TTM4195 Smart contracts in Solidity

1 Requirements

This assignment requires you to implement and demonstrate a smart contract in Solidity. You will not have to install any additional software, since we recommend the use of an online compiler. The assignment will help you understand how smart contracts work and how to deploy them. You should work in groups of 5 (4 if necessary).

In order to facilitate your meeting with Solidity programming, you will find on Blackboard a short introduction to Ethereum, Solidity and Smart Contracts. The notes will be uploaded together with this assignment. Please, refer to the Solidity website for the complete documentation of the latest version v0.8.26 (warning: the version keeps being updated).

In this assignment you will need two tools: an Ethereum wallet and Remix. A **digital wallet** (or **cryptocurrency wallet**) allows you to safely store cryptocurrencies (Ether, for this assignment). Nowadays, we have two families of wallets: **hardware wallets**, resembling an hard drive and **software wallets**. For Ethereum, the most popular (*and the one we strongly recommend you to use*) online wallet is Metamask, a browser extension that guarantees instant access to the Ethereum network (and to the testnet Goerli or Sepolia, which you will be using). Another valid option is Eidoo, from a non-custodian company, which comes in the form of app for iOs, Android and desktops.

You will make use of the **Remix IDE**, an open source software which allows you to test, debug and deploy smart contracts written in Solidity or Vyper for the Ethereum blockchain. It is available both offline and online, and we strongly recommend you to go for the online version. The interface is pretty simple and it even allows you to decide the compiler version; for any queries see the online documentation.

2 Deadlines

There are 15 marks available for this assignment:

1. **13 points for the scenario proposal and the code**, scenario proposal deadline on 4 November 23:59:59. You will have to submit the scenario proposal which outlines the context of their smart contract system by on 4 November 23:59:59. **If you do not submit your scenario proposal by the deadline, 4 points will be**

deducted from your total score, regardless of the quality of your code or final implementation.

You will have to submit the code containing your implemented smart contract by Sunday 10 November, 23:59:59. The code will be evaluated within 2 weeks. As reference for the structure of your smart contract, see section 4. Uncommented/poorly commented code will result in a penalty of 2 points.

2. **7 points for the presentation/demonstration**. Students presentations (20 minutes per group) will take place Tuesday 11 November and Thursday 13 November, during usual class hours. Each team will have to briefly explain how the smart contract works, and illustrate/motivate its approach (splitting of the workload, obstacles and difficulties, main design choices). Then each team will demonstrate a smart contract run. This can be done with a live demonstration OR by showing a recorded demonstration OR using screenshots from a test run.

Time-slot allocation will be performed on a first-come-first-served basis until Monday 10 November, 23:59:59. Reserve a time slot for your group using the link (will be given later).

Deadline for...	**on...**
Booking a presentation slot Scenario proposal submission	4 Nov, 23:59:59
code delivery	10 November, 23:59:59

3 Presentation

See Item 2 for details on how the presentation should be run and what it must contain. The presentation should last 15 minutes in total, followed by 5 minutes for questions. You will use your own laptop connected to the projector.

During your presentation you will be evaluated on

1. (2 point) clarity of the presentation, including keeping to time
2. (3 points) understanding of the concepts
3. (2 points) successful demonstration

4 The smart contract

For this assignment, we ask you to write a Solidity smart contract (SC) and an nonfungible token (NFT) for a car leasing system. Here is the theme and goal.

- **Theme**: A seller offers various NFT-based options (products or services). A buyer selects the most suitable option based on their preferences and financial situation. Once an option is chosen, a lease or purchase contract is signed via a Solidity smart contract. This contract ensures a fair and secure transaction between both parties.
- **Goal**: To build a smart contract system using Ethereum and ERC-721 (NFTs), allowing for NFT-based asset selection, contract negotiation, pricing, and lease/lifecycle management.

Before implementation, **each group must submit a max 1-page scenario proposal** that outlines the context of their smart contract system. This scenario must follow the core requirements below but can be creatively customized.

Your scenario must involve:

- A seller offering a variety of NFTs representing products or services.
- A buyer who chooses an NFT based on their taste or financial situation.
- A contract that manages this agreement on-chain (e.g., leasing, subscription, etc.).
- Optional: Events such as lease expiration, renewal, or upgrades.

Sample Scenario Ideas:

- Electric car leasing, Digital art subscriptions,Real estate short-term leasing,...

5 Technical Tasks (Common Questions)

Each group must answer the following questions by implementing them in a Solidity smart contract.

1. (3 points) NFT Option Modeling: Implement an ERC-721 NFT for each item offered by the seller. Each NFT should include metadata appropriate to your scenario. For example:
    - A car: model, color, year, value
    - A server: CPU, RAM, bandwidth, original price

- An artwork: artist, style, year, estimated value

Choose attributes that make sense for your scenario and encode them in the NFT.

2. (2 points) Dynamic Pricing Logic: Implement a view function (no gas cost) that calculates the monthly payment (or period-based fee) based on:

   - Original value of the NFT
   - Current usage or condition (e.g., mileage, data used, etc.)
   - User attributes (e.g., years of license holding, credit score, etc.)
   - Predefined options: duration, usage caps, etc.

   The formula should reflect how pricing changes with different options.

3. (2 points) Contract Signing & Fair Exchange: To ensure a fair agreement use some security mechanism something like below:

   - The buyer deposits a security deposit (e.g., 3 monthly payments) into the contract.
   - The seller confirms the agreement.
   - Upon confirmation, the agreement becomes active, and the NFT is assigned to the buyer (if applicable).

   Use locked funds and require dual confirmation for fairness. Note that such mechanism can differ depending on your scenario. You need to justify why your mechanism makes sense in your scenario.

4. (2 points) Default Protection: To protect the seller after the contract is active:

   - If the buyer misses a payment, the seller can claim the deposit and terminate the contract.

   No need to implement late fees or grace periods — just ensure the deposit can be used to cover missed payments.

5. (4 points) Contract End Options: At the end of the lease period, the buyer should be able to choose among:

   - (1 point) Terminate the lease and end the agreement.
   - (2 points) Extend the lease for another fixed period (e.g., 1 year). Monthly payments should be recalculated based on updated parameters.
   - (1 point) Start a new contract for a different NFT.