tabla 8: Comparación de frameworks backend**

| Criterio de selección           | Express.js                                   | Django                           | ASP.NET                                   |
|---------------------------------|--|----------------------------------|---|
| <b>Lenguaje de programación</b> | JavaScript, TypeScript.                      | Python.                          | C#.                                       |
| <b>Curva de aprendizaje</b>     | Baja a moderada.                             | Moderada a alta.                 | Moderada a alta.                          |
| <b>Escalabilidad</b>            | Alta, pero requiere configuración adicional. | Buena para aplicaciones grandes. | Alta, especialmente adecuado para grandes |

|  |  |   |   |
|--|--|---|---|
|  |  |   | aplicaciones empresariales.   |
| <b>Rendimiento</b>                     | Alto, muy dependiente del motor de JavaScript. | Alto, buen rendimiento con Python.                                    | Alto, rendimiento optimizado para .NET Core.                          |
| <b>Flexibilidad</b>                    | Alta, minimalista y altamente personalizable.  | Moderada, sigue una estructura rígida.                                | Moderada, sigue una estructura, pero con opciones de personalización. |
| <b>Velocidad de desarrollo</b>         | Rápida, configuración minimalista.             | Moderada, configuración más estructurada.                             | Moderada, pero muy productiva con herramientas integradas.            |
| <b>Mantenimiento y actualizaciones</b> | Frecuentes, respaldado por la comunidad.       | Frecuentes, respaldado por la comunidad y Django Software Foundation. | Frecuentes, respaldado por Microsoft.                                 |

*Fuente: Elaboración propia*

Al presentarse las características de las tres tecnologías, se optó por Express.js, debido a su velocidad de desarrollo, escalabilidad, buen rendimiento y una baja curva de aprendizaje.

### 3.2.2.5. Selección del lenguaje de programación

Al seleccionar la tecnología para el frontend y el backend de la aplicación, los lenguajes de programación a utilizar son:

- Frontend: React utiliza JavaScript como lenguaje de programación

- Backend: Express.js maneja JavaScript como principal lenguaje de programación.

### 3.2.2.6. Selección del gestor de base de datos

A continuación, se presenta una tabla comparativa entre los gestores de base de datos investigados para el desarrollo del presente proyecto.

**Tabla 9: Comparación de gestores de bases de datos**

| Criterio de selección            | MySQL  | PostgreSQL  | SQLite  |
|----------------------------------|--|---|---|
| <b>Licencia</b>                  | GPL (versión comunitaria) y comercial (versiones empresariales). | PostgreSQL License (similar a MIT).                       | Dominio público.  |
| <b>Curva de aprendizaje</b>      | Baja a moderada.   | Moderada a alta.  | Baja.   |
| <b>Escalabilidad</b>             | Alta, adecuado para grandes aplicaciones web.                    | Muy alta, adecuado para aplicaciones empresariales.       | Limitada, adecuado para aplicaciones pequeñas y medianas. |
| <b>Rendimiento</b>               | Alto rendimiento para lecturas y escrituras.                     | Alto rendimiento, especialmente en operaciones complejas. | Alto rendimiento en aplicaciones pequeñas.                |
| <b>Características avanzadas</b> | Buen soporte para transacciones, clustering y replicación.       | Soporte avanzado para transacciones,                      | Limitado, pero suficiente para aplicaciones pequeñas.     |

|                                |  |  |                  |
|--------------------------------|--|--|------------------|
|                                |  | extensiones, y replicación.                            |                  |
| <b>Seguridad</b>               | Buen soporte, pero con muchas configuraciones adicionales. | Soporte avanzado, muchas características de seguridad. | Básica.          |
| <b>Soporte de la comunidad</b> | Grande y activo.   | Grande y activo.                                       | Grande y activo. |

*Fuente: Elaboración propia*

Dadas las características presentadas, se optó por el uso de MySQL, dado a la baja curva de aprendizaje, el buen desempeño enfocado en las lecturas y escrituras y la escalabilidad.

### 3.2.2.7. Selección del visualizador de datos

A continuación, se muestra una tabla comparativa entre los visualizadores de datos para su uso en el presente proyecto.

**Tabla 10: Comparación de visualizadores de datos**

| Criterio de selección       | Tableau  | Power BI   |
|-----------------------------|--|--|
| <b>Facilidad de uso</b>     | Intuitiva, pero puede ser compleja para principiantes. | Intuitiva, más amigable para usuarios de Excel.                            |
| <b>Curva de aprendizaje</b> | Moderada a alta, depende de la complejidad de uso.     | Baja a moderada, especialmente fácil para usuarios de productos Microsoft. |

|   |  |  |
|---|--|--|
| <b>Interfaz de usuario</b>                | Interfaz rica en funcionalidades, pero puede ser abrumadora.   | Interfaz moderna y simplificada, integración con otros productos de Microsoft.                     |
| <b>Precios</b>                            | Costoso, especialmente para licencias empresariales.           | Más asequible, con una versión gratuita disponible.  |
| <b>Integración de datos</b>               | Soporte para una amplia variedad de fuentes de datos.          | Soporte para una amplia variedad de fuentes de datos, especialmente bueno con productos Microsoft. |
| <b>Integración con otras herramientas</b> | Amplia, con conectores para muchas herramientas empresariales. | Excelente, especialmente con el ecosistema de Microsoft (Excel, Azure, Dynamics 365, etc.).        |

*Fuente: Elaboración propia*

Dadas las características mencionadas, se optó por usar Power BI como visualizador de datos, dado a que sus licencias son más asequibles contando con una versión gratuita, la baja curva de aprendizaje y la excelente integración con otras herramientas de Microsoft.

### 3.2.2.8. Selección de la arquitectura de software

A continuación, se presenta una tabla comparativa entre las distintas arquitecturas de software para el desarrollo del presente proyecto.

Tabla 11: Comparación de arquitecturas de software

| Criterio de selección               | Arquitectura Cliente-Servidor   | Arquitectura en Capas  | Modelo Vista-Controlador (MVC)   |
|-------------------------------------|---|--|--|
| <b>Componentes principales</b>      | Cliente, Servidor   | Capa de Presentación, Capa de Negocio, Capa de Datos                                 | Vista, Controlador, Modelo   |
| <b>Comunicación</b>                 | Directa entre cliente y servidor.                                       | Comunicación entre capas adyacentes.   | Comunicación entre Vista y Controlador, y entre Controlador y Modelo.                                  |
| <b>Complejidad</b>                  | Baja. Sencilla de implementar en aplicaciones pequeñas.                 | Moderada. Requiere una organización clara de las responsabilidades.                  | Moderada a alta. La separación en tres componentes aumenta la complejidad inicial.                     |
| <b>Flexibilidad y Mantenimiento</b> | Moderado. Cambios en el servidor pueden afectar al cliente y viceversa. | Alta. Cada capa se puede modificar independientemente si se respetan las interfaces. | Alta. Separación clara de responsabilidades facilita el mantenimiento y la evolución de la aplicación. |
| <b>Escalabilidad</b>                | Escalabilidad vertical (aumentar potencia del servidor).                | Escalabilidad tanto vertical como horizontal.  | Escalabilidad alta al desacoplar componentes.  |

Fuente: Elaboración propia

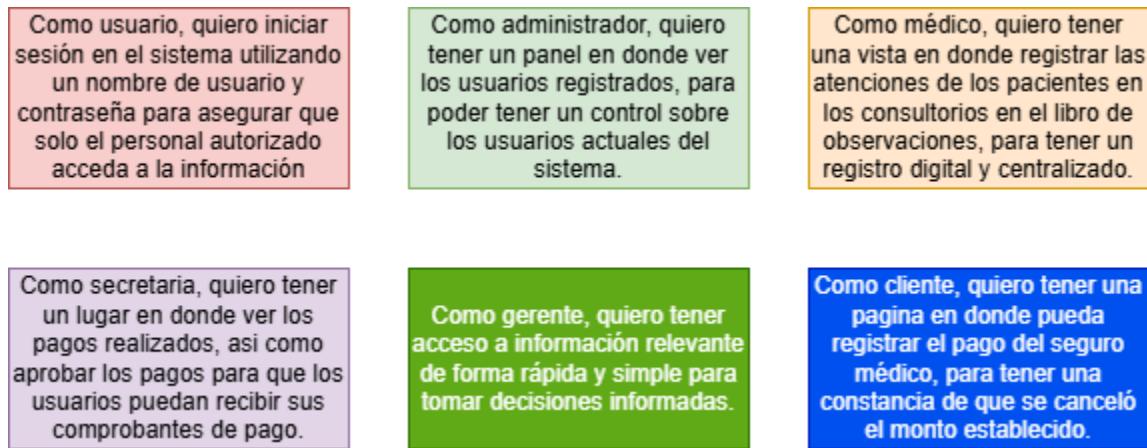
Dadas las características mencionadas, se optó por utilizar la arquitectura cliente-servidor, debido a la baja complejidad requerida para su implementación y la escalabilidad vertical.

### 3.2.3. Diseño de los requerimientos de la aplicación

#### 3.2.3.1. Historias de usuario

A continuación, se presentan las historias de usuario de manera general, en base a los requerimientos funcionales y no funcionales del sistema.

**Figura 13: Historias de usuario del sistema**



*Fuente: Elaboración propia*

A continuación, se detallan las historias de usuario mostradas en la figura 13.

**Tabla 12: Historia de usuario 1**

| <b>Historia de usuario</b>      |   |                              |                                |
|---------------------------------|---|------------------------------|--------------------------------|
| <b>Numero:</b>                  | 1   | <b>Usuario:</b>              | Administrador, médico, gerente |
| <b>Nombre de historia:</b>      | Inicio de sesión  |                              |                                |
| <b>Prioridad en negocio:</b>    | Alta  | <b>Riesgo de desarrollo:</b> | Baja                           |
| <b>Fecha de creación:</b>       | 24/05/2024  |                              |                                |
| <b>Programador responsable:</b> | Ariel Tellez Torrico  |                              |                                |
| <b>Descripción:</b>             | <p>Como usuario, quiero iniciar sesión en el sistema utilizando un nombre de usuario y contraseña para asegurar que solo el personal autorizado acceda a la información</p>   |                              |                                |
| <b>Validación:</b>              | <p>Puedo ingresar mi nombre de usuario y contraseña en la pantalla de inicio de sesión.</p> <p>Recibo un mensaje de error si el nombre de usuario o la contraseña son incorrectos. Al cuarto intento recibo un error de muchos intentos fallidos, bloqueando el acceso durante 10 minutos.</p> <p>Accedo al sistema si las credenciales son correctas.</p> <p>Las contraseñas deben contener al menos 8 caracteres, conteniendo letras, números y símbolos.</p> |                              |                                |

*Fuente: Elaboración propia*

**Tabla 13: Historia de usuario 2**

| <b>Historia de usuario</b>      |   |                              |               |
|---------------------------------|---|------------------------------|---------------|
| <b>Numero:</b>                  | 2   | <b>Usuario:</b>              | Administrador |
| <b>Nombre de historia:</b>      | Gestión de Usuarios   |                              |               |
| <b>Prioridad en negocio:</b>    | Alta  | <b>Riesgo de desarrollo:</b> | Media         |
| <b>Fecha de creación:</b>       | 24/05/2024  |                              |               |
| <b>Programador responsable:</b> | Ariel Tellez Torrico  |                              |               |
| <b>Descripción:</b>             | <p>Como administrador quiero tener un panel, en donde ver los usuarios registrados para tener un control sobre los usuarios actuales del sistema.</p> |                              |               |

**Validación:**

Puedo agregar usuarios registrando su nombre completo, nombre de usuario y contraseña. La contraseña debe estar cifrada en la base de datos.

Puedo editar el nombre, apellidos, nombre de usuario y contraseña, solo si el usuario lo solicita.

*Fuente: Elaboración propia*

**Tabla 14: Historia de usuario 3**

| <b>Historia de usuario</b>  |   |                              |        |  |  |  |
|---|---|------------------------------|--------|--|--|--|
| <b>Numero:</b>  | 3   | <b>Usuario:</b>              | Médico |  |  |  |
| <b>Nombre de historia:</b>  | Registrar atención médica en consultorio. |                              |        |  |  |  |
| <b>Prioridad en negocio:</b>  | Alta                                      | <b>Riesgo de desarrollo:</b> | Media  |  |  |  |
| <b>Fecha de creación:</b>   | 24/05/2024                                |                              |        |  |  |  |
| <b>Programador responsable:</b>   | Ariel Tellez Torrico                      |                              |        |  |  |  |
| <b>Descripción:</b>   |   |                              |        |  |  |  |
| Como médico, quiero poder registrar la atención médica de paciente en el consultorio, para tener un registro digital y centralizado; y poder ver un historial de las atenciones médicas realizadas al paciente. |   |                              |        |  |  |  |
| <b>Validación:</b>  |   |                              |        |  |  |  |
| Puedo registrar consultas médicas contenido el nombre, curso, fecha, diagnóstico, tratamiento y observaciones de cada paciente atendido.  |   |                              |        |  |  |  |
| Puedo ver un historial de todas las atenciones realizadas, filtrando por nombre completo y cédula de identidad del paciente.  |   |                              |        |  |  |  |

*Fuente: Elaboración propia*

**Tabla 15: Historia de usuario 4**

| <b>Historia de usuario</b>   |  |                              |                     |
|------------------------------|--|------------------------------|---------------------|
| <b>Numero:</b>               | 4  | <b>Usuario:</b>              | Secretaria, gerente |
| <b>Nombre de historia:</b>   | Registro y Seguimiento de Pagos Realizados |                              |                     |
| <b>Prioridad en negocio:</b> | Alta                                       | <b>Riesgo de desarrollo:</b> | Media               |
| <b>Fecha de creación:</b>    | 24/05/2024                                 |                              |                     |

|   |
|---|
| <b>Programador responsable:</b> Ariel Tellez Torrico  |
| <b>Descripción:</b>   |
| Como secretaria, quiero tener un lugar en donde ver los pagos realizados, así como aprobar los pagos. |
| <b>Validación:</b>  |

Puedo ver un historial de todos los pagos realizados y filtrar los pagos por nombre completo o cedula de identidad del estudiante, así como filtrar pagos por colegio y estado del pago (aprobado o pendiente de aprobación).

Puedo aprobar los pagos, para que el sistema genere un recibo de pago.

*Fuente: Elaboración propia*

**Tabla 16: Historia de usuario 5**

| <b>Historia de usuario</b>      |   |                                    |
|---------------------------------|---|------------------------------------|
| <b>Número:</b>                  | 5   | <b>Usuario:</b> Gerente            |
| <b>Nombre de historia:</b>      | Dashboards de control administrativo  |                                    |
| <b>Prioridad en negocio:</b>    | Alta  | <b>Riesgo de desarrollo:</b> Media |
| <b>Fecha de creación:</b>       | 24/05/2024  |                                    |
| <b>Programador responsable:</b> | Ariel Tellez Torrico  |                                    |
| <b>Descripción:</b>             | Como gerente, quiero tener acceso a información relevante de pagos y consultas, de forma rápida y simple para tomar decisiones.   |                                    |
| <b>Validación:</b>              | <p>Puedo ver métricas clave del negocio (total de pacientes atendidos, total de pagos realizados, total de pacientes derivados, total de monto recaudado).</p> <p>Puedo personalizar las métricas que se muestran en el dashboard, aplicando filtros sobre las mismas (por colegio, por gestión, por rango de fechas, por diagnóstico).</p> |                                    |

*Fuente: Elaboración propia*

Tabla 17: Historia de usuario 6

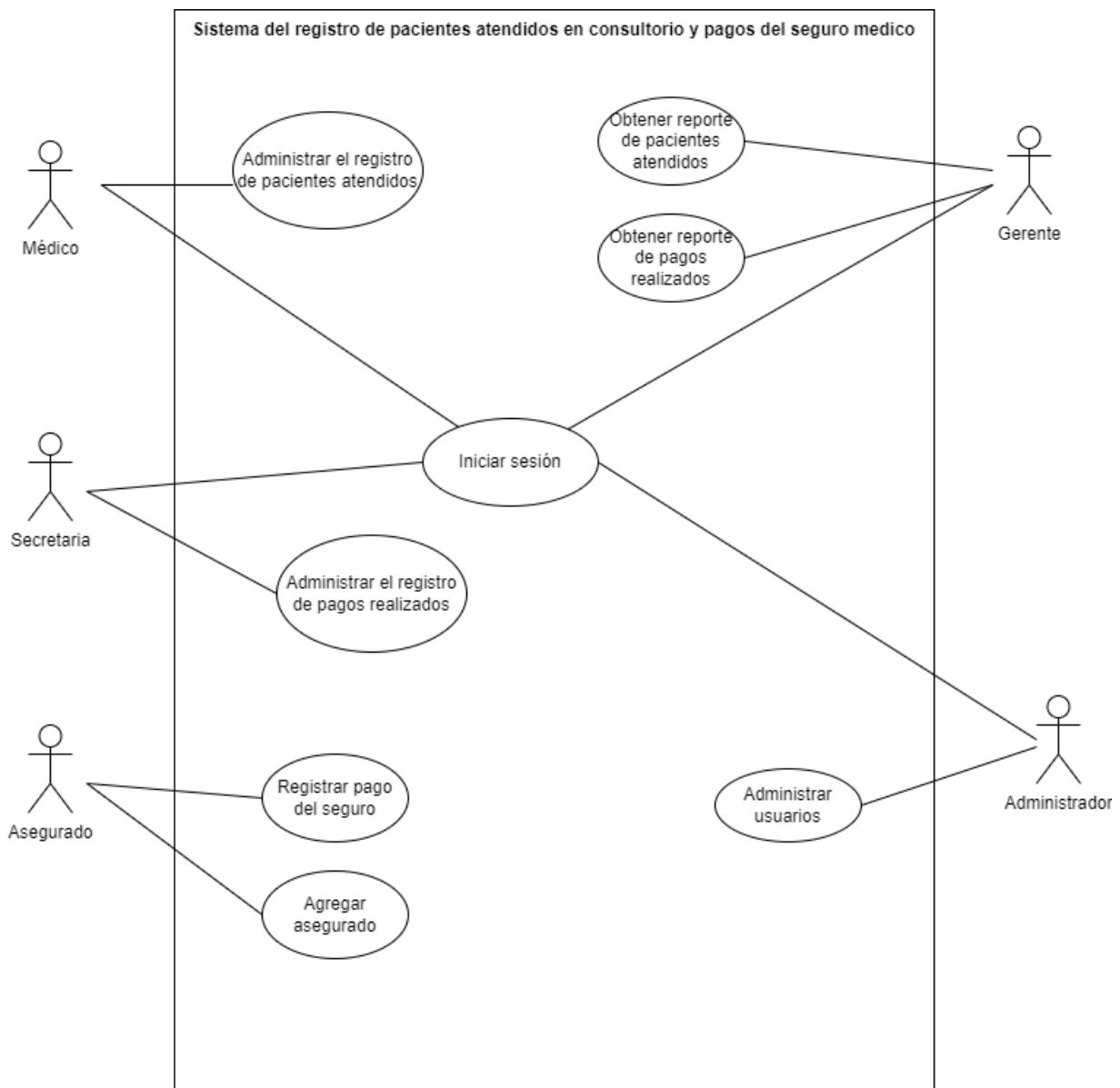
| Historia de usuario             |  |                              |         |
|---------------------------------|--|------------------------------|---------|
| <b>Numero:</b>                  | 6  | <b>Usuario:</b>              | Cliente |
| <b>Nombre de historia:</b>      | Registro de pagos del seguro médico  |                              |         |
| <b>Prioridad en negocio:</b>    | Alta   | <b>Riesgo de desarrollo:</b> | Media   |
| <b>Fecha de creación:</b>       | 24/05/2024   |                              |         |
| <b>Programador responsable:</b> | Ariel Tellez Torrico   |                              |         |
| <b>Descripción:</b>             | <p>Como cliente, quiero tener una página en donde pueda registrar el pago del seguro médico, para tener una constancia de que se canceló el monto establecido.</p>   |                              |         |
| <b>Validación:</b>              | <p>Puedo seleccionar el nombre del asegurado mediante una búsqueda por nombre completo. Si no existe el asegurado en la base de datos, puedo agregarlo.</p> <p>Puedo subir una foto del comprobante de pago, así como registrar los datos de contacto del asegurado.</p> <p>Puedo descargar el recibo de pago filtrando por nombre completo del asegurado o por su cédula de identidad</p> |                              |         |

*Fuente: Elaboración propia*

### 3.2.3.2. Diagrama de casos de uso del sistema

A continuación, se presenta el diagrama de casos de uso de todo el sistema, en la que se representan a los usuarios y todas las interacciones que tendrán con el sistema.

Figura 14: Diagrama de casos de uso del sistema



Fuente: Elaboración propia

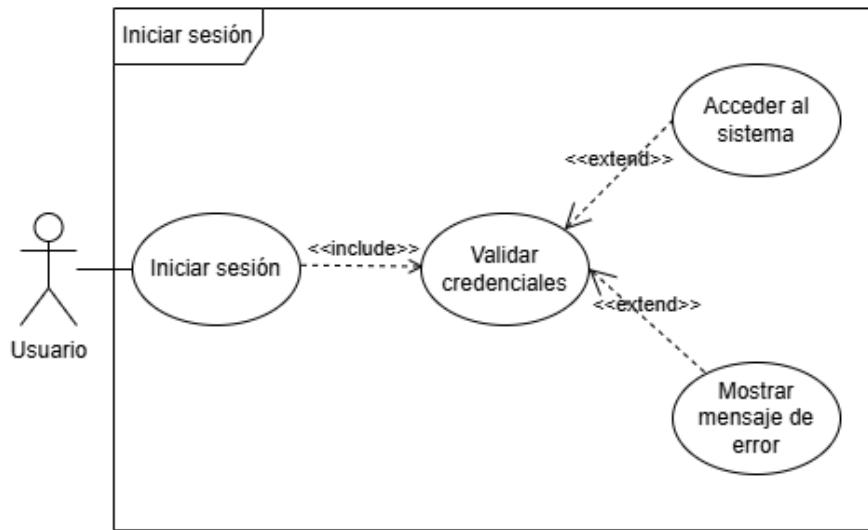
El sistema contará con 5 módulos, los cuales se describen a continuación.

### 3.2.3.3. Módulo de inicio de sesión

El módulo de inicio de sesión permitirá a los usuarios del sistema (administrador, gerente, secretaria, médico) iniciar sesión en su correspondiente página de inicio, habilitando los módulos correspondientes según el tipo de usuario. A continuación, se presentan los diagramas correspondientes.

#### a) Diagrama de caso de uso

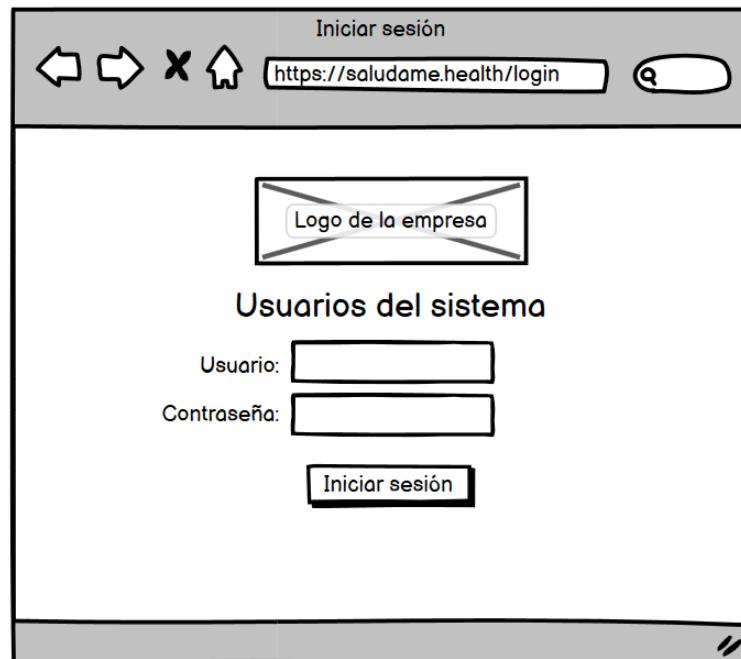
Figura 15: Diagrama de caso de uso del módulo de inicio de sesión



Fuente: Elaboración propia

## b) Wireframes del módulo

Figura 16: Wireframe de la pantalla de inicio de sesión



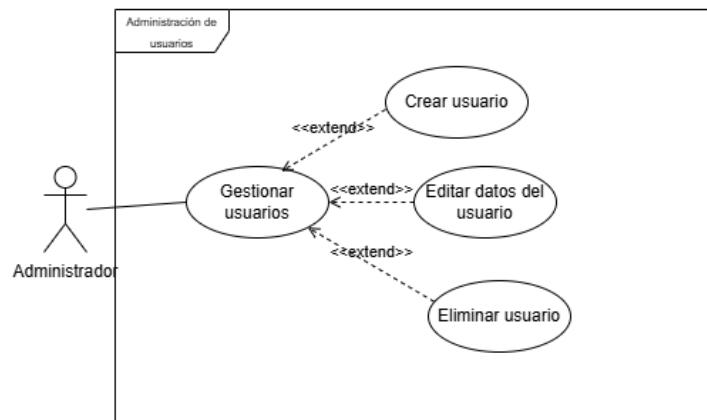
*Fuente: Elaboración propia*

### 3.2.3.4. Módulo de administración de usuarios

Este módulo permitirá al administrador gestionar los usuarios existentes en el sistema, agregando usuarios y asignándoles un rol en el sistema, el cual determina los módulos a habilitar. A continuación, se presentan los diagramas correspondientes.

## a) Diagrama de caso de uso

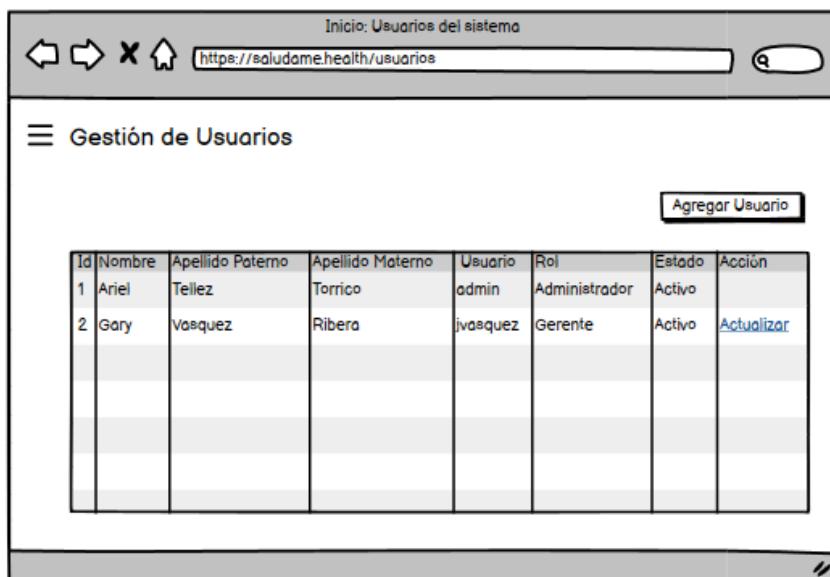
Figura 17: Diagrama de casos de uso del módulo de administración de usuarios



Fuente: Elaboración propia

## b) Wireframes del módulo

Figura 18: Wireframe de la pantalla principal de la gestión de usuarios



Fuente: Elaboración propia

Figura 19: Wireframe de la pantalla para agregar usuarios

Inicio: Usuarios del sistema  
https://saludame.health/crearUsuario

### Agregar usuario

Nombre:

Apellido Paterno:

Apellido Materno:

Usuario:

Contraseña:

Rol:

Estado:

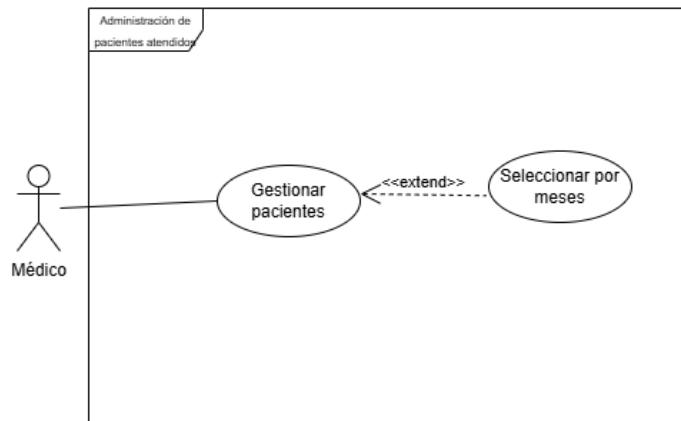
Fuente: Elaboración propia

#### 3.2.3.5. Módulo de administración de pacientes en consultorios

Este módulo permitirá al médico del consultorio registrar los datos de los pacientes atendidos en el consultorio. El médico podrá ver los registros realizados a la fecha, así como ver los registros realizados en los meses anteriores. A continuación, se presentan los diagramas correspondientes.

## a) Diagrama de casos de uso

Figura 20: Diagrama de caso de uso del módulo de administración de pacientes atendidos



Fuente: Elaboración propia

## b) Wireframes del módulo

Figura 21: Wireframe de la pantalla principal del módulo de administración de pacientes

Libro de observaciones

https://saludame.health/libroObservaciones

☰ Libro de observaciones

Colegio:  Gestión:  Mes:

| Nombre | Apellidos  | Fecha de atención | Diagnóstico | Tratamiento | Observaciones |
|--------|------------|-------------------|-------------|-------------|---------------|
| Juan   | Perez Cruz | 2024-01-01 00:00  | test        | test        | test          |

Fuente: Elaboración propia

Figura 22: Wireframe de la pantalla para agregar registro de paciente atendido

Agregar registro  
https://saludame.health/createRegi

### Agregar datos

Buscar asegurado por:

Nombre Completo  
Cédula de Identidad

Nombre:

Apellido Paterno:

Apellido Materno:

Buscar

### Agregar datos para Cliente

Fecha de atención:  / /

Diagnóstico:

Tratamiento:

Observaciones:

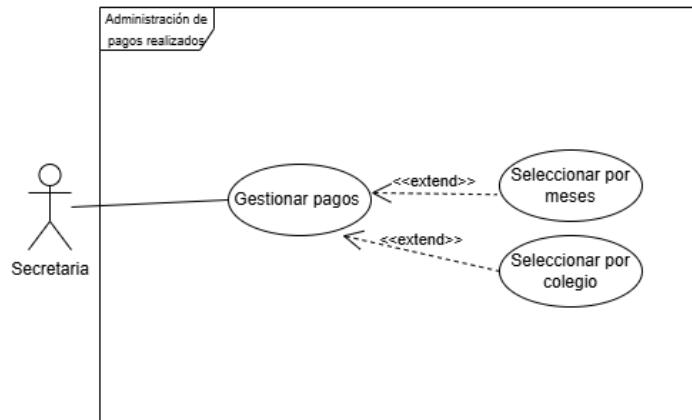
Agregar

Fuente: Elaboración propia

#### 3.2.3.6. Módulo de administración de pagos realizados

En este módulo, la secretaría podrá ver los pagos realizados a la fecha, pudiendo filtrar por nombre completo o cédula de identidad en caso de que se desee buscar a un estudiante, por estado del pago y por colegio. A continuación, se presentan los diagramas correspondientes.

## a) Diagrama de casos de uso



*Fuente: Elaboración propia*

## b) Wireframes del módulo

Figura 23: Wireframe de la pantalla principal de la administración de pagos realizados.

Wireframe de la pantalla principal de la administración de pagos realizados:

URL: <https://saludame.health/pagos>

Contenido:

≡ Pagos realizados

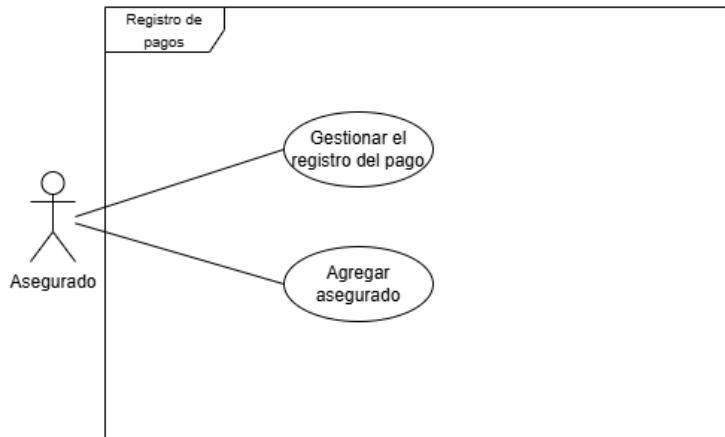
| N. | Nombre       | Colegio   | Fecha            | Gestión | Monto | Forma de pago | Estado      | Acción   |
|----|--------------|-----------|------------------|---------|-------|---------------|-------------|--|
| 1  | Juan Perez   | Don Bosco | 2024-01-01 00:00 | 2023    | 80    | Efectivo      | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar Recibo</a> |
| 2  | Sofia Suarez | Uboldi    | 2024-01-01 00:00 | 2023    | 80    | Efectivo      | Por aprobar | <a href="#">Ver comprobante</a> <a href="#">Aprobar Pago</a>     |

*Fuente: Elaboración propia*

### 3.2.3.7. Módulo del registro de pagos

En este módulo los clientes (asegurados) podrán registrar el pago realizado, en el cual podrán buscar el nombre del estudiante mediante una búsqueda por su nombre completo o cédula de identidad. En caso de que el asegurado no esté registrado en la base de datos se lo puede agregar. Luego se muestra una pantalla para realizar el pago y subir el comprobante de la transferencia a la base de datos. Una vez realizado el pago y subida la imagen, se muestra una pantalla de registro de los datos de contacto del cliente para el envío de su comprobante de pago, y termina el proceso de manera exitosa, mientras que a la secretaría le llega una notificación al sistema con los pagos por aprobar. Cuando la secretaría aprueba los pagos, el asegurado puede acceder a la aplicación para poder descargar su recibo filtrando por nombre o cédula de identidad. A continuación, se muestran los diagramas correspondientes.

#### a) Diagrama de casos de uso



*Fuente: Elaboración propia*

b) **Wireframes del módulo**

Figura 24: Wireframe de la búsqueda de asegurados

The wireframe depicts a web browser window with the following layout:

- Header:** Includes standard browser controls (Back, Forward, Stop, Home) and a URL bar containing the URL <https://saludame.health/seguro>.
- Title:** The page title is "Paga tu seguro".
- Text:** A placeholder text instructs the user to "Escribe el nombre completo del asegurado, a continuación presiona el botón Buscar".
- Form Fields:** A dropdown menu labeled "Tipo de búsqueda" with options "Nombre Completo" and "Cédula de Identidad".
- Text Fields:** Three input fields for "Nombre", "Apellido Paterno", and "Apellido Materno".
- Buttons:** A large "Buscar" button.
- Text at the bottom:** A link "No se encuentra en la lista? [Registrese Aquí](#)".

*Fuente: Elaboración propia*

Figura 25: Wireframe de la pantalla para agregar asegurados

Agregar Asegurado

Agregar asegurado

Nombre:

Apellido Paterno:

Apellido Materno:

CI:

Fecha de nacimiento:  /

Colegio:

Sexo:

Alergias:

Enfermedades base:

Fuente: Elaboración propia

Figura 26: Wireframe de la pantalla para cancelar el seguro

Seguros: Cancelar

https://saludame.health/seguro/upload

Cancelar seguro

Por favor, haz una transferencia con el banco de tu preferencia:  
Número de cuenta: XXXXX  
Nombre: SaludAME SRL  
Monto a cancelar: 80Bs

Si ya has cancelado, sube una foto de tu comprobante de pago

Fuente: Elaboración propia

Figura 27: Wireframe de la página de registro de datos de contacto

The wireframe shows a web browser window with the following details:

- Header:** "Seguro: Datos de contacto" and a URL bar with "https://saludame.health/confirm".
- Section:** "≡ Datos de contacto".
- Text:** "Por favor, registra tus datos de contacto .".
- Form Fields:** "Nombre:" followed by an input field, and "Nro de celular:" followed by another input field.
- Buttons:** A "Finalizar" button at the bottom right.

*Fuente: Elaboración propia*

Figura 28: Wireframe de la página de finalización del proceso

The wireframe shows a web browser window with the following details:

- Header:** "Seguro: Hecho" and a URL bar with "https://saludame.health/success".
- Section:** "≡ Hecho".
- Text:** "Gracias".
- Text:** "En el transcurso del día, podrás descargar tu recibo aquí."
- Buttons:** A "Volver al inicio" button at the bottom.

*Fuente: Elaboración propia*

Figura 29: Wireframe de la página para descargar recibos

The wireframe shows a web browser window with the title 'Pagos' at the top. The address bar contains the URL 'https://saludame.health/pagos'. Below the address bar, there are navigation icons (back, forward, search, etc.). The main content area is titled 'Descargar recibo'. It features a search bar with fields for 'Nombre' and 'Cédula de Identidad', and a 'Filtrar' button. Below the search bar is a table with the following data:

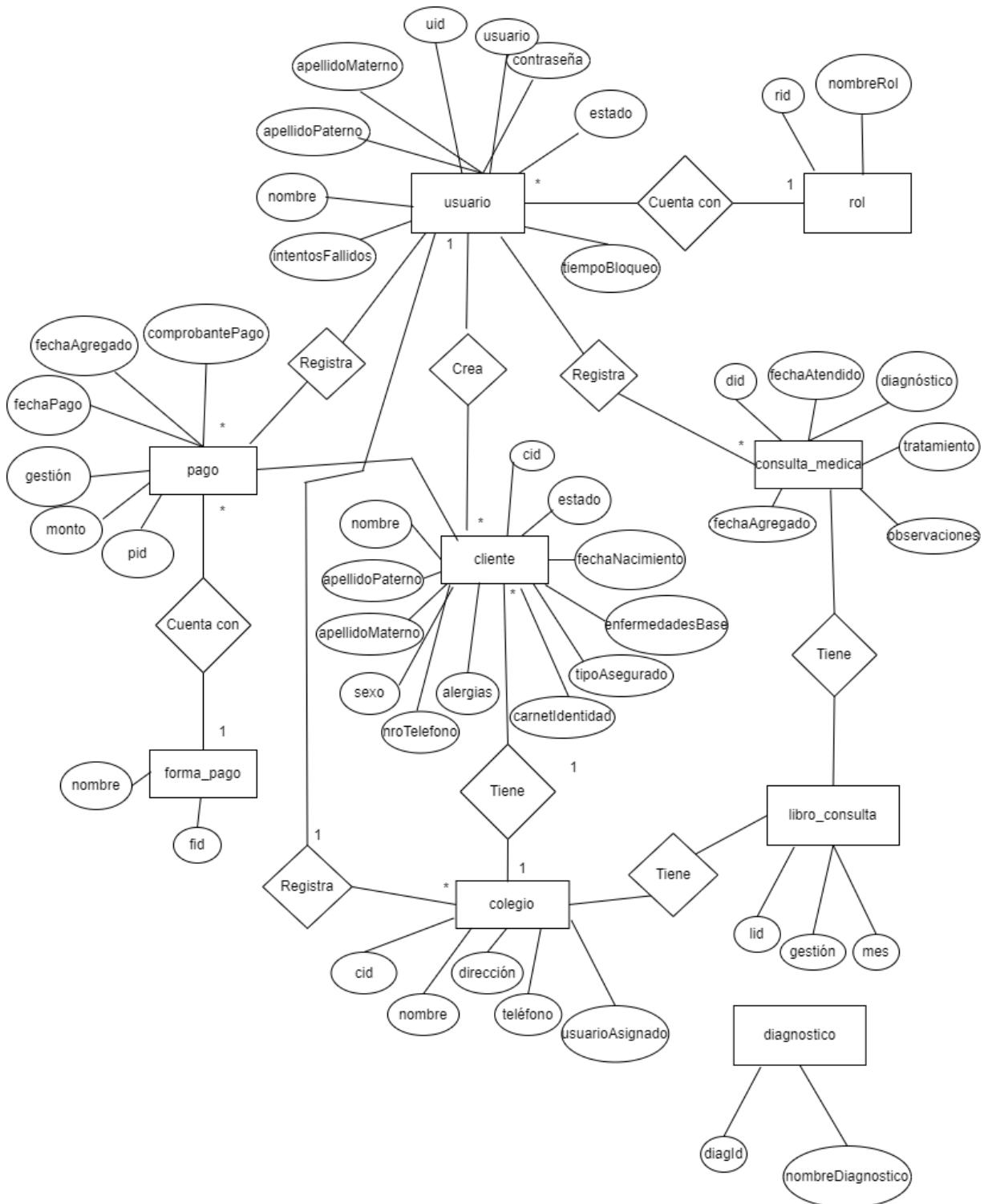
| N. | Nombre     | Cédula de Identidad | Colegio   | Gestión          | Fecha de pago | Acción                           |
|----|------------|---------------------|-----------|------------------|---------------|----------------------------------|
| 1  | Juan Perez | 9876789             | Don Bosco | 2024-01-01 00:00 | 2023          | <a href="#">Descargar Recibo</a> |

Fuente: Elaboración propia

### 3.2.3.8. Diseño conceptual de la base de datos

El sistema contará con 9 tablas, las cuales son: Usuario, Cliente, Datos de la consulta médica, Libro de consulta médica, Pago, Forma de pago, Rol y una tabla auxiliar llamada Diagnóstico. A continuación, se presenta el diseño conceptual de la base de datos, en donde se describen las tablas que contendrá el sistema con sus relaciones.

Figura 30: Diagrama conceptual de la base de datos

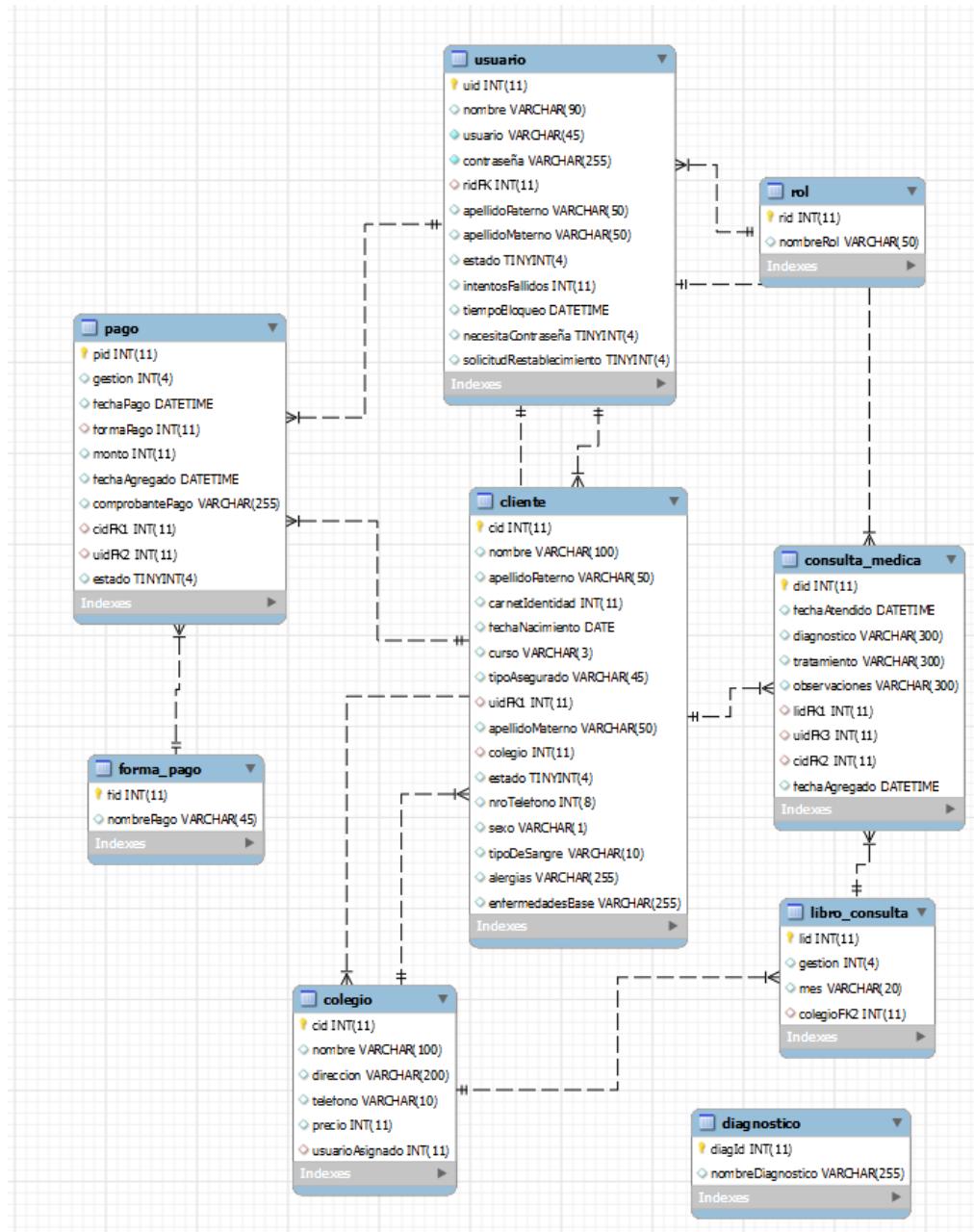


Fuente: Elaboración propia

### 3.2.3.9. Diseño lógico de la base de datos

A continuación, se presenta el diseño lógico de la base de datos, en donde se incluye los tipos de datos y llaves foráneas.

Figura 31: Diagrama lógico de la base de datos



Fuente: Elaboración propia

### 3.2.4. Implementación de los componentes de la aplicación

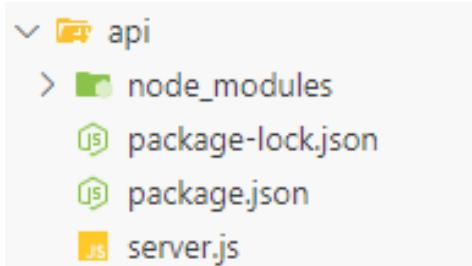
A continuación, se presenta la implementación de los módulos correspondientes.

#### 3.2.4.1. Estructura del proyecto

El proyecto está estructurado en dos carpetas: una llamada “api”, que contiene el backend de la aplicación, y otra llamada “client”, que contiene el frontend.

##### a) Backend

**Figura 32: Estructura del backend**



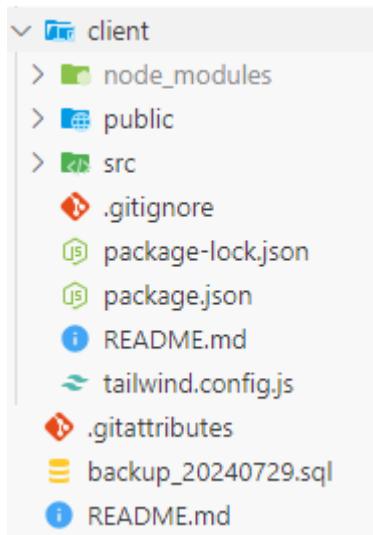
*Fuente: Elaboración propia*

En donde:

- La carpeta node\_modules incluye todos los archivos y dependencias instalados que se usan en el proyecto.
- Package.json y package-lock.json son archivos que contienen los datos y los detalles de las dependencias instalados. Express es un tipo de dependencia que se utiliza para establecer un servidor, por lo tanto, estos archivos contienen los detalles tales como la versión y la instalación.
- Server.js es el archivo principal del backend, en donde se encuentra la conexión a las base de datos y los distintos métodos para realizar operaciones con las mismas.

## b) Frontend

Figura 33:Estructura del frontend



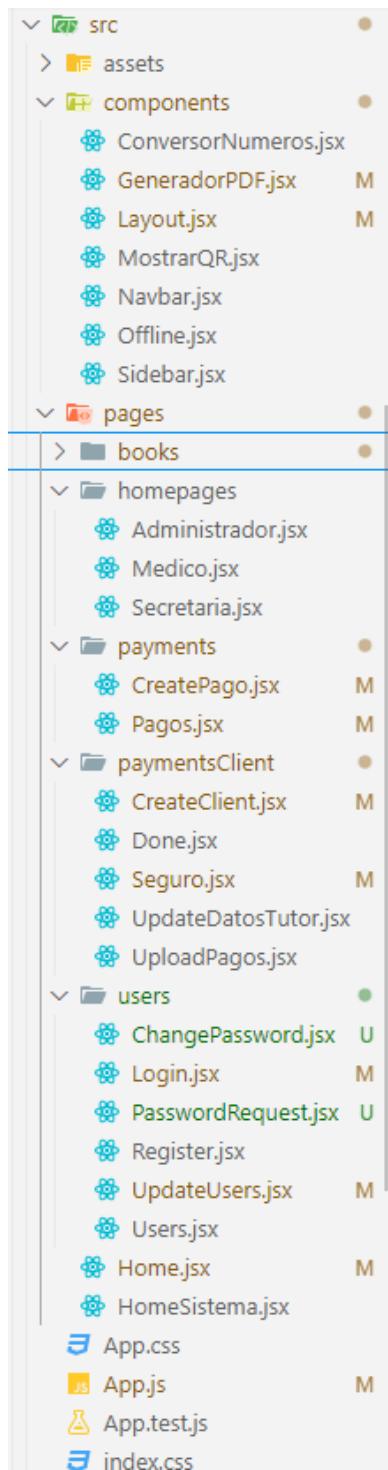
Fuente: *Elaboración propia*

En donde:

- La carpeta node\_modules incluye todos los archivos y dependencias instalados que se usan en el proyecto.
- La carpeta public contiene assets (activos, por su traducción del inglés) estáticos que no requieren procesamiento, tales como archivos HTML, el archivo manifest.json entre otros.
- Package.json y package-lock.json son archivos que contienen los datos y los detalles de las dependencias instalados.
- README.md es un archivo para colocar información del proyecto.
- .gitignore es un archivo en donde se especifica que archivos deben ser ignorados por el gestor de control de versiones.
- Tailwind.config.js es un archivo que define la configuración principal de Tailwind CSS, se pueden personalizar para adaptarse a las necesidades del proyecto.

- La carpeta src es donde se encuentra el proyecto principal, la cual se subdivide en las siguientes carpetas:

**Figura 34: Estructura de la carpeta src del frontend**



*Fuente: Elaboración propia*

- La carpeta assets contiene assets estáticos, tales como imágenes y videos. En este caso se almacena el logo de la empresa.
- La carpeta componets contiene componentes reusables y pequeños que se utilizan en todo el proyecto.
- La carpeta pages contiene componentes más grandes que representan diferentes páginas de la aplicación. En este caso se dividieron en carpetas con los nombres de los módulos a implementar.
- El archivo index.js es el archivo que se ejecuta primero en la aplicación.
- El archivo App.js es el componente raíz que es renderizado en el navegador.

#### **3.2.4.2. Elaboración del diseño físico de la base de datos**

En el archivo server.js del backend, primero se importan las librerías a utilizar en el proyecto, luego se inicializan las variables a utilizar y se declara que se utilizará Express como backend, así como también otras librerías a utilizar, como CORS para acceder a los diferentes métodos para realizar el CRUD a la base de datos desde el frontend, cookieParser, para utilizar las cookies del navegador y bcrypt para la encriptación y desencriptación de la contraseña al momento de registrar o iniciar sesión. Luego se realiza la conexión a la base de datos utilizando el método createConnection de la librería mysql, en donde se le especifica a qué servidor se conectará, el usuario, la contraseña y qué base de datos usar.

Figura 35: Archivo server.js con la conexión a la base de datos

```
1 import express from "express";
2 import mysql from "mysql";
3 import cors from "cors";
4 import jwt from "jsonwebtoken";
5 import bcrypt from "bcrypt";
6 import cookieParser from "cookie-parser";
7
8 const salt = 10;
9
10 const app = express();
11 app.use(express());
12 app.use(express.json());
13 app.use(cors({
14     origin: ["http://localhost:3000"],
15     methods: ["POST", "GET", "PUT", "DELETE"],
16     credentials: true
17 }));
18 app.use(cookieParser());
19
20
21 const db = mysql.createConnection({
22     host: "localhost",
23     user: "root",
24     password: "password",
25     database: "saludame_prod"
26 });
```

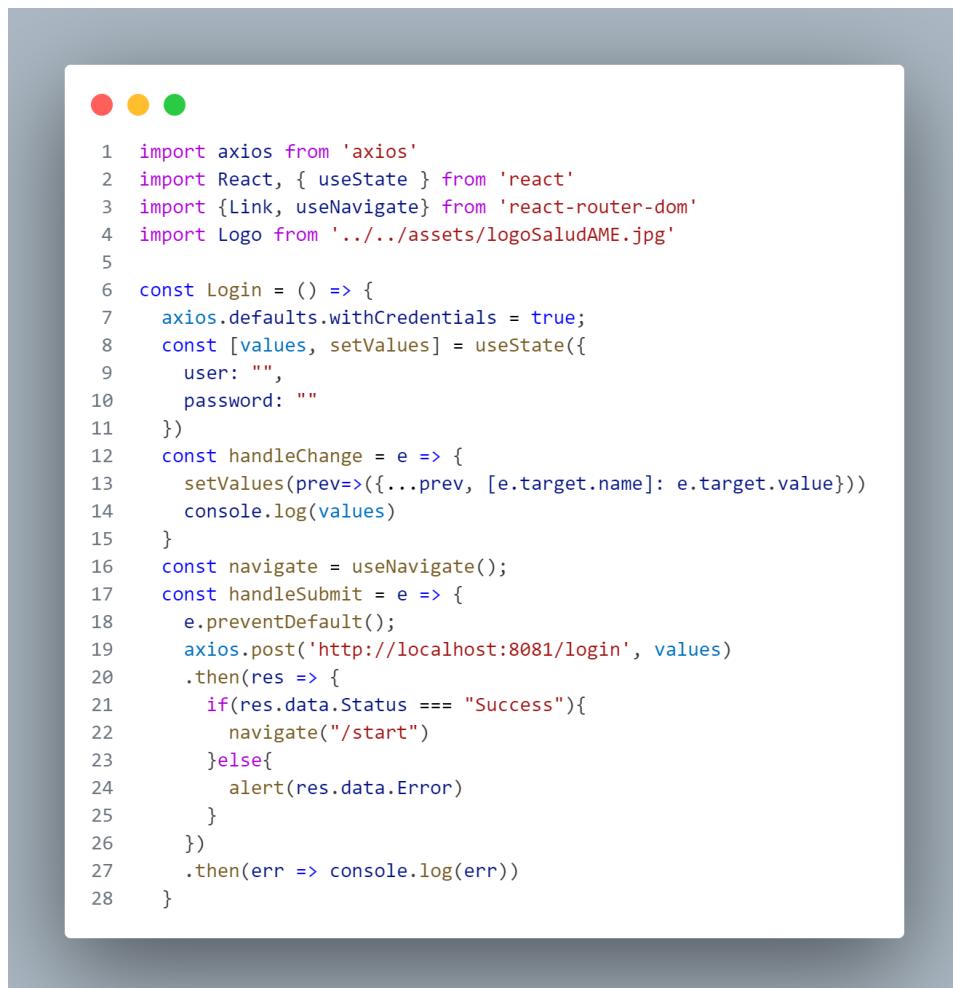
Fuente: Elaboración propia

### 3.2.4.3. Codificación del módulo de inicio de sesión

El módulo de inicio de sesión está compuesto por los siguientes elementos: el componente Login.jsx y el componente HomeSistema.jsx por parte del frontend, y una petición GET y POST en el backend.

El componente Login es un formulario de inicio de sesión que permite a los usuarios autenticarse con sus credenciales. El componente se divide en dos partes: una que maneja la parte lógica y la otra que retorna una estructura JSX que define el estilo del formulario. A continuación, se describe la estructura lógica del componente:

**Figura 36: Estructura lógica del componente Login.jsx**



```
1 import axios from 'axios'
2 import React, { useState } from 'react'
3 import {Link, useNavigate} from 'react-router-dom'
4 import Logo from '../../assets/logoSaludAME.jpg'
5
6 const Login = () => {
7   axios.defaults.withCredentials = true;
8   const [values, setValues] = useState({
9     user: "",
10    password: ""
11  })
12  const handleChange = e => {
13    setValues({...prev, [e.target.name]: e.target.value})
14    console.log(values)
15  }
16  const navigate = useNavigate();
17  const handleSubmit = e => {
18    e.preventDefault();
19    axios.post('http://localhost:8081/login', values)
20    .then(res => {
21      if(res.data.Status === "Success"){
22        navigate("/start")
23      }else{
24        alert(res.data.Error)
25      }
26    })
27    .then(err => console.log(err))
28  }

```

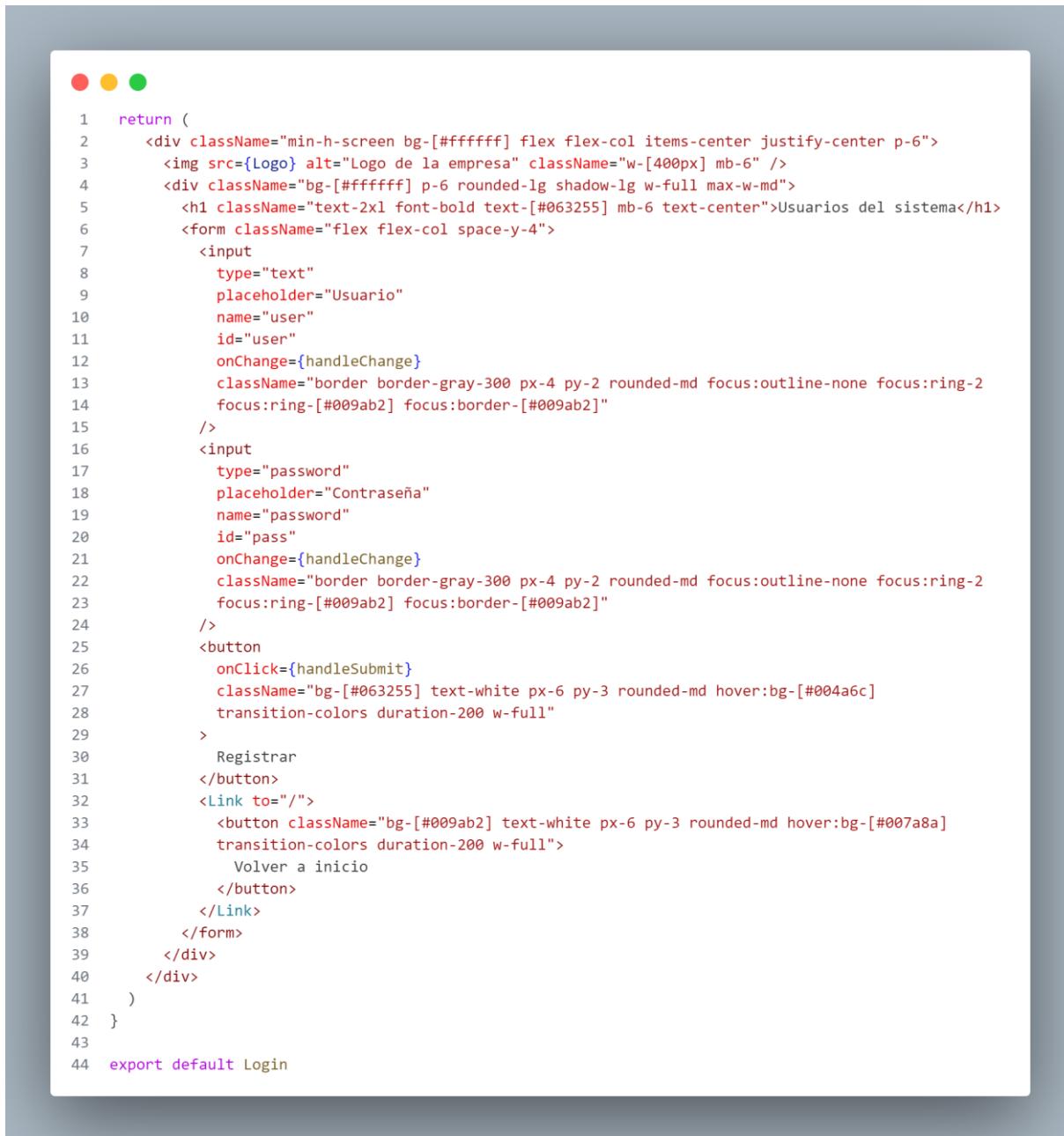
*Fuente: Elaboración propia*

En donde:

- Se importan las librerías a utilizar, las cuales son: Axios para realizar solicitudes HTTP al backend, React y useState para crear el componente y manejar su estado interno, Link y useNavigate para la navegación entre páginas, y el Logo que es una imagen que se muestra en el formulario.
- axios.defaults.withCredentials se configura para incluir las cookies en las solicitudes.
- useState se usa para manejar el estado de las entradas del formulario, inicializándolo con valores vacíos.
- handleChange es una función que se llama cada vez que hay un cambio en los campos de entrada del formulario, se encarga de actualizar el estado values con los nuevos valores introducidos por el usuario.
- La función handleSubmit se llama cuando el usuario presiona el botón de enviar: primero previene el comportamiento por defecto del formulario, luego envía una solicitud POST al backend con los valores del formulario, y si la respuesta es exitosa, redirige al usuario a la página de inicio usando la variable navigate; caso contrario muestra una alerta con el mensaje de error.

A continuación, se describe la parte de la estructura y diseño del formulario.

**Figura 37: Estructura visual del componente Login.jsx**



```
1  return (
2    <div className="min-h-screen bg-[#ffffff] flex flex-col items-center justify-center p-6">
3      <img src={Logo} alt="Logo de la empresa" className="w-[400px] mb-6" />
4      <div className="bg-[#ffffff] p-6 rounded-lg shadow-lg w-full max-w-md">
5        <h1 className="text-2xl font-bold text-[#063255] mb-6 text-center">Usuarios del sistema</h1>
6        <form className="flex flex-col space-y-4">
7          <input
8            type="text"
9            placeholder="Usuario"
10           name="user"
11           id="user"
12           onChange={handleChange}
13           className="border border-gray-300 px-4 py-2 rounded-md focus:outline-none focus:ring-2
14           focus:ring-[#009ab2] focus:border-[#009ab2]"
15           />
16          <input
17            type="password"
18            placeholder="Contraseña"
19            name="password"
20            id="pass"
21            onChange={handleChange}
22            className="border border-gray-300 px-4 py-2 rounded-md focus:outline-none focus:ring-2
23           focus:ring-[#009ab2] focus:border-[#009ab2]"
24           />
25          <button
26            onClick={handleSubmit}
27            className="bg-[#063255] text-white px-6 py-3 rounded-md hover:bg-[#004a6c]
28           transition-colors duration-200 w-full"
29           >
30            Registrar
31          </button>
32          <Link to="/">
33            <button className="bg-[#009ab2] text-white px-6 py-3 rounded-md hover:bg-[#007a8a]
34           transition-colors duration-200 w-full">
35            Volver a inicio
36          </button>
37        </Link>
38      </form>
39    </div>
40  </div>
41 )
42 }
43
44 export default Login
```

*Fuente: Elaboración propia*

En donde:

- Se utiliza Tailwind CSS para aplicar estilos.
- La imagen del logo se muestra en la parte superior del formulario.

- El formulario incluye campos de entrada para el usuario y la contraseña, un botón para enviar el formulario y un enlace para volver a la página de inicio.
- min-h-screen, bg-[#ffffff], flex, flex-col, items-center, justify-center, p-6 son clases de Tailwind CSS que configuran el diseño del formulario.
- Los campos de entrada actualizan el estado values con cada cambio.
- Al enviar el formulario, se realiza una solicitud POST para autenticar al usuario.

Por parte del backend, se realiza una solicitud POST, que hace un SELECT a la base de datos donde el nombre de usuario se lo pasa como valor en el cuerpo de la página (desde el frontend). Una vez ejecutada la consulta, verifica si es que existe el usuario y que los datos enviados sean mayores a 0, si la comprobación es correcta se verifica si la cuenta no está bloqueada comparando el campo del tiempo bloqueado de la base de datos con la fecha actual del sistema, si la comprobación es correcta se procede a comparar la contraseña pasada por el cuerpo de la página con la contraseña obtenida de la base de datos comparando los hashes de la contraseña, si la comprobación es correcta se procede a guardar el nombre, el identificador del usuario, nombre de usuario y el rol asignado a la cookie del navegador que tiene como expiración un tiempo de 1 hora y devuelve el estado “Success” (exitoso, por su traducción del inglés) y se actualizan a 0 los intentos fallidos; caso contrario de que fallaran algunas de las comprobaciones, devuelve un mensaje de error al frontend y en caso de introducirse mal la contraseña aumenta el contador de intentosFallidos, y si el usuario cuenta con más de 4 intentos fallidos, se le bloquea el acceso a la cuenta por 10 minutos. A continuación, se muestra el código que se ejecuta:

**Figura 38: Parte del backend del componente de iniciar sesión**

```
78 app.post('/login', (req, res) => {
79   const q = "SELECT * FROM usuario WHERE usuario = ? AND estado = 1";
80   const s = req.body.user;
81
82   db.query(q, s, (err, data) => {
83     if (err) return res.json({ Error: "Error en el servidor" });
84     if (data.length > 0) {
85       const ahora = new Date();
86       const usuario = data[0];
87
88       // Verificar si el usuario está bloqueado
89       if (usuario.tiempoBloqueo && ahora < new Date(usuario.tiempoBloqueo)) {
90         return res.json({ Error: "Cuenta bloqueada. Inténtelo más tarde." });
91       }
92
93       bcrypt.compare(req.body.password, usuario.contraseña, (err, response) => {
94         if (err) return res.json({ Error: "Error en el servidor" });
95
96         if (response) {
97           // Resetear intentos fallidos si el inicio de sesión es exitoso
98           const resetIntentos = "UPDATE usuario SET intentosFallidos = 0, tiempoBloqueo = NULL WHERE uid = ?";
99           db.query(resetIntentos, [usuario.uid], (err) => {
100             if (err) return res.json({ Error: "Error en el servidor" });
101           });
102
103           const name = usuario.nombre;
104           const ridFK = usuario.ridFK;
105           const uid = usuario.uid;
106           const token = jwt.sign({ name, ridFK, uid }, "jwt-secret-key", { expiresIn: '1h' });
107
108           res.cookie('token', token);
109           return res.json({ Status: "Success" });
110
111     } else {
112       // Incrementar intentos fallidos si la contraseña es incorrecta
113       let intentosFallidos = usuario.intentosFallidos + 1;
114       let tiempoBloqueo = null;
115
116       if (intentosFallidos >= MAX_INTENTOS) {
117         intentosFallidos = 0;
118         tiempoBloqueo = new Date(ahora.getTime() + TIEMPO_BLOQUEO);
119       }
120
121       const updateIntentos = "UPDATE usuario SET intentosFallidos = ?, tiempoBloqueo = ? WHERE uid = ?";
122       db.query(updateIntentos, [intentosFallidos, tiempoBloqueo, usuario.uid], (err) => {
123         if (err) return res.json({ Error: "Error en el servidor" });
124       });
125
126       return res.json({ Error: "Error: usuario y/o contraseña incorrecto" });
127     }
128   });
129 } else {
130   return res.json({ Error: "Error: usuario y/o contraseña incorrecto" });
131 }
132 });
133});
```

*Fuente: Elaboración propia*

A continuación, se presentan los mockups del módulo de inicio de sesión.

**Figura 39: Inicio de sesión**



*Fuente: Elaboración propia*

Una vez el usuario se haya autentificado de manera exitosa, se lo redirige a la página inicial del sistema, si es la primera vez que inicia sesión se le pedirá que cambie su contraseña, una vez realizado dicho paso se lo redirigirá a la página inicial del sistema.

Si el usuario se llega a olvidar la contraseña, existe un botón para restablecer la contraseña, el cual le pide al usuario ingresar su nombre de usuario, y el administrador puede cambiar su contraseña por una temporal, para después hacer que el usuario cambie nuevamente su contraseña por motivos de seguridad.

El componente de la página principal se encarga de mostrar diferentes botones de acuerdo con los roles asignados a cada usuario en la base de datos. A continuación, se detalla la implementación lógica y el funcionamiento visual de este componente:

**Figura 40: Estructura lógica de la página inicial**



```
1 import axios from 'axios';
2 import React, { useEffect, useState } from 'react'
3 import { Link, useNavigate } from 'react-router-dom'
4 import Cookies from 'js-cookie'
5 import {jwtDecode} from 'jwt-decode'
6 import Administrador from '../pages/homepages/Administrador';
7 import Medico from '../pages/homepages/Medico';
8 import Secretaria from '../pages/homepages/Secretaria';
9
10
11 const HomeSistema = () => {
12   axios.defaults.withCredentials = true;
13   const [auth, setAuth] = useState(false);
14   const [message, setMessage] = useState('')
15   const [name, setName] = useState('')
16   const [ridFK, setRidFK] = useState(null)
17   const navigate = useNavigate();
18
19   useEffect(()=> {
20     axios.get('http://localhost:8081/start')
21     .then(res => {
22       if(res.data.Status === "Success"){
23         setAuth(true);
24         setName(res.data.name)
25       }else{
26         setAuth(false)
27         setMessage(res.data.Error)
28       }
29     })
30     .then(err => console.log(err))
31   })
32
33   useEffect(() => {
34     const token = Cookies.get('token');
35     if (token) {
36       try {
37         const decodedToken = jwtDecode(token);
38         setRidFK(decodedToken.ridFK);
39       } catch (err) {
40         console.error('Invalid token');
41       }
42     }
43   }, []);

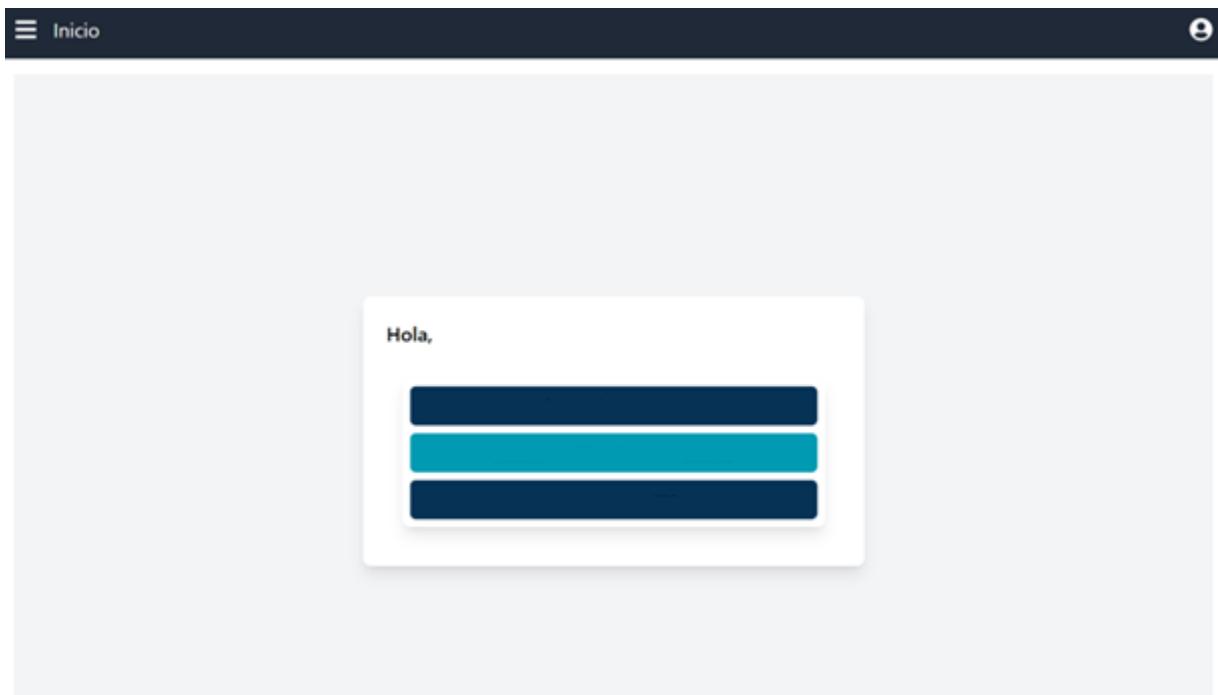
```

*Fuente: Elaboración propia*

En primer lugar, se importan las librerías y componentes necesarios y se declaran las variables y los métodos para inicializar dichas variables. Luego se realiza una petición GET al servidor en la cual se verifica si el usuario ha iniciado sesión, para colocar la variable auth en verdadero y colocar el nombre del usuario, y por último verifica si existe un token en las cookies del navegador, y si es verdadero coloca el rol en la variable correspondiente.

En la parte visual del componente, si el usuario está autenticado se muestra un mensaje de bienvenida con el nombre del usuario y se renderiza el componente según el rol; caso contrario se muestra un mensaje de error con opciones para iniciar sesión o volver a la página de inicio. A continuación, se presenta el mockup de la página inicial.

**Figura 41: Mockup de la página inicial**

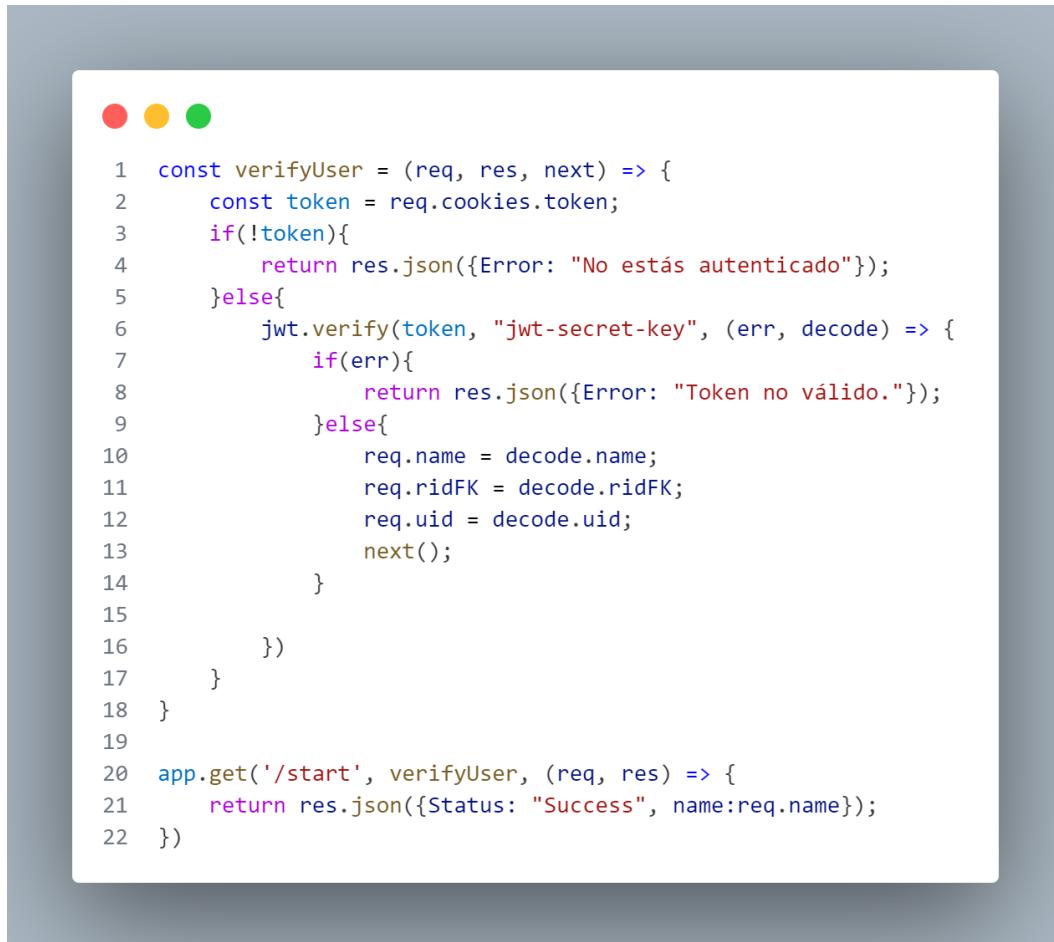


*Fuente: Elaboración propia*

En el backend, se utiliza la solicitud GET junto con la función verificarUsuario para saber si es que el usuario inició sesión verificando si existe un token almacenado en

las cookies del navegador. Si es verdadero, devuelve los valores decodificados del token y una respuesta exitosa al frontend, caso contrario, devuelve un mensaje de error. A continuación, se muestra la parte del código ejecutado.

**Figura 42: Backend del componente de la página inicial**



```
1  const verifyUser = (req, res, next) => {
2      const token = req.cookies.token;
3      if(!token){
4          return res.json({Error: "No estás autenticado"});
5      }else{
6          jwt.verify(token, "jwt-secret-key", (err, decode) => {
7              if(err){
8                  return res.json({Error: "Token no válido."});
9              }else{
10                  req.name = decode.name;
11                  req.ridFK = decode.ridFK;
12                  req.uid = decode.uid;
13                  next();
14              }
15          })
16      }
17  }
18 }
19
20 app.get('/start', verifyUser, (req, res) => {
21     return res.json({Status: "Success", name:req.name});
22 })
```

*Fuente: Elaboración propia*

#### 3.2.4.4. Codificación del módulo de usuarios

El módulo de usuarios se compone por 3 componentes diferentes: Users, Register y UpdateUsers por parte del frontend, y por parte del backend se presentan 4 solicitudes: GET, POST y PUT para realizar el CRUD a la base de datos.

El componente Users muestra a todos los usuarios registrados en una tabla con los campos ID, Nombre, Apellido Paterno, Apellido Materno, Usuario, Rol, Estado y una columna llamada Acción que contiene los botones para editar y eliminar a los usuarios. Arriba de la tabla existe un botón para crear un nuevo usuario. A continuación, se detalla la implementación lógica y visual de este componente:

**Figura 43: Estructura lógica del componente principal de usuarios**



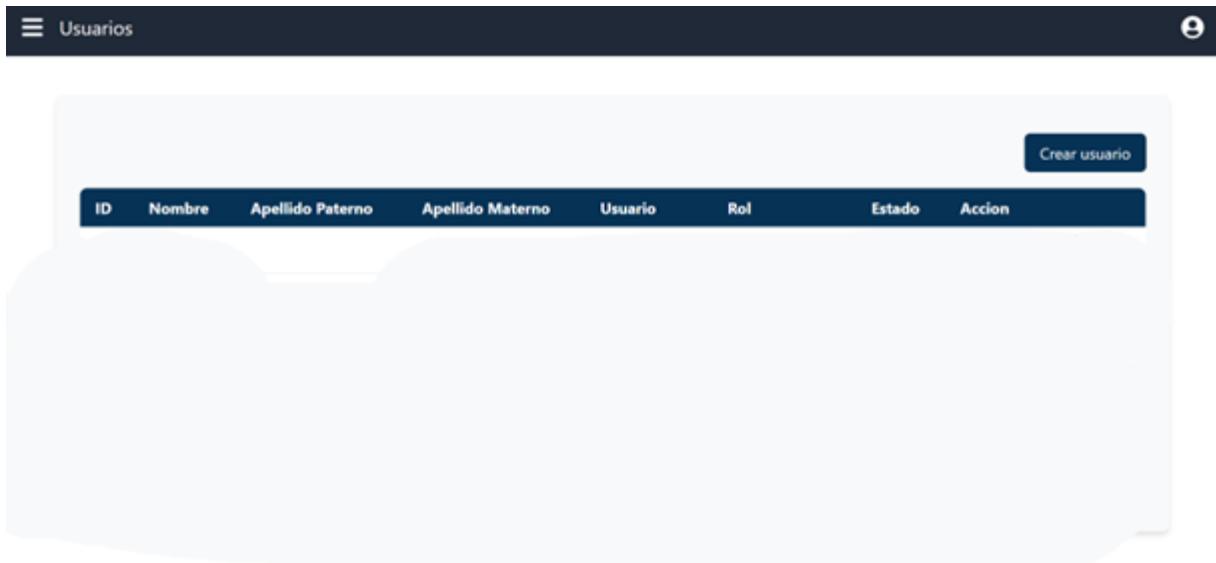
```
1 import axios from 'axios'
2 import React, { useEffect, useState } from 'react'
3 import { Link, useNavigate } from 'react-router-dom'
4
5 const Users = () => {
6     axios.defaults.withCredentials = true;
7     const [data, setData] = useState([]);
8     const navigate = useNavigate();
9
10    useEffect(() => {
11        axios.get('http://localhost:8081/users')
12            .then(res => {
13                if (Array.isArray(res.data)) {
14                    setData(res.data);
15                } else {
16                    alert("Error al cargar los datos");
17                }
18            })
19            .catch(err => alert(err))
20    }, [])
21
22    const handleDelete = (ID) => {
23        axios.delete('http://localhost:8081/deleteUser/' + ID)
24            .then(res => {
25                console.log(res)
26                window.location.reload()
27                navigate("/users")
28            })
29            .catch(err => console.log(err))
30    }
}
```

*Fuente: Elaboración propia*

Primeramente, se importan las librerías necesarias y se inicializa la variable que contendrá los datos obtenidos. Luego se utiliza un useEffect para mandar una solicitud GET al backend para obtener los datos, luego pregunta si estos datos son un array, en cuyo caso se colocan los datos en la variable data, caso contrario muestra un mensaje de error.

En la parte visual, se muestran los campos en una tabla, los cuales son el Id del usuario, el nombre completo, su nombre de usuario y contraseña, el rol asignado y el estado del usuario. A continuación, se describe el mockup del componente.

**Figura 44: Mockup del componente principal de usuarios**



| ID | Nombre | Apellido Paterno | Apellido Materno | Usuario | Rol | Estado | Accion               |
|----|--------|------------------|------------------|---------|-----|--------|----------------------|
|    |        |                  |                  |         |     |        | <b>Crear usuario</b> |

*Fuente: Elaboración propia*

Por parte del backend, se cuenta con un método GET para obtener los usuarios de la base de datos. A continuación, se presenta el código mencionado.

**Figura 45: Backend para mostrar y eliminar a los usuarios registrados**

```
141 app.get('/users', verifyUser, checkRole(1), (req,res) => {
142   const q = 'select u.uid as "ID", u.nombre, u.apellidoPaterno, u.apellidoMaterno,' +
143   ' u.usuario, r.nombreRol as "Rol", ' +
144   'case when u.estado = 1 then "Activo" else "Inactivo" end as "Estado" ' +
145   'from usuario u join rol r on u.ridFK = r.rid';
146   db.query(q, (err, result) => {
147     if(err) return res.json({Error: "Error inside server"});
148     return res.json(result);
149   })
150 })
151
```

*Fuente: Elaboración propia*

Cuando se hace clic al botón de crear usuario, se renderiza el componente Register.jsx, que carga el formulario con los campos necesarios para registrar el usuario. A continuación, se detalla el componente en su estructura lógica y visual

Figura 46: Estructura lógica del componente Register.jsx

```
1 import React from 'react'
2 import { useState } from 'react'
3 import {Link, useNavigate} from 'react-router-dom'
4 import axios from 'axios'
5
6 const Register = () => {
7   const [values, setValues] = useState({
8     name: "",
9     apellidoMaterno: "",
10    apellidoPaterno: "",
11    user: "",
12    password: "",
13    ridFK:"",
14    estado: ""
15  })
16  const handleChange = e => {
17    setValues(prev=>({...prev, [e.target.name]: e.target.value}))
18  }
19  const navigate = useNavigate();
20  const handleSubmit = e => {
21    e.preventDefault();
22    axios.post('http://localhost:8081/createUser', values)
23    .then(res => {
24      console.log(res)
25      navigate("/users")
26    })
27    .then(err => {
28      alert(err)
29      navigate("/users")
30    });
31  }
}
```

*Fuente: Elaboración propia*

En primer lugar, se importan las librerías a utilizar y se inicializan los valores a registrar, luego se declara la función para detectar los cambios en los campos y capturar los valores en las variables ya inicializadas. Cuando el usuario hace clic en el botón de registrar, se llama a la función handleSubmit, la cual hace una solicitud POST al

backend, si la respuesta es exitosa redirige al usuario a la vista de usuarios, caso contrario muestra un mensaje de error.

Para la parte visual, se cuentan con los campos para registrar el nombre completo, el nombre de usuario, la contraseña, el rol y el estado del usuario, un botón para registrar y otro para volver a la página anterior. A continuación, se presenta el mockup del componente.

**Figura 47: Mockup del componente para registrar usuario**

The form is titled "Crear usuario". It contains the following fields:

- Nombre (Name)
- Apellido Paterno (Last Name)
- Apellido Materno (Last Name)
- Usuario (User)
- Contraseña (Password)
- Selección Rol (Select Role)
- Selección Estado (Select State)

At the bottom are two buttons:

- Registrar (Register) in a dark blue box
- Volver atrás (Go back) in a teal box

*Fuente: Elaboración propia*

Por parte del backend, se cuenta con una solicitud POST que hace un INSERT en la base de datos, encriptando la contraseña en el proceso usando un hash y salt. A continuación, se muestra el código del backend.

Figura 48: Backend para la inserción de usuarios

```
125 app.post('/createUser', verifyUser, checkRole(1), (req, res) => {
126   const q = "INSERT INTO usuario('nombre', 'apellidoPaterno', 'apellidoMaterno', 'usuario', 'contraseña', 'ridFK', 'estado') VALUES (?, ?, ?, ?, ?, ?, ?)";
127   bcrypt.hash(req.body.password, salt, (err, hash) => {
128     if (err) {
129       console.error('Error hashing password:', err);
130       return res.json({ Error: "Error from hashing password" });
131     }
132     const values = [req.body.name, req.body.apellidoPaterno, req.body.apellidoMaterno, req.body.user, hash, req.body.ridFK, req.body.estado];
133     db.query(q, values, (err, result) => {
134       if (err) {
135         console.error('Error inserting data:', err);
136         return res.json({ Error: "Inserting data error in server" });
137       }
138       return res.json({ Status: "Success" });
139     });
140   });
141 })
```

Fuente: Elaboración propia

Cuando el usuario hace clic en el botón de actualizar usuario, se le pasa el id del usuario por la URL haciendo una solicitud GET, y se muestra el formulario de actualizar usuario con los datos del usuario seleccionado; cuando el usuario da clic al botón de Actualizar, se manda una solicitud PUT al backend. A continuación, se presenta el mockup del componente.

Figura 49: Mockup de la actualización de usuario

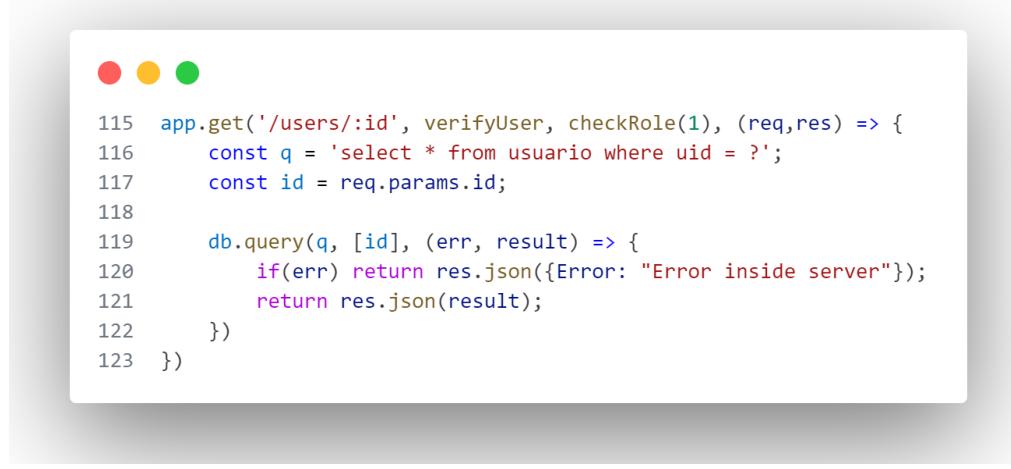
The mockup shows a user update form with the following fields:

- Text input: Marco
- Text input: Condarco
- Text input: Mirabal
- Text input: mcondarcoc
- Text input: Nueva contraseña
- Dropdown menu: Médico
- Dropdown menu: Activo
- Large blue button: Actualizar
- Large teal button: Volver atrás

Fuente: Elaboración propia

En el backend, se cuenta con la solicitud GET para devolver los datos del usuario con el id seleccionado que es enviado por parámetro mediante la URL, luego se cuenta con la solicitud PUT que se encarga de actualizar los datos en la base de datos. Si existe una contraseña en el campo, la encripta y actualiza en la base de datos. A continuación, se muestra el código ejecutado.

**Figura 50: Backend para obtener los datos del usuario seleccionado**



```

115  app.get('/users/:id', verifyUser, checkRole(1), (req,res) => {
116      const q = 'select * from usuario where uid = ?';
117      const id = req.params.id;
118
119      db.query(q, [id], (err, result) => {
120          if(err) return res.json({Error: "Error inside server"});
121          return res.json(result);
122      })
123  })

```

*Fuente: Elaboración propia*

**Figura 51: Backend para actualizar datos del usuario**



```

143  app.put('/update/:id', verifyUser,checkRole(1), (req, res) =>{
144      const id = req.params.id;
145      const { name, user, password, ridFK, estado, apellidoPaterno, apellidoMaterno } = req.body;
146
147      if (!password) {
148          // Si la contraseña está vacía, actualiza solo los otros campos
149          const q = 'UPDATE usuario SET `nombre` = ?, `usuario` = ?, `ridFK` = ?, `estado` = ?, `apellidoPaterno` = ?, `apellidoMaterno` = ? WHERE uid = ?';
150          const values = [name, user, ridFK, estado, apellidoPaterno, apellidoMaterno, id];
151          db.query(q, values, (err, result) => {
152              if (err) return res.json({ Error: "Error inside server" });
153              return res.json(result);
154          });
155      } else {
156          // Si hay una contraseña, hashearla y actualizar todos los campos
157          bcrypt.hash(password, salt, (err, hash) => {
158              if (err) return res.json({ Error: "Error from hashing password" });
159              const q2 = 'UPDATE usuario SET `nombre` = ?, `usuario` = ?, `contraseña` = ?, `ridFK` = ?, `estado` = ?, `apellidoPaterno` = ?, `apellidoMaterno` = ? WHERE uid = ?';
160              const values = [name, user, hash, ridFK, estado, apellidoPaterno, apellidoMaterno, id];
161              db.query(q2, values, (err, result) => {
162                  if (err) return res.json({ Error: "Inserting data error in server" });
163                  return res.json(result);
164              });
165          });
166      }
167  })

```

*Fuente: Elaboración propia*

### 3.2.4.5. Codificación del módulo del libro de observaciones

El módulo del libro de observaciones está compuesto por 3 componentes en el frontend para crear, actualizar y mostrar los datos de los libros de observaciones, y solicitudes GET y POST en el backend.

En el componente principal que muestra los datos de los libros de observaciones, el usuario tiene que filtrar el colegio, la gestión y el mes del libro de observaciones a mirar, se le envía una solicitud GET al backend y devuelve en una tabla los datos de todos los libros de observaciones, así como da la opción para crear un nuevo registro en ese libro de observaciones, actualizar y eliminar los datos individuales de la tabla.

A continuación, se muestra el mockup del componente:

**Figura 52: Mockup del componente del libro de observaciones**



*Fuente: Elaboración propia*

Igualmente se cuenta con un formulario para crear registro y actualizar los datos del libro de observaciones correspondiente. Para agregar datos, primero se busca al cliente mediante el nombre, apellido materno y apellido paterno, una vez encontrado el cliente, se procede con el llenado de los campos correspondientes. A continuación, se muestra el mockup del componente:

**Figura 53: Formulario para la inserción de datos en el libro de observaciones**

**Buscar Cliente y Agregar Registro**

|                       |
|-----------------------|
| Carnet de Identidad   |
| 9667582               |
| <b>Buscar Cliente</b> |

**Agregar datos para Ariel Tellez Torrico**

|                         |
|-------------------------|
| 22/10/2024              |
| Absceso                 |
| Tratamiento             |
| Observaciones           |
| <b>Agregar Registro</b> |
| <b>Volver atrás</b>     |

*Fuente: Elaboración propia*

Para el backend, se cuenta con un método GET para obtener el nombre completo del cliente junto con su id, el dato para obtener el libro del cliente al que pertenece se obtiene mediante los parámetros de las URLs, luego se cuenta con una solicitud POST para registrar los datos del cliente. A continuación, se muestra el código ejecutado.

**Figura 54: Backend para buscar al cliente en la base de datos**

```
307 app.get('/searchCliente',(req, res) => {
308     const {nombre, apellidoPaterno, apellidoMaterno} = req.query;
309     const q = "select cid, nombre, apellidoPaterno, apellidoMaterno" +
310     "from cliente where nombre = ? and apellidoPaterno = ? and apellidoMaterno = ?;";
311     db.query(q, [nombre, apellidoPaterno, apellidoMaterno], (err, result) => {
312         if(err) return res.json({Error: "Error inside server"});
313         else if (result.length > 0) {
314             return res.json(result);
315         }else{
316             res.status(404).json({Error: 'No existen pagos' });
317         }
318     })
319 })
```

*Fuente: Elaboración propia*

**Figura 55: Backend para insertar datos en libro de observaciones**

```
232 app.post('/createRegistro/:lid', (req, res) => {
233     const q = "INSERT INTO datos_observacion (`fechaAtendido`, `diagnostico`, `tratamiento`, `observaciones`, `"
234     + "cidFK2`, `uidFK3`, `lidFK1`)VALUES (?, ?, ?, ?, ?, ?, ?)";
235     const values = [req.body.fechaAtendido, req.body.diagnostico, req.body.tratamiento, req.body.observaciones,
236     req.body.cidFK2, req.body.uidFK3, req.body.lidFK1];
237     db.query(q, values, (err, result) => {
238         if (err) {
239             console.error('Error insertando datos:', err);
240             return res.json({ Error: "Inserting data error in server" });
241         }
242         return res.json({ Status: "Success" });
243     });
244 })
```

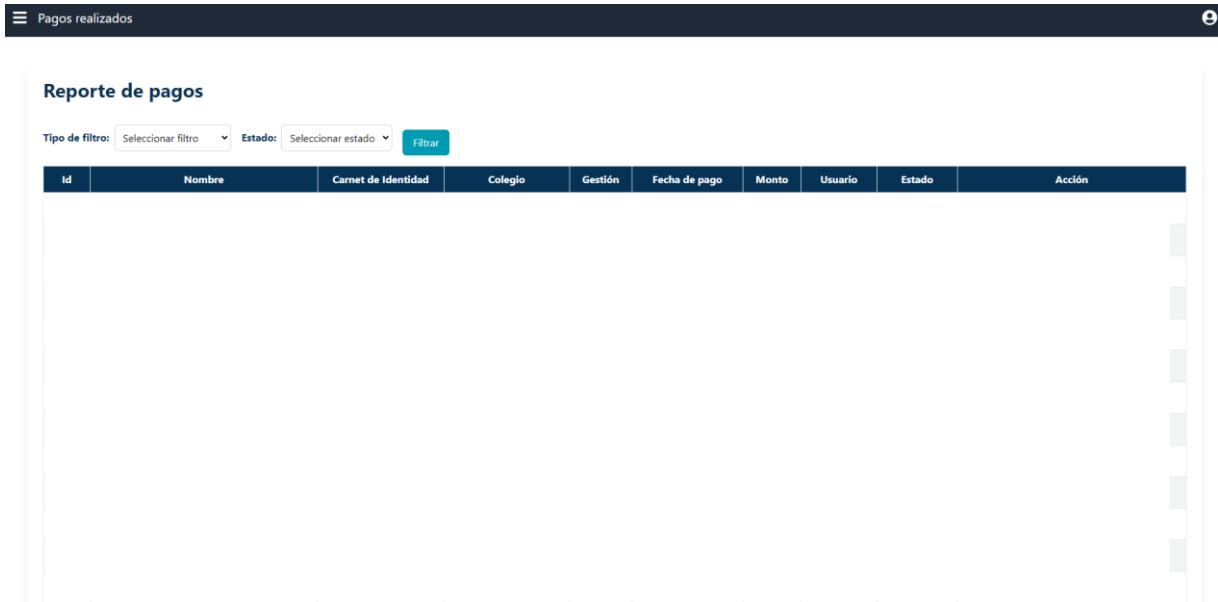
*Fuente: Elaboración propia*

### 3.2.4.6. Codificación del módulo de pagos

El módulo de pagos realizados cuenta con un componente para mostrar los pagos realizados y otro componente para registrar pagos, y solicitudes GET y POST en el backend.

El componente para mostrar la tabla de pagos devuelve una tabla desde el backend en la que se muestran todos los pagos realizados, con la opción de eliminar el pago; y encima de la tabla se muestra un botón para agregar un nuevo pago. A continuación, se muestra el mockup del componente.

**Figura 56: Mockup del componente que muestra los pagos realizados**



The mockup shows a user interface for managing payments. At the top, a dark header bar contains the text 'Pagos realizados' on the left and a user icon on the right. Below the header is a sub-header 'Reporte de pagos' in a teal box. Underneath are two dropdown menus: 'Tipo de filtro:' and 'Estado:', both with placeholder text 'Seleccionar filtro' and 'Seleccionar estado'. To the right of these is a blue 'Filtrar' button. Below these controls is a table with a dark header row containing columns: 'Id', 'Nombre', 'Carnet de Identidad', 'Colegio', 'Gestión', 'Fecha de pago', 'Monto', 'Usuario', 'Estado', and 'Acción'. The main body of the table is currently empty, showing only the column headers.

*Fuente: Elaboración propia*

En el backend, se cuenta con un método GET que devuelve una tabla con todos los campos necesarios. A continuación, se muestra el código ejecutado:

**Figura 57: Backend para mostrar la tabla de pagos realizados**

```
270 app.get('/pagos',(req, res) => {
271   const q = "select p.pid as 'Id', CONCAT(c.nombre, ' ', c.apellidoPaterno, ' ', c.apellidoMaterno) as 'Nombre', "
272   + "col.nombre as Colegio, c.curso as Curso, p.gestion as Gestion, p.fechaPago , p.monto, f.nombrePago as 'formaPago', u.usuario, p.fechaAgregado from pago p "
273   + "join cliente c on p.cidFK1 = c.cid join usuario u on p.uidFK2= u.uid join colegio col on c.colegio = col.cid join forma_pago f on p.formaPago = f.id",
274   db.query(q, (err, result) => {
275     if(err) return res.json({error: "Error inside server"});
276     else if (result.length > 0)
277       return res.json(result);
278     else{
279       res.status(404).json({Error: 'No existen pagos' });
280     }
281   })
282 })
```

*Fuente: Elaboración propia*

### 3.2.4.7. Codificación del módulo de subir pagos del asegurado

Este módulo lo conforman cuatro componentes. El primero se encarga de registrar la información del asegurado y verificar si cuenta con deudas pendientes en el sistema, en caso de que no se encuentren deudas se muestra un mensaje de error que le informa al usuario que no tiene deudas pendientes, en caso de no encontrar al asegurado en la base de datos se muestra un mensaje de error informando al usuario que no existe el asegurado en la base de datos, con la opción de agregarlo en caso de que fuera un estudiante nuevo.

Si existieran deudas pendientes, se muestra una imagen con un código QR para poder cancelar el seguro junto con un botón para subir el comprobante de pago al servidor. Una vez realizada la subida de la imagen, se procede a registrar los datos de contacto del asegurado para recibir su comprobante, y por último se muestra una pantalla para volver a la página inicial, informando que el recibo se podrá descargar desde la página correspondiente. Una vez la secretaría aprueba el pago, el usuario puede acceder al componente para descargar el recibo, filtrando por nombre o cédula de identidad.

### 3.2.4.8. Realización de pruebas funcionales

En este apartado se realizarán pruebas funcionales, para determinar el desempeño de nuestro sistema tanto en el frontend como en el backend.

## a) Prueba funcional al backend

Para llevar a cabo la prueba funcional automatizada de nuestra API REST, emplearemos Postman. Este cliente HTTP nos facilita la prueba de nuestros métodos implementados, y también permite automatizar este proceso. En la imagen siguiente, se muestra el resultado de una prueba automatizada al backend.

**Figura 58: Prueba funcional al backend**

```
GET Libros Consulta Medica
http://localhost:8081/libros
| PASS Status code is 200
200 OK 9 ms 897 B

GET Usuarios
http://localhost:8081/users
| FAIL Status code is 200 | AssertionError: expected response to have status code 200 but got 403
403 Forbidden 4 ms 329 B

GET Datos Libro Consulta Medica
http://localhost:8081/libros/3
| PASS Status code is 200
200 OK 4 ms 1.352 KB

GET Buscar Cliente
http://localhost:8081/searchCliente/?nombre=Bruno&apellidoPaterno=Melgar&apellidoMaterno=Arauz
| PASS Status code is 200
200 OK 5 ms 370 B

GET Pagos
http://localhost:8081/pagos
| PASS Status code is 200
200 OK 4 ms 1.819 KB

GET Buscar deudas cliente
http://localhost:8081/check-payment-status/9
| PASS Status code is 200
200 OK 4 ms 337 B

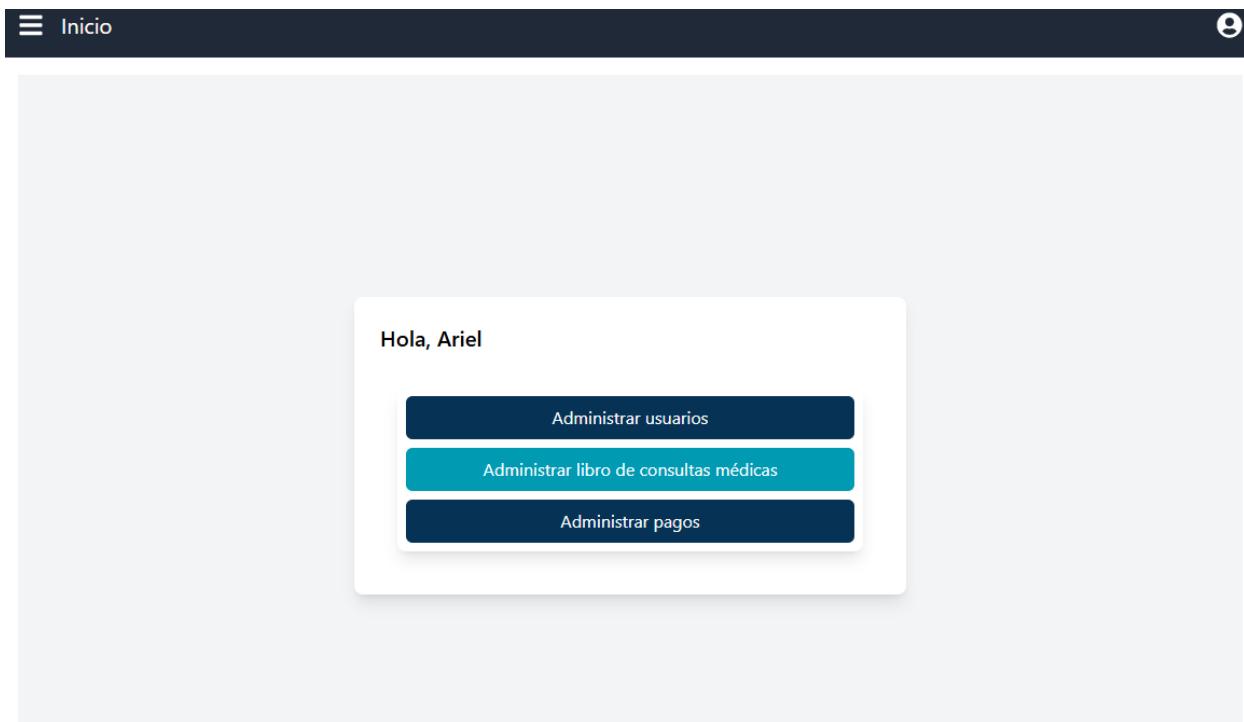
GET Nombres Colegios
http://localhost:8081/getColegios
| PASS Status code is 200
200 OK 3 ms 1.25 KB
```

*Fuente: Elaboración propia*

## b) Prueba funcional al frontend

Una vez realizada la prueba al backend, se desarrollaron las pruebas para comprobar el correcto funcionamiento del frontend con el backend. A continuación, se presentan una serie de imágenes de los resultados obtenidos.

**Figura 59: Administrador: página inicial**



*Fuente: Elaboración propia*

**Figura 60: Administrador: usuarios registrados**

Users Table Data:

| ID | Nombre   | Apellido Paterno | Apellido Materno | Usuario     | Rol           | Estado | Restablecer Contraseña? | Acción                  |
|----|----------|------------------|------------------|-------------|---------------|--------|-------------------------|-------------------------|
| 26 | Ariel    | Tellez           | Torrico          | admin       | Administrador | Activo | No                      | <button>Editar</button> |
| 29 | Gary     | Vasquez          | Ribera           | vasquezg    | Medico        | Activo | No                      | <button>Editar</button> |
| 33 | Andres   | Guerrero         | Camacho          | druguerro   | Secretaria    | Activo | No                      | <button>Editar</button> |
| 35 | Patricia | Ortiz            | Romero           | portizr     | Secretaria    | Activo | No                      | <button>Editar</button> |
| 36 | Mariel   | Tellez           | Perez            | mtellezp    | Medico        | Activo | No                      | <button>Editar</button> |
| 38 | Gloria   | Gonzales         | Fernandez        | ggonzalesf  | Secretaria    | Activo | No                      | <button>Editar</button> |
| 39 | Juan     | Perez            | Suarez           | jperezs     | Medico        | Activo | No                      | <button>Editar</button> |
| 40 | Yoelma   | Melendres        | Flores           | ymelendresf | Secretaria    | Activo | No                      | <button>Editar</button> |
| 41 | Marco    | Condarcos        | Mirabal          | mcondarcoc  | Medico        | Activo | Si                      | <button>Editar</button> |
| 42 | Risella  | Castro           | Ramos            | rcastro     | Administrador | Activo | Si                      | <button>Editar</button> |

*Fuente: Elaboración propia*

**Figura 61: Médico, libro de consultas médicas**

Search Filters:

- Colegio: Colegio Marista
- Gestión: 2024
- Mes: octubre
- Enviar
- Crear registro

Search Fields:

- Buscar asegurado por:
- Tipo de búsqueda: Nombre
- Nombre:
- Apellido Paterno:
- Apellido Materno:
- Filtrar

Consultations Table Data:

| Nombre | Apellidos      | Médico               | Fecha de atención | Diagnóstico       | Tratamiento | Observaciones       |
|--------|----------------|----------------------|-------------------|-------------------|-------------|---------------------|
| Ariel  | Tellez Cruz    | Gary Vasquez Ribera  | 20/10/2024, 00:00 | Dolor de garganta | Remedol     | Tomar cada 8 horas  |
| Ariel  | Tellez Torrico | Gary Vasquez Ribera  | 18/10/2024, 00:00 | Cefalea           | Ibuprofeno  | Tomar cada 8 horas  |
| Ariel  | Tellez Torrico | Gary Vasquez Ribera  | 18/10/2024, 00:00 | Cefalea           | Paracetamol | Volver a los 3 días |
| Ariel  | Tellez Torrico | Risella Castro Ramos | 18/10/2024, 00:00 | Dolor de cabeza   | Paracetamol | Tomar cada 8 horas  |

Page Navigation:

- Anterior
- Página 1 de 1
- Siguiente
- Exportar a Excel
- Exportar a PDF

*Fuente: Elaboración propia*

**Figura 62: Secretaria, pagos realizados**

**Reporte de pagos**

Tipo de filtro: Seleccionar filtro Estado: Seleccionar estado Colegio: Seleccionar colegio **Filtrar**

| Id   | Nombre                       | Carnet de Identidad | Colegio             | Gestión | Fecha de pago     | Monto | Usuario    | Estado      | Acción   |
|------|------------------------------|---------------------|---------------------|---------|-------------------|-------|------------|-------------|--|
| 2489 | Sofia Suarez Arana           | 3269446             | Colegio Ubaldi      | 2024    | 21/10/2024, 19:13 | 150   | admin      | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 2488 | Laura Fernandez Gutierrez    | 1413144             | Colegio Ubaldi      | 2024    | 21/10/2024, 15:45 | 150   | admin      | Por aprobar | <a href="#">Ver comprobante</a> <a href="#">Aprobar pago</a>     |
| 2487 | Cristy Agreda Vaca           | 9745632             | Colegio Ubaldi      | 2024    | 18/10/2024, 14:18 | 150   | admin      | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 2485 | Marcelo Fernandez Dorado     | 9483745             | Colegio Cristo Rey  | 2024    | 18/10/2024, 12:49 | 100   | admin      | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 2479 | Bruno Melgar Arauz           | 1293842             | Colegio Ubaldi      | 2024    | 28/9/2024, 08:44  | 150   | admin      | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 2477 | Mery Sandoval Apaza          | 9675328             | Colegio Ubaldi      | 2023    | 17/1/2024, 00:00  | 150   | ggonzalesf | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 1956 | KIARA ISABELA OREGGIA GARCIA |                     | Colegio Bertoluso   | 2023    | 13/7/2023, 11:45  | 140   | portizr    | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 2294 | MATEO PIZARRO PARADA         |                     | Colegio Cristo Rey  | 2024    | 15/11/2023, 14:13 | 100   | portizr    | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 607  | MARIANO CADARIO MERCADO      |                     | Colegio Señor Jesus | 2023    | 15/2/2023, 15:27  | 200   | admin      | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |
| 2457 | PATRICIA ELENA GODOY VILLA   |                     | Colegio Arakuarenda | 2024    | 18/12/2023, 10:10 | 150   | portizr    | Aprobado    | <a href="#">Ver comprobante</a> <a href="#">Descargar recibo</a> |

Anterior Página 1 de 244 Siguiente

Exportar a Excel Exportar a PDF

*Fuente: Elaboración propia*

**Figura 63: Asegurado: Buscarse en la base de datos**

**Paga tu seguro**

Escribe el carnet de identidad o el nombre completo del asegurado y presione buscar.

Nombre Completo

Nombre

Apellido Paterno

Apellido Materno

**Buscar asegurado**

Si es estudiante nuevo, puede registrarse [aquí](#).

**Volver al inicio**

*Fuente: Elaboración propia*

Figura 64: Asegurado, agregar a la base de datos

## Agregar Asegurado

Nombre:

Apellido Paterno:

Apellido Materno:

Cédula de identidad:

Fecha de nacimiento:

 ...

Sexo:

 ▼

Tipo de Sangre:

 ▼

Alergias:

Enfermedades base:

Colegio:

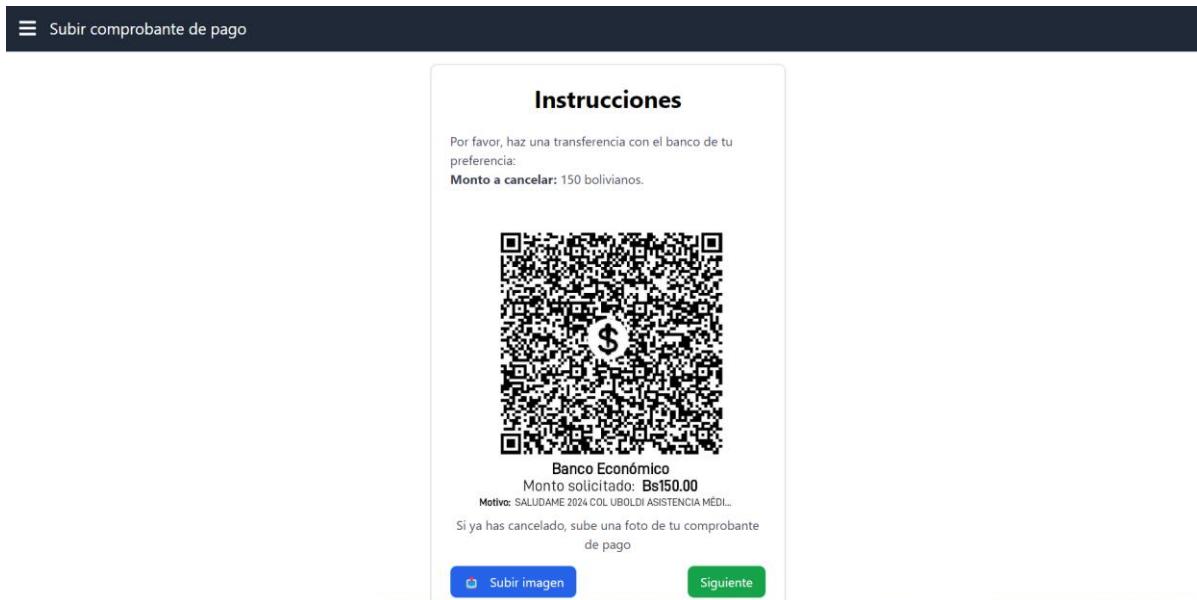
 ▼

Aregar

Volver atrás

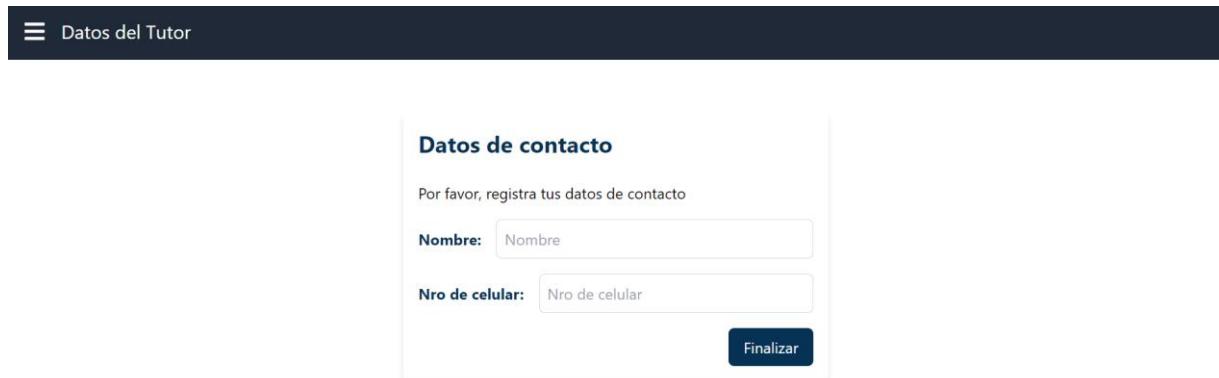
*Fuente: Elaboración propia*

**Figura 65: Asegurado: Subir imagen de la transferencia**



*Fuente: Elaboración propia*

**Figura 66: Asegurado: registrar datos de contacto**



*Fuente: Elaboración propia*

**Figura 67: Asegurado, pantalla de finalización del proceso**



*Fuente: Elaboración propia*

**Figura 68: Asegurado, pantalla para descargar recibo**



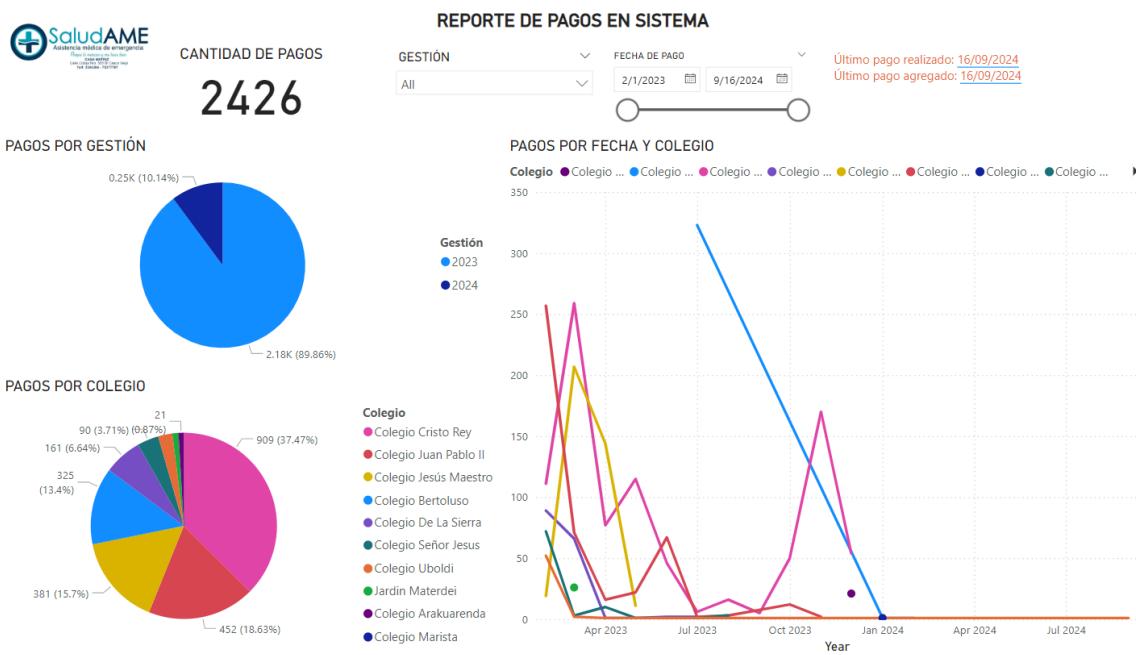
*Fuente: Elaboración propia*

Las imágenes mostradas previamente forman parte de la interfaz final del sistema.

### 3.2.4.9. Reportes del sistema

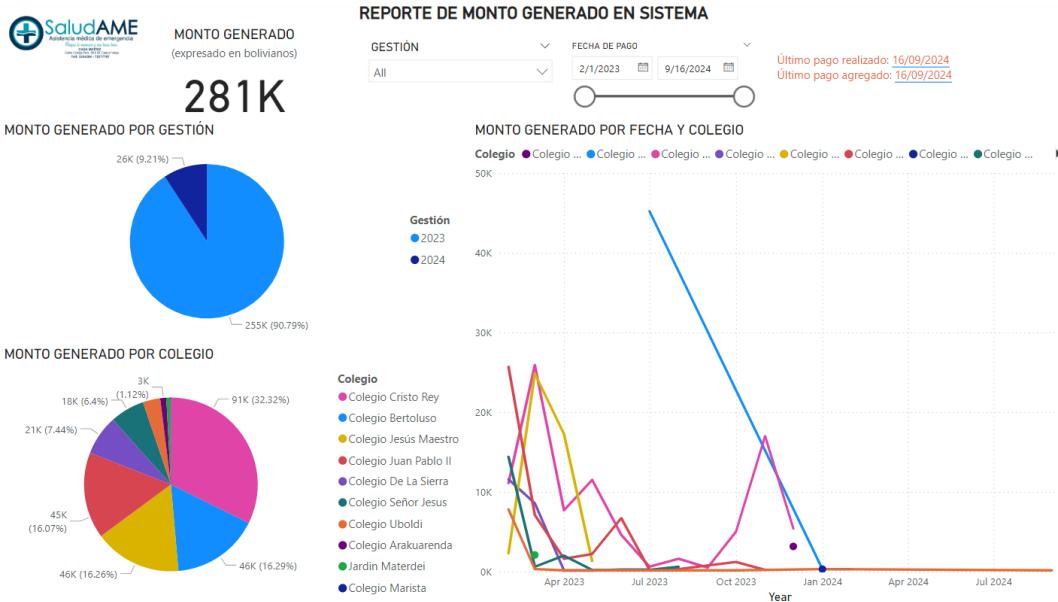
Los datos se almacenan en una base de datos MySQL, los cuales se pueden acceder desde una herramienta de visualización de datos para poder generar distintos reportes con los datos cargados. A continuación, se presentan los reportes creados a partir de los datos del sistema.

Figura 69: Reporte de pagos realizados



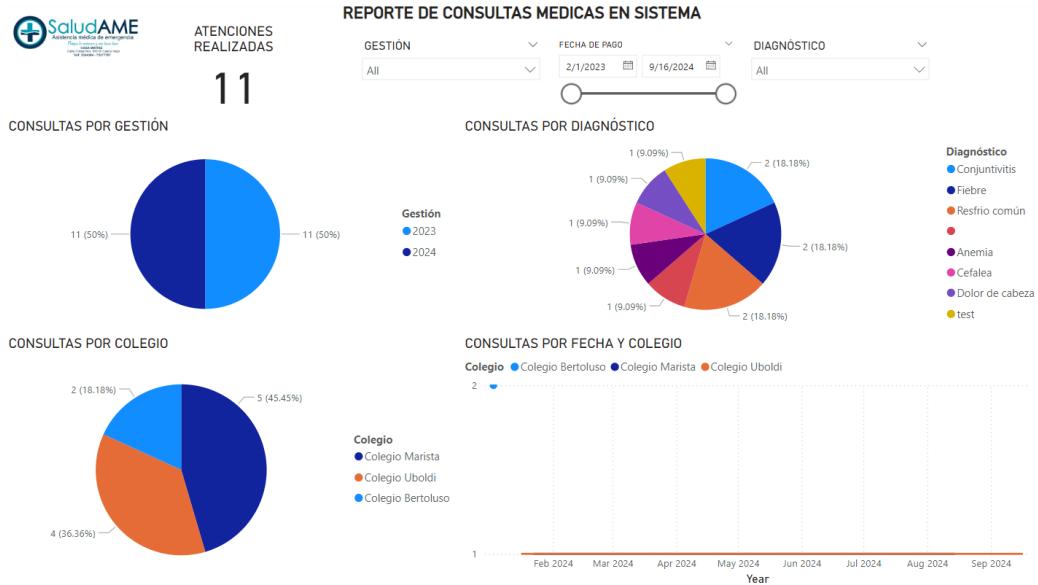
Fuente: Elaboración propia

**Figura 70: Reporte del monto generado en base a los pagos realizados**



*Fuente: Elaboración propia*

**Figura 71: Reporte de consultas atendidas**



*Fuente: Elaboración propia*

Los reportes permiten personalizar los datos mostrados, filtrando por gestión, rango de fechas y diagnóstico.

### 3.2.4.10. Aplicación Web Progresiva

Para realizar una aplicación web progresiva, se necesita que la aplicación cuente con un archivo manifest.json, que es el que nos da detalles sobre las distintas resoluciones del logo y el nombre de la aplicación, y un service worker para manejar la caché dinámica de la aplicación y agregarle funcionalidades offline.

A continuación se presenta el archivo manifest.json, el cual da detalles acerca del nombre de la aplicación y detalles sobre la resolución de los iconos:

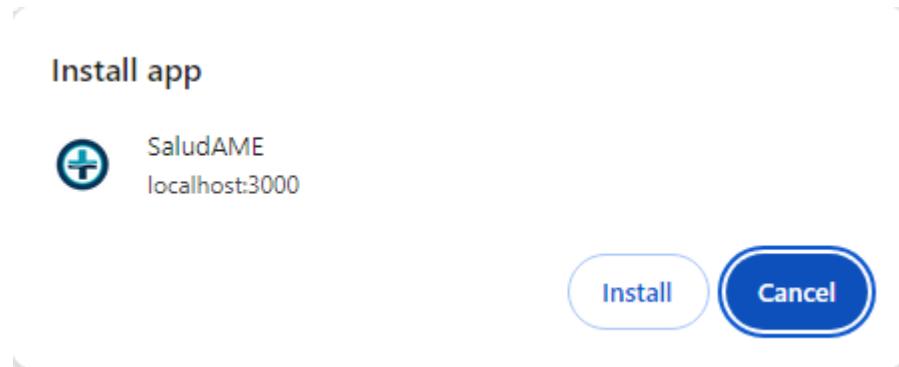
**Figura 72: Archivo manifest.json**

```
{
  "short_name": "SaludAME",
  "name": "SaludAME",
  "icons": [
    {
      "src": "favicon.ico",
      "sizes": "64x64 32x32 24x24 16x16",
      "type": "image/x-icon"
    },
    {
      "src": "logo192.png",
      "type": "image/png",
      "sizes": "192x192"
    },
    {
      "src": "logo512.png",
      "type": "image/png",
      "sizes": "512x512"
    }
  ],
  "start_url": ".",
  "display": "standalone",
  "theme_color": "#000000",
  "background_color": "#ffffff"
}
```

*Fuente: Elaboración propia*

Para determinar si una página es una aplicación web progresiva, el navegador ofrece la posibilidad de instalar la aplicación, lo cual se demuestra a continuación.

**Figura 73: Mensaje que muestra la posibilidad de instalar la aplicación.**



*Fuente: Elaboración propia*

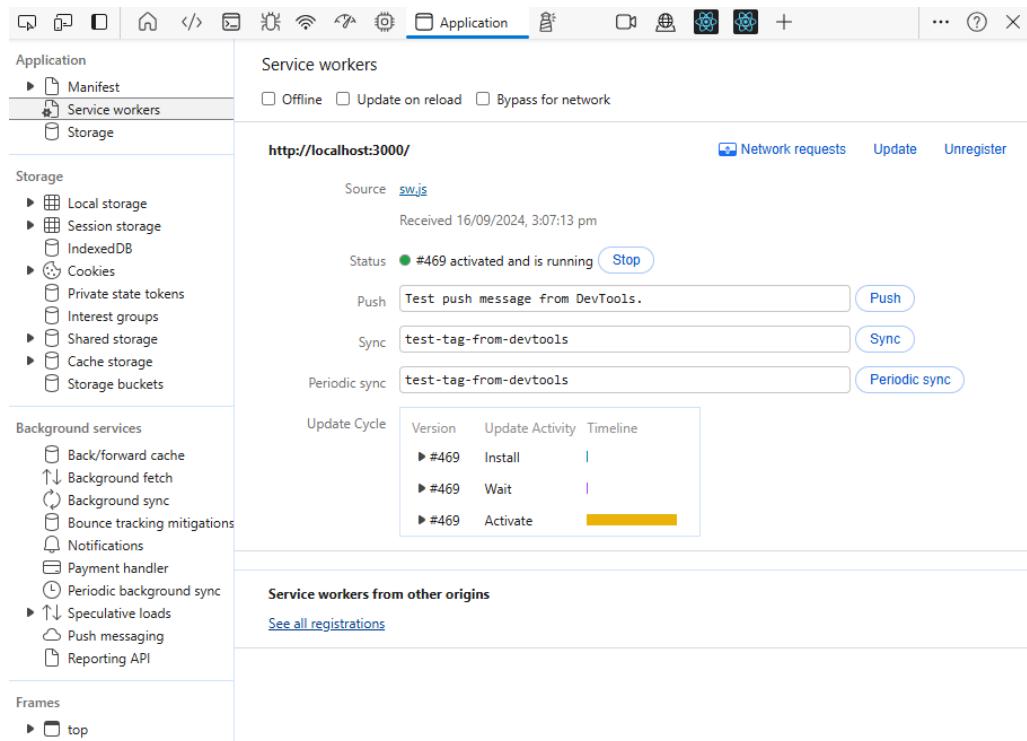
Continuando en el archivo index.html de la carpeta public, se tiene que registrar el uso del service worker en caso de que se detecte. Se puede saber que el service worker está activo verificando en la consola de desarrollador del navegador.

**Figura 74: Archivo index.html, registro del uso del service worker**

```
<script>
  if ('serviceWorker' in navigator) {
    window.addEventListener('load', function () {
      navigator.serviceWorker.register('/sw.js').then(function (registration) {
        console.log('SW registered: ', registration);
      }, function (err) {
        console.log('SW registration failed: ', err);
      });
    });
  }
</script>
```

*Fuente: Elaboración propia*

Figura 75: Consola de desarrollador, service worker funcionando



Fuente: *Elaboración propia*

El service worker nos ayuda en tareas como el caché de la página: si es que existen datos que no están disponibles, se descargan de la red, caso contrario se cargan del caché. También nos ayuda en la funcionalidad offline de la aplicación, ya que, si por alguna razón se llega a caer nuestra conexión a Internet, nos muestra un mensaje en el cual estamos sin conexión sin perder todo el contenido de la página web, lo cual permite utilizar la aplicación de manera offline. A continuación, se muestra la implementación y el uso del service worker.

Figura 76: Implementación del service worker

```
 1  /* eslint-disable no-restricted-globals */
 2  const CACHE_NAME = 'my-app-cache-v1';
 3  const urlsToCache = [
 4    '/',
 5    '/pagos'
 6  ];
 7
 8 // Instalar el Service Worker y cachear recursos
 9 self.addEventListener('install', event => {
10   event.waitUntil(
11     caches.open(CACHE_NAME)
12       .then(cache => {
13         console.log('Opened cache');
14         return cache.addAll(urlsToCache);
15       })
16     );
17   });
18
19 // Interceptar peticiones y servir recursos desde el cache
20 self.addEventListener('fetch', event => {
21   event.respondWith(
22     caches.match(event.request)
23       .then(response => {
24         if (response) {
25           return response; // Devuelve el recurso desde el caché
26         }
27         return fetch(event.request); // O continúa con la petición si no está en el caché
28       })
29     );
30   });
31
32 // Actualiza el caché cuando cambia el contenido de la app
33 self.addEventListener('activate', event => {
34   const cacheWhitelist = [CACHE_NAME];
35   event.waitUntil(
36     caches.keys().then(cacheNames => {
37       return Promise.all(
38         cacheNames.map(cacheName => {
39           if (!cacheWhitelist.includes(cacheName)) {
40             return caches.delete(cacheName);
41           }
42         })
43       );
44     })
45   );
46 });
47
48 self.addEventListener('fetch', event => {
49   event.respondWith(
50     caches.match(event.request)
51       .then(response => {
52         return response || fetch(event.request).then(fetchedResponse => {
53           return caches.open(CACHE_NAME).then(cache => {
54             cache.put(event.request, fetchedResponse.clone());
55             return fetchedResponse;
56           });
57         });
58       }).catch(() => {
59
60     })
61   );
62 });


```

Fuente: Elaboración propia

**Figura 77: Uso del service worker para la funcionalidad offline de la aplicación**

| ID | Nombre   | Apellido Paterno | Apellido Materno | Usuario     | Rol           | Estado | Restablecer Contraseña? | Acción                  |
|----|----------|------------------|------------------|-------------|---------------|--------|-------------------------|-------------------------|
| 26 | Ariel    | Tellez           | Torrico          | admin       | Administrador | Activo | No                      | <button>Editar</button> |
| 29 | Gary     | Vasquez          | Ribera           | vasquezg    | Medico        | Activo | No                      | <button>Editar</button> |
| 33 | Andres   | Guerrero         | Camacho          | druguerrero | Secretaria    | Activo | No                      | <button>Editar</button> |
| 35 | Patricia | Ortiz            | Romero           | portizr     | Secretaria    | Activo | No                      | <button>Editar</button> |
| 36 | Mariel   | Tellez           | Perez            | mtellezp    | Medico        | Activo | No                      | <button>Editar</button> |
| 38 | Gloria   | Gonzales         | Fernandez        | ggonzalesf  | Secretaria    | Activo | No                      | <button>Editar</button> |
| 39 | Juan     | Perez            | Suarez           | jperezs     | Medico        | Activo | No                      | <button>Editar</button> |
| 40 | Yoelma   | Melendres        | Flores           | ymelendresf | Secretaria    | Activo | No                      | <button>Editar</button> |
| 41 | Marco    | Condarco         | Mirabal          | mcondarcoc  | Medico        | Activo | Si                      | <button>Editar</button> |
| 42 | Risella  | Castro           | Ramos            | rcastro     | Administrador | Activo | Si                      | <button>Editar</button> |

*Fuente: Elaboración propia*

### **3.3. ANÁLISIS DE VIABILIDADAD**

En el siguiente objetivo se desarrollará la parte evaluativa del presente trabajo de grado, realizando un análisis de las características técnicas y económicas del proyecto.

### **3.3.1. Viabilidad Técnica**

En la validación técnica se especifica las características más importantes del proyecto lo cual tendrá por finalidad de dar a conocer la utilidad del software para el centro médico, las cuales son las siguientes:

#### **3.3.1.1. Usabilidad**

La usabilidad en un sistema se refiere a la facilidad con la que los usuarios pueden interactuar con un sistema o aplicación para cumplir sus objetivos. Se ha llegado a cumplir dicho criterio facilitando una interfaz intuitiva y fácil de utilizar tanto en computadora como en teléfonos inteligentes.

#### **3.3.1.2. Seguridad**

La seguridad de un sistema es fundamental para la protección de los datos usados en el sistema, para lo cual se implementaron múltiples medidas de seguridad, tales como utilizar un token de autenticación en el inicio de sesión del sistema que expira después de una hora, requiriendo volver a autenticarse al sistema; la protección de fuerza bruta implementando el bloqueo tras 3 intentos fallidos de inicio de sesión, la encriptación de las contraseñas para el guardado en la base de datos utilizando las técnicas de hashing y salting, cumpliendo con las buenas prácticas para una contraseña, y restringiendo las peticiones que se le pueden hacer al backend mediante CORS. (ver Anexo E).

#### **3.3.1.3. Funcionalidad**

La funcionalidad viene a ser la capacidad del software para cumplir con los requisitos específicos y realizar las tareas para las que fue diseñado de manera correcta y completa. En otras palabras, es el conjunto de características y capacidades que el software ofrece para resolver problemas o necesidades de los usuarios. El sistema propuesto permite registrar los datos de las atenciones médicas realizadas en los

consultorios, así como registrar los pagos realizados por parte de los asegurados, validando que no existan campos nulos con motivo de evitar errores (ver Anexo D). Con estos datos se obtienen reportes sobre la cantidad de asegurados atendidos en los consultorios de los colegios, la cantidad de pagos realizados y el monto generado; cuyo reporte se puede filtrar por un rango de fechas. Esto es de utilidad para la gerencia al momento de tomar decisiones en el momento oportuno.

### 3.3.2. Viabilidad Económica

En la viabilidad económica del proyecto se determina el coste total de implementación, lo cual determinará la viabilidad de costo del proyecto.

#### 3.3.2.1. Inversión fija

La inversión fija es la que involucra todos los activos fijos que forman parte de los módulos desarrollados, tales como dispositivos y componentes, los cuales son necesarios para la instalación y despliegue del sistema. A continuación, se detallan los componentes requeridos para el presente proyecto:

Tabla 18: Costos del hardware

| Ítem               | Descripción  | Cantidad | Costo    |              |
|--------------------|--|----------|----------|--------------|
|                    |  |          | Unitario | Total        |
| 1                  | Procesador Intel Core i5<br>Memoria RAM de 8GB<br>Disco duro de 500GB<br>Misceláneas restantes de la computadora | 1        | 3 700    | 3 700        |
| <b>Total (Bs.)</b> |  |          |          | <b>3 700</b> |

*Fuente: Elaboración propia*

Los componentes descritos previamente hacen referencia a la compra de una computadora básica con los requerimientos mínimos para el funcionamiento del

sistema. No obstante, no se está sumando el costo de las computadoras y dispositivos con los que ya cuenta el centro médico.

### 3.3.2.2. Inversión diferida

La inversión diferida se caracteriza por su intangibilidad y son derechos adquiridos y servicios necesarios para el estudio e implementación del proyecto. Esta inversión no está sujeta a desgaste físico, como las licencias y capacitación de personal. Se agregará un 5% al margen total en caso de algún imprevisto.

#### a) Licencias y servicios

En la siguiente tabla se presentarán las tecnologías usadas en el desarrollo del proyecto, la licencia y el precio de cada una.

**Tabla 19: Costo de licencias**

| Detalle            | Características   | Precio (Bs.) |
|--------------------|---|--------------|
| MySQL              | Gestor de base de datos de código abierto, con licencia GNU GPL.  | 0.00         |
| Express            | Framework para el desarrollo web enfocado en el lado del servidor basado en JavaScript, con licencia MIT. | 0.00         |
| React              | Framework para el desarrollo web enfocado en el lado del cliente basado en JavaScript, con licencia MIT.  | 0.00         |
| Microsoft Power BI | Herramienta de visualización de datos.  | 0.00         |
| <b>Total (Bs.)</b> |   | <b>0.00</b>  |

*Fuente: Elaboración propia*

El servicio en el cual se puede implementar el proyecto es un servicio de hosting web, es una alternativa ya que incluye todos los recursos necesarios para poder desplegar el proyecto, uno de los proveedores de este tipo de servicio es Hostinger. A continuación, se detallarán cada uno de sus planes con sus respectivos costos en la siguiente tabla:

**Tabla 20: Costos del servicio de hosting web**

|                                  | Alternativa 1   | Alternativa 2        | Alternativa 3        |
|----------------------------------|-----------------|----------------------|----------------------|
| Descripción                      | Premium         | Business             | Cloud Startup        |
| Dominio gratuito                 | 1er año         | 1er año              | 1er año              |
| Sitios web                       | 100             | 100                  | 300                  |
| Promedio de visitas mensuales    | 25 000          | 100 000              | 200 000              |
| Ancho de banda                   | Ilimitado       | Ilimitado            | Ilimitado            |
| Correo comercial                 | Gratis          | Gratis               | Gratis               |
| Almacenamiento                   | 100GB           | 200GB                | 200GB                |
| Certificado SSL                  | Ilimitado       | Ilimitado            | Ilimitado            |
| Copias de seguridad              | Semanales       | Diarias y a petición | Diarias y a petición |
| CDN (Content Delivery Network)   | No incluye      | Gratis               | Gratis               |
| <b>Precio/año (\$us.)</b>        | <b>143.88</b>   | <b>167.88</b>        | <b>299.88</b>        |
| <b>Total, precio/año (\$us.)</b> | <b>143.88</b>   | <b>167.88</b>        | <b>299.88</b>        |
| <b>Total, precio/año (Bs.)</b>   | <b>1 002.84</b> | <b>1 170.12</b>      | <b>2 090.16</b>      |

*Fuente: Hostinger*

### b) Costo de capacitación

Se determinó que para la capacitación se necesitarán dos días. En cuanto al coste se calcula un total de Bs. 880, siendo Bs. 55 el costo por hora de capacitación, ya que al día se capacitarán las 8 horas laborales.

**Tabla 21: Costo de capacitación**

| Tiempo (Horas)     | Costo unitario | Costo total   |
|--------------------|----------------|---------------|
| 16                 | 55             | 880.00        |
| <b>Total (Bs.)</b> |                | <b>880.00</b> |

*Fuente: Elaboración propia*

### c) Costo del desarrollo de software

Dado a que se está utilizando una metodología ágil de desarrollo, para el proceso de estimación de costo del desarrollo de software utilizaremos la estimación de costo mediante Story Points, en el cual se hace uso de las historias de usuario definidas en la etapa de los requerimientos de la aplicación, definiendo los puntos asignados a cada historia, se genera la siguiente tabla:

Tabla 22: Cálculo de Story Points

| Estimación del costo mediante Story Points |   |    |           |                                    |
|--|---|----|-----------|------------------------------------|
| Código                                     | Historia de usuario   | SP | Iteración | Duración de la iteración (semanas) |
| SA1  | Como usuario, quiero iniciar sesión en el sistema utilizando un nombre de usuario y contraseña para asegurar que solo el personal autorizado acceda a la información  | 6  | 1         | 3                                  |
| SA2  | Como administrador quiero tener un panel, en donde ver los usuarios registrados para tener un control sobre los usuarios actuales del sistema.  | 8  | 1         | 3                                  |
| SA3  | Como médico, quiero poder registrar la atención médica de paciente en el consultorio, para tener un registro digital y centralizado; y poder ver un historial de las atenciones médicas realizadas al paciente. | 8  | 1         | 3                                  |
| SA4  | Como secretaria, quiero tener un lugar en donde ver los pagos realizados, así como aprobar los pagos.   | 8  | 1         | 3                                  |
| SA5  | Como gerente, quiero tener acceso a información relevante de pagos y consultas, de forma rápida y simple para tomar decisiones.   | 8  | 1         | 3                                  |

|              |   |           |   |   |
|--------------|---|-----------|---|---|
| SA6          | Como cliente, quiero tener una página en donde pueda registrar el pago del seguro médico, para tener una constancia de que se canceló el monto establecido. | 8         | 1 | 3 |
| <b>TOTAL</b> |   | <b>46</b> |   |   |

*Fuente: Elaboración propia*

La fórmula para obtener el costo total estimado de la iteración es:

**Ecuación 2: Ecuación del costo estimado del total de las iteraciones**

$$\text{Costo estimado} = \text{Story Points} * \text{Coste de un Story Point} \quad (2)$$

*Fuente: (María et al., 2011)*

Los Story Points es el total obtenido de la tabla anterior: Story Points = 48.

El Coste de un Story Point es un valor estimado que reflejará el esfuerzo, la complejidad y el riesgo que tomará realizar cada tarea, en este caso le asignamos el siguiente valor al Coste de un Story Point = Bs320

Según la ecuación 2 se obtiene:

$$\text{Costo estimado} = 48 * 320$$

Resolviendo la operación se tiene que el costo estimado del desarrollo del proyecto es de:

$$\text{Costo estimado} = \text{Bs}15\,360$$

### 3.3.2.3. Inversión total del proyecto

Una vez determinados los distintos costos de inversión que representa el sistema, se procede al cálculo del total a invertir en el proyecto. Para esta etapa se consideran las inversiones diferidas e inversiones fijas anteriormente mencionadas. Con los criterios mencionados se obtienen los siguientes resultados:

**Ecuación 3: Ecuación de inversión total del proyecto**

$$IT = IF + ID \quad (3)$$

*Fuente: Elaboración propia*

**Tabla 23: Inversión total del proyecto**

| <b>Descripción</b>                              | <b>Precio (Bs.)</b> | <b>Total (Bs.)</b> |
|---|---------------------|--------------------|
| <b>Inversión fija</b>                           |                     |                    |
| Costo del hardware                              | 3 700.00            |                    |
| <b>Total, inversión fija</b>                    |                     | 3,700.00           |
| <b>Inversión diferida</b>                       |                     |                    |
| Costo del dominio/hosting                       | 1 002.84            |                    |
| Costo de capacitación                           | 880.00              |                    |
| Costo de desarrollo del software                | 15 360.00           |                    |
| <b>Total, inversión diferida</b>                |                     | 17 242.84          |
| Imprevisto (5%)                                 |                     | 862.14             |
| <b>Inversión total del proyecto (Bs.)</b>       |                     | <b>18 104.98</b>   |
| <b>Inversión total del proyecto (\$us 6,97)</b> |                     | <b>2 597.56</b>    |

*Fuente: Elaboración propia*

La inversión total del proyecto da un total de \$2597.56, lo cual resulta una inversión viable, ya que se cuenta con una aplicación web progresiva, el sistema del backend y la base de datos, incluyendo la ingeniería del proyecto aplicada durante el desarrollo del presente trabajo de grado.

**CAPÍTULO IV**  
**CONCLUSIONES Y**  
**RECOMENDACIONES**



## **CAPÍTULO 4.**

### **CONCLUSIONES Y RECOMENDACIONES**

#### **4.1. CONCLUSIONES**

En el presente trabajo de grado se desarrolló una aplicación web progresiva para automatizar el proceso de pago y atención médica de pacientes en los consultorios del Centro Médico SaludAME S.R.L, contribuyendo a la ineficiencia en los procesos mencionados.

Se diagnosticó el proceso actual de pagos del seguro y atenciones médicas en los consultorios, los cuales fueron obtenidos mediante la recolección de datos a través de entrevistas y la observación directa, identificando las partes de mayor relevancia las cuales nos permiten modelar los reportes generados en el proceso.

Se realizó la selección de las herramientas de software, hardware y la metodología de desarrollo de software, para lo cual se determinó utilizar React para el desarrollo de la página web, Express para la parte del Backend, MySQL como gestor de base de datos y Microsoft Power BI como herramienta de visualización de datos. De igual manera se realizó la evaluación del hardware mínimo para ejecutar el sistema en un ambiente de producción. La metodología seleccionada es Kanban.

Se implementaron los módulos que forman parte del sistema. De acuerdo con la metodología de desarrollo de software seleccionada, se obtuvieron las historias de usuario, se realizó el modelado e implementación de la base de datos y se fueron desarrollando los módulos de acuerdo con las iteraciones. Se integró los componentes del sistema utilizando una API REST comunicando el backend con el frontend validando la información obtenida mediante pruebas del sistema y se realizaron los reportes correspondientes en la herramienta de visualización de datos.

Se evaluó técnica y económicamente el proyecto con base en los requerimientos establecidos en la fase de desarrollo, se aplicó la estimación de costos por Story Points, obteniendo el coste aproximado del proyecto siendo viable realizar el proyecto económico, de igual manera se dio cumplimiento con los requerimientos técnicos del proyecto implementando la funcionalidad y seguridad en el sistema, concluyendo de esta manera que el proyecto es viable técnicamente.

#### **4.2. RECOMENDACIONES**

Durante el despliegue de la aplicación web, se recomienda utilizar servicios en la nube para asegurar la correcta escalabilidad y disponibilidad de la aplicación, esto se puede realizar utilizando servicios como Amazon Web Services o Microsoft Azure, lo que permite desplegar la aplicación en la nube.

Una vez desplegada la aplicación, se recomienda implementar el protocolo HTTPS usando el certificado SSL provisto por el servicio de hosting web, para garantizar la seguridad de la información enviada y el correcto funcionamiento de la aplicación.

Se recomienda implementar medidas de seguridad en los dispositivos en los cuales se utilizará la aplicación, tales como contar con el sistema y antivirus actualizados y utilizar un gestor de contraseñas.

Se recomienda implementar en una segunda fase de desarrollo implementar un módulo para gestionar el inventario de medicamentos en los consultorios y en el centro médico, y el envío de los recibos de pago mediante WhatsApp de forma automática.

## BIBLIOGRAFÍA



## BIBLIOGRAFÍA

- Alava, N. (2015, junio 1). *Diagramas de casos de uso*. <https://ingenieriaensoftwarenathalyalava.wordpress.com/2015/06/01/diagramas-de-casos-de-uso/>.
- Arimetrics. (2022, abril 12). *Qué es PHP*. <https://www.arimetrics.com/glosario-digital/php>.
- Arsys. (2018, junio 13). *¿Qué es PostgreSQL y por que llevarlo a Cloud?* <https://www.arsys.es/blog/postgresql-servidores#:~:text=PostgreSQL%20es%20un%20sistema%20de,consistente%20y%20tolerante%20a%20fallos>.
- Barragán, A. (2021, noviembre 26). *Qué es Vue JS y qué lo diferencia de otros frameworks*. <https://openwebinars.net/blog/que-es-vue-js-y-que-lo-diferencia-de-otros-frameworks/>.
- Bass, L., Clements, P., & Kazman, R. (2012). *Software Architecture in Practice: Software Architect Practice\_c3*. Pearson Education. <https://books.google.com.bo/books?id=-II73rBDXCYC>
- Berinato, S. (2016, junio). *Visualizations That Really Work*. <https://hbr.org/2016/06/visualizations-that-really-work>.
- BitDegree. (2024, enero 1). *Frameworks FrontEnd: ¿Cuál deberías elegir?* <https://es.bitdegree.org/tutoriales/frameworks-frontend>.
- Boardmix. (2024, febrero 1). *Guía completa sobre el diagrama de casos de uso*. <https://boardmix.com/es/articles/case-use-diagram/>.
- Bolufer, M. (2023, septiembre 17). *Análisis de las Ventajas y Desventajas de IntelliJ IDEA: Descubre por qué esta herramienta es una excelente opción para tus proyectos*. <https://ventajasydesventajastop.com/ventajas-y-desventajas-de-intellij-idea/>.

Bolufer, M. (2024a, enero 1). *Análisis de las Ventajas y Desventajas de JetBrains WebStorm: la herramienta indispensable para el desarrollo web.* <https://ventajasydesventajastop.com/jetbrains-webstorm-ventajas-y-desventajas/>.

Bolufer, M. (2024b, enero 18). *Análisis de las Ventajas y Desventajas de Microsoft Visual Studio: ¿Vale la pena utilizarlo?* <https://ventajasydesventajastop.com/microsoft-visual-studio-ventajas-y-desventajas/>.

Burin, A. (2023, abril 25). *Tipos de arquitecturas de software: Cuáles hay y en qué se diferencian.* <https://www.teclab.edu.ar/tipos-de-arquitecturas-de-software-cuales-hay-y-en-que-se-diferencian/>.

Bustos, J. L. (2023, octubre 31). *¿Qué es Spring Boot y cómo se desarrollan aplicaciones?* <https://keepcoding.io/blog/que-es-spring-boot/>.

Calvo, D. (2018, abril 7). *Metodología XP Programación Extrema (Metodología ágil).* <https://www.diegocalvo.es/metodologia-xp-programacion-extrema-metodologia-agil/>.

Cámara de Comercio de Oviedo. (2020, octubre 31). *Cómo determinar la viabilidad económica de un proyecto empresarial: aspectos clave.* <https://www.mba-asturias.com/empresas/viabilidad-economica-proyecto-empresarial/>.

Canle Fernández, E. (2022a, junio 6). *¿Qué es Django y para qué se utiliza?* <https://www.tokioschool.com/noticias/que-es-django/>.

Canle Fernández, E. (2022b, septiembre 22). *¿Qué es una base de datos relacional y cuáles son sus aplicaciones?* <https://www.tokioschool.com/noticias/base-datos-relacional/>.

Centellas Coarite, M. (2015). *Sistema de control y gestión de historiales clínicos apoyado en dispositivos móviles Caso: Centro Medico La Paz.*

Cervantes, H. (2014, julio 19). *Arquitectura de Software*.  
<https://sg.com.mx/revista/27/arquitectura-software>.

Cillero, M. (2023, agosto 2). *Diagrama de Clases*.  
<https://manuel.cillero.es/doc/metodologia/metrica-3/tecnicas/diagrama-de-clases/>.

Cloded Menendez, J. (2020, julio 21). *¿Qué es Power BI?*  
<https://www2.deloitte.com/es/es/pages/technology/articles/que-es-power-bi.html>.

Cloudflare. (2022, enero 15). *¿Qué significa lado del cliente y lado del servidor? | Lado del cliente vs. Lado del servidor*. <https://www.cloudflare.com/es-es/learning/serverless/glossary/client-side-vs-server-side/>.

codigocreativo. (2023, septiembre 24). *Diferencias entre framework front-end y back-end: lo que debes saber*. [https://codigocreativo.top/diferencias-entre-framework-front-end-y-back-end-lo-que-debes-saber/#%C2%BFQu%C3%A9\\_es\\_un\\_framework\\_back-end?](https://codigocreativo.top/diferencias-entre-framework-front-end-y-back-end-lo-que-debes-saber/#%C2%BFQu%C3%A9_es_un_framework_back-end?)

codigovanguardia. (2023, octubre 19). *Django: Ventajas y características que todo desarrollador debe conocer*. <https://codigovanguardia.com/desarrollo-web/django-ventajas-y-caracteristicas-que-todo-desarrollador-debe-conocer/>.

Coppola, M. (2023, enero 16). *Qué es Java, para qué sirve, características e historia*.  
<https://blog.hubspot.es/website/que-es-java>.

Cristancho, F. (2022, marzo 24). *Python: ventajas y desventajas*.  
<https://talently.tech/blog/python-ventajas-y-desventajas/>.

DataScientest. (2022, septiembre 2). *IDE: ¿Qué es un Entorno de Desarrollo Integrado?* <https://datascientest.com/es/ide-que-es>.

Díaz Sanjuán, L. (2010). *La observación* (Primera). • Facultad de Psicología de la Universidad Nacional Autónoma De México.

Evotic. (2023, septiembre 1). *¿Qué es Power BI? Ventajas y tipos de licencias.* <https://evotic.es/business-intelligence-bi/que-es-power-bi-ventajas-y-tipos-de-licencias/>.

flokzu. (2022, noviembre 29). *Qué es BPMN (Business Process Model and Notation).* <https://flokzu.com/es/bpm-es/new-que-es-bpmn/#FAQ>.

Flores, F. (2022, julio 22). *Qué es Visual Studio Code y qué ventajas ofrece.* <https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/>.

Fuentes Arce, E., Vázquez Bautista, M., Sánchez Ilsecas, D., Reyes Ventura, E., & Valentín Sán. (s/f). *Modelo COCOMO.* <https://www.scribd.com/presentation/454395628/Modelo-Cocomo>.

García Carmona, J. (2022, marzo 18). *Qué es ASP.NET y cuáles son sus puntos fuertes.* <https://openwebinars.net/blog/que-es-aspnet-y-cuales-son-sus-puntos-fuertes/>.

García de Zúñiga, F. (2020, mayo 27). *Qué es Vue: por qué usarlo como framework JS de referencia.* <https://www.arsys.es/blog/vuejs>.

Gonzalez Gil, J. (2018, agosto 17). *8 Características más importantes de PostgreSQL.* <https://openwebinars.net/blog/caracteristicas-importantes-de-postgresql/>.

Henry Tecnología S.A.S. (2021, agosto 3). *¿Cuáles son las etiquetas HTML más utilizadas y para qué sirven?* <https://blog.soyhenry.com/etiquetas-html-mas-utilizadas-y-para-que-sirven/>.

Hernández Sola, G. (2020, agosto 10). *¿Sabes qué son las historias de usuario?* <https://www.agile611.com/sabes-que-son-las-historias-de-usuario/?v=7b60a39fc2a4>.

Hernández, Y. (2023, enero 24). *¿Cuál es la arquitectura de una aplicación web?* <https://www.dongee.com/tutoriales/cual-es-la-arquitectura-de-una-aplicacion-web/>.

Hernández-Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2014). *Metodología de la Investigación* (Sexta). McGraw Hill.

hostingplus. (2021, octubre 1). *SQLite: ventajas y desventajas*. <https://www.hostingplus.pe/blog/sqlite-ventajas-y-desventajas/>.

IBM. (2023, marzo 27). *¿Qué es la visualización de datos?* <https://www.ibm.com/mx-es/topics/data-visualization#:~:text=La%20visualizaci%C3%B3n%20de%20datos%20es,una%20manera%20f%C3%A1cil%20de%20entender>.

Icaria Technology. (2023, abril 26). *La calidad del software y sus estándares más importantes*. <https://icariatechnology.com/la-calidad-del-software-y-sus-estandares-mas-importantes/>.

Ilerna. (2019, noviembre 12). *El modelo Entidad-Relación: el esquema de una base de datos*. <https://www.ilerna.es/blog/modelo-entidad-relacion-base-de-datos>.

Imagina. (2024, marzo 1). *¿Qué es Angular y cuáles son sus ventajas?* <https://imaginacion.com/tutoriales/conoce-las-ventajas-de-utilizar-angular>.

Instituto Tecnológico de Matehuala. (2013, noviembre 14). *2.3 Lenguajes de programación del lado del servidor*. <https://programacionwebisc.wordpress.com/2-3-lenguajes-de-programacion-del-lado-del-servidor/>.

IONOS. (2021, julio 12). *¿Qué es CSS? Definición y aplicación*. <https://www.ionos.es/digitalguide/paginas-web/diseño-web/que-es-css>.

Jesús. (2024, febrero 21). *Ventajas y Desventajas de Java Lenguaje de Programación*. <https://www.dongee.com/tutoriales/ventajas-y-desventajas-de-javascript-lenguaje-de-programacion/>.

JetBrains. (2024, abril 4). *IntelliJ IDEA overview*. <https://www.jetbrains.com/help/idea/discover-intellij-idea.html>.

Juice Studio. (2022, septiembre 16). *Arquitectura de aplicaciones web: La guía definitiva.* <https://juice-studio.com/arquitectura-de-aplicaciones-web-la-guia-definitiva/>.

Kinsta. (2019, diciembre 19). *¿Qué es Express.js? Todo lo que Debes Saber.* <https://kinsta.com/es/base-de-conocimiento/que-es-express/>.

Leroux, B. (2021, enero 18). *NODE.JS FRENTE A EXPRESS.JS: ¿CUÁL DEBERÍA UTILIZAR UNA EMPRESA DE DESARROLLO DE APLICACIONES WEB?* <https://www.startechup.com/es/blog/tospanish/>.

Lifeder. (2020, mayo 19). *Enfoque de la investigación: tipos y características.* <https://www.lifeder.com/enfoque-investigacion/>.

Lizama, O., Kindley, G., Morales, J. I. J., & Gonzales, A. (2016). Redes de computadores Arquitectura Cliente-Servidor. *Universidad Tecnica Federico Santa Maria*, 1–8.

Llamas, J. (2023, marzo 28). *Python.* <https://economipedia.com/definiciones/python.html>.

Lopez, A. M. (2024, enero 9). *Qué es ASP.NET Introducción, características y tutoriales.* <https://saberpunto.com/tecnologia/que-es-asp-net-introduccion-caracteristicas-y-tutoriales/>.

Lopez, M. (2023, junio 28). *¿Qué es un IDE? Características clave de los entornos integrados en el desarrollo web.* <https://immune.institute/blog/que-es-un-ide/>.

Lopez, M. (2024, enero 13). *MySQL: ¿qué es y cuáles son sus características?* <https://immune.institute/blog/mysql-que-es-y-sus-caracteristicas/>.

López Mendoza, M. (2020, julio 18). *Qué es un lenguaje de programación.* <https://openwebinars.net/blog/que-es-un-lenguaje-de-programacion/>.

López Mora, S. (2023, noviembre 28). *Desarrollo React: qué es, ventajas y por qué contratar a un programador React*. <https://digital55.com/blog/desarrollo-react-ventajas-contratar-programador-react/>.

Lucidchart. (2017, mayo 10). *Qué es un diagrama entidad-relación*. <https://www.lucidchart.com/pages/es/que-es-un-diagrama-entidad-relacion>.

Machuca, F. (2022, mayo 19). *Backend: ¿qué es y para qué sirve este tipo de programación?* <https://www.crehana.com/blog/transformacion-digital/que-es-el-backend-y-como-usarlo/>.

Maida, E. G., & Pacienza, J. (2015). *METODOLOGIAS DE DESARROLLO DE SOFTWARE*. Facultad de Química e Ingeniería “Fray Rogelio Bacon”. Universidad Católica Argentina.

Mancuzo, G. (2023, marzo 8). *¿Qué es Tableau? Principales funciones y características*. <https://blog.comparasoftware.com/que-es-tableau/>.

María, A., Solano, I., Simón, U., & Barranquilla -Atlántico, B. (2011). Story Points y su Importancia en la Estimación de Proyectos Bajo el Enfoque Ágil Importance of Story Points in the Estimation of Projects Under the Agile Approach. En *Investigación y Desarrollo en TIC*.

Martín, M. (2023, diciembre 30). *Requisitos funcionales versus no funcionales*. <https://www.guru99.com/es/functional-vs-non-functional-requirements.html>.

Martins, J. (2024, febrero 15). *Scrum: conceptos clave y cómo se aplica en la gestión de proyectos*. <https://asana.com/es/resources/what-is-scrum>.

Microsoft. (2018, junio 14). *¿Qué es la visualización de datos?* <https://powerbi.microsoft.com/es-es/data-visualization/>.

Microsoft. (2023, octubre 31). *¿Qué es Visual Studio?* <https://learn.microsoft.com/es-es/visualstudio/get-started/visual-studio-ide?view=vs-2022>.

Miro. (2023, septiembre 8). *Diagrama de clases UML*. <https://miro.com/es/diagrama/que-es-diagrama-clases-uml/>.

Moguel, E. A. R. (2005). *Metodología de la Investigación*. Univ. J. Autónoma de Tabasco.

Moreira, M. (2023, abril 1). *Introducción a la Programación Web: ¿Qué es y cómo funciona?* <https://univ3rsidades.com/introduccion-a-la-programacion-web-que-es-y-como-funciona/>.

Nazarevich, D. (2021, diciembre 20). *Ventajas e inconvenientes de Java*. <https://innowise.com/es/blog/benefits-and-drawbacks-of-java/>.

Nopal Consulting. (2023, abril 13). *¿Qué es la viabilidad técnica y por qué es importante?* <https://nopal.es/2023/04/13/que-es-la-viabilidad-tecnica/#:~:text=La%20viabilidad%20t%C3%A9cnica%20se%20refiere,est%C3%A1n%20disponibles%20y%20son%20factibles>.

Nużyńska, W. (2024, abril 5). *HTML*. <https://es.localo.com/marketing-dictionary/what-is-html>.

Paessler. (2022, marzo 4). *IT Explained: Bases de datos*. <https://www.paessler.com/es/it-explained/database>.

Pairazaman Esteves, L. A., & Vigo Escalante, E. A. (2017). *Sistema de información Web para el mejor control y acceso a las historias clínicas de los pacientes del centro de salud Jequetepeque*. <https://dspace.unitru.edu.pe/server/api/core/bitstreams/e510be73-024f-485a-8c67-dfac9ab8468b/content>

Palmero, E. (2021, julio 9). *Arquitectura de software: descubre en qué consiste y cuáles son las mejores prácticas*. <https://www.eykkon.com/blog/arquitectura-de-software/>.

Parra, A. (2020). *¿Qué es la recolección de datos y cómo realizarla?* <https://www.questionpro.com/blog/es/recolección-de-datos-para-investigacion/>.

- Peláez, A., Rodríguez, J., Ramírez, S., Pérez, L., Vázquez, A., & González, L. (2013). *La entrevista*. Universidad autónoma de México.[En línea].[Online].[cited 2012 Septiembre 30. Disponible en: [http://www.uam.es/personal\\_pdi/stmaria/jmurillo/InvestigacionEE/Presentaciones/Curso\\_10/E](http://www.uam.es/personal_pdi/stmaria/jmurillo/InvestigacionEE/Presentaciones/Curso_10/E).
- Pérez, A. (2021, abril 29). *La evaluación de un proyecto. Herramienta clave para evitar el fracaso*. <https://www.obsbusiness.school/blog/la-evaluacion-de-un-proyecto-herramienta-clave-para-evitar-el-fracaso>.
- Poveda Caputo, A. (2016, enero 18). *Modelamiento de Procesos con BPMN*. [https://planeacion.uniandes.edu.co/images/Arquitectura-Institucional/INS-45-1-01-01\\_Modelamiento\\_de\\_Procesos\\_con\\_BPMN1.pdf](https://planeacion.uniandes.edu.co/images/Arquitectura-Institucional/INS-45-1-01-01_Modelamiento_de_Procesos_con_BPMN1.pdf).
- Proscont. (2023, abril 13). *Ventajas y desventajas del c#*. <https://www.proscont.com/ventajas-y-desventajas-del-c/>.
- Pursell, S. (2024, marzo 28). *Análisis FODA de una empresa: qué es, cómo se hace y ejemplos*. <https://blog.hubspot.es/marketing/analisis-foda>.
- Raeburn, A. (2024, febrero 13). *La programación extrema (XP) produce resultados, pero ¿es la metodología adecuada para tí?* <https://asana.com/es/resources/extreme-programming-xp>.
- Ramos Duran, L. O. (2022). *Desarrollo de un sistema para el control y seguimiento del historial clínico de pacientes asegurados a la Corporación de Seguro Social Militar (COSSMIL) Regional Santa Cruz*.
- Robledano, A. (2019, septiembre 24). *Qué es MySQL: Características y ventajas*. <https://openwebinars.net/blog/que-es-mysql/>.
- Rodrigues, N. (2023, febrero 16). *Cómo realizar un análisis de costo-beneficio (con ejemplos)*. <https://blog.hubspot.es/sales/analisis-costo-beneficio>.
- Rodríguez, J. (2005, marzo 17). *Definición de JavaScript*. <https://www.gestiopolis.com/definicion-javascript/>.

Rodriguez, J. (2021, febrero 8). *Diferencias entre Bases de Datos relacionales y no relacionales.* <https://codenotch.com/blog/diferencias-entre-bases-de-datos-relacionales/#:~:text=Los%20principales%20sistemas%20gestores%20de%20bases%20de%20datos%20no,%3A%20MongoDB%2C%20Redis%20y%20Cassandra>

Rómmel, F. (2007, agosto). SQLite: La Base de Datos Embebida. <https://sg.com.mx/revista/17/sqlite-la-base-datos-embebida#:~:text=SQLite%20es%20una%20herramienta%20de,PDA%20o%20un%20tel%C3%A9fono%20celular.>, 12–13.

Rumbaugh, J., Jacobson, I., & Booch, G. (2007). *El Lenguaje Unificado de Modelado. Manual de Referencia.*

Salazar, C. (2021, noviembre 26). *¿Qué son las Historias de Usuario? | Cómo usarlas + Plantillas.* <https://www.iebschool.com/blog/agile-scrum/>.

Santander Universidades. (2020, diciembre 21). *Metodologías de desarrollo de software: ¿qué son?* <https://www.santanderopenacademy.com/es/blog/metodologias-desarrollo-software.html>.

Senn, J. A. (1992). *Análisis y diseño de sistemas de información.* McGraw-Hill. <https://books.google.com.bo/books?id=ZIuZQgAACAAJ>

Siderova, S. (2021, junio 21). *Método Kanban: Principios Y Ventajas.* <https://apiumhub.com/es/tech-blog-barcelona/metodo-kanban-ventajas/>.

Sinnaps. (2017, mayo 19). *SCRUM.* <https://www.sinnaps.com/blog-gestion-proyectos/metodologia-scrum>.

Sinnaps. (2020, mayo 27). *METODOLOGÍA XP O PROGRAMACIÓN EXTREMA.* <https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp>.

- Soberón, U. E. M., & Acosta, Z. (2009). Fuentes de información para la recolección de información cuantitativa y cualitativa. *Obtenido de <https://docs.bvsalud.org/biblioref/2018/06/885032/texto-no-2-fuentes-deinformacion.pdf>.*
- SOSSA, F. D. C. (2019). Aplicaciones web progresivas. *Ciencia Sur*, 5(6).
- SPNet. (2023, septiembre 7). *Tableau, una de las principales herramientas de Inteligencia de Negocios*. <https://softwarepara.net/tableau/>.
- Tandel, S., & Jamadar, A. (2018). Impact of progressive web apps on web app development. *International Journal of Innovative Research in Science, Engineering and Technology*, 7(9), 9439–9444.
- Tapia, N. (2018, noviembre 22). *Ventajas y desventajas del lenguaje PHP*. <https://baulphp.com/ventajas-y-desventajas-del-lenguaje-php/>.
- TRBL Services. (2021, octubre 20). *C# (C Sharp): Qué es, dónde se utiliza y para qué sirve*. <https://trbl-services.eu/blog-c-sharp-que-es-para-que-sirve/>.
- Universidad Francisco de Vitoria. (2023, junio 17). *BASES DE DATOS NO RELACIONALES*. <https://www.ufv.es/cetys/blog/bases-de-datos-no-relacionales/>.
- Universidad Internacional de La Rioja. (2021, octubre 27). *¿Qué es Power BI? Funciones y principales ventajas*. <https://www.unir.net/marketing-comunicacion/revista/power-bi/>.
- Vadavo. (2023, julio 6). *¿Qué es HTML y para qué sirve?* <https://www.vadavo.com/blog/html-que-es-y-para-que-sirve/>.
- Valenzuela, J. P. (2010, septiembre 20). *Análisis y Diseño Estructurado y Orientado a Objetos*. <https://www.geocities.ws/jpvalenzuelac/ads/t1.html>.
- Vidal, S. (2023, octubre 1). *¿Qué ventajas me ofrece WebStorm en el desarrollo web?* <https://tecnobits.com/en/que-ventajas-me-ofrece-webstorm-en-el-desarrollo-web/>.

Visure Solutions. (2022, julio 10). *Qué son los requisitos funcionales: ejemplos, definición, guía completa.* <https://visuresolutions.com/es/blog/requerimientos-funcionales/>.

W&B Asset Studio. (2023, septiembre 29). *Aseguramiento de la Calidad en el Software: ¿qué es y qué hace?* <https://www.wbassetstudio.com/blog/aseguramiento-de-la-calidad-en-el-software-que-es-y-que-hace/>.

WebDesing. (2022, octubre 29). *Ventajas y desventajas de Visual Studio Code 2023: ¿Es la herramienta adecuada para ti?* <https://webdesigncusco.com/ventajas-y-desventajas-de-visual-studio-code-2022/>.

Zottola, R. (2023, septiembre 14). *Arquitectura de Software. Breve introducción a Tipos y Significado.* <https://dr-zottola.medium.com/arquitectura-de-software-breve-introducci%C3%B3n-a-tipos-y-significado-144237217a36>.

## GLOSARIO



## GLOSARIO

- **API.** Interfaz de programación de aplicaciones, conjunto de definiciones y protocolos usados para diseñar e integrar el software de las aplicaciones.
- **BACK END.** Capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios, además contiene la lógica de la aplicación que maneja dichos datos.
- **FRONT END.** Tecnologías de diseño y desarrollo web que corren en el navegador y que se encargan de la interactividad con los usuarios.
- **HARDWARE.** Conjunto de componentes tangibles (o físicos) de una computadora.
- **HTTP.** Protocolo de transferencia de hipertexto, conjunto de reglas para la comunicación cliente-servidor.
- **MIDDLEWARE.** Software con el que diferentes aplicaciones pueden comunicarse entre sí.
- **MOCKUP.** También conocido como prototipo de alto nivel, es una representación visual de como quedara el sistema con tipografías y estilos.
- **REST.** Interfaz para conectar varios sistemas basados en el protocolo HTTP, sirve para obtener y generar datos, devolviendo los datos en diversos formatos.
- **SOFTWARE.** Conjunto de programas y datos con los que trabaja una computadora, el cual es inmaterial (o lógico).
- **WIREFRAME.** También conocido como prototipo de bajo nivel, es una representación visual de cómo quedará el sistema sin incluir tipografías, colores y estilos.

## ANEXOS



**ANEXO A. INFORMACIÓN  
REGISTRADA A MANO**



## **ANEXO “A”: INFORMACIÓN REGISTRADA A MANO**

**Figura 78: Hoja del registro de atenciones médicas en instituciones afiliadas al Centro Médico SaludAME S.R.L**

Fuente: Centro Médico SaludAME S.R.L.

**ANEXO B. INFORMACIÓN  
REGISTRADA EN HOJAS DE  
EXCEL**



## ANEXO “B”: INFORMACIÓN REGISTRADA EN HOJAS DE EXCEL

Figura 79: Hoja del registro de pacientes atendidos por diagnóstico en los consultorios habilitados de las instituciones afiliadas

|  <b>SaludAME</b><br>Asistencia médica de emergencia<br>Poco que te moleste y una mejor salud |   | <b>GOLDEN LION</b><br>YOSELYN ROJAS SALAS<br>2024 AGOSTO |
|---|---|--|
| <b>DIAGNOSTICOS</b>   | <b>Nº DE CASOS DEL MES ATENDIDOS EN CONSULTORIO</b> |  |
| Absceso   |   |  |
| Aftas   |   | 8  |
| Alergia   |   | 6  |
| Amigdalitis   |   | 29   |
| Ampolla   |   | 2  |
| Ansiedad  |   | 1  |
| Asma  |   | 10   |
| Cefalea   |   | 80   |
| Chikungunya   |   |  |
| Colocación de inyectable  |   |  |
| Conjuntivitis   |   | 20   |
| Constipación  |   | 3  |
| Consulta externa  |   | 3  |
| Contractura muscular  |   | 4  |
| Contusión   |   |  |
| Cuerpo extraño en boca  |   | 9  |
| Cuerpo extraño en oido  |   | 1  |
| Cuerpo extraño en ojo   |   | 6  |
| Cuerpo extraño en nariz   |   |  |
| Curación  |   | 28   |
| Dengue (Mayaro)   |   |  |
| Depresión   |   | 1  |
| Dermatitis  |   | 18   |
| Deshidratación  |   | 2  |
| Diarrea   |   | 21   |
| Dismenorrea (Menstruación)  |   | 18   |
| Disnea  |   | 13   |
| Dispepsia   |   | 7  |
| Dolor abdominal - cólico  |   | 56   |
| Epistaxis   |   | 15   |
| Escabiosis  |   | 1  |

|                                |            |
|--------------------------------|------------|
| <u>Lasceración</u>             |            |
| <u>Lumbalgia</u>               | <u>1</u>   |
| <u>Luxación</u>                |            |
| <u>Malestar general</u>        | <u>5</u>   |
| <u>Mareos</u>                  |            |
| <u>Mialgia</u>                 |            |
| <u>Nauseas</u>                 | <u>31</u>  |
| <u>Nebulización</u>            |            |
| <u>Nódulos (Gánglios)</u>      |            |
| <u>Odontálgia</u>              | <u>10</u>  |
| <u>Orzuelo</u>                 | <u>2</u>   |
| <u>Otalgia</u>                 | <u>9</u>   |
| <u>Papera</u>                  | <u>1</u>   |
| <u>Parasitosis</u>             | <u>12</u>  |
| <u>Picadura de insectos</u>    | <u>4</u>   |
| <u>Quemaduras</u>              |            |
| <u>Resfrio</u>                 | <u>28</u>  |
| <u>Retiro de puntos</u>        |            |
| <u>Sincope</u>                 |            |
| <u>Síndrome febril</u>         | <u>1</u>   |
| <u>Sinusitis</u>               | <u>4</u>   |
| <u>Taquicardia</u>             | <u>5</u>   |
| <u>TEC</u>                     |            |
| <u>Toma de PA</u>              | <u>1</u>   |
| <u>Toma de T</u>               | <u>22</u>  |
| <u>Tos</u>                     | <u>29</u>  |
| <u>Traumatismos</u>            |            |
| <u>Tunga penetrans (nigua)</u> |            |
| <u>Uñero</u>                   | <u>1</u>   |
| <u>Varicela</u>                |            |
| <u>Vómitos</u>                 | <u>13</u>  |
|                                |            |
|                                |            |
|                                |            |
| <b>TOTAL</b>                   | <b>686</b> |

*Fuente: Centro Médico SaludAME S.R.L.*

**Figura 80: Total de pacientes con seguro cancelado.**

| A    | B   | C                                 | D        | E                     | F       | G          | H             |       |
|------|---|-----------------------------------|----------|-----------------------|---------|------------|---------------|-------|
| 1    | TOTAL DE PACIENTES CON SEGURO PAGADO ENERO 2024 |                                   |          |                       |         |            |               |       |
| 2    | nro   | nombre                            | curso    | colegio               | gestion | fecha      | forma de pago | monto |
| 3    | 1   | PEINADO HERMOZA LEANDRO MANUEL    | P5B      | COLEGIO CRISTO REY    | 2024    | 02/01/2024 | TRANSFERENCIA | 100   |
| 4    | 2   | CONDORI CHOQUE JOSUE ALEJANDRO    | S5A      | COLEGIO CRISTO REY    | 2024    | 02/01/2024 | TRANSFERENCIA | 100   |
| 5    | 3   | JUSTINIANO ARTEAGA PAULA ANNETTE  | S3D      | COLEGIO CRISTO REY    | 2024    | 02/01/2024 | TRANSFERENCIA | 100   |
| 6    | 4   | APONTE FERNANDEZ PEDRO MATIAS     | S1D      | COLEGIO CRISTO REY    | 2023    | 02/01/2024 | TRANSFERENCIA | 100   |
| 7    | 5   | MELGAR JANCO ABIGAIL              | S1A      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 8    | 6   | MELGAR JANCO JOSUE HAMILTON       | S6A      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 9    | 7   | OCCA ROSADO KEREN NAOMI           | P2B      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 10   | 8   | MAMANI QUIROZ HEIDY GEORGINA      | I2B      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 11   | 9   | CACERES BANEGAS ISABELA           | P3B      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 12   | 10  | CACERES BANEGAS PAULA MILETH      | P5A      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 13   | 11  | MENESES SANTA CRUZ ARIANE TELMA   | P6B      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 14   | 12  | FLORES SANTA CRUZ LAURA BRIGITTE  | S3A      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 15   | 13  | RIBERA HUALPARA JOSE LUCAS        | P2A      | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 16   | 14  | RIBERA HUALPARA JOSE DANIEL       | NIDITO   | COLEGIO ARAKUARENDA   | 2024    | 02/01/2024 | TRANSFERENCIA | 150   |
| 17   | 15  | CORONADO TOLEDO ARATH BENJAMIN    | P1C      | COLEGIO CRISTO REY    | 2024    | 02/01/2024 | TRANSFERENCIA | 100   |
| 6829 | 6827  | SALMON MORALES ARIANE             | S6       | COLEGIO JESUS MAESTRO | 2024    | 09/04/2024 | EFFECTIVO     | 120   |
| 6830 | 6828  | TORRES UDAETA ALEJANDRA VALERIA   | S6       | COLEGIO JESUS MAESTRO | 2024    | 09/04/2024 | EFFECTIVO     | 120   |
| 6831 | 6829  | TORRICO TAPIA ANDRY               | S6       | COLEGIO JESUS MAESTRO | 2024    | 09/04/2024 | EFFECTIVO     | 120   |
| 6832 | 6830  | ESCALANTE BANZER LUIS ENRIQUE     | KINDER A | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6833 | 6831  | MENDOZA PARABA IKER YAMIR         | KINDER A | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6834 | 6832  | ESCALANTE BANZER BRIANNA VICTORIA | P3A      | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6835 | 6833  | PAZ SOTEO RENATA                  | P3A      | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6836 | 6834  | ESCALANTE BANZER JOSE RAUL        | P5A      | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6837 | 6835  | MENDOZA PARABA LANDER JOHAN       | P6A      | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6838 | 6836  | PEREDO JUSTINIANO ABRIL           | P6A      | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6839 | 6837  | ANTELO PEINADO JUAN FRANCISCO     | S2A      | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6840 | 6838  | PEÑA GUTIERREZ MARIA CELESTE      | S2A      | COLEGIO SEÑOR JESUS   | 2024    | 10/04/2024 | EFFECTIVO     | 200   |
| 6841 | 6839  | NUÑEZ VACA SANTIAGO               |          | COLEGIO JUAN PABLO II | 2024    | 10/04/2024 | TRANSFERENCIA | 100   |
| 6842 | 6840  | DIAZ VILLARRUEL OSCAR PABLO       | P6B      | COLEGIO CRISTO REY    | 2024    | 10/04/2024 | TRANSFERENCIA | 100   |
| 6843 | 6841  | LIZARDO CORTEZ VALENTINA          | P6B      | COLEGIO JUAN PABLO II | 2024    | 11/04/2024 | TRANSFERENCIA | 100   |
| 6844 | 6842  | LIZARRO CORTEZ VALENTINA          | P6B      | COLEGIO JUAN PABLO II | 2024    | 11/04/2024 | TRANSFERENCIA | 100   |
| 6845 | 6843  | GARECA COCA ALBERT AARON          | S4B      | COLEGIO JUAN PABLO II | 2024    | 12/04/2024 | TRANSFERENCIA | 100   |
| 6846 | 6844  | GARECA COCA YARA JACKELINE        | S1B      | COLEGIO JUAN PABLO II | 2024    | 12/04/2024 | TRANSFERENCIA | 100   |

Fuente: Centro Médico SaludAME S.R.L.

**ANEXO C. ENTREVISTA  
PRELIMINAR AL GERENTE  
GENERAL DEL CENTRO  
MÉDICO SALUDAME S.R.L.**



**ANEXO “C”: ENTREVISTA PRELIMINAR AL GERENTE GENERAL DEL CENTRO  
MÉDICO SALUDAME S.R.L.**

**Sección 1: Información de la empresa**

**1. A qué se dedica su empresa?**

R. La empresa se dedica a ofrecer servicios de salud integral a los pacientes, en forma de seguro contra accidentes en las instituciones.

**2. En qué sector opera su empresa?**

R. La empresa opera en el sector de seguros médicos contra accidentes y también en el sector de salud al operar en el centro médico.

**3. Cuál es la cantidad aproximada de personas que trabajan en la empresa?**

R. Actualmente son 6 personas en el centro médico.

**4. Cuáles son los principales procesos y flujos de trabajo de su empresa?**

R. El proceso de pagos y registro de pacientes en consultorios.

**5. Cuentan con alguna herramienta o recurso que facilite dichos procesos?**

R. No se cuenta con ninguna herramienta que nos facilite la tarea de registrar los pacientes en los consultorios ni en el registro de pagos.

**6. Cómo podrían mejorarse los flujos de trabajo y cuál sería el impacto esperado en términos de eficacia y calidad del trabajo?**

R. Se podría mejorar el flujo de trabajo implementando un sistema para poder registrar toda la información que actualmente se realiza de manera manual, de forma digital; para poder tener un control de los pacientes atendidos y evitar posibles pérdidas de información.

**Sección 2: Requerimientos del sistema**

**1. Cuál sería el presupuesto aproximado para el desarrollo del software?**

R. En lo posible, que se utilice software gratuito. Como presupuesto mensual para el mantenimiento unos 500 dólares mensuales.

**2. Que funcionalidades considera esenciales para el sistema?**

R. La administración y registro en el libro de observaciones para cada médico, la administración y registro de los pagos realizados, una automatización del proceso de pago por parte de los asegurados, un reporte en donde la gerencia pueda ver los pacientes atendidos de los distintos colegios y un reporte en donde se pueda visualizar la cantidad recaudada por colegios, por rango de fechas y por gestiones.

**3. Cómo describiría el volumen de datos que maneja la empresa diariamente?**

- Poco (menos de 10 registros)
- Moderado (entre 20 y 50 registros)
- Mucho (más de 50 registros)

R. En promedio se hacen entre 30 a 40 registros diariamente, incluyendo los datos del proceso de pago y registro de pacientes en los consultorios.

**4. Cuántos usuarios utilizarán el sistema simultáneamente?**

R. Se requieren unos 20 usuarios simultáneos, para que los médicos puedan registrar la información en tiempo real sin ningún tipo de colapso.

**5. Quiénes serán los principales usuarios del sistema?**

- (Por ejemplo, médicos, personal administrativo, gerentes)

R. Gerencia, secretaria, médicos de los consultorios.

**6. Cuál es el nivel de familiaridad con el uso de dispositivos electrónicos de los usuarios del sistema?**

- Bajo (necesitan capacitación extensa)

- Medio (necesitan alguna capacitación)
- Alto (pueden adaptarse rápidamente)

**R.** Bajo (necesitan capacitación extensa)

**7. Qué características de la interfaz de usuario son importantes para usted?**

- (Por ejemplo, diseño intuitivo, fácil navegación, acceso rápido a la información, personalización de vistas)

**R.** Un diseño intuitivo con acceso rápido a la información.

**8. Qué nivel de seguridad considera necesario para el acceso al sistema?**

- Básico (usuario y contraseña)
- Medio (contraseña robusta, recuperación de cuenta)
- Alto (autenticación de dos factores, cifrado de datos)

**R.** Un nivel básico, con usuario y contraseña

**9. Qué tipo de información considera sensible y debe ser protegida en el sistema?**

- (Por ejemplo, información de los estudiantes, información financiera, datos de clientes)

**R.** Los datos de los asegurados, el historial de los pagos realizados y los números telefónicos de los padres de familia.

**10. Cuántos registros de datos maneja en un día típico de trabajo?**

**R.** Diariamente se manejan aproximadamente de 10 a 15 inscripciones y unas 20 a 30 atenciones diarias repartidas entre todos los colegios.

**11. Con qué frecuencia necesita actualizar la información en el sistema?**

- Varias veces al día

- Diario

- Semanal

**R.** Diariamente

**13. Qué tan flexible necesita que sea el sistema para adaptarse a futuros crecimientos o cambios en la empresa?**

- (Muy flexible, moderadamente flexible, poco flexible)

**R.** Necesito que el sistema cuente con un moderado nivel de flexibilidad.

**14. Qué tipo de crecimiento espera en el volumen de datos o usuarios en los próximos 5 años?**

- Estable (sin mucho cambio)

- Moderado (crecimiento gradual)

- Alto (crecimiento rápido)

**R.** Un nivel moderado (crecimiento gradual).

**ANEXO D. ENTREVISTA AL  
GERENTE GENERAL DEL  
CENTRO MÉDICO  
SALUDAME S.R.L.**



**ANEXO “D”: ENTREVISTA PRELIMINAR AL GERENTE GENERAL DEL CENTRO  
MÉDICO SALUDAME S.R.L.**

**Sección 1: Información de la empresa**

**1. A qué se dedica su empresa?**

**R.** La empresa se dedica a ofrecer servicios de salud integral a los pacientes, en forma de seguro contra accidentes en las instituciones.

**2. En qué sector opera su empresa?**

**R.** Opera en el sector de seguros médicos contra accidentes, enfocado en colegios.

**3. Cuál es la cantidad aproximada de personas que trabajan en la empresa?**

**R.** Actualmente son 6 personas en el centro médico, las cuales se conforman por mi persona, la gerente administrativa, la auditora interna, la secretaria, un médico general y un traumatólogo. En los consultorios, se cuenta con 42 personas repartidas en turno mañana y tarde dentro de los 21 consultorios en los colegios.

**4. Cuáles son los principales procesos y flujos de trabajo de su empresa?**

**R.** Los dos principales procesos que se realizan son el proceso de pagos al seguro y el proceso para el registro de los pacientes atendidos. El proceso de pagos comienza con los clientes contactándose mediante el número de la empresa solicitando un QR para poder pagar el seguro médico, después de que envíen el comprobante de la transferencia se procede a emitir su recibo de forma manual y enviárselo al cliente. Por otra parte, el proceso de los pacientes atendidos en los consultorios es el siguiente: el paciente llega accidentado solicitando atención médica al consultorio, el médico de turno lo revisa y determina si es que le puede suministrar el tratamiento correspondiente o es necesario derivarlo al centro médico. Esos datos se los registra en unos libros denominados libros de observaciones, los cuales se entregan cada mes al centro médico para que se puedan sacar estadísticas de cuantos pacientes se han

atendido en el consultorio y cuantos fueron derivados al centro médico para poder determinar si es necesario aumentar o reducir el costo anual del seguro médico.

**5. Cuentan con alguna herramienta o recurso que facilite dichos procesos?**

**R.** No se cuenta con ninguna herramienta.

**6. Cómo podrían mejorarse los flujos de trabajo y cuál sería el impacto esperado en términos de eficacia y calidad del trabajo?**

**R.** Se podría mejorar el flujo de trabajo implementando un sistema para poder registrar toda la información que actualmente se realiza de manera manual.

**Sección 2: Requerimientos del sistema**

**1. Cuál sería el presupuesto aproximado para el desarrollo del software?**

**R.** En lo posible, que se utilice software gratuito. Como presupuesto mensual para el mantenimiento unos 2 000 dólares mensuales.

**2. Que funcionalidades considera esenciales para el sistema?**

**R.** La administración y registro en el libro de observaciones para cada médico, la administración y registro de los pagos realizados, una automatización del proceso de pago por parte de los asegurados, un reporte en donde la gerencia pueda ver los pacientes atendidos de los distintos colegios y un reporte en donde se pueda visualizar la cantidad recaudada por colegios, por rango de fechas y por gestiones.

**3. Cómo describiría el volumen de datos que maneja la empresa diariamente?**

- Poco (menos de 10 registros)
- Moderado (entre 20 y 50 registros)
- Mucho (más de 50 registros)

**R.** Moderado

**4. Cuántos usuarios utilizarán el sistema simultáneamente?**

**R.** Entre 15 a 20 usuarios.

**5. Quiénes serán los principales usuarios del sistema?**

- (Por ejemplo, médicos, personal administrativo, gerentes)

**R.** Gerencia, secretaria, médicos de los consultorios.

**6. Cuál es el nivel de familiaridad con el uso de dispositivos electrónicos de los usuarios del sistema?**

- Bajo (necesitan capacitación extensa)

- Medio (necesitan alguna capacitación)

- Alto (pueden adaptarse rápidamente)

**R.** Bajo

**7. Qué características de la interfaz de usuario son importantes para usted?**

- (Por ejemplo, diseño intuitivo, fácil navegación, acceso rápido a la información, personalización de vistas)

**R.** Un diseño intuitivo con acceso rápido a la información.

**8. Qué nivel de seguridad considera necesario para el acceso al sistema?**

- Básico (usuario y contraseña)

- Medio (contraseña robusta, recuperación de cuenta)

- Alto (autenticación de dos factores, cifrado de datos)

**R.** Básico.

**9. Qué tipo de información considera sensible y debe ser protegida en el sistema?**

- (Por ejemplo, información de los estudiantes, información financiera, datos de clientes)

**R.** Los datos de los asegurados y el historial de los pagos realizados.

**10. Cuántos registros de datos maneja en un día típico de trabajo?**

**R.** Diariamente se manejan aproximadamente de 10 a 15 inscripciones y unas 20 a 30 atenciones diarias repartidas entre todos los colegios. Cuando es la época de inscripción, se realizan aproximadamente 100 inscripciones diarias.

**11. Con qué frecuencia necesita actualizar la información en el sistema?**

- Varias veces al día
- Diario
- Semanal

**R.** Diariamente

**13. Qué tan flexible necesita que sea el sistema para adaptarse a futuros crecimientos o cambios en la empresa?**

- (Muy flexible, moderadamente flexible, poco flexible)

**R.** Moderadamente flexible.

**14. Qué tipo de crecimiento espera en el volumen de datos o usuarios en los próximos 5 años?**

- Estable (sin mucho cambio)
- Moderado (crecimiento gradual)
- Alto (crecimiento rápido)

**R.** Moderado.

**ANEXO E. SEGURIDAD Y  
USABILIDAD  
IMPLEMENTADA**



## ANEXO “E”: SEGURIDAD Y USABILIDAD IMPLEMENTADA

Figura 81: Validación de campos nulos en formulario

Agregar datos de consulta médica

Buscar Cliente y Agregar Registro

Carnet de Identidad

9667582

Buscar Cliente

Agregar datos para Ariel Tellez Torrico

27/09/2024

Diagnóstico

Tratamiento

Observaciones

Agregar Registro

Volver atrás

Por favor, completa todos los campos requeridos.

Fuente: Elaboración propia

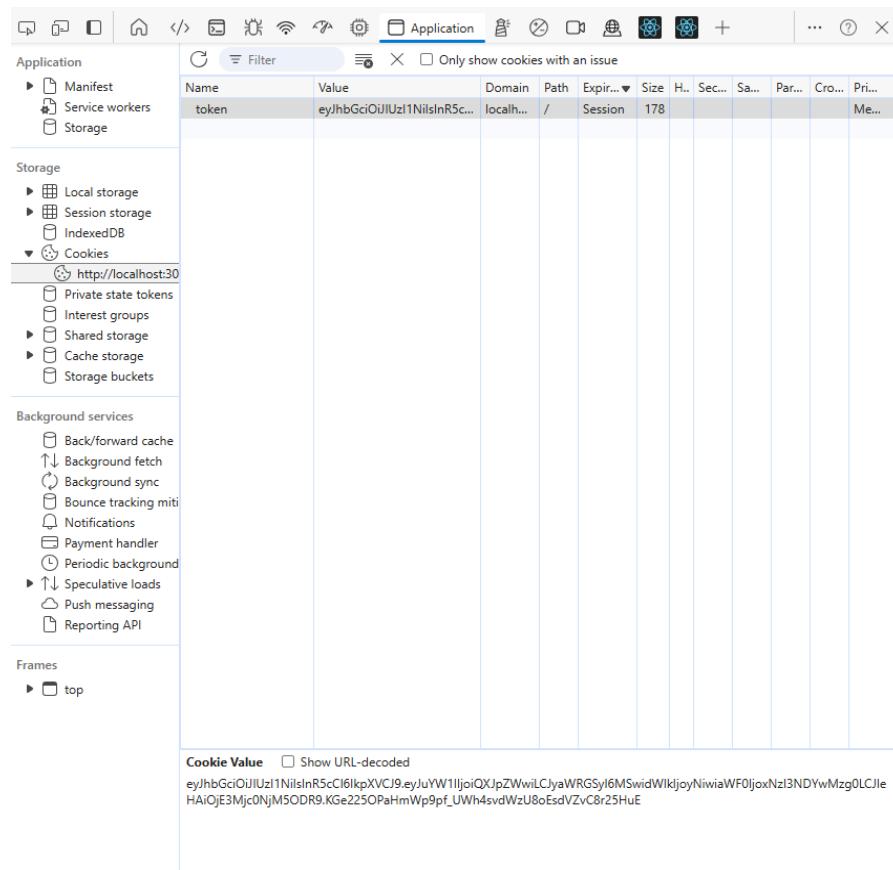
**Figura 82: Código que registra el token al momento de iniciar sesión**



```
121 const name = usuario.nombre;
122 const ridFK = usuario.ridFK;
123 const uid = usuario.uid;
124 const token = jwt.sign({ name, ridFK, uid }, "jwt-secret-key", { expiresIn: '1h' });
125
126 res.cookie('token', token);
127 return res.json({ Status: "Success" });
```

*Fuente: Elaboración propia*

**Figura 83: Token almacenado correctamente en las cookies del navegador**



*Fuente: Elaboración propia*

**Figura 84: Bloqueo de cuenta por muchos intentos fallidos**



*Fuente: Elaboración propia*

**Figura 85: Almacenamiento de contraseñas encriptadas en la base de datos**

|   | uid | nombre   | usuario     | contraseña                                     | estado | intentosFallidos | tiempoBloqueo       |
|---|-----|----------|-------------|--|--------|------------------|---------------------|
| ▶ | 26  | Ariel    | admin       | \$2b\$10\$jq2eLMOLv9yGKw0MqjnnPu66VKYejp8...   | 1      | 1                | NULL                |
|   | 29  | Gary     | vasquezg    | \$2b\$10\$FcDEjZMq7i9Mc2z4DjsbOa8KH.zYZcW...   | 1      | 0                | NULL                |
|   | 33  | Andres   | druguerrero | \$2b\$10\$6Uzy99TySSJfu/9XOIDnQOD6XAcnONr...   | 1      | 0                | NULL                |
|   | 35  | Patricia | portizr     | \$2b\$10\$aaZ20AMozlwCd3XixWYBPFODDkRQw/B...   | 1      | 0                | NULL                |
|   | 36  | Mariel   | mtellezp    | \$2b\$10\$pzptjycFCJK6zpEMgwutjulCMQ/x9dulm... | 1      | 0                | NULL                |
|   | 38  | Gloria   | ggonzalesf  | \$2b\$10\$IM7lhkRNawt0OVSCsimE9uA8dUwDW0...    | 1      | 0                | NULL                |
|   | 39  | Juan     | ioerezs     | \$2b\$10\$3zXY10aour1ZV/YiHGOuBe3hiIfrkHNwF... | 1      | 0                | 2024-09-27 14:32:16 |

*Fuente: Elaboración propia*

**Figura 86: Implementación de las buenas prácticas para una contraseña**



**Cambiar Contraseña**

.....

.....

La contraseña debe tener al menos 8 caracteres, un número y un símbolo especial.

**Cambiar Contraseña**

*Fuente: Elaboración propia*

**Figura 87: Restricción de los métodos accesibles a través de CORS**



The image shows a code editor window with a light gray background. At the top left, there are three colored circular icons: red, yellow, and green. Below them is a code snippet in a monospaced font:

```
16  app.use(cors({
17    origin: ["http://localhost:3000"],
18    methods: ["POST", "GET", "PUT", "DELETE"],
19    credentials: true
20  }));

```

*Fuente: Elaboración propia*