

Jirayu Petchhan	D10907801
Supitchar Tatiyathavornkul	M10918803
Tanaporn Chaivutitorn	M10921817

## **Assignment 2 : Blockchain application “Monster Protocol”**

### **Introduction**

Nowadays, the gaming industry is growing faster and faster including the welfare and safety of digitized assets (i.e., in-game property e.g., [1] CryptoZombies) becoming the role play of financialization and substantial community in the present. Moreover, people become more and more interested in cryptocurrency and learn to invest more on digital money.

### **Description**

“Monster Protocol”, a blockchain game in which players can enjoy fighting monsters across the galaxy. Fighting alone you might not win but fighting together you can win. In addition, conquering a monster, players will receive incentives based on how much you participate to fight this monster. This game will depict the mining coin.

### **Implementation**

We have succeeded the Virtualized crypto games (demo version) by having four stages of in-game difficulty from easy stage to hard stage to defeat them i.e., the level of baby monsters, common monsters, mini-boss and final boss. Each level of monsters requires the token to play and defeat them. Ranging from 2 MOP to 500 MOP. Besides Each monster fight will return a reward randomly. The harder boss you fight the higher chance to get more reward and higher to loss rather than gain (i.e., high risk high return). By the way, Our proposed approach is to get the participants that have satisfied our games (e.g., Airdrop, Daily gift, Monster Dungeon in each level, Ticket’s participants for burning). Our code is applied and based on IERC20 and developed on Solidity language-based programming [2] to develop the Ethereum Smart Contract environment [3], [4].

The library and contract generated to support the Monster Protocol (MOP) games are shown in Figure 1. Besides, figure 2-4 showed the solidity code for game functions which are Airdrop, Daily gift, and boss fight respectively.

```
// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0 <=0.9.0;

// ERC-20
import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/IERC20.sol";
// ERC-721
import "../deps/github/0xcert/ethereum-erc721/src/contracts/tokens/nf-token-metadata.sol";
import "../deps/github/0xcert/ethereum-erc721/src/contracts/ownership/ownable.sol";
import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/math/SafeMath.sol";

contract MonsterProtocol is IERC20 {

    mapping (address => uint256) private _balances;
    mapping (address => mapping (address => uint256)) private _allowed;
    uint256 private _totalSupply;
    string private _name;
    string private _symbol;

    // In case of fixing the static contract on only once address
    // address account;
    // uint value;

    // constructor (address _player,uint _participated_value) {
    //     account = _player;
    //     value = _participated_value;
    // }
}
```

**Fig. 1** Library and contract generated to support the Monster Protocol (MOP) games

```
// =====
//Monster minting the reward in each Level + send to incinerator
address Monster_Dungeon = 0x519A2A7A231515FeE1de8055F4230F8182732E59; // Go to incinerate wallet for burning
// address MOP_cash = 0x6cC95d2Ef35c2ad91FFf565426956Edd1a7e9b52; // Contract address of main MOR currency
uint256 none_token = 0;

// constructor (address Monster_Dungeon) {
//     _forburn = IERC20(Monster_Dungeon);
// }
//-----V3 Minting & Burning same contract-----
// 1st-drop 10K token
function Airdrop(address From_wallet) public {
    require(none_token <= _balances[From_wallet]);
    require(From_wallet != address(0));
    _totalSupply = _totalSupply + 10000;
    _balances[From_wallet] = _balances[From_wallet] +10000;
    emit Transfer(address(0), From_wallet, 10000);
}
}
```

**Fig. 2** Airdrop function

```
// daily free 10 token
function Daily_free_minter(address From_wallet) public {
    require(From_wallet != address(0));
    _totalSupply = _totalSupply + 10;
    _balances[From_wallet] = _balances[From_wallet] +10;
    emit Transfer(address(0), From_wallet, 10);
}
```

**Fig. 3** Daily gift function

```

function _miniboss_minter_burner(address From_wallet, uint256 amount) public {
    //common_monster
    if (amount == 50){
        // address Monster_Dungeon = 0x519A2A7A231515FeE1de8055F4230F8182732E59; // Go to incinerator

        //buying ticket == send to incinerator
        // require(_balances[From_wallet] >= 2, "Monster Protocol is empty, see you next times");
        require(amount <= _balances[From_wallet]);

        //minting the reward via randomish
        require(From_wallet != address(0));

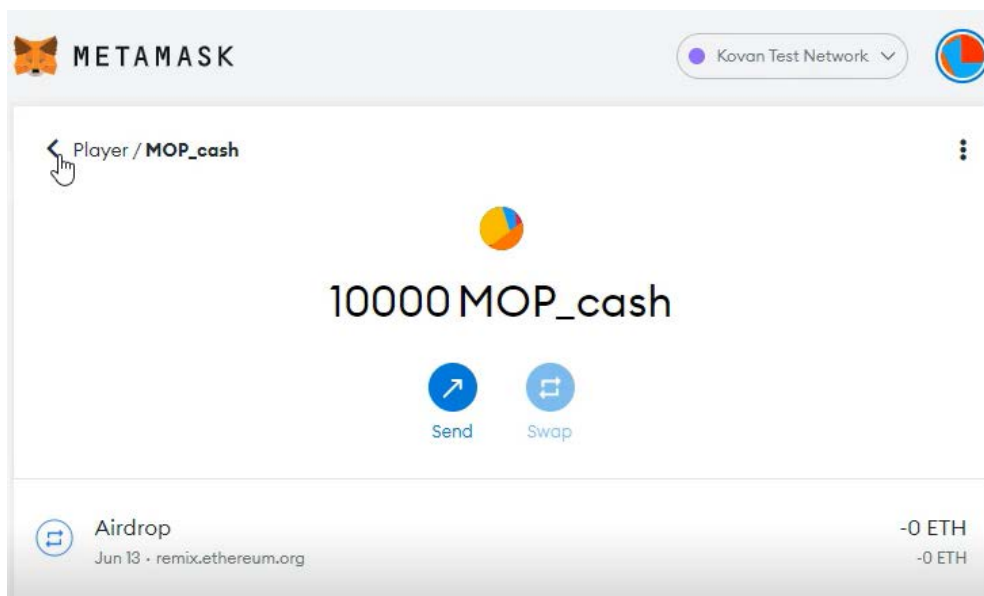
        _totalSupply = _totalSupply + block.timestamp%101 * 98/100;
        _balances[Monster_Dungeon] = _balances[Monster_Dungeon] + amount;
        _balances[From_wallet] = _balances[From_wallet] - amount + block.timestamp%101 * 98/100;
        emit Transfer(address(0), From_wallet, block.timestamp%101 * 98/100); // 2% fee reward back
        emit Transfer(address(0), Monster_Dungeon, amount);
        // return true;
    }
}

```

**Fig. 4** The example method of the function mini-boss monster level to stake and gain/loss your rewards and send ticket (MOP token) to address for burning (i.e., Monster\_Dungeon)

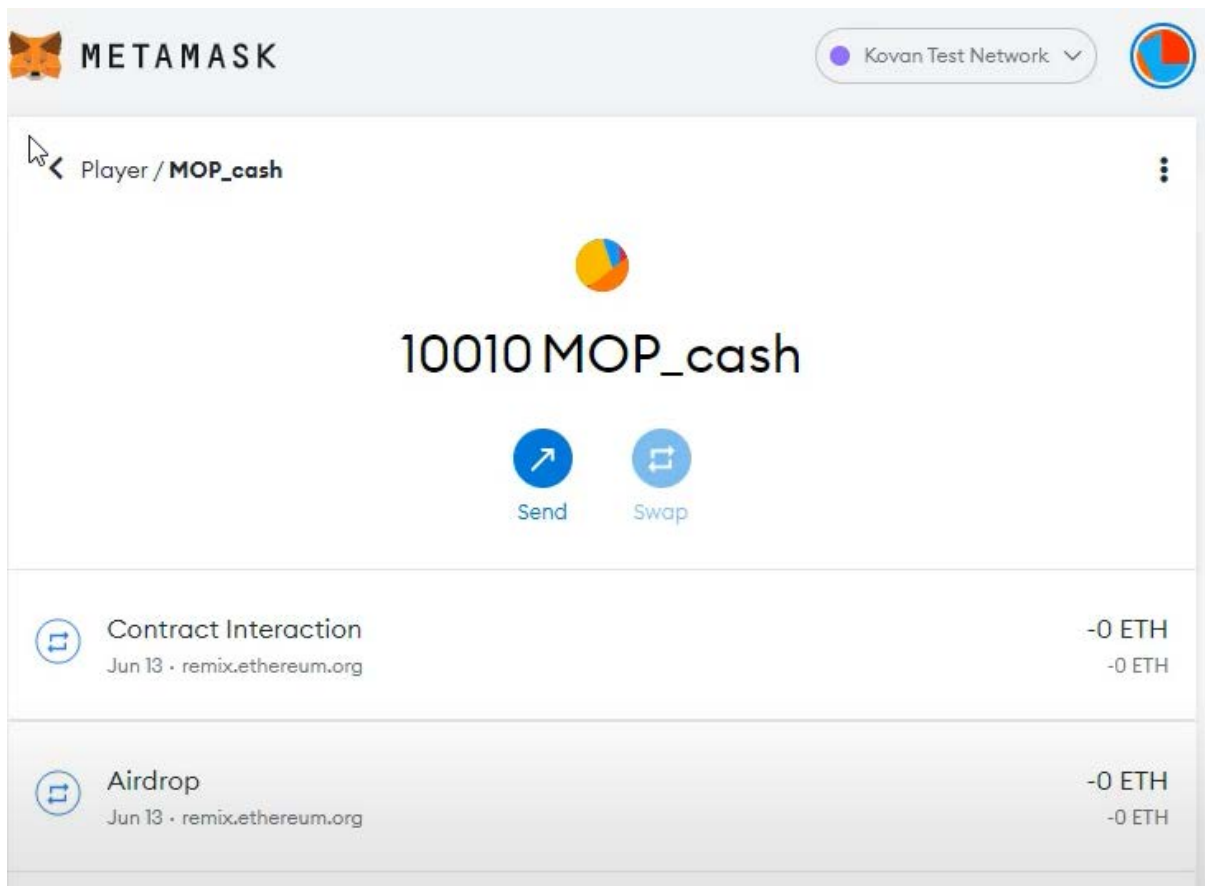
## Test Result

- Airdrop function & Top-up function : After players first register into the game, players will get the first 10,000 MOP coin. Players will get 10,000 MOP in the wallet. The top-up function will show the same result.



**Fig. 5** The first 10,000 MOP in new players' wallet

- Daily Gift: After players log in, they will get free 10 MOP every day.



**Fig. 6** Daily gift 10 MOP will transfer into players' wallet

- Fight the monster: We illustrated the Baby Monsters level and Big Boss Monsters level as an example below.
1. Baby Monsters: player will invest 2 MOP in order to fight with a baby boss monster and gain random reward ranging from 0-5 MOP.




	0xc81755a11f9de8eacc2...	51 mins ago	0x00000000000000000000...	IN	0x519a2a7a231515fee1...	2	
--	--------------------------	-------------	---------------------------	----	-------------------------	---	--

**Fig. 7** Invest on 2 MOP in Baby Monsters level




	0xc81755a11f9de8eacc2...	52 mins ago	0x00000000000000000000...	IN	0x5207324ad9609c0053...	1	
--	--------------------------	-------------	---------------------------	----	-------------------------	---	--

**Fig. 8** Randomly reward in Baby Monsters level

2. Big Boss Monsters: player will invest 500 MOP in order to fight with a big boss monster and gain random reward ranging from 101-1000 MOP.

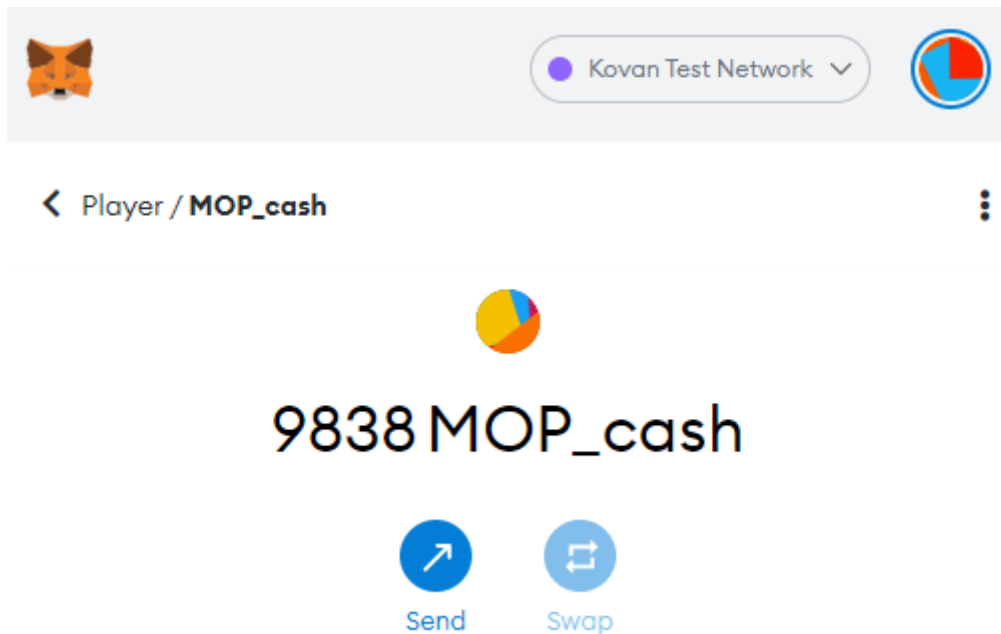
Txn Hash	Age	From		To	Value	Token
 <a href="#">0x90ea66d0875b98ad02...</a>	49 mins ago	<a href="#">0x00000000000000000000...</a>		<a href="#">0x519a2a7a231515fee1...</a>	500	 ERC-20

**Fig. 9** Invest on 500 MOP in Big Boss Monsters level

Txn Hash	Age	From		To	Value	Token
 <a href="#">0x90ea66d0875b98ad02...</a>	50 mins ago	<a href="#">0x00000000000000000000...</a>		<a href="#">0x5207324ad9609c0053...</a>	329	 ERC-20

**Fig. 10** Randomly reward in Big Boss Monsters level

3. After fighting with both of two bosses as mentioned above, the player will have remaining MOP 9838 MOP as shown in Figure 11.



**Fig. 11** Remaining MOP

## Discussion

After developing this blockchain game, we can develop 3 main functions which are Airdrop, Daily gift, and Boss fight. However, this game is only in the developing stage on Environment IDE (i.e., Remix Ethereum IDE) and Web3 interface (i.e., Metamask), the user interface still has not been developed to correspond and work efficiently with the provided solidity code.

## **Conclusion**

In this paper, we developed a novel smart contract application on Ethereum for the gaming industry which is called “Monster Protocol”. We believe that the novel application proposed in this report will leverage the gaming industry to continue to perform outstandingly. In addition, this game will help people more enjoy investing in cryptocurrency. As a result, we have learned about all the important features of blockchain technology. In addition, we covered how this Blockchain feature benefits us and applied its properties into our novel application.

## **Reference**

- [1] Cryptozombies : Learn to Code Blockchain DApps By Building Simple Games. Available[Online]: <https://cryptozombies.io/>.
- [2] Openzeppelin-contracts, Openzeppelin-contracts: A library for secure smart contract development. Build on a solid foundation of community-vetted code. Available[Online]: <https://github.com/OpenZeppelin/openzeppelin-contracts>.
- [3] Ethereum, Solidity Documentation Release 0.8.5, Jun 09, 2021. Available[Online]: <https://docs.soliditylang.org/en/v0.8.5/>.
- [4] Remix - Ethereum IDE. Available[Online]: <http://remix.ethereum.org/>.