Detector de SPAM

Desarrollo de Proyectos en IA —Universidad Autónoma de Occidente

Diego Salcedo Valencia

Oscar Arcos Zambrano

Nicolas Rubio Aponte

# Tabla de contenido

1. Contexto del proyecto
2. Objetivos y alcance
3. Cronograma, ruta crítica e hitos
4. Tablero Kanban (GitHub Projects) y tickets sugeridos
5. Marco de investigación (modelos considerados)
6. Datasets a utilizar
7. Técnicas, frameworks y librerías
8. Requisitos y restricciones de software
9. Entregables
10. Riesgos y mitigación
11. Referencias

# 1. Contexto del proyecto

Construiremos un sistema de detección de SPAM en mensajes SMS. El sistema clasificará cada SMS como SPAM o HAM (no spam). Emplearemos un enfoque comparativo entre un baseline clásico (TF‑IDF + Regresión Logística) y un modelo Transformer ligero (DistilBERT) afinado para clasificación binaria.

Modelo propuesto (Hugging Face): distilbert-base-uncased  
Enlace: https://huggingface.co/distilbert-base-uncased

Qué hace el modelo: Recibe un texto y devuelve una probabilidad por clase; se usará con una capa de clasificación y entrenamiento supervisado sobre el conjunto de entrenamiento.

# 2. Objetivos y alcance

Objetivo general:

* Diseñar, entrenar/evaluar y desplegar un detector de SPAM en SMS con interfaz web y empaquetado en contenedor.

Objetivos específicos:

• Implementar un baseline (TF‑IDF + Regresión Logística) y persistirlo como .pkl.

• Afinar DistilBERT para la tarea y comparar vs baseline.

• Desarrollar una app en Streamlit para inferencia en tiempo real.

• Dockerizar la solución y documentar ejecución.

• Lograr F1 ≥ 0.95 en el conjunto de prueba.

* Alcance (in‑scope):

• Entrenamiento/evaluación con SMS Spam Collection (UCI).

• Comparación baseline vs DistilBERT y selección del mejor.

• Demo web (Streamlit) + contenedor Docker.

* Fuera de alcance (out‑of‑scope):

• Detección de phishing en URLs o adjuntos.

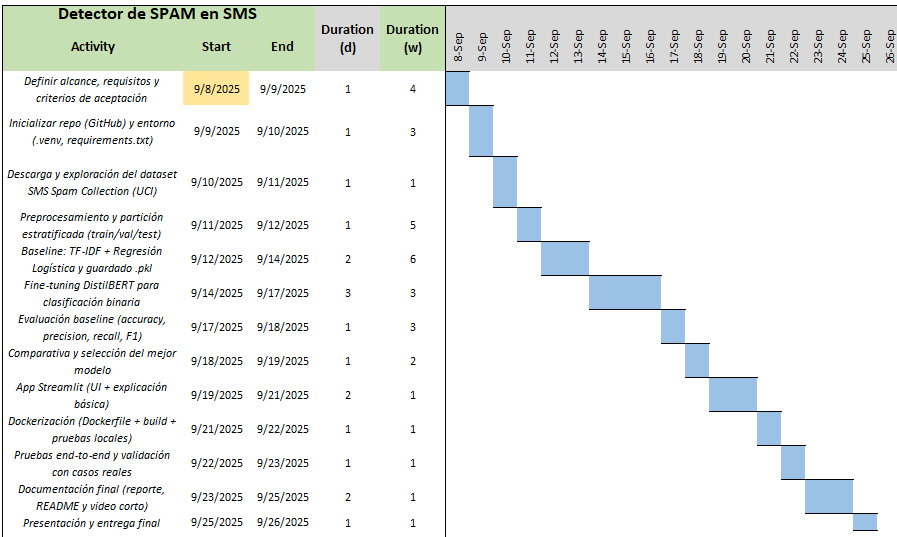
• Entrenamiento multimodal o despliegue en móvil.

# 3. Cronograma, ruta crítica e hitos

**Fecha de inicio planificada**: 2025-09-08

**Duración total estimada**: 15 días (calendario).

**Ruta crítica**: A, B, C, D, E, F, G, H, I, J, K, L, M

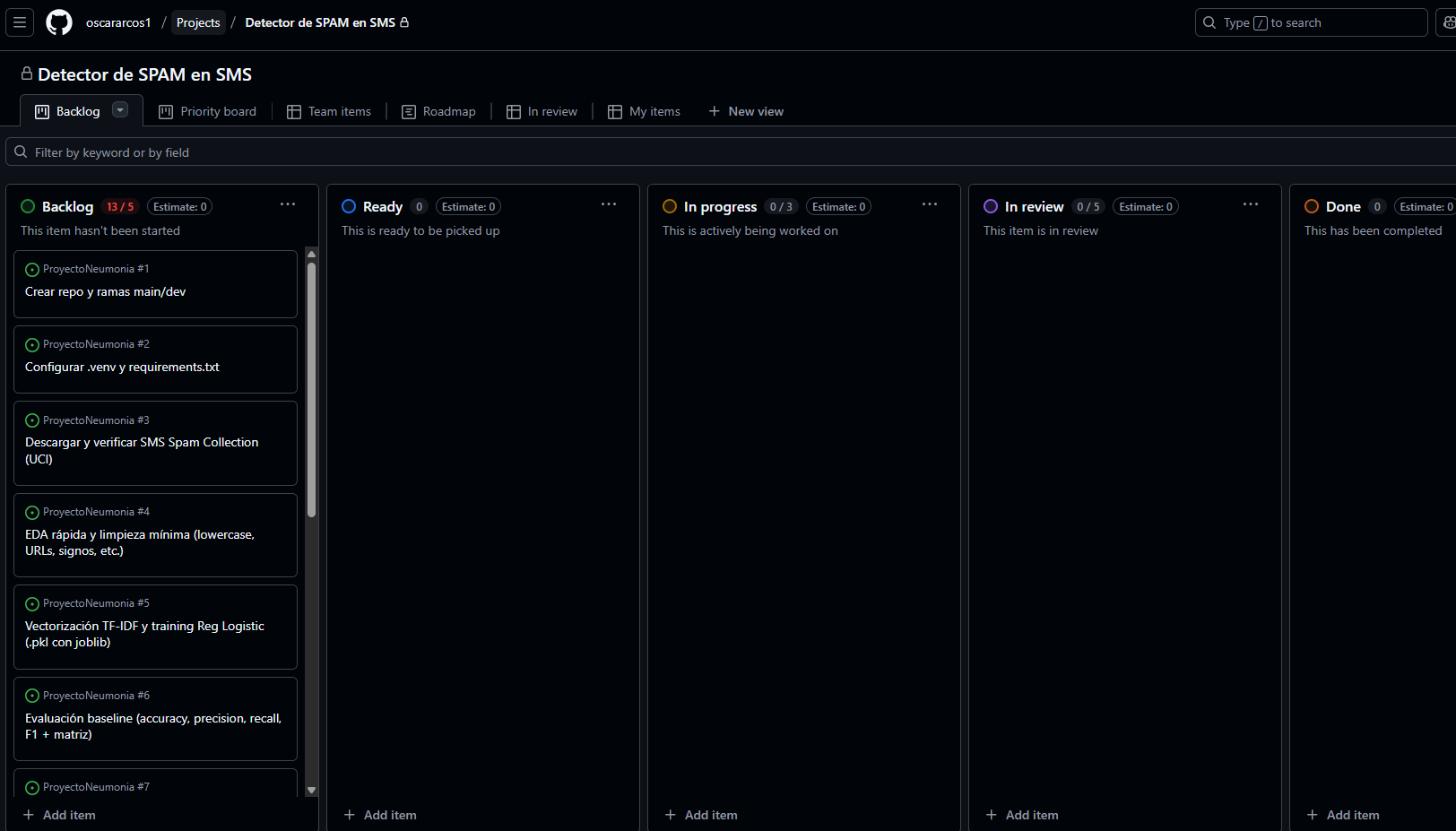


Hitos:

* Baseline entrenado y evaluado.
* DistilBERT ajustado.
* Modelo final seleccionado.
* App Streamlit funcional.
* Imagen Docker validada.
* Documentación completa.
* Presentación y entrega.

# 4. Tablero Kanban (GitHub Projects) y tickets sugeridos

El Kanban de Github está en <https://github.com/users/oscararcos1/projects/4/views/1>



# 5. Marco de investigación (modelos considerados)

• Baseline clásico: TF‑IDF + Regresión Logística (archivo .pkl).

• Deep Learning: DistilBERT (capa de clasificación) afinado para spam/ham.

• Alternativas: NB multinomial, SVM lineal o RoBERTa-base.

Criterios: F1, latencia en inferencia, tamaño del artefacto y facilidad de despliegue.

# 6. Datasets a utilizar

SMS Spam Collection (UCI) — Conjunto de mensajes SMS etiquetados como spam o ham (no spam). Formato .csv/tsv con columnas 'label' y 'message'.

UCI Machine Learning Repository: https://archive.ics.uci.edu/ml/datasets/sms+spam+collection

# 7. Técnicas, frameworks y librerías

Técnicas: limpieza básica, TF‑IDF, Regresión Logística, fine‑tuning de Transformer, validación estratificada.

Frameworks: scikit‑learn, PyTorch, Hugging Face Transformers/Datasets, Streamlit, Docker.

Librerías: numpy, pandas, scikit‑learn, torch, transformers, datasets, accelerate, matplotlib, joblib.

# 8. Requisitos y restricciones de software

• Python 3.10+.

• PyTorch 2.1+ y Transformers 4.40+ (CPU basta para inferencia; GPU acelera el fine‑tuning).

• 8 GB RAM recomendados; 2–4 GB para inferencia simple.

• SO: Windows 10/11, macOS o Linux.

• Para Docker: Docker Desktop 4.x.

# 9. Entregables

• Código fuente (GitHub) con README y requisitos.

• Modelo baseline (.pkl) y checkpoint del Transformer (carpeta HF).

• App Streamlit funcional.

• Imagen Docker o instrucciones de build/run.

• Este documento de propuesta + reporte final de resultados.

# 10. Riesgos y mitigación

Riesgos: sobreajuste, sesgos del dataset, desbalance de clases, mensajes fuera de distribución.

Mitigación: estratificación, métricas por clase, threshold tuning, documentar limitaciones, evitar uso en producción sin evaluación adicional.

# 11. Referencias

- DistilBERT (model card) — https://huggingface.co/distilbert-base-uncased

- Transformers (instalación) — https://huggingface.co/docs/transformers/installation

- SMS Spam Collection (UCI) — https://archive.ics.uci.edu/ml/datasets/sms+spam+collection

- scikit-learn: Model Persistence — https://scikit-learn.org/stable/model\_persistence.html