Universidad de Pamplona

Desarrollo de Orientado a Plataformas

Integrantes

Sebastian Oswaldo Velandia Botello -

Sebastian Perez Pacheco -

Jhon Steven Escobar Castro – 1127044232

Docente

Ing. Fanny Casadiego

Villa del Rosario

Norte de Santander

Junio del 2025

Índice General

Capítulo I

* 1. – El problema

La deserción estudiantil en educación superior representa un desafío crítico para las instituciones académicas, afectando la tasa de graduación y la eficiencia del sistema educativo. Factores como la situación económica, el nivel de preparación previa, la carga académica y la falta de acompañamiento pedagógico influyen en el abandono de los estudios. Ante esta problemática, se requiere una solución tecnológica que permita analizar grandes volúmenes de datos académicos y socioeconómicos para detectar patrones de deserción y tomar medidas preventivas eficaces.

* 1. – Objetivo General del Proyecto

Desarrollar un sistema distribuido que analice bases de datos académicas y socioeconómicas de estudiantes para identificar patrones de deserción y generar reportes que ayuden a la toma de decisiones en universidades.

Objetivos Específicos

* Diseñar la arquitectura del sistema distribuido para la recolección y análisis de datos.
* Implementar una base de datos que integre información académica y socioeconómica de los estudiantes.
* Desarrollar modelos predictivos de deserción utilizando técnicas de procesamiento paralelo y aprendizaje automático.
* Crear una interfaz gráfica para la visualización de reportes y tendencias de deserción.
* Evaluar y optimizar el desempeño del sistema mediante pruebas y ajustes iterativos.
  1. – Cronograma de Actividades

Tabla 1

*Tabla de cronograma de actividades*

|  |  |  |
| --- | --- | --- |
| Actividad | Fecha de inicio | Fecha de finalización |
| Revisión bibliográfica | 14 de marzo | 25 de marzo |
| Diseño de la arquitectura del sistema | 26 de marzo | 5 de abril |
| Desarrollo de la base de datos | 6 de abril | 16 de abril |
| Implementación de modelos predictivos | 17 de abril | 10 de mayo |
| Creación de la interfaz de usuario | 11 de mayo | 24 de mayo |
| Pruebas y ajustes del sistema | 25 de mayo | 10 de junio |
| Optimización final del sistema | 11 de junio | 21 de junio |
| Presentación de resultados | 22 de junio | 24 de junio |

**Nota:** Fechas de inicio y presentación de resultados

* 1. – La Empresa

A. Nombre

Eduplorex

B. Dirección y Teléfono

No cuenta con sede física.

+57 320 823 3679

C. Breve Reseña Histórica

Eduplorex es una empresa de educación en línea fundada en 2018 con el propósito de proporcionar formación tecnológica accesible y de alta calidad. Con una plataforma interactiva basada en inteligencia artificial, ha logrado capacitar a miles de estudiantes en áreas clave como desarrollo de software, ciencia de datos, ciberseguridad y computación en la nube. La compañía ha colaborado con universidades y empresas para cerrar la brecha de habilidades en tecnología y mejorar la empleabilidad de los profesionales del futuro.

D.- Resumen de la Actividad de la Empresa

Eduplorex ofrece cursos, certificaciones y programas especializados en tecnologías emergentes mediante un modelo de aprendizaje interactivo y basado en proyectos. Su plataforma utiliza análisis de datos y aprendizaje automático para personalizar la experiencia de aprendizaje, identificando patrones de desempeño de los estudiantes y proporcionando recomendaciones para mejorar su retención y éxito académico.

E.- Visión:

Ser la plataforma líder en educación tecnológica en línea, ofreciendo programas innovadores y personalizados que impulsen el desarrollo profesional de estudiantes y trabajadores en la era digital.

F.- Misión:

Transformar la educación mediante el uso de tecnología avanzada, ofreciendo acceso global a formación en habilidades digitales clave y reduciendo la brecha educativa en sectores tecnológicos.

G.- Organigrama (Con énfasis en el departamento de trabajo)

Capitulo II Metodología

2.1 Introducción

La metodología seleccionada para el desarrollo de este proyecto es el método de prototipado, el cual permite un desarrollo iterativo y progresivo del sistema distribuido para la identificación de patrones de deserción estudiantil. Esta metodología es ideal para proyectos en los que los requisitos pueden evolucionar, ya que facilita la retroalimentación constante de los usuarios y mejora la adaptabilidad del sistema.

El proceso de prototipado permite construir versiones funcionales iniciales del sistema, evaluar su desempeño y realizar mejoras antes de la implementación final. Esto asegura que el producto final cumpla con las expectativas de los usuarios y se adapte a sus necesidades.

2.2 Definición

El método de prototipado es un enfoque de desarrollo de software en el que se crea un modelo funcional del sistema antes de su construcción definitiva. Este prototipo es evaluado y refinado a través de múltiples iteraciones hasta alcanzar una versión óptima del sistema. Su objetivo es reducir riesgos, mejorar la comprensión de los requerimientos y garantizar la satisfacción del usuario final.

En el contexto de este proyecto, se desarrollarán prototipos del sistema distribuido para probar la integración de bases de datos, el rendimiento de los modelos predictivos y la funcionalidad de la interfaz gráfica.

2.3 Fases

El desarrollo del sistema seguirá las siguientes fases del método de prototipado:

1. Identificación de Requerimientos:

* Recolección de necesidades de las universidades sobre la detección de deserción estudiantil.
* Análisis de las fuentes de datos disponibles (bases de datos académicas y socioeconómicas).
* Definición de las funcionalidades clave del sistema.

1. Desarrollo del Primer Prototipo:

* Creación de una versión inicial del sistema con capacidades básicas de integración de datos y generación de reportes.
* Implementación de los primeros modelos de predicción de deserción.
* Diseño preliminar de la interfaz de usuario.

1. Evaluación y Retroalimentación:

* Pruebas con usuarios clave (administradores educativos y docentes).
* Recopilación de comentarios y detección de fallos en el prototipo inicial.
* Análisis del rendimiento de los modelos predictivos.

1. Refinamiento del Prototipo:

* Ajustes en la arquitectura del sistema para mejorar la escalabilidad.
* Optimización de algoritmos de predicción (K-Means y Regresión Logística).
* Mejoras en la interfaz de usuario y experiencia del usuario (UX/UI).

1. Implementación Final y Despliegue:

* Integración de todas las mejoras en una versión estable del sistema.
* Despliegue en servidores distribuidos para su uso en universidades.
* Documentación y capacitación para los usuarios finales.

2.4 Otros Conceptos

* Iteración: Cada ciclo de prueba y mejora del prototipo antes de su versión final.
* Validación de usuarios: Evaluación del sistema con usuarios reales para garantizar su funcionalidad y usabilidad.
* Escalabilidad: Capacidad del sistema para manejar grandes volúmenes de datos sin perder eficiencia.
* Aprendizaje Automático: Implementación de algoritmos predictivos para identificar patrones de deserción.
* Computación Distribuida: Uso de múltiples servidores para el procesamiento eficiente de datos académicos y socioeconómicos.

Diccionario de datos

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabla** | **Campo** | **Tipo** | **Descripción** |
| Estudiante | id | INT (PK) | Identificador único |
|  | nombre | VARCHAR | Nombre del estudiante |
|  | edad | INT | Edad del estudiante |
|  | sexo | VARCHAR(10) | Género |
|  | programa | VARCHAR | Programa académico |
|  | nivel\_socioeconomico | INT | Nivel de 1 a 6 |
| Calificación | id | INT (PK) | ID nota |
|  | id\_estudiante | INT (FK) | Relación con estudiante |
|  | materia | VARCHAR | Nombre de la materia |
|  | nota | FLOAT | Nota final |
|  | semestre | INT | Semestre cursado |
| Encuesta | id | INT (PK) | ID encuesta |
|  | id\_estudiante | INT (FK) | Relación con estudiante |
|  | trabaja | BOOLEAN | Si trabaja |
|  | vive\_con | VARCHAR | Con quién vive |
|  | tiene\_beca | BOOLEAN | Si tiene beca |
|  | discapacidad | BOOLEAN | Si presenta discapacidad |
| Resultado | id | INT (PK) | ID resultado |
|  | id\_estudiante | INT (FK) | Estudiante relacionado |
|  | probabilidad\_desercion | FLOAT | Valor entre 0 y 1 |
|  | riesgo | VARCHAR | Bajo, Medio o Alto |

Diagrama de usuarios

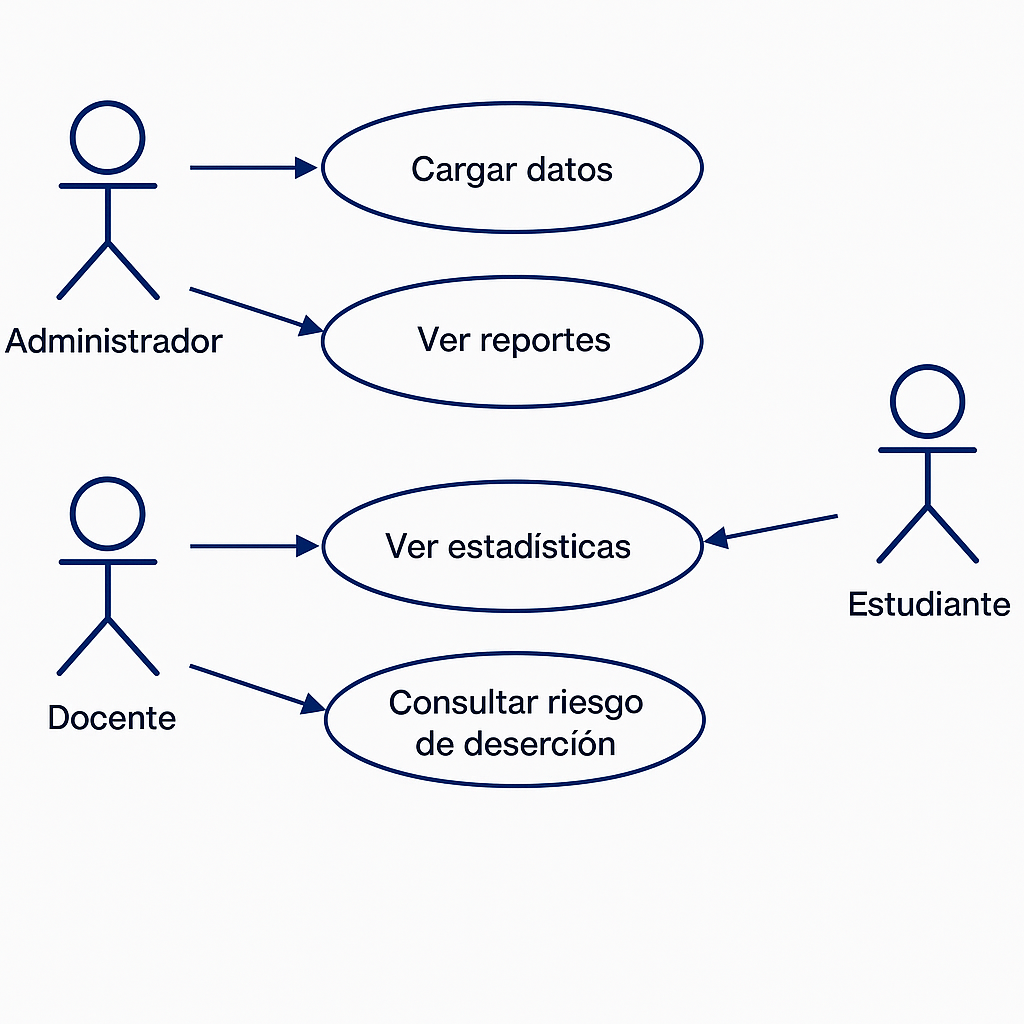
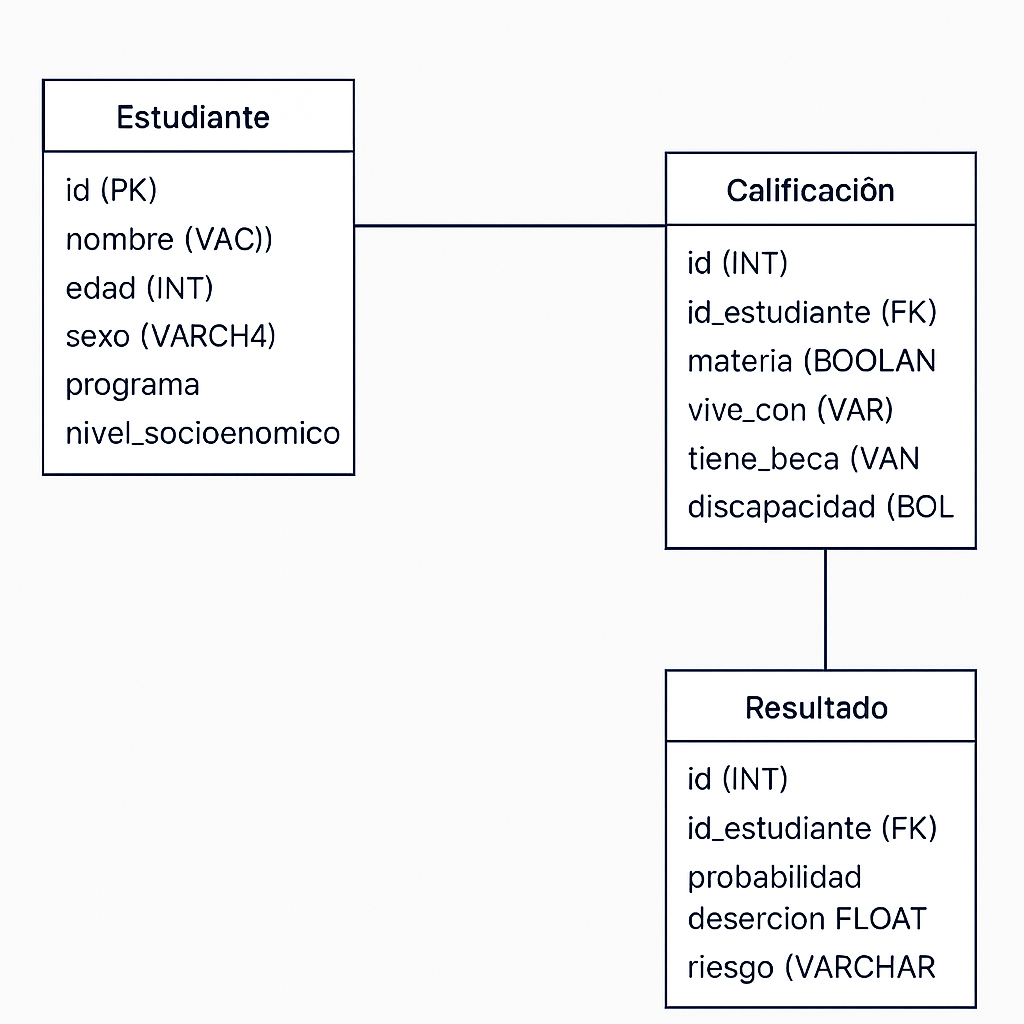


Diagrama Entidad-Relacion



Script Base de Datos

CREATE TABLE Estudiante (

id SERIAL PRIMARY KEY,

nombre VARCHAR(100),

edad INT,

sexo VARCHAR(10),

programa VARCHAR(100),

nivel\_socioeconomico INT

);

CREATE TABLE Calificacion (

id SERIAL PRIMARY KEY,

id\_estudiante INT REFERENCES Estudiante(id),

materia VARCHAR(100),

nota FLOAT,

semestre INT

);

CREATE TABLE Encuesta (

id SERIAL PRIMARY KEY,

id\_estudiante INT REFERENCES Estudiante(id),

trabaja BOOLEAN,

vive\_con VARCHAR(100),

tiene\_beca BOOLEAN,

discapacidad BOOLEAN

);

CREATE TABLE Resultado (

id SERIAL PRIMARY KEY,

id\_estudiante INT REFERENCES Estudiante(id),

probabilidad\_desercion FLOAT,

riesgo VARCHAR(10)

);

Logica Backend

OBJETIVO DEL BACKEND

* Analizar los datos de los estudiantes y generar dos tipos de gráficos:
* Gráficos de dispersión con clustering (KMeans): para identificar agrupaciones de estudiantes según variables clave.
* Gráficos de barras categóricas: para visualizar de forma más clara y legible la relación entre variables cualitativas (ej. razón de retiro, situación económica, estado actual).
* Ambos se devuelven al frontend como imágenes codificadas en base64.

1. Se recopilan los datos desde la base de datos con:

queryset = Estudiante.objects.all()

Se obtienen todos los registros del modelo Estudiante.

1. Se convierten en un DataFrame de Pandas

Esto permite aplicar análisis y procesamiento de datos fácilmente:

data = pd.DataFrame(list(queryset.values()))

1. Se codifican las variables categóricas

Como KMeans y LogisticRegression solo aceptan números, convertimos campos como:

* "estado" → estado\_binario (0 para estudiando, 1 para retirado)
* "situacion\_economica" → economica\_cod (0 baja, 1 media, 2 buena)
* "discapacidad\_o\_enfermedad" y "razon\_retiro" → códigos con .cat.codes

1. Se define la matriz X e Y

X = todas las características que pueden influir en la deserción (edad, semestre, criminal, economica\_cod, etc.).

y = el estado binario (estudiando o retirado).

1. Se aplica KMeans Clustering

Para crear agrupaciones de estudiantes con características similares:

kmeans = KMeans(n\_clusters=2, n\_init=10, random\_state=0)

clusters = kmeans.fit\_predict(X)

1. Se entrena una regresión logística

Esto permite identificar qué variables influyen más en si alguien se retira o no:

model = LogisticRegression()

model.fit(X, y)

(Este modelo no se visualiza directamente, pero se puede usar para análisis posterior o predicciones.)

7. Se generan gráficos

1. Dispersión con KMeans:

Se grafican relaciones como:

* Edad vs Semestre
* Edad vs Economía
* Enfermedad vs Razón de retiro

Usando:

plt.scatter(..., c=clusters, cmap='plasma')

2. Barras categóricas:

Usamos seaborn.countplot() para graficar:

* Razón de retiro vs Estado
* Situación económica vs Estado

8. Se convierten los gráficos a base64

Para enviarlos como JSON al frontend y mostrarlos como imágenes:

grafico\_clusters\_base64 = base64.b64encode(buffer1.read()).decode('utf-8')

9. Se devuelven los gráficos como respuesta JSON

return Response({

"mensaje": "Análisis generado exitosamente.",

"grafico\_clusters": ...,

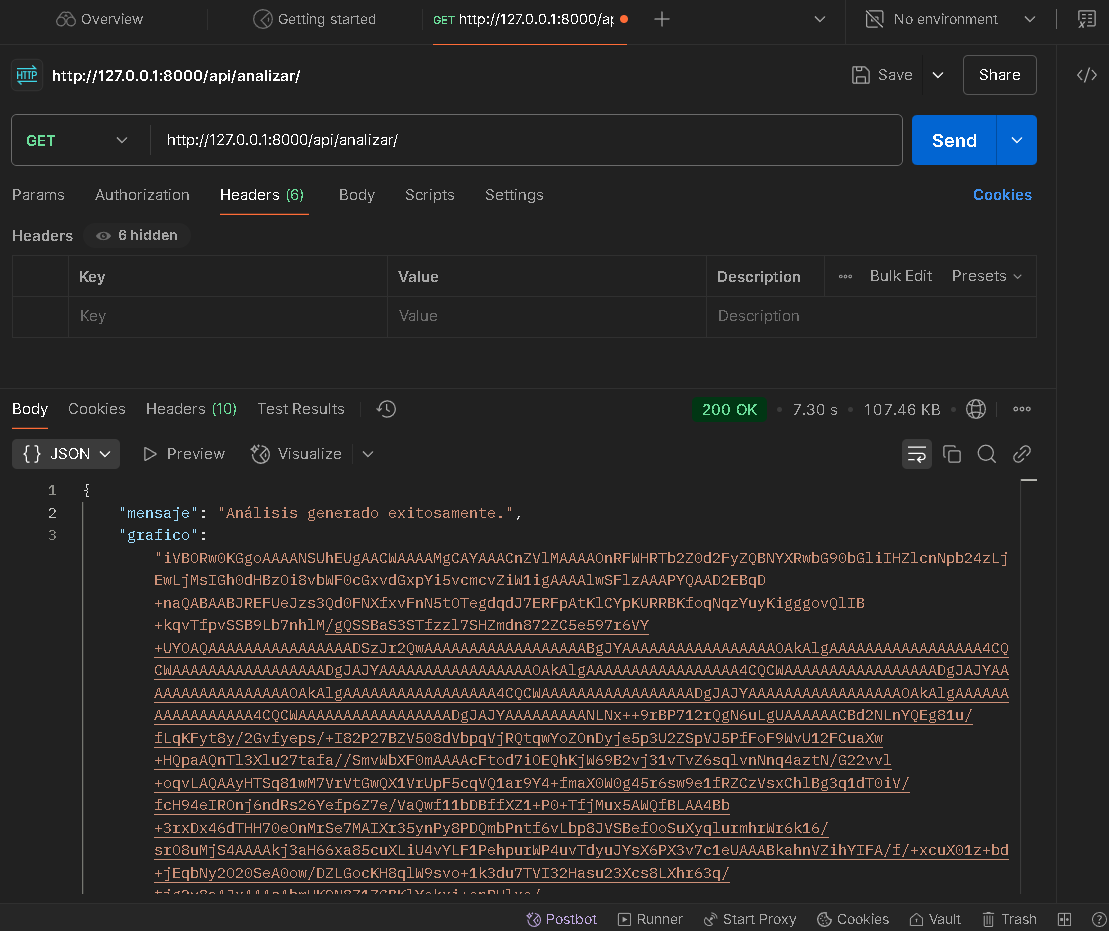
"grafico\_barras": ...

})

¿Qué puedes hacer con esto?

* Detectar patrones de deserción por grupos.
* Visualizar qué razones de retiro son más frecuentes.
* Detectar si hay más deserción en ciertos niveles económicos o de salud.

Post-Man Consumo de API



Optimizacion Front-end, Back-end y Base de Datos

Optimización en el Backend (Django)

1. Uso de pandas para procesar datos eficientemente

Utilizamos la librería panda para transformar y limpiar datos directamente desde la base de datos, lo que permite análisis vectorizado y reduce el uso de ciclos manuales.

2. Preprocesamiento de variables

Las variables categóricas fueron transformadas con .map() y .cat.codes, evitando condicionales repetitivos y facilitando el entrenamiento de los modelos de machine learning.

3. Reutilización de gráficos

Los gráficos generados se envían como imágenes codificadas en base64, lo que reduce el procesamiento y evita recalcular desde el frontend.

4. Control de errores y validaciones

Se valida que existan datos suficientes antes de entrenar modelos para evitar errores por bajo volumen de datos (if queryset.count() < 3:).

Optimización en el Frontend (React + Vite)

1. Carga asíncrona con Axios

Se implementó Axios para realizar peticiones GET al backend de forma eficiente y no bloqueante, optimizando la experiencia de usuario.

2. Separación de componentes

La visualización de gráficos se encapsuló en un componente (TrendChartPlaceholder.jsx) para mantener limpio el flujo de datos y facilitar el mantenimiento.

3. Visualización optimizada

Se emplearon gráficos de dispersión y de barras combinados, haciendo más legible la información para los usuarios y segmentando los datos por categorías relevantes.

4. Uso de imágenes base64

Los gráficos se integran como imágenes codificadas en base64, eliminando la necesidad de generar gráficos en tiempo real desde el navegador.

Optimización en la Base de Datos (SQLite)

1. Modelo de datos compacto y claro

El modelo Estudiante fue diseñado con campos precisos y relevantes, evitando redundancia y facilitando consultas específicas.

2. Carga de datos sintéticos automatizada

Se creó un script de carga (cargar\_datos.py) que genera datos aleatorios con lógica coherente, permitiendo pruebas sin necesidad de inserción manual.

3. Indexación implícita por clave primaria

Aunque SQLite maneja esto automáticamente, la definición de modelos aprovecha claves primarias (id) para consultas rápidas desde el backend.

Buenas prácticas generales

* Código organizado por responsabilidad (Modelos, Vistas, URLs).
* Evitamos lógica repetida con componentes reutilizables (React).
* Envío estructurado de respuesta JSON para permitir escalabilidad futura.