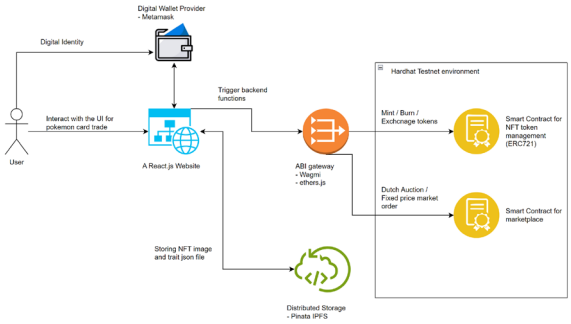
COMP0254 Assignment 1 – Project Report

A. Background And Architecture Overview



PokeAuction, is a project aims to construct a decentralized application (dApp) for trading pokemon cards. In our design (Fig.1), we have constructed a frontend website with react JS and Ant-design UI components, with a backend of smart contracts hosting in hardhat testing network. The projects included

- A smart contract which manages a collection of NFTs under ERC-721 standard.
- A smart contract which manages a marketplace allowing users to do card trades in either dutch auction or fixed price order fashion.
- A front-end portal build with React.js as user-interface and user interactions.
- Set of test cases which to ensure the smart contracts fulfilling the functional requirements of users.

We have leveraged the Wagmi and ethers.js library to communicate between backend and frontendt. For large file like the nft image, the trait json file, they are all stored in a decentralized cloud storage via IPFS. Digital identity and transaction broadcast are managed by MetaMask, a digital wallet. Summary of technology stack will be as follows:

- Solidity, React.js, Antd-UI, Ethers.js, Wagmi, Viem, Hardhat, Metamask, IPFS

B. Setup instructions

First, you will need to clone the repository into your local machine:

Git clone https://github.com/tamww/ucl_comp0254_as1.git

Frontend

1. Open a terminal and navigate to the root directory (ucl_comp0254_as1\frontend\pokemon) of the frontend and install the dependencies with:
   - *npm install*
2. Then to run the development version of the website, you can run
   - *npm run vite*
3. To run the preview (built) version of the website, you can run
   - *npm run preview*

Backend

1. First of all, you have install dependencies under directory (ucl_comp0254_as1\backend):
   - *npm install*
2. To run the test cases for smart contract, just run the following
   - *npx hardhat test*
3. Then you have to firstly kick start the test network in a standalone terminal with:
   - *npx hardhat node*
4. There will be a list of account public keys and privates key show up, mark them down for configuring the metamask in later step
5. After that, you need to deploy the smart contracts into the hardhat test network with:
   - *npx hardhat run ./ignition/modules/deploy.js --network localhost*

Configure Wallet

1. After hosting the backend and fronted, we will need to configure our wallet to be able to intergrate with the functions
2. You need to install the Metamask wallet as a browser extension: https://metamask.io/
3. Then you need to configure Metamask to connect to local hardhat testnet by following
   - Open Metamask app

- Open Network Configuration Panel and open custom RPC
- Configure RPC URL with http://127.0.0.1:8545, Chain ID: 1339, Currency symbol:ETH
- Click select account and pick “add an account or hardware wallet”. Select the import account choice
- Input the private key you noted down previously when hosting the backend test network. It is recommended to use 2 or 3 accounts. The following accounts are pre-selected as admin role during deployment and recommended to include in your metamask in order to replicate some of the features
  - *0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266*
  - *0x8626f6940E2eb28930eFb4CeF49B2d1F2C9C1199*
  - *0x70997970C51812dc3A010C7d01b50e0d17dc79C8*

4. Then you can enjoy the trading nft pokemon card under http://localhost:5173/main

C. Core functionality

Users can perform the following actions on this PokeAuction NFT marketplace.

Create an NFT

Users can use the “create” page to create an NFT token by uploading their image and configuring related trait objects. Once minted, the card will appear in digital form on the website under its creator’s wallet. The card will be open to review by everyone and free to sell to anyone with a digital wallet connected to this website.

Create an order

Users can trade their Pokemon NFT with others by creating a listing in the market. Once someone pays a price equal to or higher than the current platform calculated price, the ownership will be transferred to the new owner. There are two types of listing available

- the Dutch auction: this order requires a pair of open and close prices in which the price decreases gradually from open to close price in the given duration at a steady depreciation rate.
- Fixed price trade: this order only has one price, the start price, and it remains constant throughout the trading period.

Cancel an order / Withdraw a card

If users don’t want to trade the Pokemon card anymore, they can terminate the listing at any time on the card detail page unless someone has purchased it. They can also “burn” the card, which removes it permanently from the platform when the card is in minted stage.

Buy an NFT

Users can pay any price higher than the calculated current price shown to purchase a Pokemon NFT. The remains will then be kept in “pendingToWithdraw” for users to collect back.

Claim funds

Either if the users sold an NFT or had changed from previous transactions, they can withdraw it anytime on the “MyCard” page.

Major security features and technical decisions

There are multiple security features implemented to ensure the robustness of our application.

Prevention of Front-running

Our application uses the hardhat testing network to simulate the real-life Ethereum network. Transactions are selected based on gas fees in the mempool. Thus, there might be a case when a seller sees a buyer submit a transaction to buy the NFT, the seller will front-run the transaction to transfer the NFT to another place. As a result, the seller will earn the money while still keeping the NFT somewhere else.

To prevent this, we have implemented a strategy to immediately transfer the NFT from the seller’s wallet to the marketplace contract for temporary storage during the selling period. The ownership checks will show the marketplace contract as the owner, not the original seller.

```
metadataContract.changeOwnership(msg.sender, address(this), tokenId);
```

Thus, the seller cannot transfer the NFT to another address once listed to prevent seller transfer after listing. Also, as the NFT is already in escrow and price calculation is deterministic on block timestamp. Other buyers cannot front-run legitimate purchases by higher gas transactions after seeing a purchase intent.

Role-based Access and Control rights

```
function setAdmin(address _newAdmin) public onlyRole(ADMIN_ROLE) {
    setApprovalForAll(_newAdmin, true);
```

All critical functions are designed with role-based modifiers to only allow accounts with ADMIN_ROLE to contact those functions. Also, modifiers to check that only the market is not paused are added to mint and purchase to ensure only during operation hours, users can trade. Also, token ownership checks are added to prevent malicious from pretending the users to transfer the NFT out.

```
/// @notice execute purchase
function executePurchase(uint256 tokenId) public payable whenNotPaused nonReentrant{
    Listing memory _listing = listings[tokenId];

    require(_listing.seller != _msgSender(), "buy nft: buyer & seller shall be different.");
    require(!metadataContract.checkBurned(tokenId), "buy nft: NFT has been burned");
    require(_listing.isActive, "buy nft: not on sales.");
    require(block.timestamp <= _listing.endTime, "buy nft: out of sales period");
```

Reentrancy protection

To prevent the users from reentering the withdraw function to double withdraw the money. ReentrancyGuard library has been adopted to provide check for reentrancy. Also, we deduct the value on the mapping before sending out money to minimize loss of double withdraw.

```
function safeWithdraw() external nonReentrant {
    require(address(this).balance > 0, "Withdraw: no funds for withdraw.");
    uint256 toBeWithdraw = pendingWithdraw[_msgSender()];
    require(address(this).balance >= toBeWithdraw, "Withdraw: insufficient funds for withdraw.");
    pendingWithdraw[_msgSender()] = 0;
    payable(_msgSender()).transfer(toBeWithdraw);
    emit WithdrawFairly(_msgSender(), toBeWithdraw);
}
```

Event Emission

Various events have been set to recorded major activities happened in the operation, like the mint of new token, burn of token, a sales start or completed, or an emergency withdraw of token in case of platform emergency.

```
//*****************************************************//
//                       Event                         //
//*****************************************************//
// event Listed(uint256 indexed tokenId, SaleType saleType);
// event Sold(uint256 indexed tokenId, address buyer);
event EmergencyPaused(address indexed admin);
event EmergencyWithdraw(uint256 indexed tokenId);
// for withdraw
event WithdrawFairly(
    address user,
    uint256 amount
);
// for auction
event TradeStarted(
    uint256 tokenId,
    uint256 startTime,
    uint256 endTime,
    uint256 duration,
    uint256 startPrice,
    uint256 endPrice,
    address seller,
    bool isActive,
    SaleType cardType
);

event TokenSold(
    uint256 tokenId,
    address originalOwner,
    address newOwner,
    uint256 price,
    uint256 time,
    SaleType cardType
);

event TradeCancel(
    uint256 tokenId,
    address owner,
    uint256 time,
    SaleType cardType
);
```

Emergency Stop Functions

Emergency function is implemented to allow admin to bring a token out of trade when it gets stuck in certain situations. The token will firstly transfer to the owner of contract to ensure the token NFT will be in dangling situation.

Work distribution And Use of AI

Name	Task
Wui Wo TAM	• Deliver the demo video presentation • Frontend development and UI design • Backend contract design for PokemonCard, PokemonMarkeplace • Doucmentation • Unit and integration testing of frontend and backend
Zhen YANG	• Backend contract design for PokemonCard, PokemonMarkeplace • Documentation • Unit and integration testing of frontend and backend

Generative AI included ChatGPT and deepseek has been used to assist the frontend development specifically for the filter panel for filtering related NFT tokens, form used for creating new nft tokens,, and prompts for image generation. Microsoft Designer has been used to generate the NFT token images.