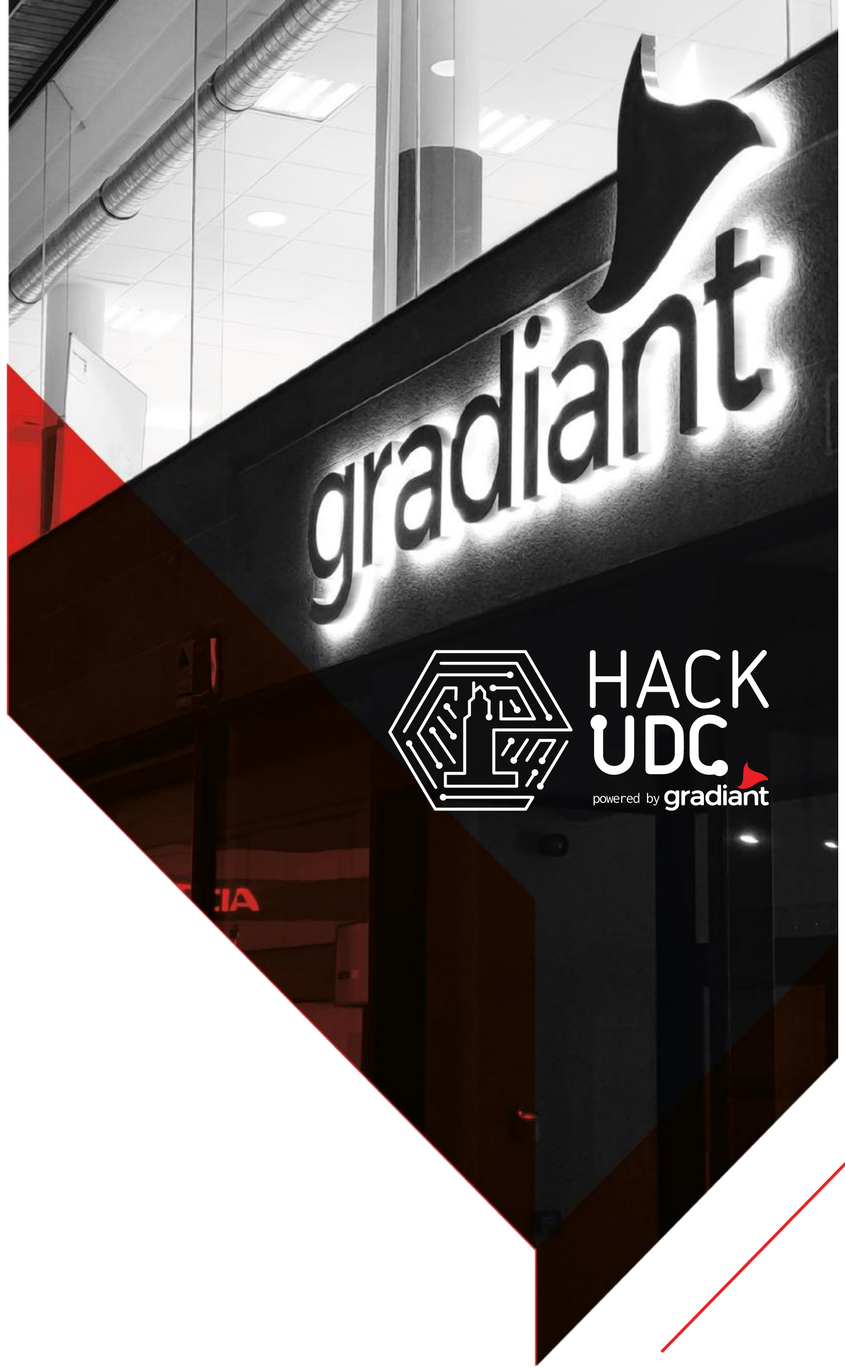
gradiant
Connectivity · Intelligence · Security



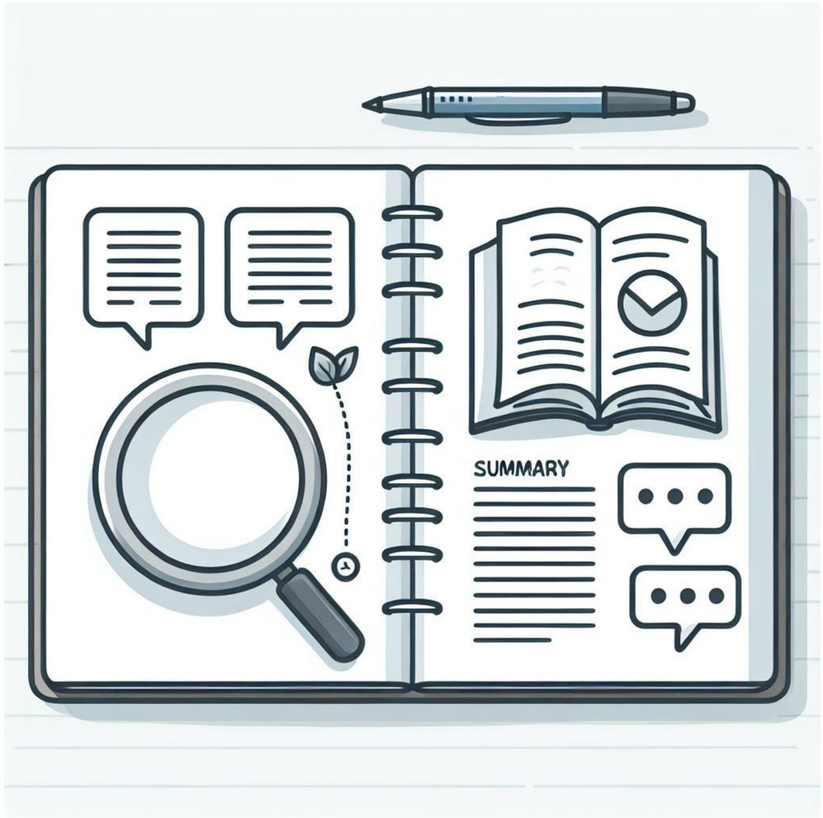
CompetenciApp

Potenciando tu aplicación con datos e IA

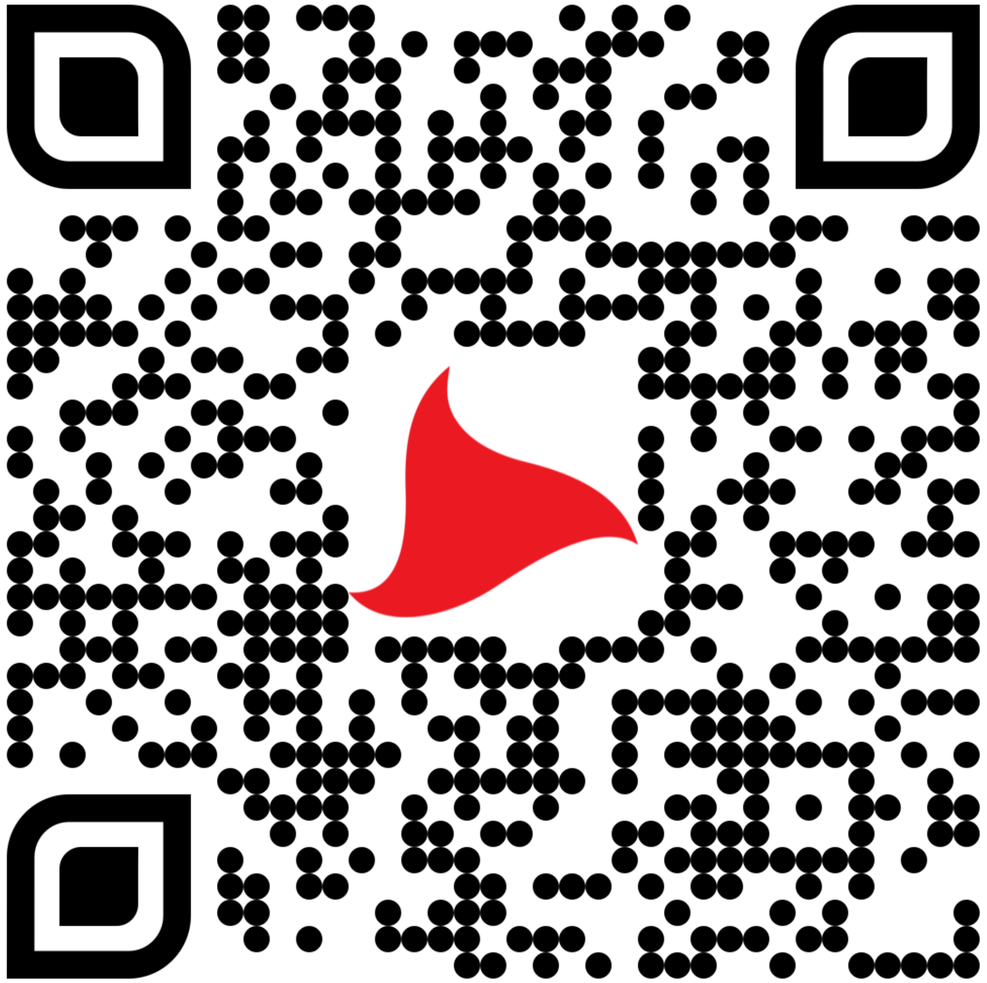
HackUDC 22 febrero 2025

Contenido

1. Ideas de mejora de CompetenciApp
2. Añadiendo IA
   - LLMs en local: Ollama
   - Plantillas de prompt
   - Datos estructurados
3. Nuevas fuentes de datos
   - Utilizando información de los repositorios de código
4. Dudas y consultas



Repositorio



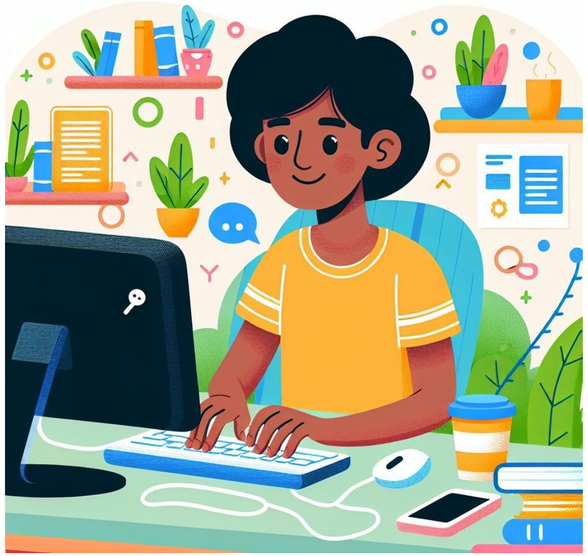
https://github.com/Gradiant/hackudc2025

Ideas de mejora de CompetenciApp (I)

Sistema de registro y consulta de competencias técnicas personales

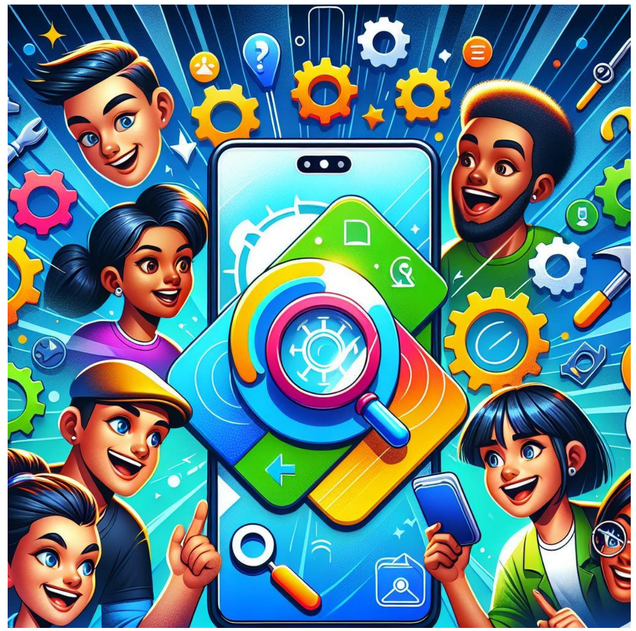
- Registrar lo que cada persona conoce
- Registrar lo que cada persona aprende
- Registrar los recursos con los que aprende
- Consultar qué personas saben de qué
- Consultar qué recursos tengo disponibles para aprender algo

- Sencillo de utilizar
- Rápido de utilizar
- Atractivo de utilizar



Ideas de mejora de CompetenciApp (II)

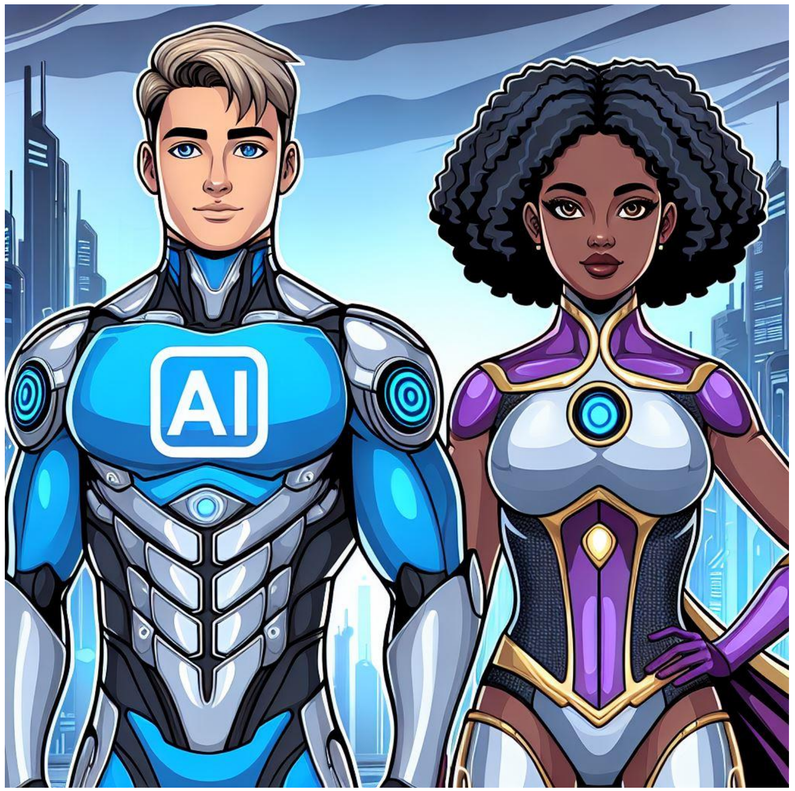
- Formulario con alta de ítems
- Reconocimiento de ítems similares
- Entrada en lenguaje natural
- Chatbot
- Jerarquía o grafo de dependencia de los conocimientos
- Ranking de conocimiento entre usuarios
- Recomendaciones de repaso
- Recomendación de recursos a otros usuarios



Añadiendo IA

- Interacción sencilla mediante lenguaje natural

- Ejemplos:
  - Simplificación de la entrada de datos
  - Ayuda con consultas complejas
  - Generación de respuestas elaboradas

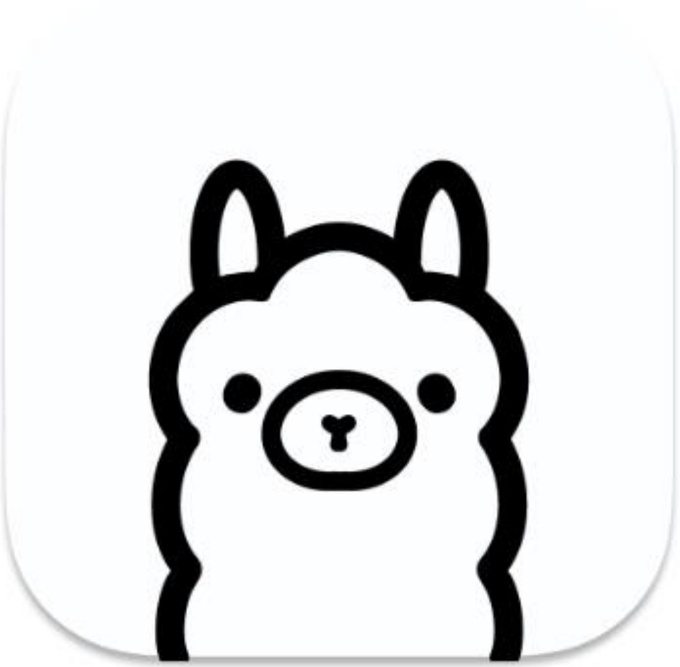


Añadiendo IA

+

Ollama

- Ollama - https://github.com/ollama/ollama
- Servidor de LLMs
  - Permite cargar diferentes modelos de lenguaje
  - REST API para acceso mediante HTTP
  - SDK
    - Python: https://github.com/ollama/ollama-python
    - JavaScript: https://github.com/ollama/ollama-js



Añadiendo IA

+

Ollama Run

```
HackUDC$> ollama serve
```

```
HackUDC$> ollama run llama3.2:1b
>>> 'Send a message (/? for help)'
```

Añadiendo IA

Ollama REST API: https://github.com/ollama/ollama/blob/main/docs/api.md

```
curl http://localhost:11434/api/chat -d '{
  "model": "llama3.2:1b",
  "messages": [
    {
      "role": "user",
      "content": "why is the sky blue?"
    }
  ],
  "stream": false
}'
```

```
import requests

url = 'http://localhost:11434/api/chat'

data = {
  "model": "llama3.2:1b",
  "messages": [
    {
      "role": "user",
      "content": "why is the sky blue?"
    }
  ],
  "stream": False
}

response = requests.post(url, json=data)
print(response.text)
```

AGENDA 2030

Añadiendo IA

Ollama SDK: https://github.com/ollama/ollama-python

```
pip install ollama
```

```python
from ollama import chat
from ollama import ChatResponse

response: ChatResponse = chat(model='llama3.2:1b', messages=[
  {
    'role': 'user',
    'content': 'Why is the sky blue?',
  },
])

print(response.message.content)
```

AGENDA 2030

Añadiendo IA

LangChain: https://github.com/langchain-ai/langchain

- Pipelines de acciones con LLMs
  - Permite interactuar con diferentes sistemas
  - Integración con Ollama

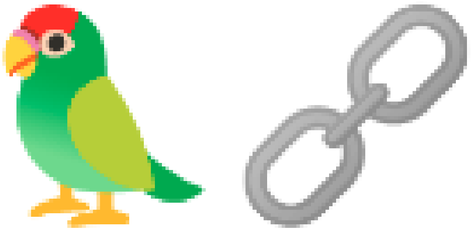
```
pip install langchain
```

```
pip install langchain-ollama
```

```
from langchain_ollama.chat_models import ChatOllama

model = ChatOllama(model = "llama3.2:1b")

messages = [
    {
        "role": "user",
        "content": "Why is the sky blue?"
    }
]

response = model.invoke(messages)

print(response.content)
```

Añadiendo IA

+



LangChain: Plantillas

```python
from langchain_ollama.chat_models import ChatOllama
from langchain_core.prompts import ChatPromptTemplate

model = ChatOllama(model = "llama3.2:1b", temperature = 0)

system_template = "Translate the following from English into {language}"

prompt_template = ChatPromptTemplate.from_messages(
    [("system", system_template), ("user", "{text}")]
)

message = {"language": "Spanish", "text": "Car is blue"}

prompt = prompt_template.invoke(message)

response = model.invoke(prompt)

print(response.content)
```

Añadiendo IA

LangChain: Datos estructurados

```python
from langchain_ollama.chat_models import ChatOllama
from langchain_core.prompts import ChatPromptTemplate

model = ChatOllama(model = "llama3.2:1b", temperature = 0)

from pydantic import BaseModel, Field
class ResponseFormatter(BaseModel):
    item: str = Field(description="The object refered in the sentence")
    color: str = Field(description="The color of the object")

model_with_structure = model.with_structured_output(ResponseFormatter)

message = "The car is blue."

structured_output = model_with_structure.invoke(message)

print(structured_output.model_dump_json())
```

AGENDA 2030

Añadiendo IA

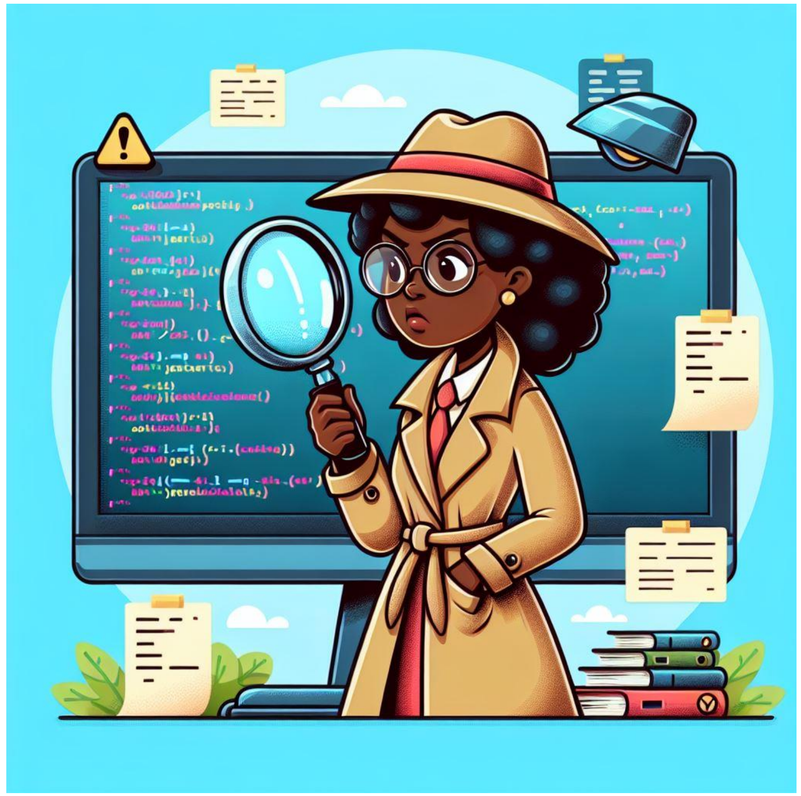
Simplificación de la entrada de datos

- Usando lenguaje natural
- "He visto un vídeo de microservicios hace 2 días"
- ¿Cuándo? - Hace 2 días
- ¿Qué? - Un vídeo de microservicios

```python
from langchain_ollama.chat_models import ChatOllama
from langchain_core.prompts import ChatPromptTemplate

model = ChatOllama(model = "llama3.2:1b", temperature = 0)

from pydantic import BaseModel, Field
class ResponseFormatter(BaseModel):
    action: str = Field(description="The action that took place")
    item: str = Field(description="The element that the action refers to")
    date: str = Field(description="Relative days ago when the action took place")

model_with_structure = model.with_structured_output(ResponseFormatter)

message = "I've watched a microservices tutorial two days ago."

structured_output = model_with_structure.invoke(message)

print(structured_output.model_dump_json())
```

Otras fuentes de datos

- No todo tienen que ser entradas de usuario
- Ejemplos
  - CV
  - Historial de navegación
  - Análisis de RRSS
  - Repositorios de código



Otras fuentes de datos

+ Extraer información de repositorios de código

- Git ofrece mucha información "oculta"
  - ¿Quién ha tocado qué ficheros?
  - ¿Cuándo?
  - ¿Qué dependencias ha introducido?

- Con la herramienta CLOC podríamos conocer lenguajes que se están utilizando

AGENDA 2030

Otras fuentes de datos

Extraer información de repositorios de código: CLOC

- CLOC: https://github.com/AlDanial/cloc
- Ejemplo:

```
cloc ./ --by-file --csv --quiet --report-file=cloc_report.csv
```

Otras fuentes de datos

Extraer información de repositorios de código

- git log
- Ejemplo: Quién ha tocado un fichero

```
git log --pretty=format:'%an' -- README.md | sort -u
```

Otras fuentes de datos

Extraer información de repositorios de código

- git blame
- Ejemplo: Quién ha introducido una línea de código

```
git blame --porcelain -L 5,5 README.md | sed -n 's/^author //p'
```

Dudas y consultas

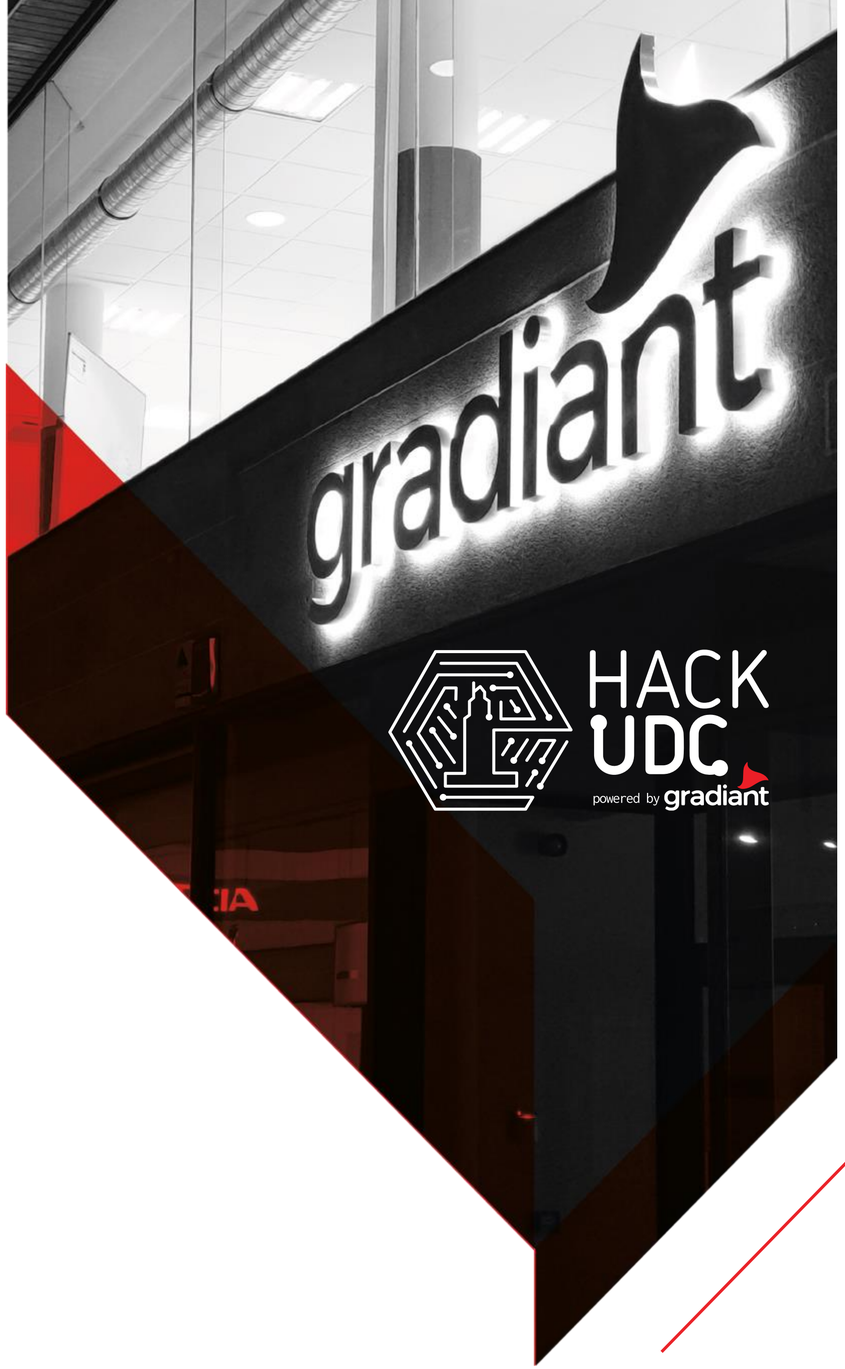
gradiant
Connectivity · Intelligence · Security

David Fernández Hermida
Alejandro Pereira Carballo
Javier Abia Álvarez

dfernandez@gradiant.org
apereira@gradiant.org
jabia@gradiant.org

/gradiant

/gradiant



(+34) 986 120 430 | gradiant@gradiant.org | www.gradiant.org