# Blockchain: Cryptomon Homework


## Done by: János Gorondi & Benedek Andrási


**Project summary:**
The Cryptomon project is a blockchain-based game that leverages Ethereum smart contracts to facilitate a trading and battling ecosystem for unique digital monsters, each represented as NFTs. Players can buy and battle Cryptomons using various smart contracts that handle transactions, game logic, and player interactions. The game includes a detailed battle system where Cryptomons can engage in turn-based combat, employing strategies through normal and special actions influenced by their stats and evolutionary stages. Each player's actions are governed by state machines ensuring actions such as buying or entering battles are taken at appropriate times. This ecosystem is managed through interconnected contracts that handle everything from minting NFTs and managing even multiple battles, to tracking player states and facilitating transactions.

**Design decisions:**
In the Cryptomon project, several key design decisions were made to ensure a robust and engaging gameplay experience:

- **1. Smart Contract Architecture:** The game utilizes a modular smart contract architecture where different aspects of the game logic are encapsulated in separate contracts, such as `MintManager` for NFT management, `TransactionManager` for handling transactions, and `GameManager` for battle mechanics. This separation enhances maintainability and scalability, allowing each component to be updated or replaced independently.

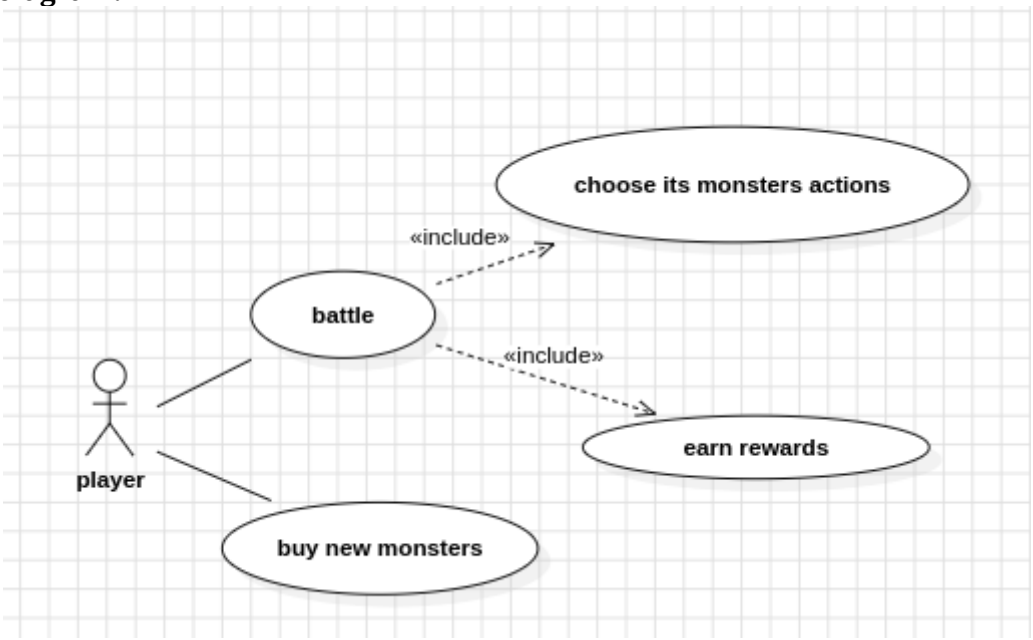
- **2. State Management:** `PlayerStates` contract is employed to manage different states of a player (e.g., IDLE, BATTLING), ensuring that actions are only performed when appropriate. This helps prevent errors and exploits in gameplay, such as buying new Cryptomons while engaged in battle.

- **3. Use of Enumerations and Structs:** The project extensively uses enums to manage states and events, and structs for organizing related data, such as player information and Cryptomon stats. These Solidity features help in writing cleaner, more organized, and more efficient code.

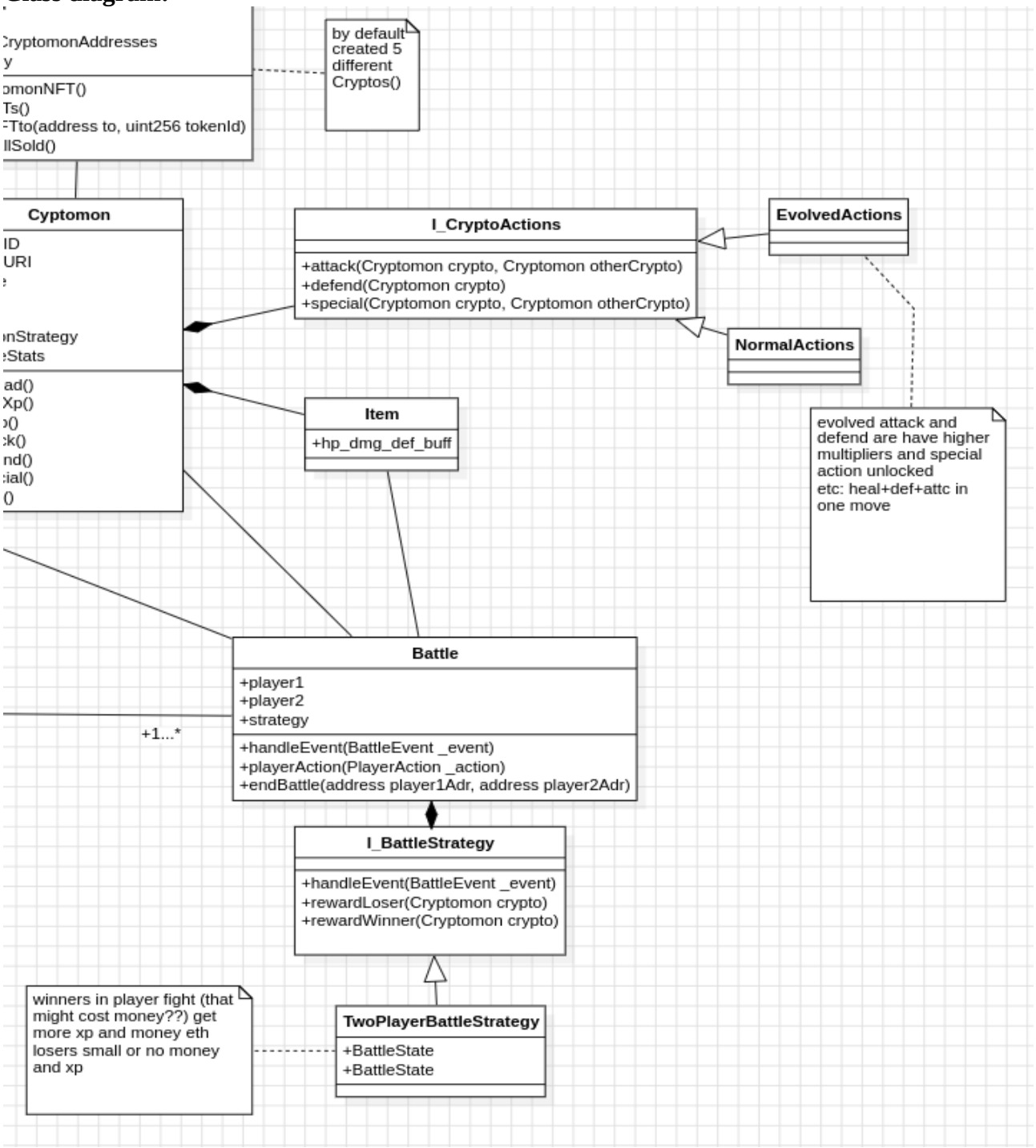
- **4. Security Considerations:** Functions that mutate state or involve asset transfers are protected with access control measures (e.g., `onlyOwner` modifier), ensuring that only authorized users can perform certain operations. The contract also handles potential reentrancy attacks by adhering to checks-effects-interactions patterns, especially in transactional methods.

- **5. Interactivity and Engagement:** The battle system is designed to be interactive and engaging, incorporating strategies through an `IBattleStrategy` interface that allows for different battle tactics and rewards, providing a dynamic and varied gaming experience.
- **6. Future improvements:** Items are already in the game, they have just not been implemented, (it would be pretty quick). Players could fight against NPCs, this is easily enabled by the use of BattleStrategy interface.

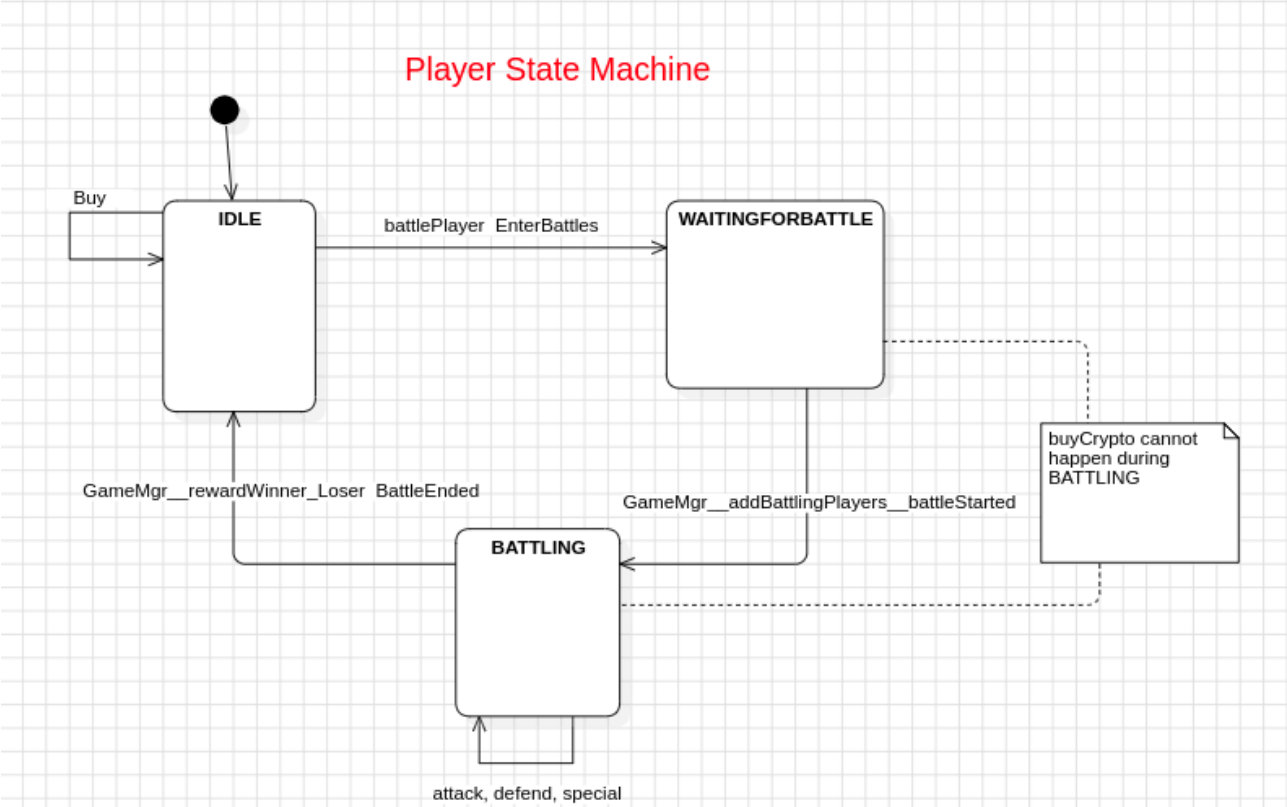
**Data models:**
**Use case diagram:**



## Class diagram:



**State machines:**




**API of the smart contract:**
It is detailed in the ignition/console/consoleREADME.md how a user could test the game.

**Essential implementation details:**
The user of the game interacts through the Player contract. This contracts public functions detail the interface of the game.
We tried to divide responsibility among contacts and used design principles that create easily extensible flexible code.
I created 5 NFTs and hosted them on IPFS as well, here are their links:
**Picures of Cryptomons:**
https://ipfs.io/ipfs/QmNSLi68SDYJMhTdWuemV3Rdw25D1rZ2Dwp4ZbG46borua
https://ipfs.io/ipfs/QmSosMGbzjeKvfnMEERQXQMFT7YoSgUY7VXqBMyQj7tRr1
https://ipfs.io/ipfs/QmYTQDzruheocj8LWYMCjD5hKrx4xcaAsfQKTh6Pqats3h

https://ipfs.io/ipfs/QmT6TYcaSy8taWpeKw8JbJNBXeoMbNE7MRQFxczAyDjN7Y
https://ipfs.io/ipfs/QmQzM8iYKeitqBYriHKHJzjRNFtmF2r3QsfCyrPMBWH8JK

**Metadata of Cryptomons**:
https://ipfs.io/ipfs/QmZ17y3ju3yav3T1LqrcF9o1vct5U5J28ZWPSbSzTFAtpx
https://ipfs.io/ipfs/QmSWuLmVkBzRWTiuqYu3f8gKKRQafuhq8fR3ZJaid1Hb34
https://ipfs.io/ipfs/QmfF1Tv7ZytfEfm3ZrhEfwFrSnewJLwxdZ8iMF7g2rPYB6
https://ipfs.io/ipfs/QmT6TYcaSy8taWpeKw8JbJNBXeoMbNE7MRQFxczAyDjN7Y
https://ipfs.io/ipfs/QmPv6Xycdz4bo6f9PijKuqmJc5eJLVbdc4khgtvEaJAK1g

**Test cases:**
In the beginnig of development we tried to test every functionality, but later on as the possibilities and execution paths grew, and time passed we also became a little lazy. But we believe the amount of tests that we wrote should be enough, it covers ALL mayor use cases.
The code is well commented, fell free to read them if any questions arise.

**Instructions for bootstrapping the project:**
After cloning the repo:
git clone https://github.com/GJanos/Blockchain-HF-GJ-AB.git && cd Blockchain-HF-GJ-AB/

execute these:
npm init -y && npm install && npx hardhat test