# Final Project – Tokenized Asset Management Platform

⚠️ **READ CAREFULLY – you will lose points if you forget or skip parts of the assignment. Every requirement matters.**

## Objective

You will build a **Tokenized Asset Management Platform** on **either**:

- **Any EVM-compatible blockchain**, **or**
- The **XRP Ledger**

-> You must be able to justify your choice (technical decision, users you want to target, etc)

Your platform must allow users to tokenize real-world assets (RWAs), manage ownership, and trade tokens securely on-chain.

## Group Rules

- You can create groups of **3–4 people**
- You are encouraged to **discuss, exchange ideas, and even share parts of your code** with other groups.
- ❌ However, **brutal copy/paste between groups will result in point loss**. Each group must demonstrate their own understanding and implementation.

**Total estimated workload:** ~120–150 hours per group over 5 weeks.

## Core Requirements

### 1. Tokenization of Real-World Assets

- Students must choose a **real-world asset** or a **class of assets** (please don't tokenize Pokemon cards or random objects)
- Platform must support **two types of tokenization**:
  - **Fungible tokens (ERC-20 / IoU, MPT XLS33 on XRPL)** → e.g. company shares, real estate shares.
  - **Non-fungible tokens (NFTs, ERC721 / XLS-20)** → e.g. unique artwork, collectibles, diamonds.

### 2. Compliance: On-Chain KYC & Whitelisting/Blacklisting

- Implement a **simple KYC system**.
- Only **whitelisted addresses** can hold and/or trade the tokenized assets.
- Include a **blacklist mechanism** (e.g. revoking access if needed and deny even with KYC).
- **Enforce this logic on-chain** (not just in the frontend).

### 3. Token Trading (On-Chain)

- Token must be **tradable on-chain**.
- Trading allowed **only between whitelisted users**.

- Create **at least one liquidity pool on a DEX** (e.g. Uniswap or XRP Ledger built-in AMM).
- Provide initial liquidity yourself (reuse DEX pool code from class).

## 4. Real-Time On-Chain Awareness (Indexer)

- Your frontend must reflect the **real state of the blockchain**.
- If a user swaps directly on the DEX (outside your UI), the change must appear in your app
- Build a **simple indexer** (runs e.g. every minute) that syncs on-chain events to your app's backend/frontend.
- (You can reuse the indexer code you built in class).

## 5. Oracles

- Add an **on-chain oracle** for at least one token/collection. Example: oracle provides price data for a real-world asset or NFT collection. (You can reuse oracle code from class if needed).

## Technical Guidelines

- **Blockchain choice: XRP Ledger** or any **EVM-compatible blockchain** (you must be able to justify your choice).
- **Onchain logic:** Use **Solidity** (EVM) smart contracts **or XRPL primitives**.
- **Frontend:** Just make it clear and intuitive.
- **Backend/Indexer:** Simple service to query blockchain events and keep the frontend in sync.
- **DEX Integration:** Uniswap v3/v4 or forks, Sushiswap, or XRPL AMM (depending on your chosen chain).

Allowed Programming languages:

- Javascript / Typescript (recommended because of the very complete and intuitive web3 libraries like ethers, viem, web3, …)

- Rust / Go

- Python

*You can use any other programming language but only you asked me and I accepted it*

## Deliverables

- **Fully working platform** deployed on chosen chain (testnet only) (Hosting the frontend is mandatory, you can use vercel for free).
- **Source code** Available in a single github repository (public).
- **Documentation** (how to use + design choices), a complete README.md file can be enough.
- **Demo presentation** showing tokenization, compliance, trading, oracle, and on-chain syncing.

## Bonus Points

- Support **Multisig wallets** (as protocol admins and/or as users)
- **Gas sponsoring** (meta-transactions, relayers, erc4337, or any other way)
- Anything you want to do that is not required and is linked with what we studied

**Important:** The project is not just about coding — documentation, and design matter.

If you have any question, contact me at **nathan@hervier.capital**
**Happy Coding !**