# Challenge Overview

Imagine it’s the year 2030. Most companies now rely on AI agents to handle their financial operations—everything from budgeting to negotiating with banks. These agents act like smart CFOs, constantly scanning the market for the best financial products.

## Problem Statement:

Let’s say Company X needs a line of credit. Its AI agent sends out a digital request (called an Intent) describing what it needs—amount, duration, purpose, etc. This request is picked up by several banks, each of which has its own AI agent listening for such opportunities.

Each bank agent:

* Verifies the identity of the sender and reviews the request
* Runs necessary checks (e.g., creditworthiness, risk)
* Prepares a quote with terms like interest rate, repayment schedule, and ESG (Environmental, Social, Governance) impact
* Packages this into a machine-readable signed response and sends it back to the company’s agent

Now, the company’s agent receives multiple offers and must choose the best one based on:

* Lowest carbon-adjusted interest rate
* Other financial terms – amount approved, repayment period etc.
* A clear, human-readable ESG summary

Your mission in this hackathon is to build the basic version of this system.

# Expectations from Participants

At the end of the hackathon, you should be able to demo this agent-to-agent interaction and negotiation, resulting in a winning offer. Following are the main tasks for this hackathon

* 1. Define the Intent and Response protocol (WFAP)
     + Define the WFAP protocol - Markdown
     + Document schemas to verify compliance (json/yaml etc.)
     + Include fields for identity, terms, ESG, and regulatory compliance
  2. Build multiple Bank Agents (WFAP Server)
     + Build an agent that listens for credit requests
     + Accepts a signed structured Intent (JSON format)
     + Returns a signed Offer Packet with:
       - Interest rate and terms
       - ESG summary (auto generated using an LLM)
       - Token identifying the bank
  3. Consumer Agent (WFAP Client)
     + Build an agent that:
       - Sends out a signed credit request (Intent)
       - Receives offers from at least 3 bank agents
       - Picks the best offer
  4. Audit/Reasoning Traces
  5. Bank Agent: For each intent received, provide traces/logs for
     1. How the agent verified the client’s identity/consent
     2. Steps taken by the agent to arrive at the offer
     3. How the agent asserted bank’s identity
  6. Consumer Agent: For each offer received, provide traces for
     1. How the agent verified the bank’s identity
     2. Steps taken by the agent to validate the offer from the banks
     3. Reasoning behind the agent choosing one of the Banks offer

# Technical Constraints & Guidelines

* Use Wells Fargo approved software/packages that are available in artifactory
* Tachyon API keys will be provided

# Sample Test Data Set

This problem statement does not include any pre-defined datasets. Participants are encouraged to generate synthetic data tailored to their implementation needs. Suggested data categories include:

* Consumer Requirements: Details such as the desired credit amount, preferred interest rate, repayment duration, and purpose of the loan.
* Consumer Internal Policies: A document outlining the consumer’s decision-making criteria—what factors are prioritized (e.g., ESG impact, interest rate) and which are less important.
* Bank Internal Policies: Guidelines specifying how much the bank is willing to lend, acceptable interest rate ranges, minimum credit score thresholds, and any industry exclusions.

# Submission Guidelines

Participants must submit:

* A GitHub repository with the code, including a README file with setup instructions.
* A demo video (mandatory) showcasing the solution in action (in the artifacts/demo folder)
* A presentation (pdf/ppt) explaining the approach, design, challenges and results (in the artifacts/arch folder). You can use this folder to share any other reference documents that you may have used – for example
  + Carbon foot printing calculations/formula
  + ESG score calculations/formula
  + Line of Credit specific specs
* The protocol definition in a machine-readable format – either schema.json or schema.yaml (in the artifacts/arch folder)
* All your code (in the code/src folder) – recommend splitting the consumer agent & bank agent in separate folders
* All your tests (in the code/test folder)

Git folder structure for Hackathon

/*<Team Repo>*

|-- artifacts/

|-- arch/ #architecture documents

|-- demo/ #presentation or Demo to support

|-- code

|-- src/ #source code

|-- test/ #test cases

# Resources & References

* **Lines of Credit for Businesses:** [**Bankrate Guide to Business Lines of Credit**](https://www.bankrate.com/loans/small-business/what-is-a-business-line-of-credit/)
* **Carbon Footprint Calculation for Financial Products:** [**Carbon Footprinting Demystified – MSCI**](https://www.msci.com/documents/10199/a57d8424-34a9-3ea4-5a74-030349845bb3)
* **ESG Scoring Frameworks and Standards:** [**IFC ESG Guidebook (PDF)**](https://www.ifc.org/content/dam/ifc/doc/mgrt/ifc-esg-guidebook.pdf)
* **Privacy-Preserving Credit Score Verification:** [**Privacy-Preserving Credit Scoring – Springer**](https://link.springer.com/article/10.1007/s12083-025-01963-4)
* **Carbon-Adjusted Interest Rates:** [**Emission-Based Interest Rates – ETH Zurich**](https://ethz.ch/content/dam/ethz/special-interest/mtec/cer-eth/resource-econ-dam/documents/research/sured/sured-2020/Emission-based%20Interest%20Rates%20and%20the%20transition%20to%20a%20low%20carbon%20economy.pdf)