# Título del reto

Análisis de vulnerabilidad de un asistente conversacional.

# Descripción del caso de uso

## Contexto

Desde Madrid Digital se está desarrollando un asistente conversacional basado en modelos de lenguaje (LLMs), con el objetivo de proporcionar a los ciudadanos información clara, accesible y actualizada sobre los principales trámites, gestiones y servicios de la Comunidad de Madrid.

El uso de tecnologías generativas permite ofrecer una atención más natural e inmediata a la ciudadanía. Sin embargo, este tipo de sistemas también presentan riesgos específicos asociados a su comportamiento abierto, como la manipulación conversacional, la desinformación o el uso indebido del lenguaje.

## Necesidad

Además de desarrollar un asistente funcional en español, es fundamental garantizar su seguridad. Para ello, el reto incluye una fase de Red Teaming, orientada a detectar vulnerabilidades mediante ataques simulados, y una de Blue Teaming, centrada en aplicar defensas que refuercen su comportamiento ante intentos de manipulación. El sistema debe garantizar:

* Mantener un comportamiento seguro y apropiado.
* No difundir información errónea, inapropiada o sensible.
* Ser robusto frente a intentos de manipulación.

# Propósito del Hackathon

El reto no se limita a construir un asistente conversacional funcional, sino también a garantizar su seguridad frente a usos indebidos.

Para ello, se plantean dos fases complementarias:

* **Red Teaming**: Simulación de ataques conversacionales para detectar vulnerabilidades, como respuestas inapropiadas, fuga de información o manipulación del asistente.
* **Blue Teaming**: Implementación de mecanismos de defensa que refuercen la seguridad y fiabilidad del sistema ante esos ataques.

El objetivo es desarrollar asistentes conversacionales robustos, seguros y responsables. Además, es fundamental comprender y explicar los resultados obtenidos, así como los comportamientos y decisiones del asistente, para garantizar su trazabilidad, fiabilidad y alineación con los objetivos del sistema.

# Herramientas recomendadas

A continuación, se listan las diferentes tecnologías y herramientas recomendadas, así como su propósito de uso.

* **Lenguaje de programación**: Python.
* **Frameworks para LLMs**, que pueden facilitar el desarrollo y despliegue de aplicaciones basados en estos modelos: LangChain, LlamaIndex.
* **Modelos de lenguaje**: Llama 3, Mistral, Phi-3, Hugging Face Transformers.
* **Frameworks API**: Flask, FastAPI.
* **Procesamiento de datos y librerías** ampliamente utilizadas en aplicaciones de inteligencia artificial: Pandas, NumPy, NLTK, spaCy, Scikit-Learn.
* **Bases de datos vectoriales**: ChromaDB, FAISS, LanceDB.
* **Control de versiones y repositorio** de evaluación: GitHub.
* **Entorno de desarrollo** **(IDE)**: Jupyter Notebooks, VS Code, NotebookLM, Google Colab.
* **Interfaces de usuario** (opcional): Streamlit, Gradio.
* **Generación de imágenes** (opcional): Stable Diffusion, Kandinsky.

**\*\* Nota:** Se recomienda asistir al evento con el entorno de desarrollo ya instalado y configurado.

1. Antes de comenzar, asegúrate de tener instalado:

* **Git**
* **Python 3.9+**
* **Editor de código**: se recomienda Visual Studio Code con las extensiones: Python y Jupyter
* **Cuenta GitHub**
* **Conda o venv**

1. Configuración del entorno virtual:

|  |
| --- |
| # Crear un entorno virtual  python -m venv .venv  # Activar entorno (Windows)  .venv\Scripts\activate  # Activar entorno (Mac/Linux)  source .venv/bin/activate |

# Logros

## Logro 1: Creación del asistente conversacional básico

### Propósito

Desarrollar un asistente conversacional básico, basado en un modelo de lenguaje, capaz de mantener una conversación sencilla y coherente con el usuario. Puede ser de propósito general o estar centrado en una temática concreta.

Se trata de construir un sistema funcional pero poco complejo, que sirva como base para las siguientes fases del reto, aunque funcionalidades como la generación de imágenes o una interfaz de usuario web se valorarán positivamente. Sí es obligatorio que el asistente sea capaz de mantener una conversación básica (3–4 turnos), con un propósito definido. Puede ser un asistente conversacional genérico, aunque se valorará positivamente aquel que se especialice en un tema concreto, utilizando técnicas de RAG (Retrieval-Augmented Generation).

### Entregable

Un prototipo funcional del asistente conversacional que incluya:

* **Código ejecutable** accesible desde el repositorio proporcionado.
* **Interfaz mínima** **resultante** del propio desarrollo (por ejemplo, línea de comandos o API simple).
* **Documentación técnica** del diseño, la arquitectura y las decisiones tomadas.
* **Explicación de los resultados** obtenidos y del comportamiento del asistente durante las pruebas básicas de conversación.

### Información de partida

Se acompañará con un ejemplo del asistente conversacional institucional que está siendo desarrollado desde Madrid Digital, que podrá servir como inspiración o referencia para los equipos participantes.

### **Criterios de evaluación**

Aspectos a valorar:

* **Calidad de la conversación**: coherencia, naturalidad, mantenimiento del contexto.
* **Implementación Técnica y Arquitectura**: calidad del código, uso efectivo de LLMs y su memoria, robustez.
* **Documentación**: tanto del diseño y desarrollo como de resultados y respuestas obtenidos.
* **Innovación y características adicionales**: generación de imágenes, interfaz avanzada, sistema RAG de información.

## Logro 2: Análisis de seguridad del asistente (Red Team)

### Propósito

Forzar el fallo del asistente conversacional, es decir, encontrar sus límites, inducir errores y poner a prueba su comportamiento ante interacciones maliciosas o manipuladas.

Se busca simular ataques realistas para detectar vulnerabilidades que puedan comprometer su utilidad, su seguridad o su alineación con los objetivos del sistema.

Los tipos de ataques más comunes que pueden aplicarse incluyen:

* **Generación de contenido inapropiado o sesgado**: inducir al asistente a emitir respuestas ofensivas, discriminatorias, éticamente dudosas o parcializadas.
* **Jailbreaking / manipulación de identidad**: lograr que el asistente ignore sus instrucciones originales (por ejemplo, actuar como otro personaje o revelar su rol interno).
* **Fuga de datos sensibles**: extraer información no destinada al usuario, como instrucciones internas, respuestas en modo debug o contenido de otras conversaciones.
* **Prompt injection**: insertar instrucciones ocultas o contradictorias dentro del input del usuario para alterar el comportamiento del modelo.
* **Invención de hechos (alucinaciones)**: provocar que el asistente afirme información incorrecta como si fuera cierta.
* **Rechazo injustificado de respuestas válidas**: hacer que el asistente eluda preguntas legítimas por una interpretación errónea de sus restricciones. Este proceso debe estar guiado por una estrategia clara, orientada a provocar fallos y documentar su aparición de forma detallada.

### Entregables

Un análisis documentado del comportamiento del asistente frente a entradas maliciosas, que incluya:

* **Código ejecutable** (en el mismo repositorio del asistente) que permita replicar los ataques realizados.
* **Documentación técnica con la estrategia seguida**: justificación de los casos probados, registro de los ataques aplicados (prompts utilizados).
* **Explicación de resultados obtenidos**: análisis de vulnerabilidades y comportamientos inesperados, así como su impacto potencial en el funcionamiento y fiabilidad del asistente.

### Información de partida

Se proporcionará una breve explicación de los tipos de ataques conversacionales más comunes, junto con una pequeña selección de ejemplos de prompts que han conseguido inducir fallos en el asistente desarrollado desde Madrid Digital. Estos materiales servirán como referencia para que los equipos comprendan el enfoque del Red Teaming y diseñen sus propios casos de prueba.

### Criterios de evaluación

En esta fase se valorará:

* **Número y severidad de vulnerabilidades** válidas identificadas.
* **Documentación de la estrategia de ataque** propuesta.
* **Documentación de los ataques realizados**, así como las vulnerabilidades detectadas.

## Logro 3: Refuerzo del asistente conversacional (Blue Team)

### Propósito

Diseñar e implementar mecanismos de protección que mitiguen o neutralicen los fallos detectados en la fase anterior, o nuevos que hayan podido surgir. El objetivo es reforzar la seguridad y estabilidad del asistente, evitando que repita los comportamientos problemáticos identificados durante el Red Teaming.

Los participantes podrán emplear diversas estrategias defensivas, como, por ejemplo:

* **Filtrado de entradas para bloquear instrucciones maliciosas o sensibles, y así prevenir instrucciones maliciosas y fuera de contexto**: una posible estrategia que se podría adoptar es el uso de “guardarraíles” (guardrails), la cual permite mantener al modelo dentro de límites seguros, éticos y coherentes.
* **Validación o reformulación de respuestas** antes de enviarlas al usuario.
* **Refuerzo o reajuste del prompt** base para delimitar mejor el comportamiento del modelo: una opción es incluir ejemplos de respuestas correctas e incorrectas frente a posibles ataques. De esta manera, el modelo recibe información adicional que le permite anticipar y defenderse mejor ante estos ataques.
* **Implementación de lógica defensiva externa**: como capas de control previas o posteriores al modelo.
* **Red Teaming inverso**: aplicando técnicas automáticas para generar ataques simulados y poner a prueba las defensas del sistema de forma continua. Un punto de partida eficaz sería defenderse ante los prompts específicos utilizados en la fase anterior.

### Entregable

Los entregables que se esperan para validar este logro son:

* **Código ejecutable** (en el mismo repositorio del asistente) que permita probar los mecanismos de protección implementados.
* **Documentación técnica con la estrategia defensiva seguida**: justificación de las soluciones aplicadas, descripción de los mecanismos de protección implementados.
* **Explicación de resultados obtenidos**: análisis del impacto de las defensas sobre los ataques previamente detectados, efectividad observada y, en su caso, limitaciones o mejoras pendientes.

### Información de partida

Se proporcionará una breve descripción de técnicas comunes de protección conversacional, así como ejemplos ilustrativos de defensas aplicadas sobre casos reales del asistente desarrollado desde Madrid Digital. Estos materiales servirán como referencia para que los equipos comprendan el enfoque del Blue Teaming y diseñen sus propios mecanismos de refuerzo.

### Criterios de evaluación

Se valorará:

* **Robustez de la seguridad**: detección y clasificación de prompt maliciosos.
* **Documentación de las estrategias defensivas adoptadas**: vulnerabilidades reforzadas, respuesta ante prompts detectados como maliciosos.
* **Documentación de las respuestas proporcionadas** ante ataques maliciosos tras implementar medidas de seguridad.