



TRAIN-THE-TRAINERS IA · DÍA 5

AI Engineering *Hackathon.*

Hoy todo lo aprendido en la formación se convierte en producto: una Wallet funcional, segura, trazable y demostrable ante el jurado.

AUDIENCIA

Train the trainers

Completaron los días 1 a 4

DURACIÓN

Jornada · 9:00–18:00

Build iterativo · Demo Arena

META DEL DÍA

Resolver el challenge

Producto · Build · Validar · Demo

Contenido.

Cuatro paradas: del porqué del reto a la Demo Arena. Cada bloque responde una pregunta que el equipo debe poder defender.

01 Overview

Qué integramos hoy de los días 1 a 4.

02 Today

Cronograma, metodología, equipos y reglas.

03 Challenge

Retos de la Hackathon.

04 Evaluation

Demo Arena, criterios y rúbrica.

1 Overview.

Objetivo general de hoy

Hoy todo lo aprendido *se convierte en producto.*

El reto es integrar lo desarrollado en los cuatro días para construir, validar y defender una solución funcional, segura, trazable y con impacto en el negocio.

DÍA 1	DÍA 2	DÍA 3	DÍA 4
Prompting & Secure AI Engineering Construir con IA bajo control. Contexto, restricciones, buenas prácticas y validación con evidencia. TRAES Criterio para usar IA de forma segura y con calidad.	Agent Skills, Templates & MCP Conectar capacidades. Prototipos convertidos en skills reutilizables, con contratos, templates y tools gobernadas. TRAES Capacidades y herramientas para integrar.	SDD + Agentes Diseñar y gobernar agentes. Definir antes de construir: issue → proposal → specs → tasks → validate, con control humano. TRAES Estructura, trazabilidad y control de decisiones.	Agentic Engineering in Practice Operar el sistema de agentes. De un agente a un sistema: orquestación, evaluación, métricas y seguridad. TRAES Cómo orquestar, medir y proteger un sistema real.
DÍA 5 HACKATHON	Integrar todo de forma <i>funcional, segura, trazable y demostrable.</i> ✓ Funciona Flujo end-to-end ejecuta en local. ✓ Se puede probar Tests, guardrails y evidencia. ✓ Se puede defender Decisiones, trade-offs y trazabilidad. ✓ Genera valor Resuelve un problema real del negocio.		

2 Today.

Cronograma de la Hackathon · sesión 5

Un día para construir, *validar y demostrar*.

Cada etapa suma hacia una historia completa, funcional y con evidencia. **La arquitectura es libre; las seis historias, los controles, la evidencia y la hora de cierre no lo son.**

HORA	DURACIÓN	ETAPA	ACTIVIDAD PRINCIPAL
08:30 – 09:00	15 min	Coffee & registro	Bienvenida y networking.
09:00 – 09:20	20 min	Kick-off	Reglas, reto, entregables, agenda y rúbrica.
09:20 – 10:00	40 min	1 · Alinear	Entender problema, usuario y valor; enfoque inicial.
		2 · Priorizar	Alcance construible, arquitectura mínima y criterios de aceptación.
10:00 – 10:15	15 min	G1 · Scope Check	Revisión rápida de alcance con mentores.
10:15 – 12:30	2h 30 min	3 · Construir · Sprint 1	Construcción end-to-end por slices; agents/tools y código.
12:30 – 13:30	1 h	Almuerzo	Pausa y networking.
13:30 – 13:45	15 min	G2 · Scope Check	Revisión rápida de alcance con mentores.
13:45 – 15:15	1h 30 min	3 · Construir · Sprint 2	Completar funcionalidades, integraciones y casos pendientes.
15:15 – 15:45	30 min	G3 · Freeze	Congelar versión: GitHub, README y URL pública; revisión con mentores.
15:45 – 16:00	15 min	Coffee break	Preparación final.
16:00 – 17:30	1h 30 min	5 · Demo Arena	Presentaciones (6 min) / preguntas (2 min) por mesa y deliberación del jurado.
17:30 – 18:00	30 min	Cierre	Entrega de certificados y ceremonia de cierre.

Iterar en ciclos, *con controles claros.*

Slices verticales: diseño, construcción y validación se solapan. Solo las reglas de seguridad y el cierre horario son obligatorios.

1 Alinear OBJETIVO Entender problema, usuario y valor antes de construir. CHECKLIST ¿Qué problema resolvemos? ¿Para quién? ¿Qué resultado esperamos? CAPACIDADES D1 D3	2 Priorizar OBJETIVO Convertir el problema en un alcance construible hoy. CHECKLIST ¿Qué sí y qué no? ¿Cuál es el flujo mínimo? ¿Cómo sabremos que funciona? CAPACIDADES D3 D4	G1 y G2 Scope Check OBJETIVO Confirmar que el alcance es viable. No bloquea. CHECKLIST ¿Cabe en el tiempo? ¿Arquitectura viable? ¿Qué entregables generarás? CAPACIDADES D3 D4	3 Construir OBJETIVO Flujo end-to-end por slices, con trazabilidad en commits y PR. CHECKLIST ¿Funciona end-to-end? ¿Tools/Skills operativos o que herramientas aplicaste? ¿Qué criterios de aceptación de HU completaste? CAPACIDADES D1 D2 D3 D4	G3 Freeze OBJETIVO Congelar una versión demostrable y verificable. CHECKLIST ¿GitHub y URL listos? ¿README o lo que gustes se presenta el caso desarrollado? ¿Versión estable? CAPACIDADES D1 D3 D4	5 Demo Arena OBJETIVO Probar el valor de lo construido ante el jurado. CHECKLIST ¿Impacto de negocio? ¿Demo funcional? ¿Tecnología aplicada? CAPACIDADES D1 D2 D3 D4
---	---	---	---	--	---

Las clases no son etapas: son capacidades que los equipos aplican donde aportan valor. D1 Prompting & Secure AI · D2 Skills, Templates & MCP · D3 SDD + Agentes · D4 Agentic Engineering

Libertad para construir, *reglas sin excepciones.*

Los equipos deciden cómo construir el reto; las reglas fijan los límites, los mocks y la evidencia que debe quedar.

ROLES DEL EQUIPO

Product Owner

Prioriza historias, define el alcance y mantiene criterios claros.

Arquitecto/a

Define módulos, mocks, permisos y decisiones técnicas.

AI Engineer

Implementa incrementos con IA y revisa cada diferencia.

QA & Security

Prueba flujos, bordes, fallos y guardrails desde el inicio.

Presenter / Timekeeper

Cuida tiempo, evidencia y demo.

REGLAS DE JUEGO

- ✓ **Mentoría – 8 tickets por equipo:** cada equipo contará con **8 tickets de consulta**, equivalentes a **8 dudas como máximo durante toda la hackathon**. El equipo decide **cuándo utilizar cada ticket y en qué etapa**, administrándolos estratégicamente según sus necesidades.
- ✓ **Pitch final:** **6 min de exposición + 2 min de preguntas** del jurado por equipo.
- ✓ **Tiempo controlado:** cada fase de la hackathon tendrá un **cronómetro visible y un tiempo límite definido**.
- ✓ **Cierre de fase:** al finalizar el cronómetro, los equipos deberán **detener la actividad y pasar a la siguiente etapa**.
- ✓ **Mismas condiciones:** todos los equipos tendrán **los mismos tiempos y reglas** durante la competencia.

EQUIPO DE FACILITACIÓN

Instructores (mentores)

Apoyo en dudas técnicas; uno por cada dos equipos.

Jurado

Miradas de técnica, producto y seguridad.

3 Challenge.

El desafío a resolver hoy

Una Wallet demostrable, *no un core bancario.*

Un flujo end-to-end para pagos cotidianos de bajo monto en Perú, con AI Agents que aporten valor en la experiencia, el control y la gestión de excepciones.

CONTEXTO

Una entidad financiera busca una Wallet simple y segura: registrarse, pagar a personas o comercios y conocer el estado de cada operación, incluso ante rechazos, duplicados o fallas.

USUARIO

Persona que necesita pagar rápido y entender al instante si su operación fue realizada, rechazada o requiere alguna acción.

OBJETIVO

Demostrar un flujo funcional de inicio a fin, con agentes donde generan valor, ante operaciones exitosas, duplicadas, rechazadas o fallidas.

TIEMPO

40 min

Alinear + Priorizar

4 h

Build en dos sprints

Demo Arena 16:00

FLUJO MÍNIMO END-TO-END

1

Alta de usuario

Registro simple con validaciones básicas.

2

Wallet ejecutable

Activar y ver saldo.

3

Pago a persona o comercio

Bajo monto, por QR o alias.

4

Seguimiento de estado

Consultar el estado de la operación.

5

Excepciones y recuperación

Rechazos, duplicados o fallas, con guía del agente.

Qué es real, qué es mock *y qué es éxito.*

El dominio Wallet se construye de verdad; todo lo externo se simula de forma controlable y reproducible.

REAL · LO QUE SE CONSTRUYE

- Dominio Wallet y sus estados
- Idempotencia de pagos y callbacks
- Ledger local
- Conciliación con fixtures
- Datos sintéticos, sin secretos ni sistemas reales

MOCK · LO QUE SE SIMULA

- Identidad, KYC, OTP y biometría
- Pagos externos, CCE, gateway, QR y callbacks
- Core, settlement, reversos monetarios y ledger corporativo
- Estados: APPROVED, DECLINED, PENDING, TIMEOUT y callback duplicado

CRITERIOS DE ÉXITO

Funcionalidad: un flujo end-to-end ejecuta en local.

Calidad: tests de comportamiento e idempotencia.

Trazabilidad: historia → task → commit → PR → evidencia.

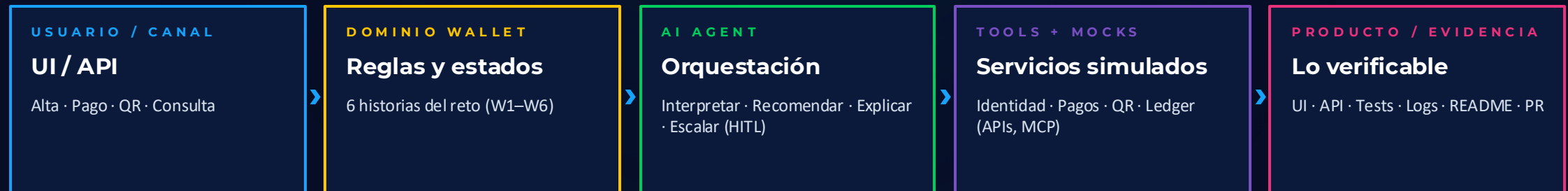
Defensa: mocks, decisiones y backlog pendiente.

ARTEFACTOS MÍNIMOS

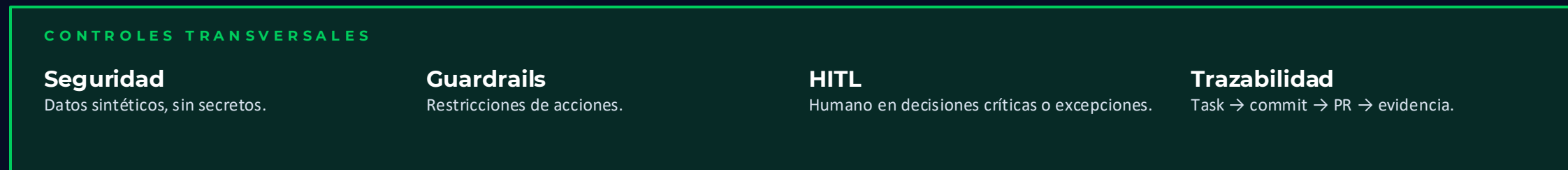
- Wallet ejecutable (app o API local)
- Flujo vertical end-to-end
- Tests + mocks con datos sintéticos
- README y arquitectura
- PR + evidence.md

Una Wallet, seis historias, *controles transversales*.

Cada equipo integra las seis historias en una solución end-to-end. La arquitectura es libre, con servicios simulados y evidencia local.



SEIS HISTORIAS EN EL DOMINIO WALLET



Seis historias, *libertad de implementación*.

El objetivo es completar el flujo funcional de las seis historias. Ustedes deciden cómo implementarlas, qué tecnologías usar y qué mockear.

W1 Onboarding Wallet

Alta, validaciones y estados de cuenta de prueba.

- ✓ Registro de usuario (UI o API)
- ✓ Validaciones básicas y KYC mock
- ✓ Estado: activa, pendiente, rechazada

COMPONENTES UI · API · KYC mock · HITL

W2 Pago a contacto

Crear intención, confirmar resultado e historial con proveedor simulado.

- ✓ Pago a persona o comercio (mock)
- ✓ Confirmación y comprobante
- ✓ Historial de transacciones

COMPONENTES API · Estados · Payment mock

W3 Pago QR comercio

Generar o leer QR de prueba, iniciar el pago y reflejar su resultado.

- ✓ Generación de QR (mock)
- ✓ Lectura de QR (cámara o archivo)
- ✓ Proceso de pago y comprobante

COMPONENTES QR local · Acquiring mock

W4 Idempotencia

Repetir requests y callbacks sin duplicar operaciones. Esta lógica sí es real.

- ✓ Reintentos con idempotency key
- ✓ Callbacks duplicados
- ✓ Validación en base de datos o store

COMPONENTES Clave idempotente · Tests

W5 Fallas y reversos

Timeout, rechazo, pending y duplicados; reversos simulados y escalamiento.

- ✓ Casos de error: timeout, rechazo, pending
- ✓ Simulación de reversos
- ✓ Escalamiento a soporte humano (HITL)

COMPONENTES Fault injection · HITL

W6 Conciliación

Comparar Wallet, proveedor mock y ledger local; detectar diferencias.

- ✓ Conciliación entre fuentes (mock)
- ✓ Detección de diferencias
- ✓ Estado final de las operaciones

COMPONENTES Reconciliador real · Datos mock

El README es tu *carta de presentación*.

Debe permitir que el jurado entienda, ejecute y evalúe la solución en pocos minutos. Claro, completo y honesto: es parte de la evaluación.

1

Resumen del proyecto

Qué resuelve, para quién, reto y valor de negocio.

2

Cobertura del reto

Tabla W1–W6: estado, implementación y evidencia.

3

Arquitectura y decisiones

Diagrama, componentes, flujo, alternativas y versiones.

4

Uso de IA

Prompts, skills/tools, MCP, agentes, guardrails y por qué.

5

Cómo ejecutar

Requisitos, .env.example, comandos, URL y datos de ejemplo.

6

Evidencia y pruebas

Tests, happy path, caso borde, logs y guardrails.

7

Trade-offs y aprendizajes

Qué no se hizo, deuda conocida, riesgos y siguiente iteración.

8

Equipo

Integrantes, roles y principales contribuciones.

ESTRUCTURA DEL REPOSITORIO

```
wallet-hackathon/  
├─ src/ agents/ skills/  
├─ infra/ tests/ docs/  
├─ demo/  
├─ README.md  
├─ evidence.md  
├─ .env.example  
└─ requirements.txt · docker-compose.yml
```

CHECKLIST ANTES DE ENTREGAR

- ✓ README y cobertura del reto
- ✓ Diagrama e instrucciones que funcionan
- ✓ Código en GitHub y video de la demo (o en vivo)
- ✓ Tests, evidencia y trade-offs

4 Evaluation.

¿Qué se espera en la Demo Arena?

6 minutos para *demostrar y 2 defender.*

No cuentes todo lo que hicieron. Muestra qué funciona, cómo lo controlaron y por qué tomaron esas decisiones.

1 (10%)

Impacto de negocio

- Problema y contexto
- Usuario y necesidad
- Resultado esperado y valor

2 (60%)

Tecnología usada

- Arquitectura y componentes
- Prompts, skills, tools/MCP, single o multi-agent
- Por qué esas decisiones

3 (30%)

Demo en vivo

- Flujo end-to-end funcionando
- Alta de usuario
- Wallet ejecutable
- Pago a persona o comercio
- Seguimiento de estado
- Excepciones y recuperación

Formato: 6 min de demo/pitch + 2 min de preguntas por equipo.

Orden sorteado antes de empezar; demo en vivo, video solo como contingencia técnica.

Todos pueden ser preguntados: no se evalúa solo a la persona que presenta.

Tres bloques, 100 puntos.

Se puntúa la calidad de las decisiones y la evidencia, no la cantidad de tecnologías.

BLOQUE 1

10%

Impacto de negocio

- Problema y contexto
- Usuario y necesidad
- Resultado esperado y valor

EVIDENCIA

Presentación del proyecto (pitch)

EL JURADO OBSERVA

Problema claro, usuario identificado y valor esperado medible.

10%

BLOQUE 2

60%

Arquitectura y componentes · Prompts, skills, tools/MCP, single o multi-agent · Por qué esas decisiones

Tecnología usada

DÍA 1

CRITERIO 01

IA con Control

Prompting & Secure AI Engineering

10%

Blueprint, ADR, prompts con contexto y restricciones, tests.

DÍA 2

CRITERIO 02

Capacidades que Escalan

Agent Skills, Templates & MCP

10%

Demo en vivo, skills/tools con contrato, templates reutilizados, código y tests.

DÍA 3

CRITERIO 03

Del Spec al Agente

SDD + Agentes

20%

Spec → tasks, pipeline, reporte del agente revisor y hallazgos.

DÍA 4

CRITERIO 04

Agentes en Acción

Agentic Engineering in Practice

20%

Orquestación, harness, PR, README, evidence.md y prompts versionados.

Σ 10 + 10 + 20 + 20 = 60%

Tecnología · 60%

BLOQUE 3

30%

Demo en vivo

- Flujo end-to-end funcionando
- Happy path
- Alta de usuario
- Wallet ejecutable
- Pago a persona o comercio
- Seguimiento de estado
- Excepciones y recuperación

EL JURADO OBSERVA

Demo ejecutándose en vivo, sin intervención manual y con manejo de excepciones.

Demo en vivo · 30%

© 2026 NTT DATA, Inc.

BCP · Train the Trainers · Rúbrica de evaluación

