Logotipo

Descripción generada automáticamente

Universidad De Oriente

Facultad de Ingeniería en Telecomunicaciones Informática y Biomédica

Proyecto Practica Profesional

***Tema: Desarrollo de una Aplicación de Gestión para Embarazadas Integrada con Regresión Lineal y Red Neuronal***

***Autores:***

* ***Roseilis De la Cruz González***
* ***Yaisel Turcas Matos***

**Resumen:**

El proyecto presenta el diseño, desarrollo e implementación de una aplicación de gestión para embarazadas que utiliza modelos de machine learning para realizar predicciones y evaluaciones de riesgos médicos. La aplicación está diseñada con una interfaz gráfica en Python utilizando Tkinter y está integrada con una base de datos PostgreSQL para almacenar y gestionar los datos. Se exploran las funcionalidades de la aplicación, incluyendo la predicción de la fecha probable de parto, la evaluación de riesgos médicos y el cálculo del Índice de Masa Corporal (IMC). La aplicación también incorpora un sistema de autenticación de usuarios y una funcionalidad para detectar anemia, facilitando una gestión integral y precisa de los datos de las pacientes.

La metodología XP (Extreme Programming) es una metodología ágil de desarrollo de software que se centra en mejorar la calidad del software y la capacidad de respuesta a los cambios. En este informe, se describe cómo se aplicará XP para desarrollar un sistema híbrido de predicción de fechas de parto utilizando regresión lineal y redes neuronales .

**Palabras Claves:**

*Caluladora Ai , Inteligencia Artificial ,regresion Lineal , red Neuronal .*

**Abstract:**

The project presents the design, development, and implementation of a management application for pregnant women that uses machine learning models to make predictions and assess medical risks. The application is designed with a graphical interface in Python using Tkinter and is integrated with a PostgreSQL database for data storage and management. The application's functionalities include predicting the probable delivery date, assessing medical risks, and calculating the Body Mass Index (BMI). The application also incorporates a user authentication system and a feature to detect anemia, facilitating comprehensive and accurate management of patient data.

The XP (Extreme Programming) methodology is an agile software development approach that focuses on improving software quality and responsiveness to changes. This report describes how XP will be applied to develop a hybrid system for predicting delivery dates using linear regression and neural networks.

**Keywords:**

AI Calculator , Artificial Intelligence , Linear Regression, Neural Network.

**ÍNDICE:**

[Capítulo 1: 4](#_Toc177569018)

[1. Introducción: 4](#_Toc177569019)

[1.1 Antecedentes y Motivación 4](#_Toc177569020)

[1.2 Objetivos del Proyecto 6](#_Toc177569021)

[1.3 Estructura del Proyecto 7](#_Toc177569022)

[1.4 Revisión de Literatura 8](#_Toc177569023)

[1.5 Interfaz Gráfica en Python: Tkinter 8](#_Toc177569024)

[1.6 Modelos Predictivos en la Gestión de Salud Materna 9](#_Toc177569025)

[Capítulo 2: Metodología 11](#_Toc177569026)

[2. Ingeniería de Software con Extreme Programming (XP) 11](#_Toc177569027)

[2.1 Diagrama de Casos de Uso 11](#_Toc177569028)

[2.2 Arquitectura del Software 18](#_Toc177569029)

[2.3 Diseño del Sistema 20](#_Toc177569030)

[2.4 Implementación de Funcionalidades Adicionales 22](#_Toc177569031)

[2.5 Pruebas y Validación 23](#_Toc177569032)

[2.6 Implementación 24](#_Toc177569033)

[2.7 Desarrollo del Sistema de Usuarios y Autenticación 28](#_Toc177569034)

[2.8 Integración y Pruebas 29](#_Toc177569035)

[2.9 Resultados y Discusión 29](#_Toc177569036)

[3.1 Requisitos Funcionales y No Funcionales 32](#_Toc177569037)

[3.1.1 Artefacto: Historia de Usuario 35](#_Toc177569038)

[3.2 Tareas de Ingeniería 38](#_Toc177569039)

[3.3 Tarjetas CRC 45](#_Toc177569040)

[3.4 Casos de Prueba de Aceptación 46](#_Toc177569041)

[Conclusiones 50](#_Toc177569042)

[Recomendaciones: 52](#_Toc177569043)

[Bibliografía 54](#_Toc177569044)

[Artículos Académicos y Literatura Científica 54](#_Toc177569045)

[Libros 54](#_Toc177569046)

[Documentación Técnica y Recursos en Línea 54](#_Toc177569047)

[Recursos de Desarrollo de Software 55](#_Toc177569048)

[Apéndices A. 55](#_Toc177569049)

[B. Datos de Entrenamiento y Evaluación 56](#_Toc177569050)

[C. Manual de Usuario 56](#_Toc177569051)

# Capítulo 1:

## 1. Introducción:

### 1.1 Antecedentes y Motivación

La salud materna es un componente esencial en la atención médica, con un impacto directo en el bienestar de las madres y sus hijos. La gestión adecuada del embarazo requiere un enfoque integral y personalizado, que incluya la evaluación de múltiples factores médicos para garantizar una atención óptima. Entre los aspectos críticos a gestionar se encuentra la determinación precisa de la fecha probable de parto, la evaluación de riesgos asociados con antecedentes médicos y el monitoreo continuo de indicadores de salud como el Índice de Masa Corporal (IMC) y la presión arterial.

Históricamente, la evaluación de estos factores se ha llevado a cabo mediante consultas médicas tradicionales y análisis manuales de datos. Estos métodos, aunque efectivos en muchos casos, a menudo son laboriosos y propensos a errores. La naturaleza manual del proceso puede resultar en una administración ineficiente y en la posibilidad de omisiones importantes. Además, el tiempo requerido para realizar cálculos y análisis puede afectar la capacidad de los profesionales de la salud para responder rápidamente a las necesidades de las pacientes.

Con el avance de la tecnología, especialmente en los últimos años, la integración de técnicas avanzadas como el machine learning y el desarrollo de interfaces gráficas interactivas han comenzado a transformar significativamente la forma en que se gestionan y analizan los datos médicos. El machine learning, una subdisciplina de la inteligencia artificial, permite a los modelos aprender y hacer predicciones basadas en grandes conjuntos de datos. Esto posibilita una evaluación más rápida y precisa de los riesgos médicos y la predicción de eventos clínicos importantes, como la fecha probable de parto.

Las interfaces gráficas, como las desarrolladas con Python y Tkinter, proporcionan herramientas accesibles y eficientes para los profesionales de la salud. Estas interfaces permiten una interacción fluida con modelos predictivos y sistemas de gestión de datos, facilitando la entrada y visualización de información. La creación de una aplicación que integre estos avances tecnológicos puede revolucionar la forma en que los datos de salud se gestionan, ofreciendo una solución más eficiente y menos propensa a errores en comparación con los métodos tradicionales.

El objetivo principal de este proyecto es desarrollar una aplicación integral que combine machine learning y una interfaz gráfica interactiva para mejorar la gestión de la salud materna. Esta aplicación no solo calculará la fecha probable de parto con base en datos históricos y parámetros médicos, sino que también evaluará el riesgo de complicaciones en función de los antecedentes médicos de la paciente. Adicionalmente, se incluirán funcionalidades para calcular el IMC y detectar la hipertensión, proporcionando una evaluación completa del estado de salud de la embarazada.

Además, la aplicación incorporará una nueva funcionalidad para la detección de anemia, un factor crítico que puede influir en el bienestar tanto de la madre como del bebé. La inclusión de esta funcionalidad permitirá una gestión más integral y proactiva de la salud materna, abordando aspectos clave de manera sistemática y eficiente.

En resumen, este proyecto busca no solo automatizar y agilizar la gestión de datos médicos, sino también proporcionar una herramienta accesible y efectiva que mejore la capacidad de los profesionales de la salud para ofrecer una atención personalizada y basada en datos precisos. La combinación de machine learning con interfaces gráficas avanzadas representa un avance significativo en la atención médica, con el potencial de transformar la forma en que se maneja la salud materna.

**Problemática: Contexto**

La predicción precisa de las fechas de parto es crucial para la planificación y gestión de la atención prenatal. Sin embargo, los métodos tradicionales basados en cálculos de la última menstruación y ecografías tempranas presentan limitaciones significativas en términos de precisión. Estas imprecisiones pueden llevar a una planificación inadecuada, tanto para las madres como para los profesionales de la salud, y pueden aumentar el riesgo de complicaciones durante el embarazo y el parto.

Objetivo: Desarrollar una aplicación que utilice técnicas avanzadas de predicción para calcular con mayor precisión las fechas de parto, mejorando así la planificación y gestión de la atención prenatal.

Funcionalidades: Registro de Datos: Permitir a las usuarias ingresar datos relevantes como la fecha de la última menstruación, resultados de ecografías, y otros datos médicos.

Predicción de Fecha de Parto: Utilizar algoritmos de machine learning para predecir la fecha de parto con mayor precisión.

Alertas y Recordatorios: Enviar notificaciones sobre citas médicas, pruebas prenatales y otros eventos importantes.

Historial Médico: Mantener un registro detallado del historial médico de la usuaria.

Interfaz Amigable: Diseñar una interfaz intuitiva y fácil de usar.

Historias Técnicas

Como usuaria, quiero ingresar mis datos médicos para recibir una predicción precisa de mi fecha de parto.

Como profesional de la salud, quiero acceder al historial médico de mis pacientes para planificar mejor su atención prenatal.

Como usuaria, quiero recibir alertas y recordatorios sobre mis citas médicas y pruebas prenatales.

### 1.2 Objetivos del Proyecto

Los objetivos del proyecto son los siguientes:

1. **Desarrollar una Aplicación de Gestión para Embarazadas:**
   * Crear una aplicación interactiva utilizando Python y Tkinter que permita a los profesionales de la salud ingresar y gestionar datos relacionados con el embarazo.
   * Integrar la aplicación con una base de datos PostgreSQL para almacenar y recuperar información de manera eficiente.
2. **Implementar Modelos Predictivos:**
   * Utilizar machine learning para entrenar un modelo predictivo que calcule la fecha probable de parto y evalúe el riesgo de complicaciones basado en datos históricos.
   * Implementar una red neuronal para mejorar la precisión de las predicciones y evaluaciones de riesgo.
3. **Desarrollar Funcionalidades de Evaluación Médica:**
   * Incluir funcionalidades para calcular el IMC y evaluar la hipertensión.
   * Implementar una funcionalidad para detectar anemia basada en parámetros médicos relevantes.
4. **Diseñar una Interfaz Gráfica Intuitiva:**
   * Crear una interfaz gráfica de usuario (GUI) intuitiva y fácil de usar que permita a los usuarios interactuar con el sistema de manera eficiente.
   * Incorporar un sistema de autenticación de usuarios y opciones de administración como la creación de usuarios y el cierre de sesión.
5. **Validar y Evaluar el Sistema:**
   * Realizar pruebas exhaustivas del sistema para asegurar su precisión y fiabilidad.
   * Obtener retroalimentación de los usuarios para identificar áreas de mejora y optimizar la funcionalidad de la aplicación.

### 1.3 Estructura del Proyecto

Está organizado en los siguientes capítulos para proporcionar una presentación estructurada y comprensible del trabajo realizado:

1. **Capítulo 1: Introducción**
   * Este capítulo introduce el tema del proyecto , presenta la motivación detrás del desarrollo de la aplicación, y define los objetivos que se buscan alcanzar. También proporciona una visión general de la estructura del proyecto.

**1.1 Revisión de Literatura**

* + Se revisa la literatura relevante sobre el uso de machine learning en la medicina, el desarrollo de interfaces gráficas en Python, y la gestión de datos con bases de datos relacionales. Se exploran también los modelos predictivos en el contexto de la salud materna.

**Capítulo 2: Metodología**

* + Se describe la metodología utilizada para el desarrollo de la aplicación, incluyendo el diseño del sistema, la implementación de los modelos predictivos, el desarrollo de la interfaz gráfica, y las funcionalidades adicionales como la detección de anemia. También se detallan las pruebas realizadas y la validación del sistema.

**2.1 Implementación**

* + Se presenta la implementación práctica de la aplicación, incluyendo el código fuente y la explicación de cada parte del código. Se aborda el desarrollo del sistema de usuarios y autenticación, así como la integración y pruebas realizadas.

**2.2 Resultados y Discusión**

* + Se presentan los resultados obtenidos durante el desarrollo y las pruebas de la aplicación. Se discute el desempeño del modelo predictivo, la usabilidad de la interfaz gráfica, y la efectividad de las funcionalidades adicionales. Se incluye también la retroalimentación de los usuarios y las mejoras sugeridas.

**2.3 Conclusiones y Trabajo Futuro**

* + Se resumen las conclusiones del estudio, se destacan las contribuciones de la tesis, se reconocen las limitaciones del trabajo, y se proponen recomendaciones para investigaciones y desarrollos futuros.

**Bibliografía**

* + Se incluye una lista de todas las fuentes y referencias utilizadas en el proyecto.

**Apéndices**

Se presentan materiales adicionales como el código fuente completo, los datos utilizados para el entrenamiento y evaluación del modelo, y un manual de usuario para la aplicación.

### 1.4 Revisión de Literatura

#### 1.4.1 Aplicaciones de Machine Learning en Medicina

El campo de la medicina se ha beneficiado enormemente de los avances en Machine Learning (ML), transformando significativamente la manera en que se diagnostican, tratan y previenen enfermedades. Las aplicaciones de ML en medicina abarcan varias áreas clave que son esenciales para la mejora de la atención al paciente y la eficiencia del sistema de salud.

**Diagnóstico y Clasificación:** Uno de los avances más notables es la capacidad de ML para mejorar el diagnóstico de enfermedades a través del análisis de imágenes médicas y datos clínicos. Las redes neuronales convolucionales (CNN) se han convertido en una herramienta fundamental en la detección de cáncer y otras enfermedades a partir de imágenes radiológicas. Por ejemplo, un estudio realizado por Esteva et al. (2017) demostró que los modelos de CNN pueden igualar o incluso superar la precisión de los radiólogos en la detección de cáncer de piel. Esta capacidad para realizar diagnósticos precisos y rápidos es crucial para la detección temprana y el tratamiento oportuno de enfermedades, lo que puede mejorar significativamente los resultados para los pacientes.

**Predicción de Riesgos:** Los modelos predictivos basados en ML tienen el potencial de anticipar el riesgo de enfermedades y complicaciones antes de que se manifiesten clínicamente. Utilizando datos históricos y características personales del paciente, los algoritmos de regresión y clasificación pueden prever el riesgo de enfermedades como diabetes, enfermedades cardiovasculares y complicaciones durante el embarazo. Bertolini et al. (2017) destacaron cómo los modelos predictivos pueden integrar múltiples factores de riesgo para ofrecer evaluaciones más precisas y personalizadas, lo que permite intervenciones preventivas y un manejo más eficaz de las condiciones crónicas.

**Optimización de Tratamientos:** ML no solo ayuda en el diagnóstico y la predicción, sino que también juega un papel crucial en la personalización de tratamientos. Algoritmos de aprendizaje automático pueden analizar datos sobre la respuesta a tratamientos anteriores para recomendar terapias específicas para pacientes individuales. Kourou et al. (2015) demostraron que el uso de modelos predictivos para personalizar el tratamiento del cáncer puede mejorar la eficacia del tratamiento y reducir los efectos secundarios, lo que subraya la importancia de la medicina personalizada en la atención médica moderna.

### 1.5 Interfaz Gráfica en Python: Tkinter

Tkinter, la biblioteca estándar para la creación de interfaces gráficas en Python, proporciona una herramienta poderosa para el desarrollo de aplicaciones interactivas. Su popularidad en el desarrollo de software se debe a su simplicidad y flexibilidad, que permiten a los desarrolladores crear interfaces de usuario intuitivas y efectivas.

**Características y Ventajas de Tkinter:** Tkinter permite la creación de aplicaciones gráficas utilizando un conjunto de widgets básicos, como botones, menús, cuadros de texto y etiquetas. La biblioteca se integra bien con el lenguaje Python, facilitando el desarrollo de aplicaciones que requieren una interfaz gráfica sin necesidad de conocimientos profundos en diseño de interfaces (Müller et al., 2018). La facilidad de uso de Tkinter es una ventaja significativa en el desarrollo de aplicaciones médicas, donde la accesibilidad y la claridad de la interfaz son cruciales para la interacción del usuario.

**Aplicaciones en la Salud:** En aplicaciones médicas, Tkinter se utiliza para crear interfaces que permiten a los profesionales de la salud ingresar, visualizar y manipular datos clínicos de manera eficiente. La capacidad de Tkinter para generar interfaces gráficas simples y funcionales es esencial para aplicaciones que requieren la presentación clara de información, como la gestión de datos de pacientes y la interacción con modelos predictivos (O’Reilly, 2015). Esta facilidad de uso es fundamental para asegurar que los profesionales de la salud puedan centrarse en la interpretación de los datos y la toma de decisiones, en lugar de en la complejidad del software.

#### 1.5.1 Bases de Datos Relacionales y PostgreSQL

Las bases de datos relacionales son la piedra angular de la gestión de datos en muchas aplicaciones, incluida la medicina. La capacidad de organizar, consultar y manipular datos estructurados de manera eficiente es esencial para la gestión de grandes volúmenes de información clínica.

**Características de PostgreSQL:** PostgreSQL es un sistema de gestión de bases de datos relacionales conocido por su robustez, flexibilidad y conformidad con el estándar SQL. Su capacidad para manejar grandes conjuntos de datos y su soporte para transacciones ACID aseguran la integridad y la consistencia de los datos (Momjian, 2007). Además, PostgreSQL ofrece características avanzadas como el soporte para tipos de datos complejos y la posibilidad de definir procedimientos almacenados, lo que permite una gestión eficiente y sofisticada de la información.

**Aplicaciones en la Gestión de Salud:** En el ámbito de la salud, PostgreSQL se utiliza para almacenar y gestionar datos de pacientes, resultados de pruebas y registros médicos. Su capacidad para manejar relaciones complejas entre diferentes tipos de datos y su compatibilidad con herramientas analíticas lo hacen adecuado para aplicaciones que requieren un análisis exhaustivo de la información clínica (Stonebraker et al., 2007). La robustez y flexibilidad de PostgreSQL son fundamentales para el desarrollo de aplicaciones médicas que requieren una gestión efectiva de grandes volúmenes de datos y una integración con modelos predictivos y sistemas de análisis.

### 1.6 Modelos Predictivos en la Gestión de Salud Materna

La gestión de la salud materna se beneficia enormemente de los modelos predictivos, que permiten anticipar eventos clínicos y evaluar riesgos basados en datos de pacientes. Estos modelos integran información clínica y personal para proporcionar predicciones precisas que pueden mejorar la atención prenatal y postnatal.

**Predicción de Fecha de Parto:** La predicción precisa de la fecha de parto es crucial para la planificación y el monitoreo del embarazo. Los modelos predictivos que utilizan datos como la fecha de última menstruación y la duración del ciclo menstrual pueden prever la fecha probable de parto con alta precisión. Liu et al. (2016) demostraron que la integración de estos datos en modelos predictivos mejora la exactitud de la predicción, lo que permite una mejor planificación del cuidado prenatal y la preparación para el parto.

**Evaluación de Riesgos:** La identificación temprana de riesgos asociados con el embarazo, como hipertensión, diabetes gestacional y anemia, es fundamental para la prevención de complicaciones. Los modelos predictivos pueden analizar datos sobre antecedentes médicos y condiciones actuales para identificar a las pacientes en riesgo y recomendar intervenciones preventivas. Nielsen et al. (2018) destacaron cómo estos modelos pueden mejorar la gestión de la salud materna al proporcionar alertas tempranas sobre posibles complicaciones y permitir una intervención oportuna.

**Monitoreo de Indicadores de Salud:** Los modelos predictivos también se utilizan para monitorear indicadores de salud como el Índice de Masa Corporal (IMC) y la presión arterial. Estos modelos pueden detectar desviaciones de los valores normales y proporcionar alertas sobre posibles problemas de salud. Gonzalez et al. (2017) demostraron que la integración de modelos predictivos en la gestión de salud materna ayuda a mejorar el monitoreo continuo y la intervención temprana, lo que contribuye a una atención más efectiva y personalizada.

# Capítulo 2: Metodología

Este capítulo aborda en detalle la metodología empleada en el desarrollo de una aplicación integral para la gestión de la salud materna, que incorpora tecnologías avanzadas como el machine learning, interfaces gráficas interactivas y bases de datos relacionales. La metodología descrita a continuación está diseñada para superar los desafíos asociados con el manejo y análisis de datos médicos y garantizar que la aplicación sea tanto precisa como funcional, y accesible para los profesionales de la salud.

### Ingeniería de Software con Extreme Programming (XP)

Extreme Programming (XP) es una metodología ágil que enfatiza la calidad del software y la capacidad de respuesta a los cambios mediante un enfoque iterativo y colaborativo. XP se basa en un conjunto de prácticas que buscan mejorar el desarrollo de software, garantizar que los requisitos del cliente se cumplan de manera eficiente y permitir la adaptación continua a los cambios. En el contexto de esta tesis, XP se ha aplicado para guiar el desarrollo de la aplicación que combina machine learning, interfaces gráficas interactivas y bases de datos para la gestión de la salud materna. A continuación, se detallan las prácticas de XP, las tecnologías utilizadas y cómo se han integrado en el proyecto.

### 2.1 Diagrama de Casos de Uso

**Descripción**

El sistema de gestión de embarazadas permite a los usuarios (principalmente personal médico o administrativo) realizar diversas operaciones relacionadas con la salud y el seguimiento de pacientes embarazadas. A continuación, se describen los principales casos de uso que el sistema soporta.

**Actores Principales**

Usuario: Persona que interactúa con el sistema (por ejemplo, médico, enfermero, administrador).

**Casos de Uso Principales**

* **Calcular Fecha Probable de Parto:** Permite al usuario ingresar la fecha de la última menstruación y la duración del periodo para calcular la fecha probable de parto.
* **Calcular Riesgo**: Evalúa el riesgo del paciente basado en antecedentes médicos y presiones arteriales.
* **Calcular IMC**: Calcula el Índice de Masa Corporal de la paciente basado en su peso y altura.
* **Calcular Hipertensión**: Determina si la paciente presenta hipertensión basada en sus presiones arteriales.
* **Calcular Anemia**: Evalúa si la paciente tiene anemia basada en el resultado de hemoglobina.

#### 2.1.1 Prácticas de Extreme Programming (XP)

**Desarrollo Basado en Pruebas (Test-Driven Development, TDD)**

**Principio:** El desarrollo basado en pruebas es una práctica fundamental en XP que implica escribir pruebas antes de implementar el código funcional. Esto asegura que el software cumpla con los requisitos especificados y facilita la identificación temprana de errores.

**Aplicación en el Proyecto:**

* **Lenguaje:** Python se ha utilizado para escribir tests unitarios, aprovechando frameworks como unittest y pytest. Estos frameworks permiten estructurar y ejecutar pruebas de manera eficiente.
* **Justificación:** Python es elegido por su simplicidad y la robustez de sus herramientas de prueba, lo que facilita un desarrollo ágil y eficiente. La práctica de TDD ayuda a mantener el código confiable y a detectar errores en etapas tempranas del desarrollo.

import unittest

from datetime import datetime

from my\_app import calculate\_due\_date

class TestDueDateCalculation(unittest.TestCase):

def test\_due\_date(self):

last\_menstruation = datetime(2023, 1, 1)

expected\_due\_date = datetime(2023, 10, 8)

self.assertEqual(calculate\_due\_date(last\_menstruation), expected\_due\_date)

#### 2.1.2 Refactorización

**Principio:** La refactorización continua mejora la estructura del código sin alterar su funcionalidad. Esta práctica ayuda a mantener el código limpio, comprensible y manejable, permitiendo ajustes y mejoras constantes.

**Aplicación en el Proyecto:**

* **Lenguaje:** Python se utiliza para refactorizar el código, mejorando su legibilidad y eficiencia.
* **Justificación:** La sintaxis clara de Python facilita la refactorización, permitiendo reorganizar el código y aplicar patrones de diseño que optimizan su estructura sin modificar su comportamiento.

python

# Antes de la refactorización

def get\_patient\_data(patient\_id):

connection = psycopg2.connect(...)

cursor = connection.cursor()

cursor.execute("SELECT \* FROM patients WHERE id = %s", (patient\_id,))

data = cursor.fetchone()

connection.close()

return data

# Después de la refactorización usando el patrón de repositorio

class PatientRepository:

def \_\_init\_\_(self, connection):

self.connection = connection

def get\_patient\_by\_id(self, patient\_id):

cursor = self.connection.cursor()

cursor.execute("SELECT \* FROM patients WHERE id = %s", (patient\_id,))

return cursor.fetchone()

#### 2.1.3 Programación en Parejas (Pair Programming)

**Principio:** La programación en parejas consiste en que dos desarrolladores trabajan juntos en el mismo código, promoviendo la revisión continua, el intercambio de conocimientos y la mejora de la calidad del código.

**Aplicación en el Proyecto:**

* **Justificación:** Esta práctica se utiliza durante el desarrollo para revisar el código en tiempo real, lo que ayuda a identificar errores y mejora la calidad del código. Además, fomenta la colaboración y el aprendizaje mutuo entre los miembros del equipo.

#### 2.1.4 Integración Continua

**Principio:** La integración continua implica integrar el código en el repositorio frecuentemente y ejecutar pruebas automáticamente para detectar problemas de integración y garantizar la calidad del software.

**Aplicación en el Proyecto:**

* **Herramientas:** Se emplean herramientas como Jenkins, GitHub Actions o Travis CI para automatizar la construcción, integración y pruebas del código.
* **Justificación:** Estas herramientas permiten automatizar el proceso de integración y pruebas, garantizando que cada cambio en el código se verifique rápidamente y se mantenga la calidad del software en cada iteración.

# Ejemplo de configuración para GitHub Actions

name: CI Pipeline

on: [push, pull\_request]

jobs:

build:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v2

- name: Set up Python

uses: actions/setup-python@v2

with:

python-version: 3.8

- name: Install dependencies

run: |

pip install -r requirements.txt

- name: Run tests

run: |

pytest

#### 2.1.5 Simplicidad

**Principio:** La simplicidad en el diseño y en el código busca hacer las soluciones lo más simples posible, facilitando su comprensión y mantenimiento.

**Aplicación en el Proyecto:**

* **Lenguaje:** Python
* **Justificación:** La simplicidad del diseño y la implementación de funciones ayudan a mantener el código fácil de entender y modificar. Utilizar funciones simples para cálculos evita la sobrecomplicación y mejora la mantenibilidad del código.

# Implementación simple para calcular la fecha probable de parto

def calculate\_due\_date(last\_menstruation):

return last\_menstruation + timedelta(weeks=40)

#### 2.1.6 Retroalimentación Continua

**Principio:** La retroalimentación continua permite ajustes iterativos basados en las necesidades reales del usuario y mejora el software conforme a sus expectativas.

**Aplicación en el Proyecto:**

* **Método:** Se realizan sesiones de prueba y se recopila feedback para ajustar la interfaz gráfica y los modelos predictivos.
* **Justificación:** La retroalimentación permite alinear la aplicación con las expectativas de los usuarios, mejorando su funcionalidad y usabilidad. Las iteraciones basadas en feedback aseguran que el producto final cumpla con los requisitos del usuario.

#### 2.1.7 Lenguajes, Tecnologías y metodologías Utilizadas

**Programación: Python**

* **Justificación:** Python se utiliza por su simplicidad, facilidad de uso y la disponibilidad de bibliotecas robustas para machine learning (scikit-learn, TensorFlow) y desarrollo de interfaces gráficas (Tkinter). Estas características facilitan un desarrollo ágil y efectivo.

**Base de Datos: PostgreSQL**

* **Justificación:** PostgreSQL es una base de datos relacional poderosa y flexible que maneja eficientemente grandes volúmenes de datos y es adecuada para almacenar y consultar datos médicos de manera segura.

**Interfaces Gráficas: Tkinter**

* **Justificación:** Tkinter se emplea para crear una interfaz gráfica de usuario intuitiva y accesible. Es una biblioteca estándar de Python que permite el diseño rápido y fácil de interfaces gráficas, facilitando la interacción del usuario con la aplicación.

##### 2.1.7.1 Metodología de desarrollo XP

1. La Metodología de Programación Extrema (Extreme Programming, XP) es un enfoque ágil de desarrollo de software que se centra en la entrega temprana y frecuente de software funcional, la colaboración estrecha entre los miembros del equipo y la capacidad de respuesta ante cambios y requisitos emergentes. Fue creado por Kent Beck a fines de la década de 1990 y ha ganado popularidad por su enfoque pragmático y orientado a resultados. A continuación, se describen algunos de los principios y prácticas clave de XP:
2. **Comunicación constante:** XP promueve una comunicación intensiva y en tiempo real entre los miembros del equipo, incluyendo a los desarrolladores, clientes y gerentes. Esto se logra mediante prácticas como la programación en parejas, donde dos programadores trabajan juntos en la misma tarea, y reuniones regulares cara a cara.
3. **Feedback rápido:** La retroalimentación es fundamental en XP. Los ciclos de desarrollo son cortos y se fomenta la entrega temprana y continua de software funcional al cliente o al usuario final. Esto proporciona una oportunidad para recibir comentarios valiosos y realizar ajustes según sea necesario.
4. **Cambios y adaptabilidad:** XP reconoce que los requisitos y las prioridades cambian con el tiempo. En lugar de resistirse a los cambios, XP se enfoca en abrazarlos. Los cambios y las nuevas prioridades se pueden incorporar fácilmente a través del enfoque incremental y la planificación ágil.
5. **Pruebas automatizadas:** La calidad del software es primordial en XP. Se enfatiza la escritura de pruebas automatizadas en todos los niveles, desde pruebas unitarias hasta pruebas de integración. Estas pruebas garantizan que el software cumpla con los requisitos y funcione correctamente, incluso después de los cambios o mejoras.
6. **Diseño simple:** En XP, el diseño simple es preferido sobre la complejidad innecesaria. Esto se logra a través de la refactorización continua y la eliminación de código duplicado. Al mantener el diseño simple y modular, se mejora la legibilidad del código y la capacidad para introducir cambios sin grandes dificultades.
7. **Programación en parejas:** XP fomenta la colaboración y el intercambio de conocimientos mediante la práctica de la programación en parejas. Dos programadores trabajan juntos en la misma tarea, compartiendo una sola pantalla y teclado. Esto permite una revisión constante del código, el intercambio de ideas y la detección temprana de posibles problemas.

Estos son solo algunos de los principios y prácticas clave de XP. La metodología XP se basa en la idea de que el éxito del desarrollo de software radica en la colaboración, el enfoque en los resultados y la adaptabilidad ante los cambios. Es importante destacar que XP puede adaptarse y ajustarse para adaptarse a las necesidades y características específicas de cada proyecto y equipo.

La metodología XP consta de cuatro etapas de desarrollo, las cuales se

muestran en la siguiente Figura.

Diagrama, Escala de tiempo

Descripción generada automáticamente

**Fase de Planificación:** Es la etapa inicial de todo proyecto en XP. En este punto se comienza a interactuar con el cliente y el resto del grupo de desarrollo para descubrir los requerimientos del sistema. También se identifican el número y tamaño de las iteraciones al igual que se plantean ajustes necesarios a la metodología según las características del proyecto.

**Fase Diseño:** Durante esta fase, se prioriza cada historia de usuario y los programadores estiman el esfuerzo para cada una. El cliente determina el orden en el que se implementan las historias en la iteración.

**Fase de Codificación:** Es un proceso que se realiza en forma paralela con el diseño. Es la única actividad de la que no se puede prescindir, sin código fuente no hay programa. Por tanto, necesitamos codificar y plasmar nuestras ideas a través del código. La programación en XP es en pareja, donde el código expresa tu interpretación del problema, y así lo podemos utilizar para comunicar, y por tanto para aprender.

**Fase Pruebas:** Se enfatiza mucho los aspectos relacionados con las pruebas, clasificándolas en diferentes tipos y funcionalidades específicas, indicando quién, cuándo y cómo deben ser implementadas y ejecutadas. Sólo se liberará una nueva versión si esta ha pasado todas las pruebas.

#### 2.1.8 Herramientas de Integración Continua

* **Herramientas:** Jenkins, GitHub Actions, Travis CI
* **Justificación:** Estas herramientas automatizan la construcion integración y pruebas del código, permitiendo la detección temprana de errores y 2.1.9 **Aplicación Práctica de XP en el Proyecto**

**Planificación y Iteraciones:** El proyecto se organiza en iteraciones cortas que permiten adaptar continuamente el enfoque y las prácticas según los resultados y la retroalimentación obtenida. Esto asegura que la aplicación evolucione en función de las necesidades del usuario y las mejoras detectadas durante el desarrollo.

**Colaboración:** La programación en parejas y la revisión continua del código fomentan la colaboración entre desarrolladores, promoviendo la calidad del código y la transferencia de conocimientos.

**Adaptabilidad:** XP permite ajustar el enfoque y las prácticas del proyecto basándose en resultados de pruebas y feedback de los usuarios, garantizando que el producto final esté alineado con las expectativas del usuario y mejore continuamente.

Esta metodología ha sido fundamental para el desarrollo de una aplicación robusta, flexible y capaz de adaptarse a las necesidades cambiantes del entorno de la salud materna.

### 2.2 Arquitectura del Software

La arquitectura del software describe la estructura organizativa de un sistema y la forma en que sus componentes interactúan. Para el proyecto de gestión de la salud materna, la arquitectura está diseñada para integrar tecnologías avanzadas, como machine learning, interfaces gráficas y bases de datos, y garantizar que el sistema sea escalable, modular y mantenible. La siguiente es una descripción detallada de la arquitectura del software propuesta:

#### 2.2.1. Visión General de la Arquitectura

La arquitectura del software se basa en una estructura en capas que organiza los componentes del sistema en distintos niveles de abstracción. Esto permite una separación clara de responsabilidades, facilita el mantenimiento y mejora la escalabilidad. Las principales capas y componentes incluyen:

1. **Capa de Presentación (Interfaz de Usuario)**
2. **Capa de Aplicación (Lógica de Negocio)**
3. **Capa de Datos (Persistencia de Datos)**
4. **Capa de Integración y Servicios Externos**

**2. Capa de Presentación (Interfaz de Usuario)**

**Descripción:** Esta capa es responsable de la interacción con el usuario. Utiliza Tkinter para proporcionar una interfaz gráfica intuitiva y accesible.

**Componentes:**

* **Ventanas y Formularios:** Para ingresar y visualizar datos, como la información de las pacientes y resultados de cálculos.
* **Menús y Botones:** Para navegación y selección de funcionalidades, incluyendo opciones para crear un usuario, cerrar sesión y realizar cálculos.
* **Validaciones y Mensajes de Error:** Para asegurar que los datos ingresados sean correctos y proporcionar retroalimentación al usuario.

**Tecnología Utilizada:** Tkinter (biblioteca estándar de Python para interfaces gráficas).

**3. Capa de Aplicación (Lógica de Negocio)**

**Descripción:** Esta capa contiene la lógica de negocio del sistema, encargada de realizar cálculos y procesar datos. Implementa los modelos predictivos y las reglas de negocio asociadas con la gestión de la salud materna.

**Componentes:**

**Modelos Predictivos:** Incluye el modelo de regresión lineal y la red neuronal para calcular la fecha probable de parto y evaluar riesgos. Los modelos se entrenan con datos históricos y se utilizan para realizar predicciones en tiempo real.

* **Servicios de Cálculo:** Funciones para calcular el Índice de Masa Corporal (IMC), evaluar la hipertensión y detectar anemia.
* **Controladores:** Encargados de coordinar las solicitudes entre la interfaz de usuario y la capa de datos.

**Tecnología Utilizada:** Python (para la implementación de la lógica de negocio y modelos predictivos), bibliotecas de machine learning como scikit-learn y TensorFlow.

**4. Capa de Datos (Persistencia de Datos)**

**Descripción:** Esta capa se encarga de almacenar y recuperar datos de manera eficiente y segura. Utiliza una base de datos relacional para manejar grandes volúmenes de datos médicos.

**Componentes:**

**Modelo de Datos:** Definición de esquemas y tablas en PostgreSQL para almacenar información sobre pacientes, historial médico, resultados de cálculos, etc.

**Repositorio de Datos:** Clases y funciones para realizar operaciones de lectura y escritura en la base de datos.

**Tecnología Utilizada:** PostgreSQL (sistema de gestión de bases de datos relacional).

**5. Capa de Integración y Servicios Externos**

**Descripción:** Esta capa gestiona la interacción con servicios externos y la integración con otros sistemas si es necesario.

**Componentes:**

* **Servicios de Autenticación:** Mecanismos para la creación y gestión de usuarios, incluyendo el inicio y cierre de sesión.
* **Integración Continua:** Herramientas para automatizar el proceso de integración y pruebas.

**Tecnología Utilizada:** Herramientas de integración continua como Jenkins, GitHub Actions, Travis CI.

**7. Justificación de la Arquitectura**

La arquitectura propuesta permite una clara separación de responsabilidades entre las distintas capas del sistema, lo que facilita el mantenimiento y la escalabilidad. La utilización de tecnologías y prácticas adecuadas para cada capa garantiza que el sistema sea eficiente, seguro y fácil de usar. La elección de Python para la lógica de negocio y el desarrollo de la interfaz gráfica se basa en su facilidad de uso y la robustez de sus bibliotecas. PostgreSQL proporciona una base de datos relacional sólida para gestionar los datos médicos, y las herramientas de integración continua aseguran que el código se mantenga de alta calidad a lo largo del ciclo de desarrollo.

Esta arquitectura no solo permite la implementación efectiva de las funcionalidades necesarias, sino que también proporciona una base sólida para futuras expansiones y mejoras del sistema.

### 2.3 Diseño del Sistema

El diseño del sistema es fundamental para garantizar que todos los componentes de la aplicación funcionen en armonía y puedan manejar las necesidades específicas del análisis de salud materna. La arquitectura del sistema se organiza en varias capas, cada una con funciones y responsabilidades claramente definidas.

#### 2.3.1 Arquitectura General de la Aplicación

La arquitectura general de la aplicación sigue un modelo de tres capas:

1. Capa de Datos: En esta capa se encuentra PostgreSQL, una base de datos relacional robusta que almacena toda la información médica relevante, como los antecedentes de las pacientes, resultados de pruebas y datos históricos. PostgreSQL es elegido debido a su capacidad para manejar grandes volúmenes de datos y su compatibilidad con transacciones complejas y consultas avanzadas.
2. Capa de Aplicación: Esta capa incluye el procesamiento de datos y la lógica de machine learning. Aquí, se implementan los modelos predictivos de machine learning, como la regresión lineal y la red neuronal, utilizando herramientas y librerías de Python. Esta capa es responsable de la integración de datos, la aplicación de modelos predictivos y la generación de resultados que luego serán presentados en la interfaz gráfica.
3. Capa de Presentación: Esta capa se encarga de la interfaz gráfica del usuario (GUI), construida con Tkinter. Tkinter permite crear una interfaz interactiva y accesible para los profesionales de la salud, facilitando la entrada de datos y la visualización de resultados. Esta capa se conecta con la capa de aplicación para recibir datos procesados y mostrar los resultados de manera intuitiva.

#### 2.3.2 Integración con Base de Datos PostgreSQL

La integración con PostgreSQL es esencial para la gestión eficiente y segura de los datos médicos. La base de datos almacena información crítica, como los antecedentes médicos de las pacientes, datos sobre sus consultas y resultados de análisis. La aplicación se conecta a PostgreSQL mediante una interfaz de conexión que permite la inserción, actualización y consulta de datos en tiempo real. Se implementan medidas de seguridad para proteger la confidencialidad de la información y se asegura que la base de datos sea escalable para manejar futuros incrementos en el volumen de datos.

#### 2.3.3 Desarrollo del Modelo de Machine Learning

El desarrollo del modelo de machine learning es una parte crucial de la aplicación, ya que proporciona las capacidades predictivas que permiten la evaluación de riesgos y la predicción de eventos clínicos.

#### 2.3.4 Preprocesamiento de Datos

El preprocesamiento de datos es el primer paso para garantizar que el modelo de machine learning pueda aprender de datos limpios y relevantes. Esto incluye la limpieza de datos, la imputación de valores faltantes y la normalización de características. Los datos se transforman y se codifican para que sean aptos para el entrenamiento del modelo. En este proyecto, el preprocesamiento también implica la codificación de variables categóricas, como los antecedentes médicos, para su uso en el modelo predictivo.

#### 2.3.5 Entrenamiento del Modelo de Regresión Lineal y Red Neuronal

Se utilizan dos tipos de modelos predictivos:

* Regresión Lineal: Se emplea para modelar relaciones lineales entre variables y prever valores continuos, como el riesgo asociado con ciertos antecedentes médicos.
* Red Neuronal: Se utiliza para capturar patrones complejos y no lineales en los datos. La red neuronal se entrena con datos históricos para predecir la fecha probable de parto y otros resultados clínicos.

Ambos modelos se entrenan utilizando conjuntos de datos divididos en entrenamiento y prueba, y se ajustan mediante técnicas de validación cruzada para optimizar su rendimiento.

#### 2.3.6 Evaluación del Modelo

La evaluación del modelo es crucial para determinar su precisión y eficacia. Se utilizan métricas de evaluación como la precisión, la sensibilidad y la especificidad para medir el rendimiento del modelo. Los modelos se prueban con datos no vistos durante el entrenamiento para asegurar que generalicen bien y no sobreajusten los datos de entrenamiento.

#### 2.3.7 Desarrollo de la Interfaz Gráfica

El desarrollo de la interfaz gráfica es esencial para proporcionar una experiencia de usuario intuitiva y eficiente. Tkinter se utiliza para crear una GUI que permita a los profesionales de la salud interactuar con la aplicación de manera efectiva.

#### 2.3.8 Implementación de Funcionalidades

La interfaz gráfica se diseña para permitir la entrada de datos, la selección de opciones y la visualización de resultados. Se implementan formularios para ingresar información sobre la paciente, botones para ejecutar cálculos y paneles para mostrar los resultados. La interfaz se organiza de manera que sea fácil de usar y navegar, con una estructura clara y controles bien definidos.

#### 2.3.9 Sistema de Autenticación de Usuarios

Para asegurar el acceso a la aplicación y proteger la información médica, se implementa un sistema de autenticación de usuarios. Los médicos deben iniciar sesión con credenciales válidas antes de acceder a las funcionalidades de la aplicación. Este sistema garantiza que solo usuarios autorizados puedan utilizar la aplicación y consultar datos sensibles.

### 2.4 Implementación de Funcionalidades Adicionales

La implementación de funcionalidades adicionales en la aplicación de gestión de salud materna se orienta a proporcionar una evaluación integral que vaya más allá de los aspectos básicos de predicción de la fecha probable de parto y evaluación de riesgos. Estas funcionalidades adicionales se enfocan en aspectos críticos de la salud materna, tales como la detección de anemia y el cálculo del Índice de Masa Corporal (IMC), lo cual es fundamental para una valoración más completa de la salud de las pacientes. A continuación se detallan las razones y la justificación para cada una de estas funcionalidades.

#### 2.4.1 Detección de Anemia

Justificación: La anemia es una condición médica prevalente durante el embarazo que puede tener graves consecuencias para la salud de la madre y el bebé. La detección temprana de anemia es esencial para prevenir complicaciones y para la implementación de estrategias de manejo adecuadas. Las mujeres embarazadas con anemia corren un riesgo mayor de parto prematuro, bajo peso al nacer del bebé y complicaciones en el parto. Incorporar un módulo de detección de anemia en la aplicación permite identificar estos riesgos de manera temprana y tomar medidas preventivas.

Implementación: La funcionalidad de detección de anemia se implementa integrando datos clínicos relevantes, como los niveles de hemoglobina y hematocrito, que son indicadores clave de la anemia. La aplicación utilizará estos datos para calcular los niveles de anemia y clasificar a las pacientes en diferentes categorías (por ejemplo, anemia leve, moderada, severa) utilizando umbrales clínicamente aceptados. Esto se puede hacer mediante la creación de algoritmos que comparen los valores ingresados con los rangos de referencia para detectar y clasificar la anemia.

**Beneficios:**

* Mejora la precisión del diagnóstico al proporcionar una herramienta automatizada que complementa el juicio clínico.
* Permite un monitoreo constante de los indicadores de anemia, facilitando la intervención temprana y el seguimiento adecuado.
* Ofrece una visión más completa de la salud de la paciente, mejorando el cuidado y la gestión durante el embarazo.

#### 2.4.2 Cálculo de IMC y Riesgos Médicos

Justificación: El Índice de Masa Corporal (IMC) es una medida clave utilizada para evaluar el estado nutricional y el riesgo de diversas condiciones médicas durante el embarazo. Un IMC fuera del rango normal puede estar asociado con riesgos como la hipertensión gestacional, la diabetes gestacional y otros problemas que pueden complicar el embarazo. La incorporación de la funcionalidad para calcular el IMC en la aplicación permite una evaluación continua del estado nutricional y la identificación de riesgos potenciales.

Implementación: Para calcular el IMC, la aplicación requerirá la entrada de datos como el peso y la altura de la paciente. Usando la fórmula estándar del IMC (peso en kilogramos dividido por el cuadrado de la altura en metros), el sistema calculará automáticamente el IMC y lo clasificará en categorías como bajo peso, peso normal, sobrepeso y obesidad. Además, se integrarán algoritmos que evalúen el riesgo de complicaciones basado en el IMC y otros factores médicos ingresados.

**Beneficios:**

* Proporciona una evaluación continua del estado nutricional, permitiendo a los profesionales de la salud intervenir rápidamente si se detectan problemas.
* Facilita la identificación de riesgos asociados con el IMC y permite una mejor planificación y manejo del embarazo.
* Mejora la comunicación entre la paciente y el proveedor de salud al proporcionar datos claros y precisos sobre el estado de salud de la paciente.

La implementación de estas funcionalidades adicionales en la aplicación no solo amplía su capacidad para gestionar la salud materna, sino que también mejora la calidad del cuidado proporcionado. La detección de anemia y el cálculo del IMC son herramientas cruciales para una evaluación integral y precisa, que facilita la intervención temprana y una mejor gestión de los riesgos durante el embarazo. Estas funcionalidades contribuyen a una atención más personalizada y basada en datos, alineándose con los objetivos de proporcionar una solución efectiva y completa para la salud materna.

### 2.5 Pruebas y Validación

**La fase de pruebas y validación es crucial para garantizar que la** aplicación funcione correctamente y cumpla con los requisitos establecidos.

* Pruebas Unitarias: Se realizan pruebas unitarias para verificar que cada componente de la aplicación funcione de manera independiente y correcta.
* Pruebas de Integración: Se llevan a cabo pruebas de integración para asegurar que todos los componentes del sistema interactúen correctamente entre sí.
* Validación con Datos Reales: Se valida el modelo y la aplicación utilizando datos reales para confirmar su precisión y eficacia en condiciones de uso real.
* Pruebas de Usabilidad: Se realizan pruebas de usabilidad con usuarios finales para asegurar que la interfaz gráfica sea intuitiva y satisfaga sus necesidades.

Estas pruebas y validaciones garantizan que la aplicación sea fiable, precisa y adecuada para su uso en un entorno clínico.

### 2.6 Implementación

El Capítulo 4 describe en detalle el proceso de implementación de la aplicación para la gestión de la salud materna. Este capítulo aborda la ejecución práctica de la metodología descrita en el Capítulo 3, proporcionando una visión clara del código fuente, el desarrollo del sistema de usuarios y autenticación, y los procesos de integración y pruebas. La implementación es crucial para transformar los conceptos y diseños en una aplicación funcional y operativa.

#### 2.6.1 Código Fuente y Explicación

La implementación del código fuente es fundamental para el desarrollo de una aplicación eficaz y robusta en la gestión de la salud materna. Cada componente del código está diseñado para cumplir con objetivos específicos, desde la carga y preparación de datos hasta la creación de una interfaz gráfica amigable y la integración de modelos predictivos. A continuación, se proporciona una explicación detallada y justificada de cada componente clave del código fuente.

#### 2.6.2 Carga y Preparación de Datos

**Justificación:** La correcta carga y preparación de datos es esencial para asegurar que la aplicación funcione de manera efectiva y produzca resultados precisos. Los datos médicos, como los antecedentes de salud, medidas físicas y resultados de pruebas, deben ser procesados de manera que sean compatibles con los modelos predictivos y las funcionalidades de la aplicación.

**Explicación:** El código para la carga de datos se encarga de importar datos desde diversas fuentes, como archivos CSV, bases de datos PostgreSQL o entradas manuales. Esto se realiza utilizando bibliotecas de Python como pandas para la manipulación de datos y sqlalchemy para la conexión con bases de datos.

Carga de Datos: Se utilizan bibliotecas como pandas para leer datos desde fuentes externas. El código está diseñado para manejar diferentes formatos de archivos y posibles errores en la carga de datos. Por ejemplo:

import pandas as pd

try:

data = pd.read\_csv('datos\_maternos.csv')

except FileNotFoundError:

data = pd.DataFrame() # Manejo de error si el archivo no se encuentra

2.Limpieza de Datos: Se eliminan duplicados, se imputan valores faltantes y se corrigen errores en los datos. Esto asegura que el modelo de machine learning trabaje con datos limpios y fiables. El código para esta tarea puede incluir operaciones como:

data.drop\_duplicates(inplace=True)

data.fillna(method='ffill', inplace=True)

3.Transformación de Datos: Se convierten las variables categóricas en numéricas mediante técnicas como la codificación one-hot y senormalizanlas variables continuas para que el modelo pueda aprender de manera más eficiente. Por ejemplo:

data = pd.get\_dummies(data, columns=['antecedentes\_medicos'])

Estas etapas aseguran que los datos estén listos para el entrenamiento y la evaluación de los modelos predictivos.

#### 2.6.3 Modelo Predictivo y Escalado

**Justificación:** El modelo predictivo es el núcleo de la aplicación, ya que permite realizar cálculos basados en datos históricos para predecir la fecha probable de parto y evaluar riesgos médicos. La precisión y la eficiencia del modelo dependen en gran medida del diseño y entrenamiento adecuados del mismo.

**Explicación:** El desarrollo del modelo predictivo implica varios componentes clave:

**Entrenamiento del Modelo:** Se utilizan técnicas de machine learning como la regresión lineal y redes neuronales para predecir eventos y evaluar riesgos. El código incluye la definición de la arquitectura del modelo, la selección de funciones de pérdida y el uso de algoritmos de optimización.

**Escalado de Datos:** Los datos deben ser escalados para asegurar que las características numéricas estén en un rango adecuado para el entrenamiento del modelo. Se utilizan técnicas de escalado como la normalización o estandarización para mejorar la convergencia y el rendimiento del modelo.

**Evaluación del Modelo:** Después del entrenamiento, el modelo se evalúa utilizando métricas de rendimiento como la precisión, el recall, la F1-score y el error cuadrático medio (MSE). Esto garantiza que el modelo generalice bien y proporcione predicciones precisas.

El código para la implementación del modelo incluye la integración de librerías de machine learning como scikit-learn para la regresión lineal y TensorFlow o Keras para las redes neuronales. También se realiza un ajuste de hiperparámetros para optimizar el rendimiento del modelo.

Entrenamiento del Modelo: Se crean y entrenan dos modelos: la regresión lineal y la red neuronal. Utilizando bibliotecas como scikit-learn para la regresión lineal y tensorflow o keras para la red neuronal. El código para el entrenamiento puede ser:

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)

model = LinearRegression()

model.fit(X\_train, y\_train)

1. Escalado de Datos: La normalización o estandarización de datos se realiza para asegurar que los modelos no se vean afectados por escalas de datos distintas. Esto se logra utilizando StandardScaler o técnicas similares:

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

1. Evaluación del Modelo: Se evalúa el rendimiento de los modelos utilizando métricas como el error cuadrático medio (MSE) para la regresión lineal y la precisión, la sensibilidad y la especificidad para la red neuronal. Por ejemplo:

y\_pred = model.predict(X\_test)

mse = mean\_squared\_error(y\_test, y\_pred)

Estas actividades aseguran que los modelos sean precisos y puedan realizar predicciones fiables.

#### 2.6.4 Interfaz Gráfica y Funcionalidades

**Justificación:** Una interfaz gráfica de usuario (GUI) bien diseñada es crucial para la usabilidad y accesibilidad de la aplicación. Permite a los usuarios interactuar con el sistema de manera intuitiva y realizar las operaciones necesarias para gestionar la salud materna de manera eficiente.

**Explicación:** La implementación de la interfaz gráfica se realiza utilizando la biblioteca Tkinter, que proporciona herramientas para crear interfaces gráficas interactivas en Python. El código para la interfaz gráfica incluye:

**Diseño de la Interfaz:** Creación de ventanas, formularios y controles (botones, etiquetas, campos de entrada) para permitir a los usuarios introducir datos, visualizar resultados y acceder a funcionalidades.

**Implementación de Funcionalidades:** Integración de las funcionalidades de la aplicación con la interfaz gráfica. Esto incluye la interacción con los modelos predictivos, la visualización de resultados y la ejecución de cálculos médicos.

**Sistema de Autenticación de Usuarios:** Implementación de un sistema de inicio de sesión y registro para asegurar que solo usuarios autorizados puedan acceder a las funciones de la aplicación. Esto se realiza mediante la gestión de credenciales y permisos de usuario.

La interfaz gráfica debe ser intuitiva y fácil de usar para garantizar una experiencia de usuario fluida. La implementación del código se basa en prácticas de diseño de interfaces amigables, asegurando que los usuarios puedan navegar y utilizar la aplicación con facilidad.

Diseño de la Interfaz: Se crean ventanas, formularios y botones utilizando Tkinter para permitir a los usuarios ingresar datos y ver resultados. El código para crear una ventana simple puede ser:

import tkinter as tk

def mostrar\_resultados():

resultado\_label.config(text="Resultado del cálculo")

ventana = tk.Tk()

ventana.title("Aplicación de Salud Materna")

resultado\_label = tk.Label(ventana, text="")

resultado\_label.pack()

calcular\_button = tk.Button(ventana, text="Calcular", command=mostrar\_resultados)

calcular\_button.pack()

ventana.mainloop()

1. Implementación de Funcionalidades: Se integran funcionalidades para ingresar datos, ejecutar cálculos y mostrar resultados en la interfaz gráfica. El código maneja eventos y actualiza la interfaz en función de las acciones del usuario:

def calcular\_fecha\_parto():

1. Validación de Entrada: Se implementa validación para asegurar que los datos ingresados por el usuario sean correctos y completos antes de procesarlos. Esto incluye la verificación de campos obligatorios y formatos de datos:

def validar\_datos():

if not peso\_entry.get() or not altura\_entry.get():

tk.messagebox.showerror("Error", "Por favor, complete todos los campos")

return False

return True

Estas funcionalidades garantizan que la aplicación sea fácil de usar y que proporcione una experiencia de usuario efectiva.

### 2.7 Desarrollo del Sistema de Usuarios y Autenticación

El sistema de usuarios y autenticación es crucial para proteger la información sensible y garantizar que solo los usuarios autorizados tengan acceso a la aplicación. El desarrollo incluye:

1. Sistema de Registro: Permite a los usuarios crear nuevas cuentas proporcionando credenciales seguras. El código para el registro puede incluir:

def registrar\_usuario(username, password):

hashed\_password = hash\_password(password)

# Guardar usuario y contraseña en la base de datos

1. Sistema de Autenticación: Verifica las credenciales de los usuarios durante el inicio de sesión y asegura el acceso a la aplicación. El código para la autenticación puede ser:

def autenticar\_usuario(username, password):

stored\_password = obtener\_contraseña\_usuario(username)

return verificar\_password(password, stored\_password)

1. Gestión de Sesiones: Se maneja el inicio y cierre de sesión para controlar el acceso a la aplicación y proteger la información del usuario. El código para gestionar sesiones puede incluir:

Este sistema garantiza la seguridad de los datos y la privacidad de los usuarios.

### 2.8 Integración y Pruebas

La integración y pruebas son pasos finales críticos para asegurar que todos los componentes de la aplicación funcionen correctamente juntos.

1. Integración de Componentes: Se integran la interfaz gráfica, los modelos predictivos y la base de datos en un sistema cohesivo. Esto implica la conexión de los modelos predictivos con la GUI para que los resultados se muestren correctamente en función de los datos ingresados.
2. Pruebas de Funcionalidad: Se realizan pruebas exhaustivas para verificar que todas las funcionalidades de la aplicación funcionen como se espera. Esto incluye pruebas unitarias para cada componente individual y pruebas de integración para el sistema en su totalidad.
3. Pruebas de Usabilidad: Se lleva a cabo una evaluación de la experiencia del usuario para garantizar que la interfaz gráfica sea intuitiva y fácil de usar. Esto puede incluir pruebas con usuarios finales para obtener retroalimentación sobre la usabilidad y realizar ajustes según sea necesario.
4. Validación de Resultados: Se valida la precisión de los resultados generados por los modelos predictivos utilizando datos de prueba y comparando los resultados con expectativas conocidas. Esto asegura que la aplicación proporcione resultados precisos y fiables en un entorno clínico real.

En conclusión, el Capítulo 4 detalla la implementación técnica de la aplicación, desde la carga y preparación de datos hasta el desarrollo de la interfaz gráfica, el sistema de usuarios y autenticación, y los procesos de integración y pruebas. Cada componente es crítico para asegurar que la aplicación sea funcional, segura y útil para los profesionales de la salud en la gestión de la salud materna.

### 2.9 Resultados y Discusión

En el Capítulo 5 se analizan y discuten los resultados obtenidos de la implementación de la aplicación para la gestión de la salud materna, así como la efectividad y usabilidad de las diferentes componentes del sistema. Este capítulo se centra en evaluar el desempeño del modelo predictivo, la usabilidad de la interfaz gráfica, la efectividad de las funcionalidades adicionales, y la retroalimentación recibida de los usuarios, con el objetivo de identificar áreas de mejora y validar la efectividad general del sistema.

#### 2.9.1 Desempeño del Modelo Predictivo

El desempeño del modelo predictivo es crucial para la efectividad de la aplicación, ya que determina la precisión de las predicciones realizadas sobre la fecha probable de parto y el riesgo de complicaciones.

1. **Evaluación del Modelo de Regresión Lineal:** El modelo de regresión lineal se evaluó utilizando métricas como el error cuadrático medio (MSE) y el coeficiente de determinación (R²). Estos indicadores permiten medir cuán bien el modelo se ajusta a los datos históricos y su capacidad para generalizar a datos no vistos. En nuestras pruebas, el MSE se mantuvo dentro de un rango aceptable, indicando que el modelo tiene un buen desempeño en términos de precisión. El R² también mostró un valor alto, sugiriendo que el modelo explica una proporción significativa de la variabilidad en los datos.

from sklearn.metrics import mean\_squared\_error, r2\_score

y\_pred = model.predict(X\_test)

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

1. **Evaluación del Modelo de Red Neuronal:** La red neuronal se evaluó utilizando métricas como la precisión, la sensibilidad y la especificidad para clasificar los riesgos médicos y detectar anemia. Estos indicadores son esenciales para garantizar que la red neuronal pueda realizar predicciones correctas en un contexto clínico. La red neuronal mostró una alta precisión y sensibilidad en la detección de riesgos y anemia, lo que valida su utilidad en la aplicación.

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

1. **Comparación de Modelos:** Se compararon los resultados obtenidos de ambos modelos para determinar cuál proporciona las mejores predicciones. La red neuronal, debido a su capacidad para modelar relaciones no lineales y aprender características complejas, mostró una ventaja en la detección de patrones específicos en los datos médicos, mientras que la regresión lineal fue efectiva para la predicción de variables continuas como la fecha probable de parto.

#### 2.9.2 Usabilidad de la Interfaz Gráfica

La usabilidad de la interfaz gráfica es crucial para la aceptación y efectividad de la aplicación. La interfaz debe ser intuitiva y fácil de usar para los profesionales de la salud, quienes interactuarán con la aplicación de manera regular.

1. **Diseño de la Interfaz:** Se evaluó la efectividad del diseño de la interfaz gráfica mediante pruebas de usabilidad y la retroalimentación de usuarios. La interfaz debe permitir una entrada de datos rápida y sencilla, así como una visualización clara de los resultados. La implementación de menús y botones de navegación contribuyó a una experiencia de usuario fluida. Se realizaron ajustes basados en pruebas para mejorar la disposición de los elementos y la navegación.

def actualizar\_resultados(resultados):

resultado\_label.config(text=f"Fecha probable de parto: {resultados['fecha\_parto']}")

1. **Pruebas de Usabilidad:** Se realizaron pruebas con usuarios finales para evaluar la facilidad de uso de la interfaz. Las pruebas revelaron que la mayoría de los usuarios pudieron completar tareas sin dificultades significativas. Se identificaron y abordaron problemas menores relacionados con la navegación y la entrada de datos, lo que mejoró la usabilidad general.

# Ejemplo de entrada de datos

def ingresar\_datos():

try:

peso = float(peso\_entry.get())

altura = float(altura\_entry.get())

# Validar y procesar datos

except ValueError:

tk.messagebox.showerror("Error", "Entrada inválida para peso o altura")

#### 2.9.3 Evaluación de Funcionalidades Adicionales

Las funcionalidades adicionales, como la detección de anemia y el cálculo del Índice de Masa Corporal (IMC) y los riesgos médicos, son componentes clave de la aplicación. Su efectividad se evalúa de la siguiente manera:

1. **Detección de Anemia:** La funcionalidad de detección de anemia se evalúa en términos de precisión y relevancia clínica. La capacidad de la aplicación para identificar correctamente los casos de anemia a partir de los datos ingresados es crucial para su utilidad en el manejo de la salud materna. Los resultados mostraron que la detección de anemia es efectiva y permite una intervención temprana para mejorar los resultados de salud.

def detectar\_anemia(nivel\_hemoglobina):

if nivel\_hemoglobina < umbral\_anemia:

return "Anemia detectada"

return "Nivel de hemoglobina normal"

1. **Cálculo de IMC y Riesgos Médicos:** El cálculo del IMC y la evaluación de riesgos médicos basados en antecedentes y otros parámetros se realizaron con precisión. Estas funcionalidades permiten una evaluación integral de la salud de la paciente y la identificación de posibles complicaciones. Los resultados obtenidos se compararon con los estándares médicos para asegurar la exactitud.

def calcular\_imc(peso, altura):

return peso / (altura \*\* 2)

#### 2.9.4 Retroalimentación de Usuarios y Mejoras

La retroalimentación de los usuarios es esencial para identificar áreas de mejora y garantizar que la aplicación satisfaga sus necesidades. La retroalimentación se recopiló a través de encuestas y entrevistas con profesionales de la salud que utilizaron la aplicación en un entorno clínico.

1. **Comentarios Recibidos:** Los usuarios proporcionaron comentarios sobre la funcionalidad y la usabilidad de la aplicación. La mayoría de los comentarios fueron positivos, destacando la precisión de las predicciones y la facilidad de uso. Sin embargo, también se identificaron áreas para mejorar, como la necesidad de una mayor personalización de las funcionalidades y una mejor integración con sistemas existentes.
2. **Mejoras Implementadas:** Basado en la retroalimentación, se realizaron mejoras en la interfaz gráfica para hacerla más intuitiva y se ajustaron las funcionalidades adicionales para satisfacer mejor las necesidades clínicas. También se implementaron nuevas características solicitadas por los usuarios, como la capacidad de exportar informes y la integración con otros sistemas de gestión de salud.

def exportar\_informe(resultados):

with open('informe\_resultados.txt', 'w') as file:

file.write(f"Fecha probable de parto: {resultados['fecha\_parto']}\n")

## 3.1 Requisitos Funcionales y No Funcionales

**1. Registro de Paciente**

- Descripción:El sistema debe permitir el registro de nuevas pacientes.

- Detalles:

- Captura de datos personales (nombre, edad, peso, altura, fecha de última menstruación).

- Selección de antecedentes médicos (sida, hipertensión, diabetes, etc.).

- Validación de datos antes del almacenamiento.

- Confirmación del registro exitoso.

**2. Cálculo de Fecha Probable de Parto**

- Descripción:El sistema debe calcular la fecha probable de parto basada en la fecha de última menstruación.

- Detalles:

- Ingreso de la fecha de última menstruación.

- Cálculo de la fecha probable de parto utilizando un modelo predictivo.

- Presentación del resultado en la interfaz gráfica.

- Almacenamiento de la fecha calculada en el historial de la paciente.

**3. Evaluación de Riesgos Médicos**

- Descripción:El sistema debe evaluar el riesgo de complicaciones basado en los antecedentes médicos.

- Detalles:

- Selección de antecedentes médicos.

- Cálculo del riesgo (bajo, medio, alto).

- Presentación de resultados en la interfaz gráfica.

- Almacenamiento de la evaluación de riesgo en la base de datos.

**4. Detección de Anemia**

- Descripción: El sistema debe permitir la detección de anemia mediante criterios predefinidos.

- Detalles:

- Ingreso de resultados de análisis de sangre.

- Cálculo de la presencia de anemia basado en los resultados.

- Presentación de los resultados en la interfaz gráfica.

- Almacenamiento de la información sobre anemia en el historial médico.

**5. Sistema de Autenticación de Usuarios**

- Descripción:El sistema debe permitir a los usuarios registrarse, iniciar sesión y cerrar sesión de manera segura.

- Detalles:

- Registro de nuevos usuarios y almacenamiento seguro de credenciales.

- Inicio de sesión con autenticación de usuario y contraseña.

- Cierre de sesión y redirección a la pantalla de inicio de sesión.

- Mensajes de error para credenciales incorrectas.

**6. Gestión de Sesiones**

- Descripción: El sistema debe gestionar las sesiones de usuario de manera segura.

- Detalles:

- Implementación de un botón de "Cerrar Sesión" en la interfaz gráfica.

- Cierre de sesión seguro que borre todos los datos de sesión.

- Prevención de la reanudación de sesión sin autenticación.

**Requisitos No Funcionales**

**1. Usabilidad**

- Descripción:La aplicación debe ser intuitiva y fácil de usar para los profesionales de salud.

- Detalles:

- Interfaz gráfica clara y accesible utilizando Tkinter.

- Manual de usuario detallado para facilitar la adaptación.

**2. Rendimiento**

- Descripción:La aplicación debe operar de manera eficiente y rápida.

- Detalles:

- Tiempo de respuesta para cálculos y consultas no debe exceder los 2 segundos.

- Eficiencia en el manejo de grandes volúmenes de datos médicos.

**3. Seguridad**

- Descripción:La aplicación debe garantizar la seguridad y privacidad de los datos médicos.

- Detalles:

- Encriptación de credenciales y datos sensibles en la base de datos.

- Cumplimiento con las normativas de protección de datos (como GDPR o HIPAA).

**4. Mantenibilidad**

- Descripción: El código debe ser fácil de mantener y actualizar.

- Detalles:

- Aplicación de prácticas de refactorización y desarrollo basado en pruebas.

- Documentación clara y completa del código y del sistema.

**5. Escalabilidad**

- Descripción: La aplicación debe ser capaz de manejar un aumento en el número de usuarios y datos sin degradar el rendimiento.

- Detalles:

- Diseño modular que permita agregar nuevas funcionalidades sin afectar el sistema existente.

- Uso de PostgreSQL para manejar grandes volúmenes de datos.

**6. Compatibilidad**

- Descripción: La aplicación debe ser compatible con diferentes entornos y dispositivos.

- Detalles:

- Compatibilidad con sistemas operativos como Windows, macOS y Linux.

- Interfaz gráfica adaptable a diferentes tamaños de pantalla.

### 3.1.1 Artefacto: Historia de Usuario

Una historia de usuario es una breve descripción de una característica del software desde la perspectiva del usuario final. Está diseñada para capturar lo que el usuario quiere lograr y por qué es importante para él. A continuación, se presentan las historias de usuario para el proyecto de gestión de la salud materna.

**Artefacto de Historia de Usuario**

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 1** | **Nombre de la Historia de Usuario : Registro de Paciente** | |
| **Modificación o Extensión de Historia de Usuario y Número: Ninguna** | | |
| **Usuario: Administrador, Médico** | | **Iteración asignada: 1** |
| **Prioridad del Negocio: Alta** | | **Puntos Estimados: 3 meses** |
| **Riesgo en Desarrollo: Media** | | **Puntos Reales: 3 meses** |
| **Programador responsable: Yaisel Turcas Matos** | | |
| **Descripción:**  1. El sistema debe permitir ingresar datos personales, incluyendo nombre, edad, peso, altura, y fecha de última menstruación.  2. Debe ser posible añadir antecedentes médicos relevantes mediante casillas de verificación (sida, hipertensión, diabetes, etc.).  3. La información debe ser validada antes de ser almacenada en la base de datos.  4. El sistema debe mostrar un mensaje de confirmación al registrar con éxito una nueva paciente. | | |
| **Observaciones:** Los datos deben almacenarse en la base de datos PostgreSQL.  - La interfaz debe ser diseñada con Tkinter para facilitar el ingreso de datos. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número: 2** | **Nombre de la Historia de Usuario:** Cálculo de Fecha Probable de Parto | |
| **Modificación o Extensión de Historia de Usuario y Número: Ninguna** | | |
| **Usuario: Médico** | | **Iteración asignada: 2** |
| **Prioridad del Negocio: Alta** | | **Puntos Estimados: 2 días** |
| **Riesgo en Desarrollo: Media** | | **Puntos Reales: 1 día** |
| **Programador responsable: Roseilis de La Cruz** | | |
| **Descripción:** 1. El sistema debe permitir ingresar la fecha de última menstruación.  2. El sistema debe calcular la fecha probable de parto utilizando el modelo predictivo.  3. El resultado debe ser mostrado en la interfaz de usuario de manera clara y precisa.  4. Debe ser posible guardar el cálculo en el historial de la paciente. | | |
| **Observaciones:**  Utilizar el modelo de regresión lineal para realizar el cálculo.  - La fecha debe ser calculada y presentada en la interfaz gráfica mediante Tkinter. | | |

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | | |
| **Número:3** | **Nombre de la Historia de Usuario:** Evaluación de Riesgos Médicos | |
| **Modificación o Extensión de Historia de Usuario y Número: Ninguna** | | |
| **Usuario: Médico** | | **Iteración asignada: 3** |
| **Prioridad del Negocio: (**Alta/Media/ Baja) : Alta | | **Puntos Estimados: 3 días** |
| **Riesgo en Desarrollo: : (**Alta/Media/ Baja)  Media | | **Puntos Reales: 2 días** |
| **Programador responsable: Roseilis de la Cruz** | | |
| **Descripción:** 1. El sistema debe permitir seleccionar múltiples antecedentes médicos y evaluar el riesgo (bajo, medio, alto).  2. Debe ser posible ver el resultado de la evaluación de riesgo en la interfaz de usuario.  3. Los riesgos deben ser almacenados junto con el perfil de la paciente en la base de datos. | | |
| Observaciones: - Utilizar un algoritmo para calcular el riesgo en función de los antecedentes médicos.  - Mostrar los resultados en la interfaz gráfica con Tkinter. | | |

Estas historias de usuario sirven como base para el desarrollo de la aplicación, guiando la implementación de funcionalidades y asegurando que el sistema satisfaga las necesidades y expectativas de los usuarios finales.

### 3.2 Tareas de Ingeniería

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 1** | **Historia de Usuario (Número y Nombre): HU 5 Registro de paciente.** | |
| **Nombre de Tarea: Diseñar y desarrollar el formulario de ingreso de datos.** | | |
| **Tipo de Tarea (Desarrollo/Corrección): Desarrollo**  **Mejora/Otra (especificar) ): Diseño** | | **Puntos Estimados: 1 día** |
| **Fecha de Inicio: 8/septiempre/2024** | | **Fecha de Finalización: 8 / septiembre /2024** |
| **Riesgo en desarrollo: Media** | | **Puntos Reales: 1 día** |
| **Programador responsable: Roseilis De La Cruz** | | |
| **Descripción: : Desarrollar una interfaz para la entrada de características maternas (edad, altura, peso, presión arterial, edad gestacional, historial de partos previos).** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número:2** | **Historia de Usuario (Número y Nombre): HU 5 Registro de Paciente** | |
| **Nombre de Tarea: Implementar validaciones de datos para asegurar la precisión y completitud.** | | |
| **Tipo de Tarea (Desarrollo/Corrección) : Desarrollo**  **Mejora/Otra (especificar) ):** | | **Puntos Estimados: 2 días** |
| **Fecha de Inicio: 8/septiembre /2024** | | **Fecha de Finalización: 9/septiembre/2024** |
| **Riesgo en desarrollo: Media** | | **Puntos Reales: 2 días** |
| **Programador responsable: Roseilis de La Cruz** | | |
| **Descripción: Se deben implementar validaciones de datos para que sean más exactos .** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 3** | **Historia de Usuario (Número y Nombre):HU 5 Registro de Paciente .** | |
| **Nombre de Tarea: Crear pruebas automatizadas para validar la entrada de datos.** | | |
| **Tipo de Tarea (Desarrollo/Corrección/**  **Mejora/Otra (especificar) ): Diseño** | | **Puntos Estimados: 3 días** |
| **Fecha de Inicio: 8 /septiembre / 2024** | | **Fecha de Finalización: 10 / septiembre /2024** |
| **Riesgo en desarrollo: Medio** | | **Puntos Reales: 3 días** |
| **Programador responsable: Yaisel Turcas Matos** | | |
| **Descripción: Se deben crear las pruebas automatizadas para validar la entrada de los datos.** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 1** | **Historia de Usuario (Número y Nombre): HU 5 Calculo de Fecha probable de parto** | |
| **Nombre de Tarea: Definir las reglas de negocio para el cálculo.** | | |
| **Tipo de Tarea (Desarrollo/Corrección/**  **Mejora/Otra (especificar) ):** | | **Puntos Estimados: 1 día** |
| **Fecha de Inicio: 8/septiempre/2024** | | **Fecha de Finalización: 9/ septiembre /2023** |
| **Riesgo en desarrollo: Media** | | **Puntos Reales: 1 día** |
| **Programador responsable: Yaisel Turcas** | | |
| **Descripción: Implementar un sistema experto para el cálculo de la fecha probable de parto .** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 2** | **Historia de Usuario (Número y Nombre): HU 5 Calculo de fecha probable de parto** | |
| **Nombre de Tarea: Desarrollar el algoritmo para el cálculo de fecha probable de parto** | | |
| **Tipo de Tarea : Desarrollo**  **Mejora/Otra (especificar) ): Diseño** | | **Puntos Estimados: 2 días** |
| **Fecha de Inicio: 9 / septiembre/2024** | | **Fecha de Finalización: 10/septiembre/2024** |
| **Riesgo en desarrollo: Media** | | **Puntos Reales: 1 día** |
| **Programador responsable: Roseilis de La Cruz** | | |
| **Descripción: Crear y desarrollar un algoritmo para realizar los cálculos pertinentes.** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 3** | **Historia de Usuario (Número y Nombre): HU 5 Cálculo de fecha probable de parto .** | |
| **Nombre de Tarea: Validar el cálculo de la fecha probable de parto .** | | |
| **Tipo de Tarea (Desarrollo/Corrección/**  **Mejora/Otra (especificar) ): Diseño** | | **Puntos Estimados: 3 días** |
| **Fecha de Inicio: 8 / septiembre/2024** | | **Fecha de Finalización: 9 / septiembre/2024** |
| **Riesgo en desarrollo: Media** | | **Puntos Reales: 1 dia** |
| **Programador responsable: Roseilis De La Cruz** | | |
| **Descripción: Hacer la validación de los cálculos realizados de las probabilidades de fecha de parto .** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 1** | **Historia de Usuario (Número y Nombre): HU 5 Evaluacion de riesgos médicos** | |
| **Nombre de Tarea: Recopilar y preprocesar los riegos médicos para la validación.** | | |
| **Tipo de Tarea (Desarrollo/Corrección) : Desarrollo**  **Mejora/Otra (especificar) ):** | | **Puntos Estimados: 3 días** |
| **Fecha de Inicio: 8/septiempre/2024** | | **Fecha de Finalización: 10 / septiembre/2024** |
| **Riesgo en desarrollo: Media** | | **Puntos Reales: 3 días** |
| **Programador responsable: Roseilis De La Cruz** | | |
| **Descripción: Se desarrolla la Evaluación de Riesgos Médicos.** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 2** | **Historia de Usuario (Número y Nombre): HU 5 Evaluacion de riesgos médicos** | |
| **Nombre de Tarea: Entrenar modelos de regresión lineal y redes neuronales (LSTM).** | | |
| **Tipo de Tarea (Desarrollo/Corrección) Desarrollo**  **Mejora/Otra (especificar) ):** | | **Puntos Estimados: 2 meses** |
| **Fecha de Inicio: 8/septiempre/2024** | | **Fecha de Finalización: 8/ octubre / 2024** |
| **Riesgo en desarrollo: Alta** | | **Puntos Reales: 1 mes** |
| **Programador responsable: Yaisel Turcas Matos** | | |
| **Descripción: Entrenar al sistema con los modelos de regresión lineal y redes neuronales (LSTM)** | | |

|  |  |  |
| --- | --- | --- |
| **Tarea de Ingeniería** | | |
| **Número: 3** | **Historia de Usuario (Número y Nombre): HU 5 Evaluacion de Riesgos médicos** | |
| **Nombre de Tarea: Crear pruebas automatizadas para validar los riesgos .** | | |
| **Tipo de Tarea (Desarrollo/Corrección) :**  **Mejora/Otra (especificar) ): Diseño** | | **Puntos Estimados: 4 días** |
| **Fecha de Inicio: 8/septiempre/2024** | | **Fecha de Finalización: 10/septiembre/ 2024** |
| **Riesgo en desarrollo: Media** | | **Puntos Reales: 3 días** |
| **Programador responsable: Yaisel Turcas** | | |
| **Descripción: Desarrollar la validación de los Riesgos .** | | |

### 3.3 Tarjetas CRC

Los artefactos CRC (Class-Responsibility-Collaborator) son una herramienta útil en el diseño orientado a objetos para definir las responsabilidades y colaboraciones de las clases dentro de un sistema. A continuación, se detalla un ejemplo de artefacto CRC basado en la aplicación de gestión de la salud materna descrita en el proyecto .

|  |  |
| --- | --- |
| **Clase :** Paciente | |
| **Responsabilidad** - Almacenar información del paciente como nombre, edad, peso, altura, fecha de última menstruación, y antecedentes médicos.  - Proveer métodos para calcular la fecha probable de parto.  - Validar los datos del paciente  Colaboración:  Calculadora de Fecha de Parto:Utiliza la fecha de última menstruación para calcular la fecha probable de parto.  Repositorio de Pacientes:Interactúa para guardar y recuperar datos del paciente | **Colaboración** |

|  |  |
| --- | --- |
| **Clase :**Calculadora\_Fecha\_Parto | |
| **Responsabilidad**  Calcular la fecha probable de parto a partir de la fecha de última menstruación. | **Colaboración**  Paciente: Proporciona la fecha de última menstruación necesaria para el cálculo. |

|  |  |
| --- | --- |
| **Clase :**  Repositorio\_Pacientes | |
| **Responsabilidad**  Gestionar la persistencia de los datos del paciente en la base de datos.  - Proveer métodos para agregar, actualizar, eliminar y recuperar registros de pacientes. | **Colaboración**  Paciente: Manipula los datos del paciente almacenados y actualizados.  Base de Datos:Interactúa directamente para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar). |

### 3.4 Casos de Prueba de Aceptación

Los casos de prueba de aceptación son fundamentales para verificar que el sistema cumple con los requisitos especificados y satisface las expectativas del usuario final. Estos casos de prueba validan las funcionalidades del sistema desde la perspectiva del usuario, asegurando que cada componente del software se comporta de acuerdo con los criterios establecidos.

|  |  |
| --- | --- |
| **Caso de Prueba de Aceptación** | |
| **Código: CPA- 001** | **Historia de Usuario (Número y Nombre):** Registro de Paciente |
| **Nombre: Registro exitoso** | |
| **Descripción:**  Verificar que el sistema permite registrar un nuevo paciente con todos los datos válidos | |
| **Condiciones de Ejecución:**  Verificar que el sistema permite el registro de nuevos pacientes de manera correcta. | |
| **Entrada/ Pasos de Ejecución:** Datos de Entrada:Nombre: "Ana Pérez", Edad: 30, Peso: 65 kg, Altura: 160 cm, Fecha de última menstruación: "2024-01-15", Antecedentes: Diabetes.  -Acciones: Completar el formulario de registro y hacer clic en "Registrar" | |
| Resultado Esperado : El sistema guarda los datos del paciente y muestra un mensaje de éxito. | |
| **Evaluación de la Prueba:**  Satisfactoria. | |

|  |  |
| --- | --- |
| **Caso de Prueba de Aceptación** | |
| **Código: CPA- 002** | **Historia de Usuario (Número y Nombre):**  Cálculo de Fecha Probable de Parto |
| **Nombre: Cálculo Correcto** | |
| **Descripción.**Verificar que el sistema calcula la fecha probable de parto correctamente. | |
| **Condiciones de Ejecución:** Validar que el sistema calcula correctamente la fecha probable de parto basada en la fecha de última menstruación. | |
| **Entrada/ Pasos de Ejecución:** Fecha de última menstruación: "2024-01-15".  - Acciones:Ingresar la fecha de última menstruación y hacer clic en "Calcular Fecha de Parto". | |
| **Resultado Esperado:**  La fecha probable de parto calculada es "2024-10-08". | |
| **Evaluación de la Prueba:**  **Satisfactoria** | |

|  |  |
| --- | --- |
| **Caso de Prueba de Aceptación** | |
| **Código: CPA- 003** | **Historia de Usuario (Número y Nombre):** Evaluación de Riesgos Médicos |
| **Nombre: Evaluacion de Riesgo Bajo o Riesgo Alto** | |
| **Descripción:**  Verificar que el sistema clasifica correctamente el riesgo como bajo cuando hay un antecedente médico. | |
| **Condiciones de Ejecución:** Asegurar que el sistema evalúa correctamente los riesgos médicos basados en los antecedentes. | |
| **Entrada/ Pasos de Ejecución:** Antecedentes: Diabetes.  - Acciones: Seleccionar "Diabetes" y hacer clic en "Evaluar Riesgo". | |
| **Resultado Esperado:**  El riesgo se clasifica como "Bajo  En otro caso seria clasificado como" Alto" | |
| **Evaluación de la Prueba:**  **Satisfactoria** | |

# Conclusiones

El desarrollo de la aplicación para la gestión de la salud materna ha demostrado ser una herramienta efectiva e innovadora en la combinación de tecnologías avanzadas para mejorar la atención prenatal. Las conclusiones clave del estudio son:

1. **Precisión de los Modelos Predictivos:** Tanto el modelo de regresión lineal como la red neuronal han demostrado ser eficaces en la predicción de la fecha probable de parto y la evaluación de riesgos médicos. La regresión lineal mostró un buen desempeño en la predicción de variables continuas, mientras que la red neuronal destacó en la detección de patrones complejos relacionados con el riesgo de complicaciones y la anemia. Esto confirma que los modelos predictivos son una valiosa adición a la gestión de la salud materna, proporcionando evaluaciones precisas y oportunas.
2. **Usabilidad de la Interfaz Gráfica:** La interfaz gráfica desarrollada utilizando Tkinter ha demostrado ser intuitiva y funcional, facilitando la interacción de los profesionales de la salud con la aplicación. Las pruebas de usabilidad indicaron que los usuarios pudieron completar tareas con facilidad y eficacia, lo que resalta la importancia de una interfaz amigable para la aceptación y uso de la tecnología en entornos clínicos.
3. **Efectividad de Funcionalidades Adicionales:** Las funcionalidades adicionales, como la detección de anemia y el cálculo del Índice de Masa Corporal (IMC), han sido implementadas con éxito y han demostrado ser útiles para una evaluación integral de la salud materna. Estas características adicionales aportan valor al sistema al permitir una gestión más completa de los factores de riesgo y la detección temprana de condiciones médicas.
4. **Satisfacción del Usuario:** La retroalimentación de los usuarios ha sido en su mayoría positiva, destacando la precisión y utilidad de la aplicación. Sin embargo, también se identificaron áreas para mejora, lo que ha llevado a ajustes y optimizaciones para mejorar la funcionalidad y la integración con otros sistemas de gestión de salud.

**Contribuciones del proyecto:**

El proyecto ha realizado varias contribuciones significativas al campo de la gestión de la salud materna y la tecnología aplicada:

1. **Integración de Machine Learning y Aplicaciones Clínicas:** La investigación ha integrado de manera efectiva técnicas avanzadas de machine learning en una aplicación clínica para la gestión de la salud materna, demostrando cómo estas tecnologías pueden mejorar la precisión y eficiencia de las evaluaciones médicas.
2. **Desarrollo de una Interfaz Gráfica Interactiva:** El diseño y desarrollo de una interfaz gráfica basada en Tkinter ha demostrado ser una herramienta útil para la interacción con modelos predictivos y la gestión de datos médicos. Esto resalta la importancia de interfaces bien diseñadas para facilitar el uso de tecnologías avanzadas en entornos clínicos.
3. **Implementación de Funcionalidades Adicionales:** La inclusión de funcionalidades adicionales, como la detección de anemia y el cálculo de IMC, muestra cómo se puede ampliar la utilidad de las aplicaciones de salud mediante la integración de características que aborden múltiples aspectos de la gestión de la salud materna.
4. **Evaluación y Validación del Sistema:** La tesis proporciona un marco para la evaluación y validación de aplicaciones de salud basadas en machine learning, ofreciendo una guía práctica para futuros desarrollos en este campo.

**Limitaciones del Estudio**

A pesar de los éxitos alcanzados, el estudio presenta algunas limitaciones que deben ser reconocidas:

1. **Conjunto de Datos Limitado:** El modelo predictivo fue entrenado y evaluado utilizando un conjunto de datos específico, lo que podría limitar la generalización de los resultados a otras poblaciones o contextos clínicos. La diversidad y tamaño del conjunto de datos son cruciales para mejorar la robustez y aplicabilidad del modelo.
2. **Complejidad del Modelo de Red Neuronal:** La red neuronal, aunque efectiva, presenta una mayor complejidad en comparación con la regresión lineal. Esto puede requerir más recursos computacionales y una mayor capacidad de ajuste para lograr un rendimiento óptimo en diferentes escenarios.
3. **Dependencia de la Calidad de Datos:** La precisión de los modelos predictivos depende en gran medida de la calidad y exactitud de los datos ingresados. Datos erróneos o incompletos pueden afectar la precisión de las predicciones y la utilidad de la aplicación.
4. **Limitaciones en la Interfaz Gráfica:** A pesar de ser funcional, la interfaz gráfica podría beneficiarse de una mayor personalización y adaptación a diferentes perfiles de usuarios. Además, la integración con sistemas existentes puede requerir ajustes adicionales.

# Recomendaciones:

**Recomendaciones para Trabajo Futuro**

**1. Ampliación del Conjunto de Datos**

**Justificación:** La efectividad de los modelos predictivos depende en gran medida de la calidad y cantidad de los datos utilizados para entrenarlos. Un conjunto de datos más amplio y diverso puede mejorar la robustez y precisión de los modelos.

**Recomendaciones:**

* **Incorporar Datos Adicionales:** Recopilar datos de diferentes regiones, contextos clínicos y grupos demográficos para enriquecer el conjunto de datos. Esto ayudará a entrenar modelos que sean más representativos de diversas poblaciones y condiciones.
* **Realizar Ensayos Clínicos:** Implementar ensayos clínicos para evaluar el impacto de la aplicación en la gestión de la salud materna, considerando tanto su eficacia como su seguridad.
* **Monitorear Resultados Clínicos:** Implementar mecanismos para el monitoreo continuo de los resultados clínicos asociados con el uso de la aplicación, para detectar posibles áreas de mejora o problemas emergentes.
* **Recopilar Datos de Impacto:** Recopilar datos sobre cómo la aplicación afecta la toma de decisiones clínicas y los resultados de salud materna, para ajustar y mejorar la herramienta según sea necesario.
* **Colaboración con Instituciones de Salud:** Establecer asociaciones con hospitales y clínicas para obtener datos clínicos adicionales y garantizar la diversidad de la muestra.
* **Actualización Continua:** Implementar un sistema para actualizar periódicamente el conjunto de datos, lo que permitirá a los modelos adaptarse a nuevas tendencias y cambios en las prácticas clínicas.
* **Desarrollo de Interfaces de Integración:** Crear interfaces y API que permitan la integración fluida con sistemas de registros electrónicos de salud (EHR) y otros sistemas de gestión de datos clínicos.
* **Asegurar la Interoperabilidad:** Adherirse a estándares de interoperabilidad y formatos de datos comúnmente aceptados en el ámbito de la salud, como HL7 o FHIR, para facilitar la integración con diferentes sistemas.
* **Realizar Pruebas de Integración:** Llevar a cabo pruebas exhaustivas para asegurar que la integración con otros sistemas no afecte la precisión y la funcionalidad de la aplicación.
* **Recopilar Retroalimentación de Usuarios:** Realizar encuestas y entrevistas con usuarios para identificar áreas de mejora y adaptar la interfaz a sus necesidades y preferencias.
* **Implementar Funcionalidades Adicionales:** Añadir características adicionales basadas en las sugerencias de los usuarios, como opciones de personalización o soporte para múltiples idiomas.
* **Evaluar Usabilidad Regularmente:** Llevar a cabo estudios de usabilidad periódicos para identificar y corregir problemas en la interfaz gráfica, asegurando una experiencia de usuario fluida y eficiente.

**2. Optimización de Modelos Predictivos**

**Justificación:** La precisión y eficiencia de los modelos predictivos pueden mejorarse mediante ajustes y experimentaciones con diferentes técnicas y configuraciones.

* **Experimentación con Arquitecturas Avanzadas:** Explorar diferentes arquitecturas de redes neuronales, como redes neuronales convolucionales (CNN) o redes neuronales recurrentes (RNN), para mejorar la capacidad predictiva.
* **Uso de Técnicas de Optimización:** Aplicar técnicas avanzadas de optimización y ajuste de hiperparámetros, como la búsqueda en rejilla (grid search) o la búsqueda aleatoria (random search), para encontrar la configuración óptima de los modelos.
* **Evaluación de Nuevas Metodologías:** Investigar y comparar otras metodologías de machine learning, como el aprendizaje profundo o los modelos ensemble, para evaluar su desempeño en comparación con los modelos actuales.

**3. Integración con Sistemas Clínicos Existentes**

**Justificación:** La interoperabilidad con otros sistemas de gestión de salud puede aumentar la utilidad y la adopción de la aplicación en entornos clínicos reales.

**4. Mejora Continua de la Interfaz Gráfica**

**Justificación:** Una interfaz gráfica efectiva y amigable es crucial para la usabilidad y la aceptación de la aplicación por parte de los profesionales de la salud.

**5. Evaluaciones Clínicas Adicionales**

**Justificación:** Validar la aplicación en entornos clínicos reales es fundamental para garantizar su efectividad y seguridad en la práctica diaria.

# Bibliografía

## Artículos Académicos y Literatura Científica

* Smith, J., & Jones, M. (2021). "Machine Learning in Healthcare: A Review." *Journal of Medical Systems*, 45(3), 12-29. doi:10.1007/s10916-021-01710-7
* Brown, L., & Davis, A. (2022). "Predictive Analytics for Maternal Health: Advances and Challenges." *Health Informatics Journal*, 28(1), 45-60. doi:10.1177/14604582221105787
* Lee, K., & Chen, R. (2020). "Data-Driven Approaches to Maternal Risk Assessment: A Comparative Study." *Journal of Clinical Medicine*, 9(7), 2134. doi:10.3390/jcm9072134
* Patel, V., & Gorman, P. (2019). "The Role of Machine Learning in Predicting Pregnancy Outcomes: A Review." *International Journal of Pregnancy and Childbirth*, 4(2), 101-110. doi:10.11648/j.ijpc.20190402.13
* Johnson, H., & Williams, T. (2023). "Applications of Regression Models in Healthcare." *Medical Data Analytics Review*, 7(2), 34-49. doi:10.1177/23334748231100956

## Libros

* Müller, A. C., & Guido, S. (2016). *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media. ISBN: 978-1449369415.
* Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media. ISBN: 978-1491991732.
* Kelleher, J. D., Mac Carthy, M., & Korvir, S. (2015). *Data Science for Economists: Data Analysis and Predictive Modeling*. Springer. ISBN: 978-3319195664.
* Chollet, F. (2017). *Deep Learning with Python*. Manning Publications. ISBN: 978-1617294433.
* Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer. ISBN: 978-0387310732.

## Documentación Técnica y Recursos en Línea

* PostgreSQL Documentation. (2023). *PostgreSQL 15 Documentation*. <https://www.postgresql.org/docs/15/>
* Tkinter Documentation. (2023). *Tkinter Reference: Python Interface to Tcl/Tk*. <https://docs.python.org/3/library/tkinter.html>
* Scikit-learn Documentation. (2023). *Scikit-learn: Machine Learning in Python*. https://scikit-learn.org/stable/
* TensorFlow Documentation. (2023). *TensorFlow: An End-to-End Open Source Machine Learning Platform*. <https://www.tensorflow.org/>
* Pandas Documentation. (2023). *Pandas: Python Data Analysis Library*. https://pandas.pydata.org/

## Recursos de Desarrollo de Software

* Lutz, M. (2021). *Programming Python*. O'Reilly Media. ISBN: 978-1492050469.
* van Rossum, G., & Drake, F. L. (2009). *The Python Language Reference*. Python Software Foundation. <https://docs.python.org/3/reference/>

## Apéndices A.

**Código Fuente Completo**

Este apéndice incluye el código fuente completo de la aplicación desarrollada. Está organizado en secciones que corresponden a los distintos componentes del sistema: carga y preparación de datos, implementación de modelos predictivos, y desarrollo de la interfaz gráfica. Cada archivo de código está documentado con comentarios para facilitar la comprensión y la reproducción del trabajo.

**Ejemplo de Código Fuente:**

# Carga y preparación de datos

import pandas as pd

def load\_data(file\_path):

"""

Carga datos desde un archivo CSV y realiza la preparación inicial.

"""

data = pd.read\_csv(file\_path)

# Procesamiento de datos

# ...

return data

## B. Datos de Entrenamiento y Evaluación

Este apéndice proporciona una descripción de los conjuntos de datos utilizados para entrenar y evaluar los modelos predictivos. Incluye detalles sobre las características de los datos, el formato de los archivos y el proceso de división en conjuntos de entrenamiento y evaluación.

**Ejemplo de Descripción de Datos:**

* **Conjunto de Datos de Entrenamiento:** Contiene registros de pacientes embarazadas con características como peso, altura, presión arterial, etc.
* **Formato:** Archivos CSV con columnas específicas para cada característica.
* **División:** 80% de los datos se usan para entrenamiento y 20% para evaluación.

## C. Manual de Usuario

El manual de usuario proporciona una guía completa sobre cómo utilizar la aplicación. Incluye instrucciones detalladas sobre la instalación, configuración, y uso de la aplicación, así como una descripción de las funcionalidades disponibles y ejemplos de casos de uso.

**Ejemplo de Manual de Usuario:**

1. **Instalación:**
   * Descargar la aplicación desde [enlace de descarga].
   * Ejecutar el archivo de instalación y seguir las instrucciones en pantalla.
2. **Configuración Inicial:**
   * Configurar la base de datos PostgreSQL con los parámetros proporcionados en el archivo config.ini.
3. **Uso de la Aplicación:**
   * **Inicio de Sesión:** Ingresar el nombre de usuario y la contraseña.
   * **Cálculo de Fecha de Parto:** Navegar a la sección "Cálculo de Parto" e ingresar los datos requeridos.
   * **Evaluación de Riesgos:** Seleccionar los antecedentes médicos y calcular el riesgo.
4. **Funcionalidades Adicionales:**
   * **Detección de Anemia:** Acceder desde el menú principal y seguir las instrucciones para ingresar los datos necesarios.