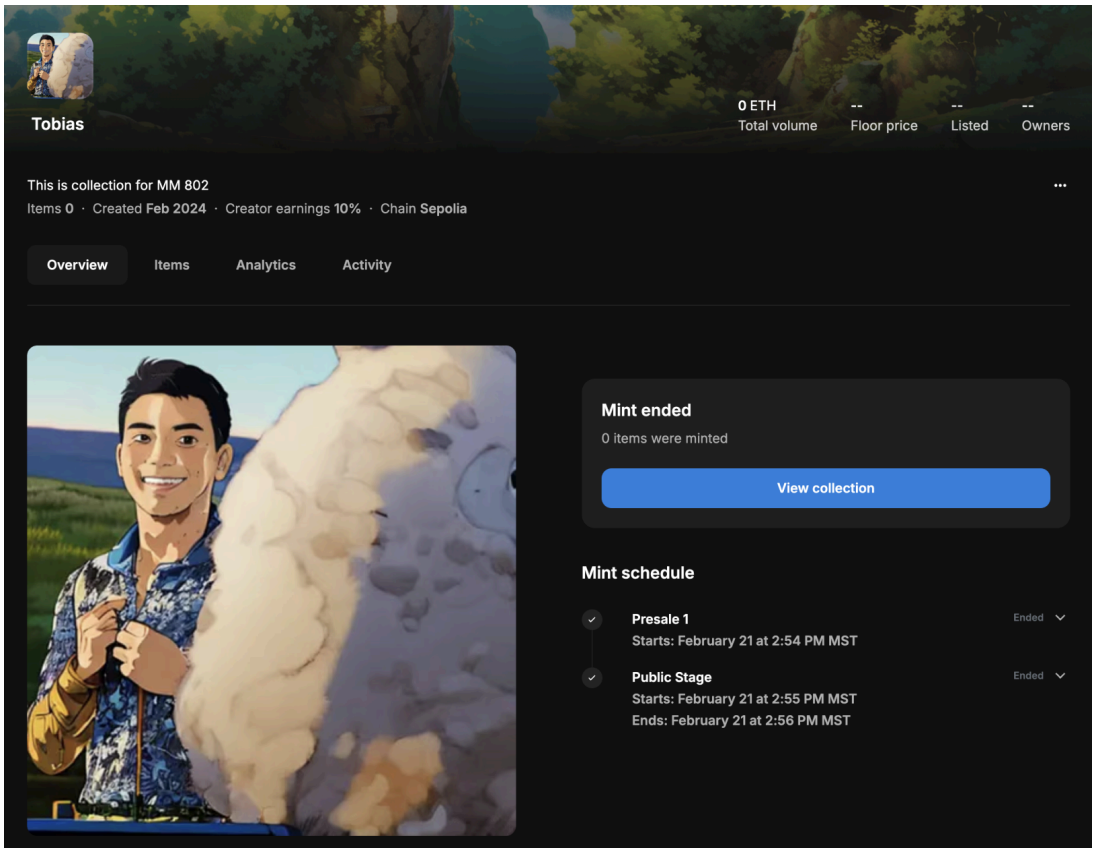
OpenSea (Testnet) collection link

https://testnets.opensea.io/collection/ttobias



Account address that with some balance on Sepolia testnet :
0x5d3d5f8bacd8090c741ede66244ec8f816203227

Contract Address for T1 :
0x41CF4a5e0855959810609ee845934E6016a911D7

Verified : 0x3D7ab43a7781b14658bF15BBF58b1d93Cd9710f2

Transaction Hash for T2 :
0x9448a487f87d0740c92a5197840bc5dbd1ed3c6a0057489ad2cee17fb78cb097

New :
0xbacea42dd4e48a3ac95cfe7bccfbe32084bf7f8782e0d14d9b5531ece6c2123d

contract source code :

```Python
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0;

contract Coin {
    // The keyword "public" makes variables
    // accessible from other contracts
    address public minter;
    string public owner_email;
    mapping(address => uint) public balances;

    // Constructor code is only run when the contract
    // is created
    constructor(string memory email) {
        minter = msg.sender;
        // ###     2 points        ### !!!!!!!!
        // assign owner_email to contract state
        // put your code here
        owner_email = email; // Set the email value
        // ###       END           ###
    }

    // Sends an amount of newly created coins to an address
    // Can only be called by the contract creator
    function mint(address receiver, uint amount) public {
        require(msg.sender == minter);
        require(amount < 1e60);
        balances[receiver] += amount;
    }

    // Sends an amount of existing coins
    // from any caller to an address
    function send(address receiver, uint amount) public {
        require(amount <= balances[msg.sender], "Insufficient balance.");
        balances[msg.sender] -= amount;
        balances[receiver] += amount;
    }

    // ###     2 points        ### !!!!!!!!
    // Return a string as "Hello from " + owner_email
    function hellofromOwner() public view returns (string memory) {
        // put your code here
        return string(abi.encodePacked("Hello from ", owner_email));
```

```
    }
    // ###       END           ###

    // ###     6 points        ### !!!!!!!!
    // Return first 10 integers with a given input {0<number<100} that
satisfied NUM % {number} = 0
    function calculate(uint number) public view returns (uint[] memory) {
        require(number > 0);
        require(number < 100);
        uint[] memory ret = new uint[](10);
        // put your code here
        for (uint i = 0; i < 10; i++) {
            ret[i] = i * number;
        }
        return ret;
    }
    // ###       END           ###

}
```