

# 區塊鏈技術與應用

## Blockchain Techniques and Applications

### Homework 01

#### Guest Lecture: Smart Contract



March 30, 2025

## Week 01

Date: March 11, 2025

### Task & Check Points

CP	Task
✓	準備 MetaMask Wallet。
✓	加入 Holesky & Sepolia。
✓	透過 Faucet 領取 Holesky & Sepolia 的錢。
✓	製作 NFT 的組圖（至少 16 張），並上傳至 IPFS。

#### ✓ 準備 MetaMask Wallet

1. 在 Chrome Web Store 搜索「MetaMask」。
2. 點擊「加到 Chrome」。



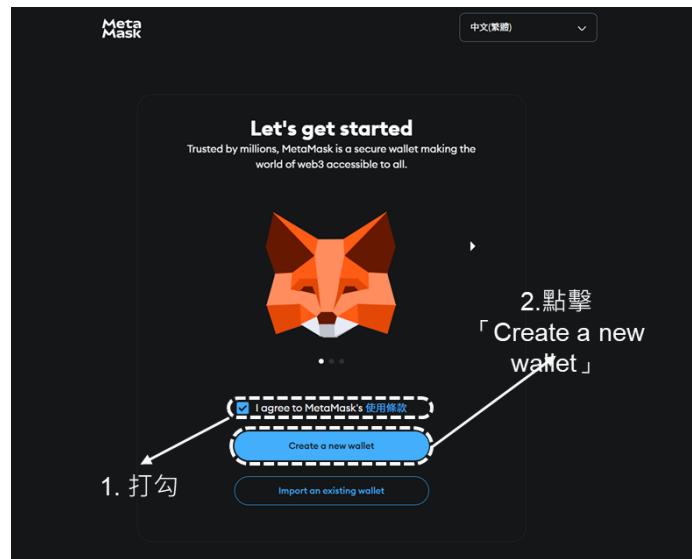
3. 點擊「新增擴充功能」。



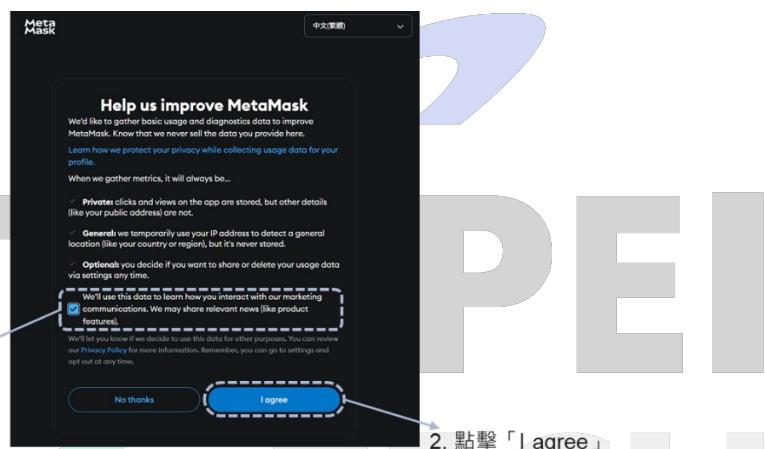
4. 檢查「擴充功能」清單，是否出現 MetaMask，並釘選於佇列。



5. 將「使用條款」進行，以及進行「產生新的錢包」。



6. 將「推播資訊」打勾，並且進行「我同意」。



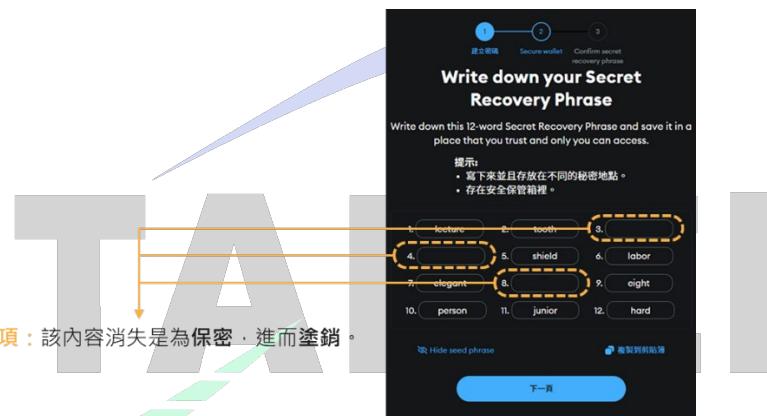
7. 進行建立密碼，接著將「聲明資訊」進行打勾，接著「產生新的錢包」。



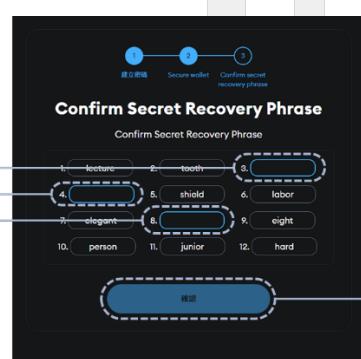
8. 進行「保護我的錢包」的按鈕，接著 Windows 10 和 11 會跳出在 Google Chrome 進行保護政策。



- 接著會出現 12 個「助憶詞」，可以透過截圖，將其順序與內容進行存取，使用複製到剪貼簿雖然方便，但是順序有時會跑掉。

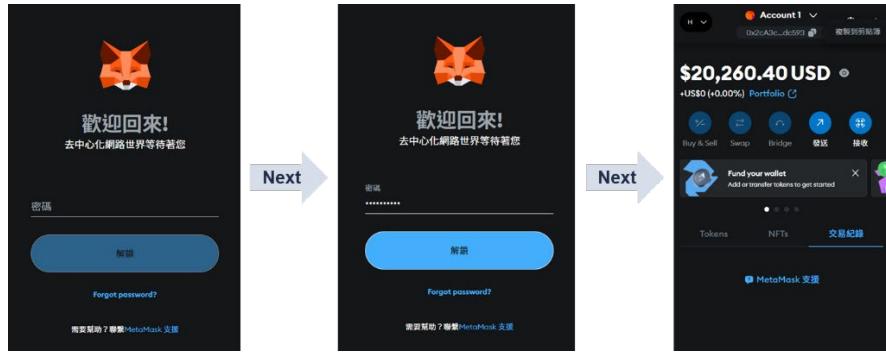


- 紀錄完畢後，進行確認。
- 輸入挖空的「助憶詞」，進行驗證。



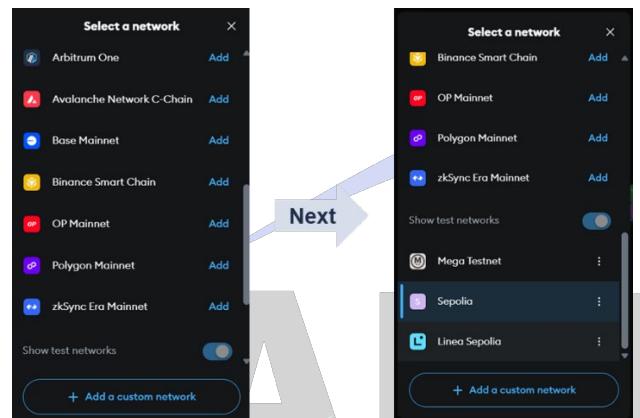
2. 完成後，請點擊「確認」

- 輸入密碼即可進入錢包。

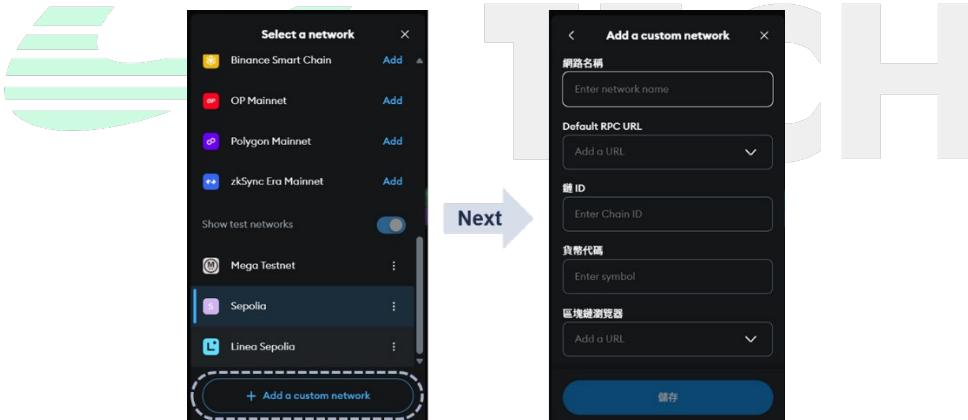


## ✓ 加入 Holesky & Sepolia

1. 進入錢包後，先進行 Sepolia 的新增，因此需將列表最下面的「顯示測試網路」進行啟動，接著 Sepolia 就能出現。



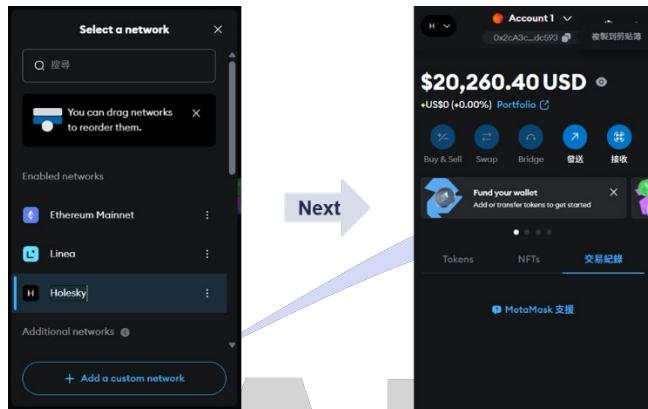
2. Holesky 的新增需要使用「加入客製化網路」，因此點擊其按鈕，接著看到許多欄位。



3. 對應欄位將依序輸入，相關內容已於下圖展示，將不另贅述。

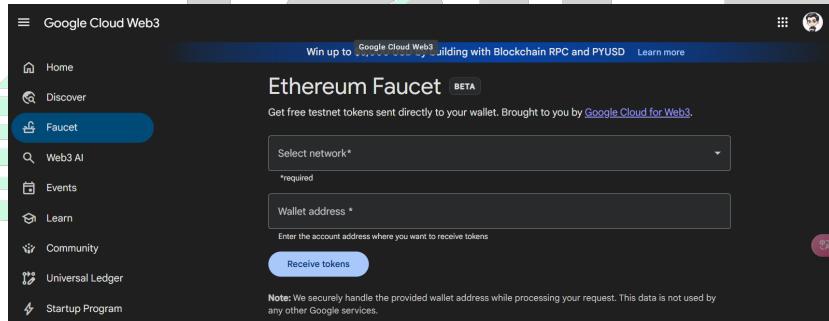


4. 完成第 3 步驟，將會看見有 Holesky 的錢包。

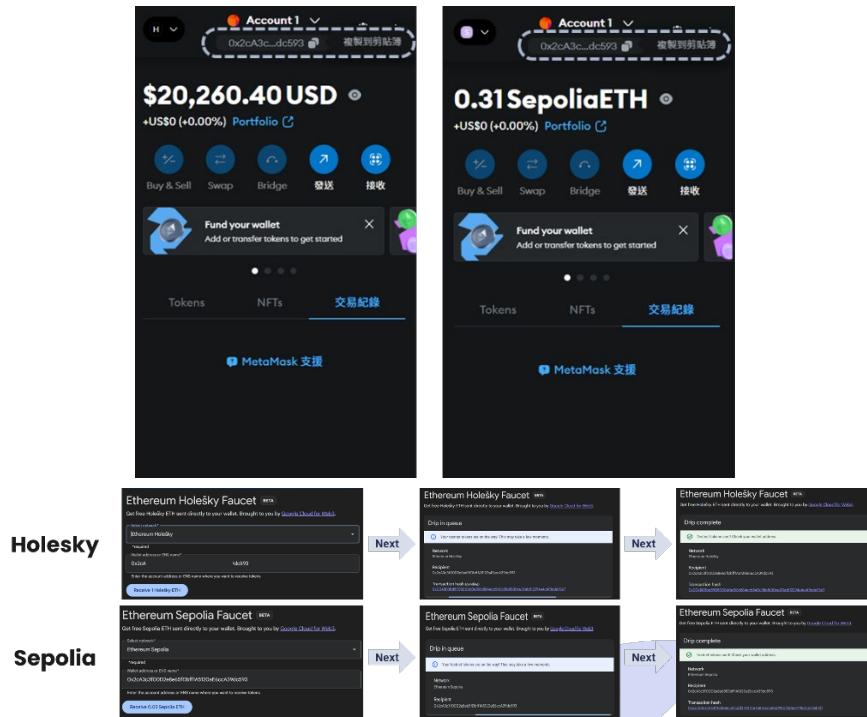


#### ✓ 透過 Faucet 領取 Holesky & Sepolia 的錢

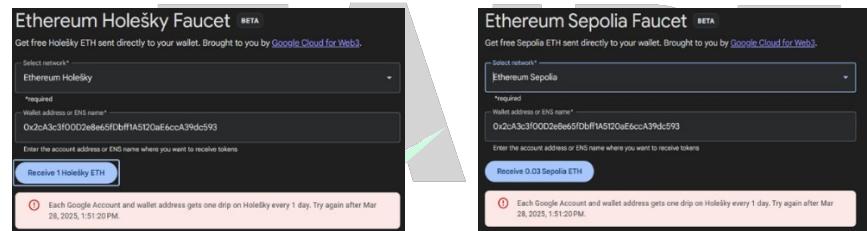
1. 將前往 Google Cloud Web3 – Ethereum Faucet 這個網頁進行領錢的手續。



2. 在選單中將分別針對 Holesky 和 Sepolia 進行選擇，並填入 Wallet Address，接著完成提領，這過程將會需要數秒鐘的等待。



3. 當領取完畢，並再次點擊領取時，將會出現需要一天後，才能再次領取的警告。

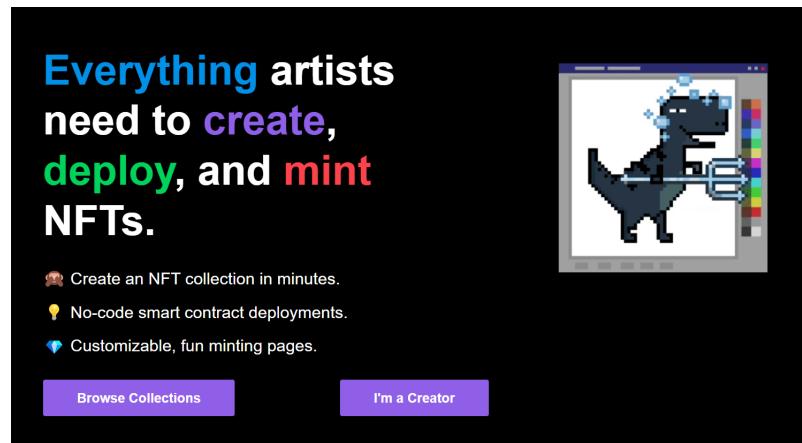


✓ 製作 NFT 的組圖（16 張），並上傳至 IPFS

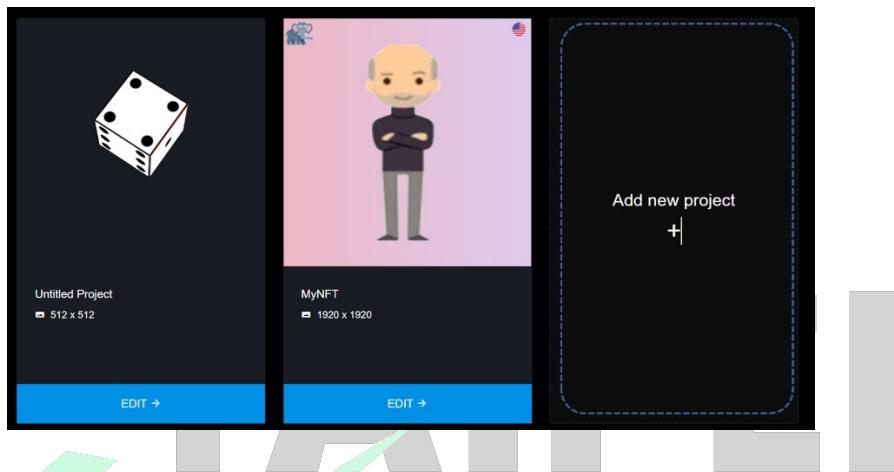
1. 先完成 Background、Flag、Body、Animal 等圖片的蒐集，並將尺寸為 1920 x 1920，並對於 Background 外的其他圖片需要去除背景，避免產生時遭到圖層覆蓋。
2. 由於 4 個 Layer，我對於每個類別規劃將做出 6 張圖，因此將會獲得  $6 \times 6 \times 6 \times 6 = 1296$  張的隨機產生圖。

Layer	Count	Image
Background	6	[Six colored squares: light green, medium green, pink, yellow, blue, purple]
Animal	6	[Six empty white boxes]
Body	6	[Six human figures: man, woman, child, etc.]
Flag	6	[Six empty white boxes]

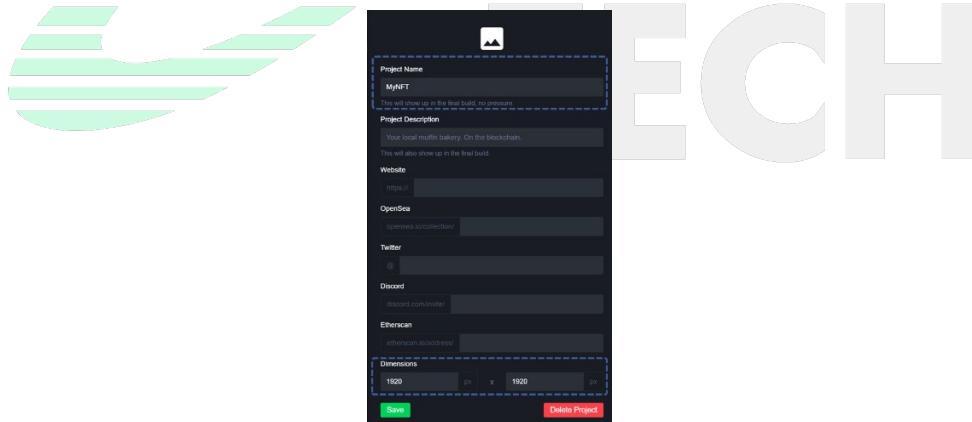
3. 網路搜尋「Mintrables」，並註冊。



4. 首先建立一個「我的專案」。



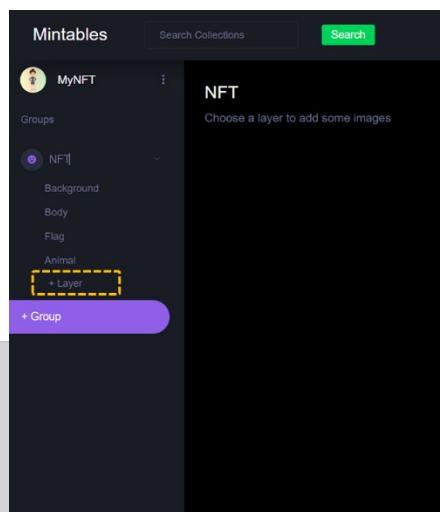
5. 接著對專案名稱命名，並設定尺寸為圖片所用的 1920x1920。



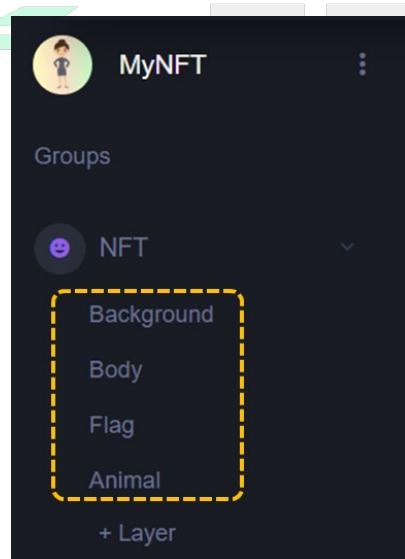
6. 在專案中「新增群組 (+Group)」。



7. 在群組中依據圖片的類別 Background、Flag、Body、Animal 等，新增圖層 (Layer)，其順序相當重要，需要避免 Background 蓋到其他圖層。



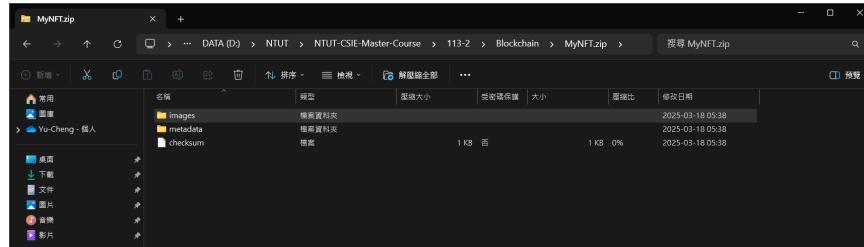
8. 對於每一個圖層 (Layer) 進行圖片新增。



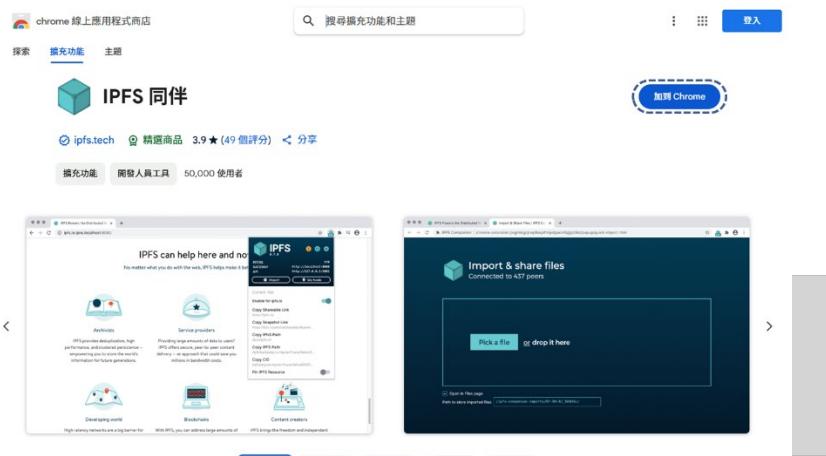
9. 接著進行圖片產生。

10. 圖片產生時預設為 10000，單因排列組合僅有 1296，故不能大於 1296，但避免後續使用的 FileBase 等限制，故僅產生 18 張，其編號將由 0 至 17 進行編號。

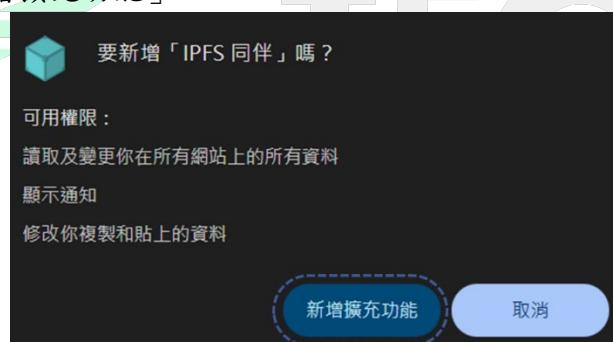
11. 產生完畢後，將會在電腦本地端收到與專案名稱一樣的壓縮檔。
12. 壓縮檔進行解壓縮後，將會獲得與壓縮檔同名的資料夾。
13. 在該資料夾下，會有 metadata 和 images 兩個資料夾。分別儲存 0 ~ 17 共 18 張的資料及圖片。



14. 在 Chrome Web Store 搜索「IPFS 同伴」，並完成安裝。



15. 點擊「新增擴充功能」。



16. 將會看到歡迎頁。

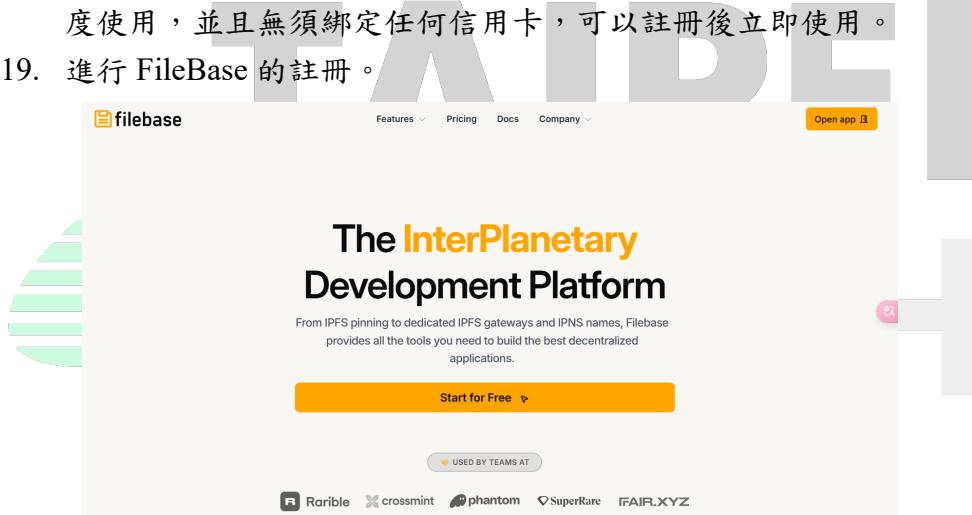


17. 在「擴充功能」清單中也能找到該擴充套件，點擊後將能看到以下畫面。

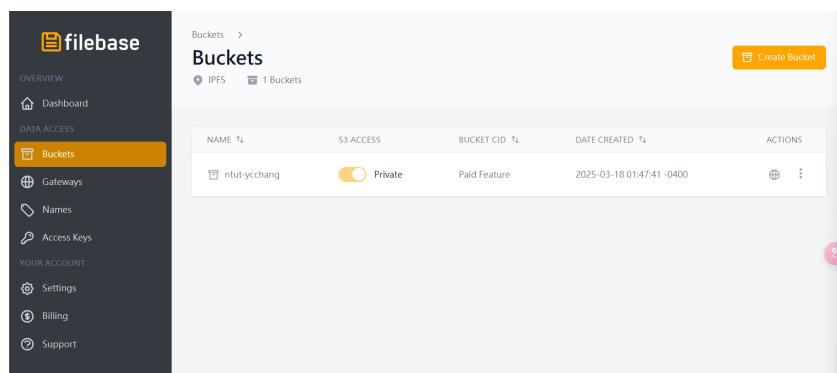


18. 搜索與前往 FileBase 的網站，在該網站中提供 5GB 及 1000 個檔案的額度使用，並且無須綁定任何信用卡，可以註冊後立即使用。

19. 進行 FileBase 的註冊。



20. 註冊完畢將進行 Bucket 的建立，其名稱可以自行命名，Storage Network 將為預設的 IPFS (Always public)。

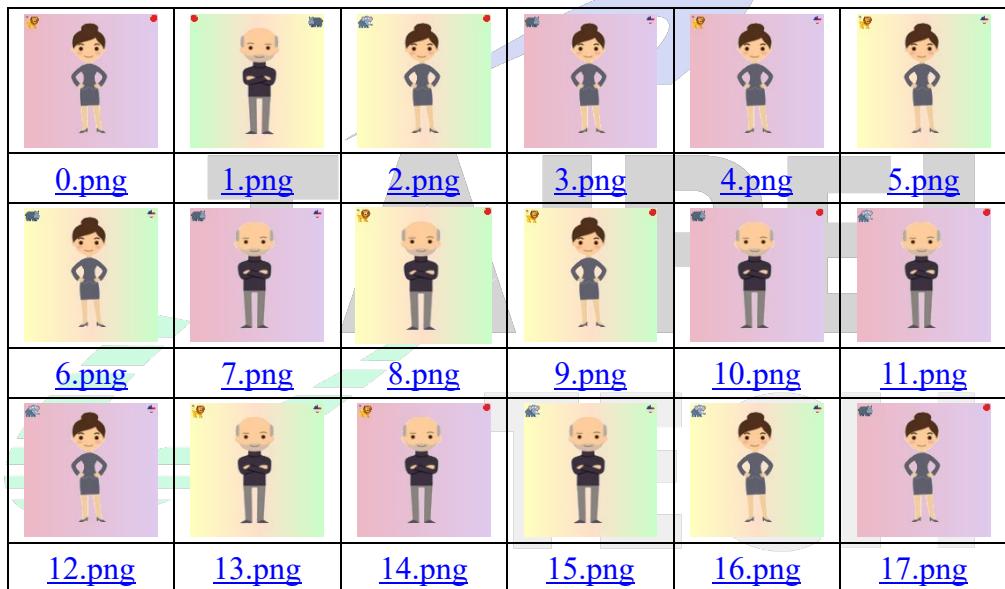


21. 進入建立完成的 Bucket 後進行上傳，上傳內容為第 13 步驟的 images 的資料夾。
22. 上傳完畢後，每張圖片皆有 IPFS 網址，將其與 metadata 中的圖片之 images 的連結進行更換。
23. 進行上傳 metadata 資料夾。

The screenshot shows the Firebase Storage interface. On the left, there's a sidebar with options like Overview, Dashboard, Buckets, Gateways, Names, Access Keys, Settings, Billing, and Support. The main area shows the 'ntut-ycchang' bucket. It has a summary bar at the top with 'IPFS' (14.2 MB), '2 Objects'. Below it is a table with columns: NAME, CID, STATUS, LAST MODIFIED, SIZE, and ACTIONS. Two objects are listed:

NAME	CID	STATUS	LAST MODIFIED	SIZE	ACTIONS
images	QmULZjDji6U8BZCrUVXJyAoZAPSUTXAEGevY673fcPACaU	Pinned	2025-03-18 01:49:47 -0400	14.1 MB	
metadata	QmNUsqEod9QrYrxn8mhyGkcybo...	Pinned	2025-03-18 01:57:59 -0400	9.91 KB	

24. 接著可以檢查看看 metadata 是否有產生資料夾的 IPFS。



➤ IPFS Gateway URL

<https://shallow-salmon-marmot.firebaseio.com/ipfs/QmULZjDji6U8BZCrUVXJyAoZAPSUTXAEGevY673fcPACaU>

## Week 02

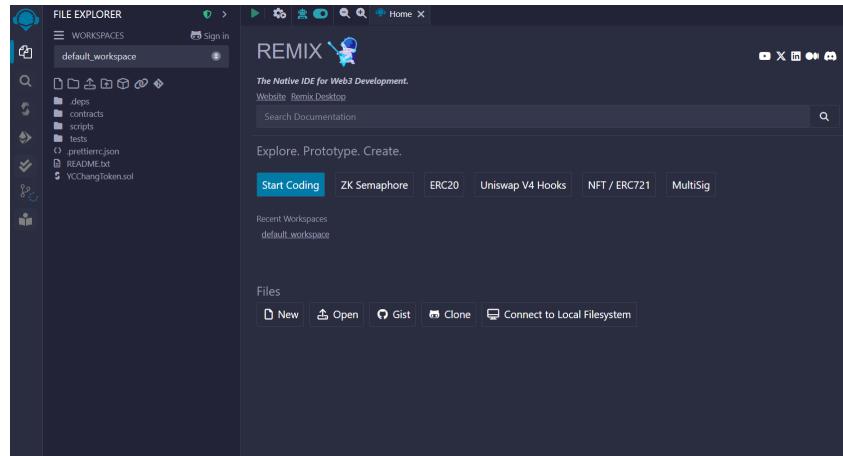
Date: March 18, 2025

### Task & Check Points

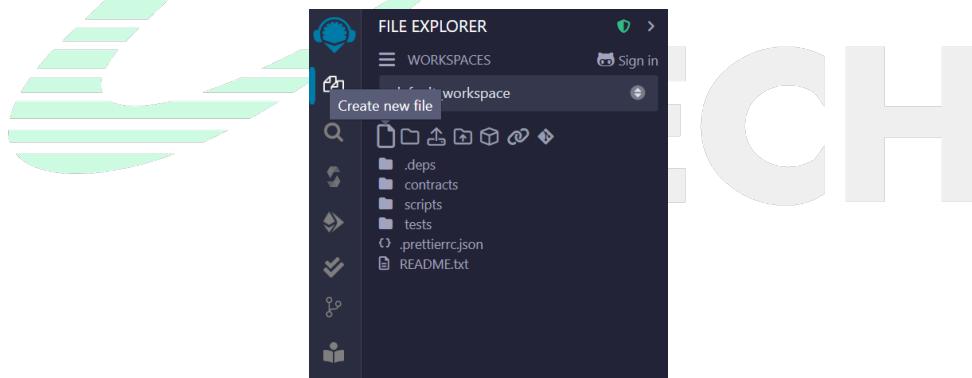
CP	Task
✓	以 ERC-20 佈建 Sepolia 的鏈。
✓	使用合約各項功能(須包含多個使用者)。
✓	額外功能 Buy & Burn。
✓	OnlyOwner 相關功能測試，請給出不是 Owner 的失敗測試。

- ✓ 以 ERC-20 佈建 Sepolia 的鏈

- 進入「Remix - Ethereum IDE」網頁。



- 點擊「建立新檔案」，並命名為「YCToken.sol」，這名稱可自行命名。



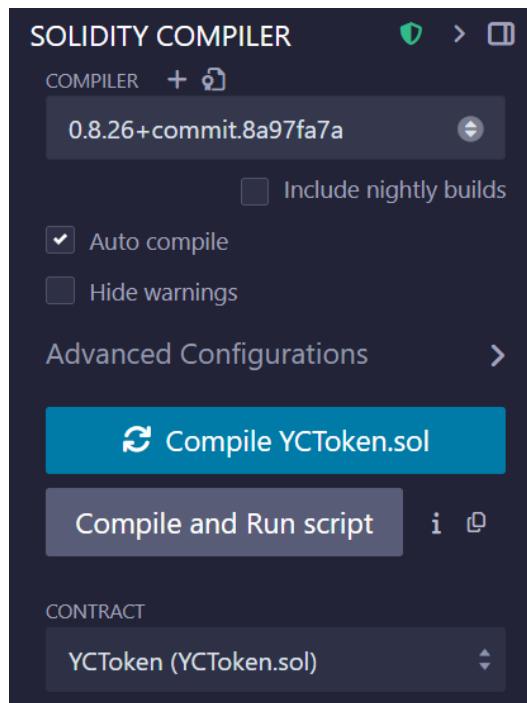
- 對於「YCToken.sol」完成程式撰寫。

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

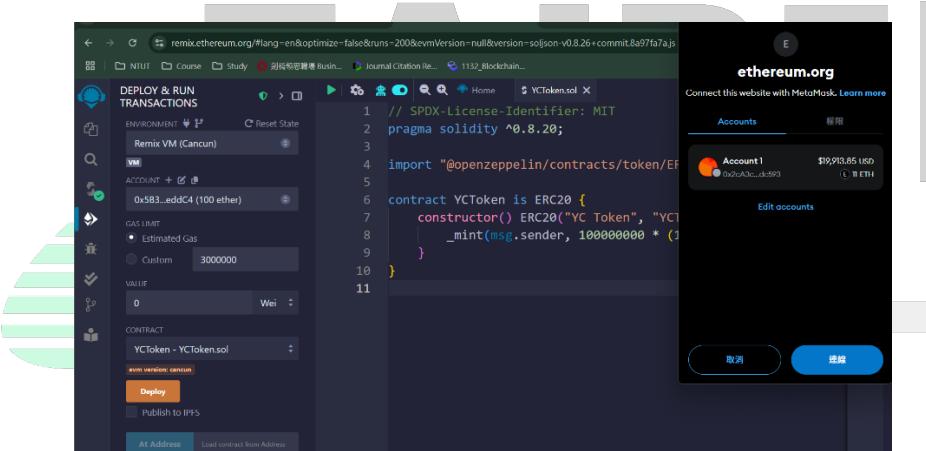
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract YCToken is ERC20 {
    constructor() ERC20("YC Token", "YCT") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }
}
```

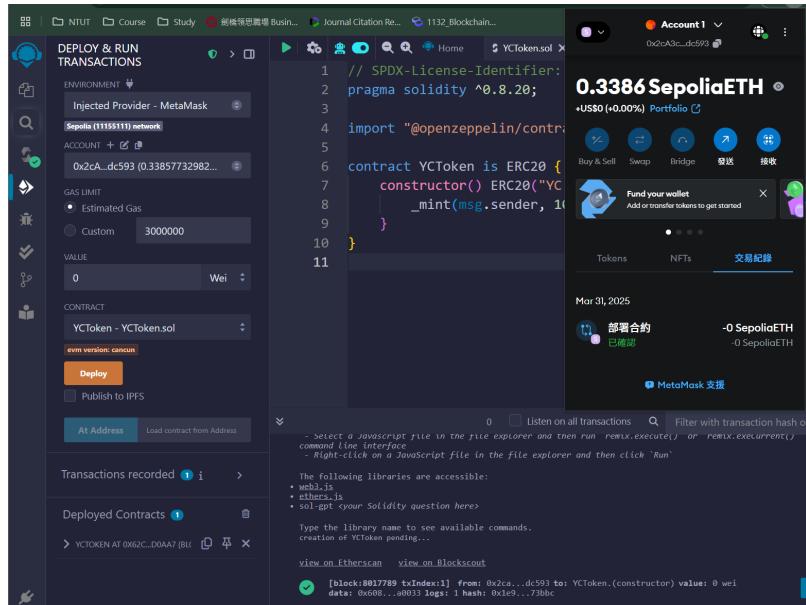
- 勾選「Auto Complie」，並點擊「Complie YCToken.sol」。



5. 會發現在 File Explorer 多出許多檔案。
6. 接著到「Deploy & Run Transactions」的分頁中，進行連線與 Deploy。



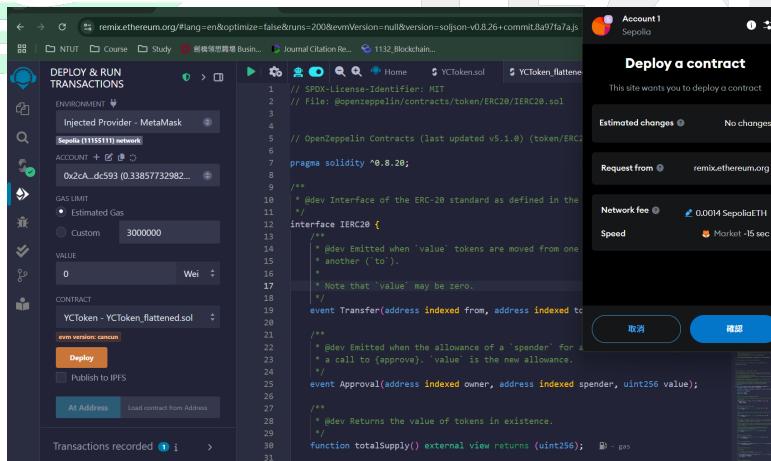
7. 經過和交易與核准後，將會得到以下畫面。



➤ View Sepolia Transaction

<https://sepolia.etherscan.io/tx/0x2f68a11cff02b4980ff0b13e28a724102b7e48ea9642a69eb95342c5636731e5>

8. 對「YCToken.sol」進行 Flatten。
9. 在產生的「YCToken\_flattened.sol」會缺少「// SPDX-License-Identifier: MIT」，需要協助補上。
10. 到 Solidity Compiler 分頁設定「Auto Complie」和「Context (YCToken\_flattened.sol)」，並進行 Complie。
11. 進行設定為「Injected Provider - MetaMask」與「YCToken – YCToken\_flattened.sol」後進行 Deploy 和確認。



➤ View Sepolia Transaction

<https://sepolia.etherscan.io/tx/0xb835b6c7421141e019f6a7446e202744f00bd44260bcd591c8099e25911303ac>

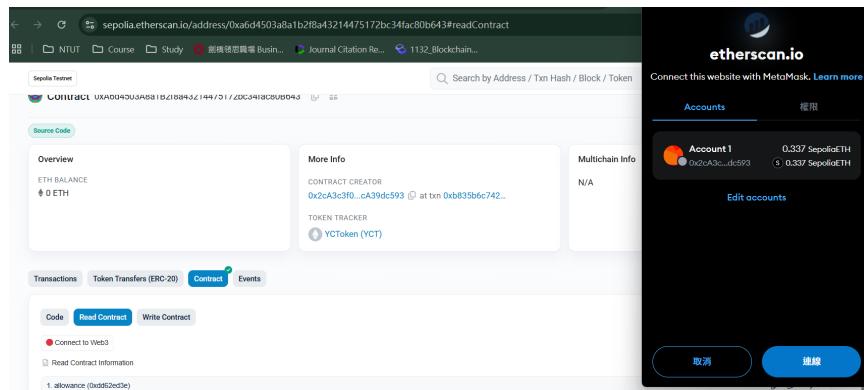
12. 透過交易後 Address 網頁找到 Contract，並進入 Verify and Public 超連結，進行驗證與公開。

13. 驗證時，需填寫 Address 和 Solidity (Single file)，並選擇 v0.8.26+commit 8a97fa7a 和 MIT License (MIT)，接著貼上「YCToken\_flattened.sol」裡面的 code 進行驗證，接著就會公開你的原始碼。

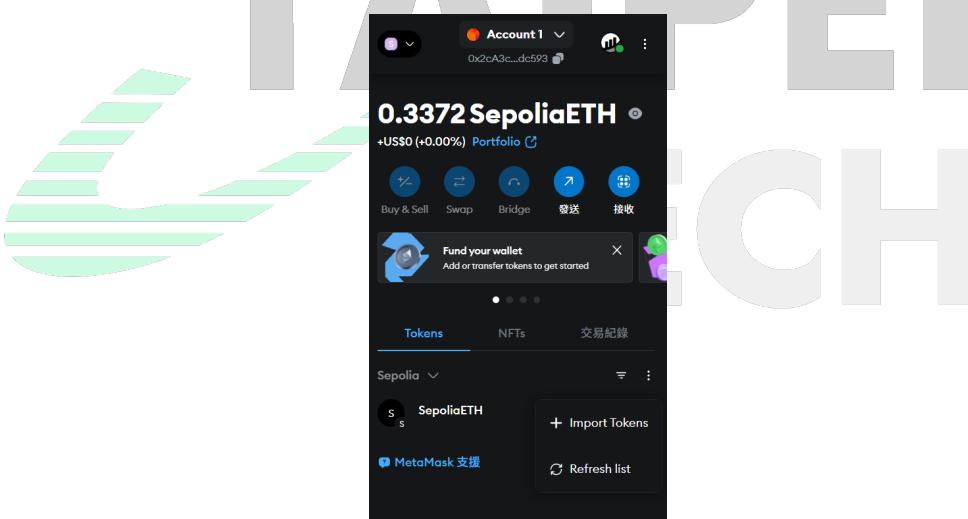
➤ View Contract Source code

<https://sepolia.etherscan.io/address/0xa6d4503a8a1b2f8a43214475172bc34fac80b643#code>

14. 連接 MetaMask Wallet。



15. 在網頁進行 blanceOF，在此需填入 address。  
 16. 返回 Deploy & Run Transactions，下方也要進行 blanceOF。  
 17. 進行 Import Tokens 之前需要到網頁複製。  
 18. 進行 Import Tokens 的時候會找不到選單，可遵循下圖找到該項目。

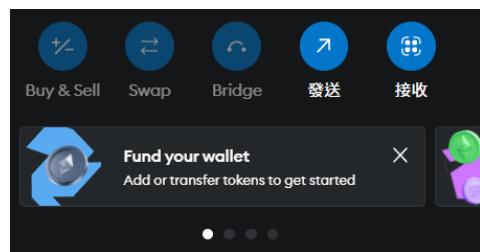


19. 接著只要貼上 address，並點擊下一步即可，成果如下。



- ✓ 使用合約各項功能(須包含多個使用者)

1. 傳送 Token 的環節，僅需點擊發送。



2. 設定好金額與接收對象，就能進行轉帳。



✓ 預外功能 Buy & Burn

1. 在 Buy 的部分前期與上個小結相同，僅將程式碼改為下圖。

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract YCToken is ERC20 {
    uint256 price = 0.0000001 ether;

    constructor () ERC20("YCToken", "YCT") {
        _mint(msg.sender, 10000000 * (10 ** uint256(decimals())));
    }

    function buy() external payable {
        require(msg.value > 0, "You must send some ether");
        _mint(msg.sender, msg.value * 10 ** decimals() / price);
    }

    function burn(uint256 amount) external {
        _burn(msg.sender, amount);
    }
}
```

➤ View Contract Source code

<https://sepolia.etherscan.io/address/0x9ffe9583727864a7e6bf795e594d3265b5f7b742#code>

2. Buy Token 與上述相同，不再贅述。

➤ View Token YCToken

<https://sepolia.etherscan.io/token/0x9ffe9583727864a7e6bf795e594d3265b5f7b742>

### 3. Burn Token 與上述相同，不再贅述。

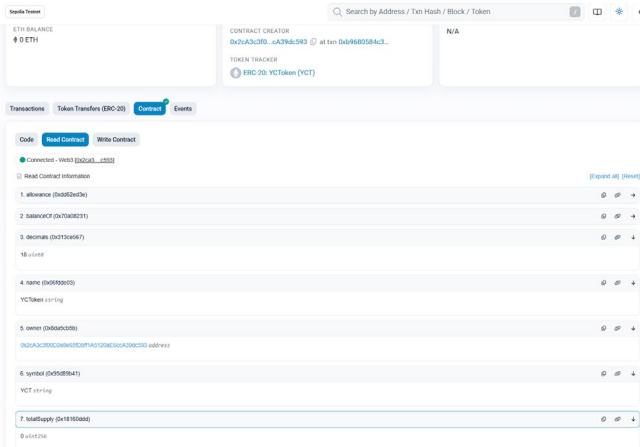
➤ View Token YCToken

<https://sepolia.etherscan.io/token/0x9ffe9583727864a7e6bf795e594d3265b5f7b742>

- ✓ OnlyOwner 相關功能測試，請給出不是 Owner 的失敗測試。

### 1. Openzeppelin v4.9.0 將程式碼修改為下圖樣式。

### 2. 經過如上設定後驗證交易。



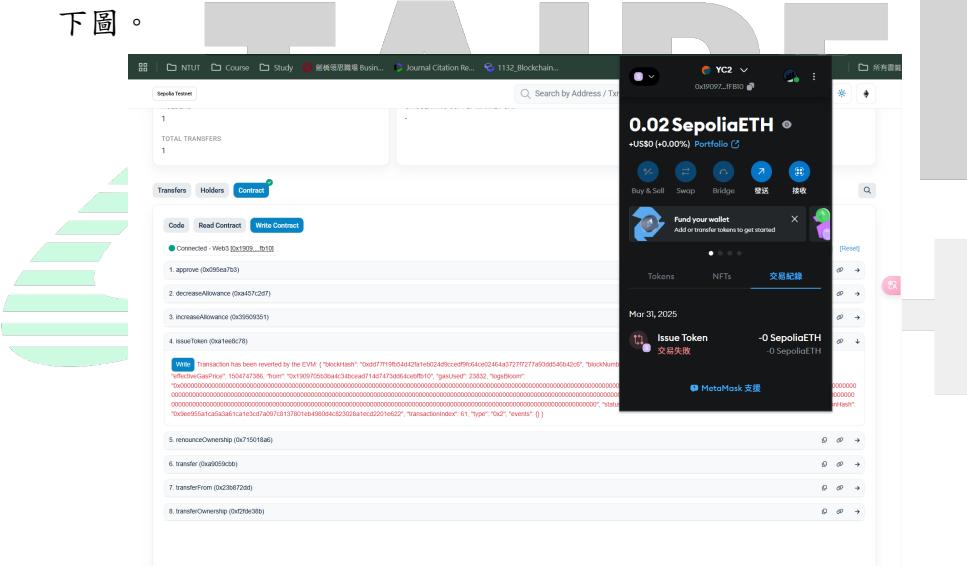
➤ View Contract Source code

<https://sepolia.etherscan.io/address/0x6502cdd191cf926fbcd78374c7d0fa24ce50a626#code>

➤ View Transaction

<https://sepolia.etherscan.io/token/0x6502cdd191cf926fbcd78374c7d0fa24ce50a626>

3. 切換到另一帳號，進行 Write Contract，由於不是 Owner 所以失敗，如下圖。



➤ View Transaction Details

<https://sepolia.etherscan.io/tx/0x9ee955a1ca5a3a61ca1e3cd7a097c8137801eb4980d4c823028a1ecd2201e622>

4. Openzeppelin v5.1.0

The screenshot shows the Sepolia Etherscan interface for a specific Ethereum contract. The address is 0x50d6841ad9602e1f50796e031ecb9544bf8542b. The page includes sections for Overview, More Info (Contract Creator: 0x2a3c3f0...cA39dc593 at tx 0x83297dfc599...), and Multichain Info (N/A). The Transactions section shows one transaction: Issue Token (tx 8018684) from 0x2cA3c3f0...cA39dc593 to 0x50d6841a...4BfE8542b with 0 ETH amount and 0.00010542 gas fee. A note at the bottom states: "A contract address hosts a smart contract, which is a set of code stored on the blockchain that runs when predetermined conditions are met. Learn more about addresses in our Knowledge Base." There are download options for CSV and JSON.

➤ View Contract Source code

<https://sepolia.etherscan.io/address/0x50d6841ad9602e1f50796e031ecb9544bf8542b#code>

➤ View Token YCToken

<https://sepolia.etherscan.io/token/0x50d6841ad9602e1f50796e031ecb9544bf8542b>

5. 切換到另一帳號，進行 Write Contract，一樣交易失敗。

The screenshot shows the Sepolia Etherscan interface with a transaction request overlay. The target contract is 0x50d6841ad9602e1f50796e031ecb9544bf8542b. The transaction request window displays an error message: "Estimated changes: This transaction is likely to fail". It also shows the transaction details: Request from sepolia.etherscan.io, Interacting with 0x50d6841a...4BfE8542b, Network fee 0.0189 SepoliaETH, and Speed Market - 15 sec. The transaction itself is listed in the history with a status of "Failed".

## Week 03

Date: March 25, 2025

### Task & Check Points

CP	Task
✓	ERC-721 °

### Extra Points (Bonus)

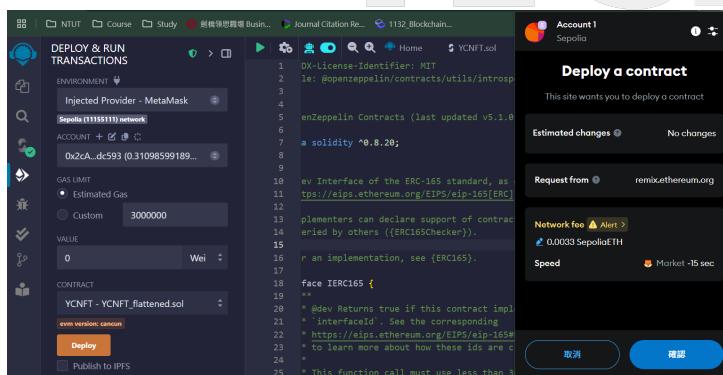
CP	Task
✓	ERC-1155
✓	ERC 721A

✓ ERC-721

- 建立「YCNFT.sol」，如下圖之程式碼。

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3 import "@openzeppelin/contracts/token/ERC721/ERC721.sol";
4 import "@openzeppelin/contracts/utils/Counters.sol";
5 import "@openzeppelin/contracts/access/Ownable.sol";
6 import "@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.sol";
7
8 contract SaluNFT is ERC721URIStorage, Ownable(address(msg.sender)) {
9
10     using Counters for Counters.Counter;
11     Counters.Counter private _tokenIds;
12
13     constructor() ERC721("YCNFT", "NFT") {}    // infinite gas 1893000 gas
14
15     function mintNFT(address recipient, string memory tokenURI) public onlyOwner returns (uint256)
16     {
17         _tokenIds.increment();
18         uint256 newItemId = _tokenIds.current();
19         _mint(recipient, newItemId);
20         _setTokenURI(newItemId, tokenURI);
21         return newItemId;
22     }
23
24 }
25
26 }
```

- 進行 Flatten。
- 進行 Complie。
- 進行 Deploy & Run Transactions。
- Deploy a contract 確認。



- 進行驗證與公開交易原始碼。

➤ View Constant Source Code

<https://sepolia.etherscan.io/address/0x2186e828141a53e9fd15b85efba3fddb38f7d7dc#code>

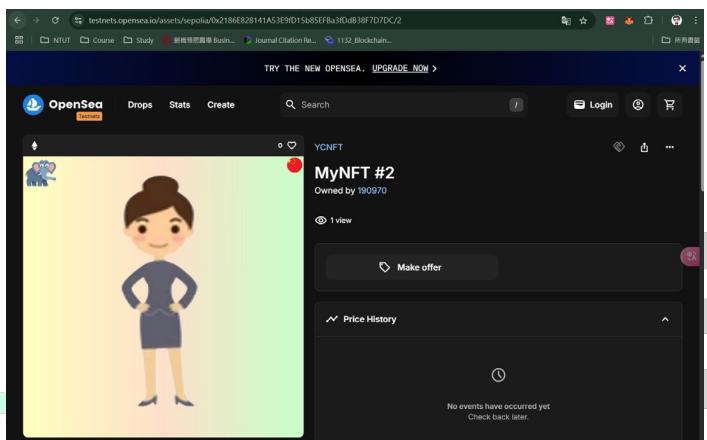
- Query tokenURI

```

1. balanceOf (0x70a08231)
2. getApproved (0x01812fcf)
3. isApprovedForAll (0xe985e9c5)
4. name (0x0fddde03)
5. owner (0xbda5cb5b)
6. ownerOf (0x6352211e)
7. supportsInterface (0x01ffc9a7)
8. symbol (0x95d89b41)
9. tokenURI (0xc87b56dd)
tokenId (uint256)
2
Query
└ string
[ tokenURI(uint256) method Response ]
└> string : https://shallow-salmon-marmot.firebaseio.com/pfs/QmNUSqEod9QzYrxon9lmhGKcYbomB2GF5sEd7ufB9XGpW2

```

## 8. 使用 OpenSea testnet 查看。



➤ OpenSea URL

<https://testnets.opensea.io/assets/sepolia/0x2186E828141A53E9fD15b85EFBa3fDd38F7D7DC/2>

## ✓ ERC-1155

### 1. 先將程式碼改為以下圖片的形式。

```

// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
import "https://sepolia.org/contracts/foundry/ERC1155.sol";
contract YCGameToken is ERC1155 {
    uint256 public constant YC_NFT = 1;
    uint256 public constant YC_NFT_2 = 2;
    uint256 public constant YC_NFT_3 = 3;
    uint256 public constant YC_NFT_4 = 4;
    string public symbol;
    string public name;
    constructor() ERC1155("https://nftgen-retired-cardsuite-908.appspot.com/contracts/erc1155/ERC1155.sol") {
        symbol = "YC_NFT";
        name = "YC_NFT";
        _mint(msg.sender, YC_NFT, 1);
        _mint(msg.sender, YC_NFT_2, 1);
        _mint(msg.sender, YC_NFT_3, 1);
        _mint(msg.sender, YC_NFT_4, 1);
    }
    function uri(uint256 tokenId) override public view returns (string memory) {
        return string(abi.encodePacked("https://nftgen-retired-cardsuite-908.appspot.com/contracts/erc1155/ERC1155.sol?tokenid=",
            Strings.toString(tokenId), ".json"));
    }
    function contractURI() public view returns (string memory) {
        return string(abi.encodePacked("https://nftgen-retired-cardsuite-908.appspot.com/contracts/erc1155/ERC1155.sol?contract=",
            "description='Welcome to my first ERC1155 collection project.'",
            "external_link='https://www.google.com/'",
            "image='https://nftgen-retired-cardsuite-908.appspot.com/contracts/erc1155/ERC1155.sol?image=1.jpg&name=ERC1155%20Logo%20Image%201.jpg'",
            "feature_image='https://nftgen-retired-cardsuite-908.appspot.com/contracts/erc1155/ERC1155.sol?image=2.jpg&name=ERC1155%20Logo%20Image%202.jpg'",
            "collaborators='[{"name": "nftgen", "url": "https://nftgen.com"}]'"));
    }
}

```

### 2. 其他與先前的設定方式相同。

➤ View Contract Source Code

<https://sepolia.etherscan.io/address/0xd2f7f0f1bd7beb8bd5ba69d4727e0e26b1a4781c#code>

Sepolia Testnet		Search by Address / Txn Hash / Block / Token	
<input type="checkbox"/>	Read Contract Information		<span>[Expand all] [Reset]</span>
1.	GOLD (0x6ebeef8)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
2.	SHELD (0x502725ed)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
3.	SILVER (0xe3e5508)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
4.	SWORD (0x7ac1d0f9)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
5.	ThORG_HAMMER (0xd26a2c24)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
6.	balanceOf (0x009505e)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
7.	balanceOfEth (0x4e127f3)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
8.	contractURI (0x7eaf3d4b)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
9.	isApprovedForAll (0x7080cfc2)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
10.	name (0x909de0c)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
11.	supportsInterface (0x1f1fc37)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
12.	symbol (0x50d9f541)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
13.	uri (0x6989341c)	<span>0</span>	<span>0</span> <span>0</span> <span>+</span>
_skewer (0x1f246)			<span>0</span> <span>0</span> <span>+</span>
0			
<b>Query</b>			
<input checked="" type="checkbox"/>	string		
<input type="checkbox"/>	uint256		
0x	string	<a href="https://rinkeby.sairento.marinaxy.net/api/tx?txHash=2750f715c1a52a944d204f1bf1baa4551fb4234b75.jso">https://rinkeby.sairento.marinaxy.net/api/tx?txHash=2750f715c1a52a944d204f1bf1baa4551fb4234b75.jso</a>	

#### ➤ OpenSea URL

<https://testnets.opensea.io/assets/sepolia/0xd2f7f0f1bd7beb8bd5ba69d4727e0e26b1a4781c/0>

✓ ERC-721A

1. 前期須將「YC721A.sol」程式碼修改為下圖的形式。

2. 進行 Flatten。
  3. 進行 Complie。
  4. Deploy & Run Transaction 的部分依據過往形式選擇，但 Deploy 需要展開選單並且填寫 Name 和 Symbol，如下圖。

**DEPLOY**

NAME: YC721A

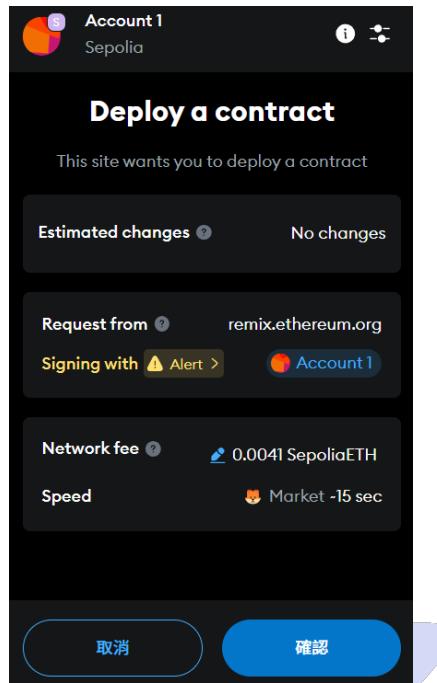
symbol: MILU

Calldata     Parameters    **transact**

Publish to IPFS

**At Address**    Load contract from Address

5. 進行 Transact。
  6. 進行確認。



- View Contract Source Code  
<https://sepolia.etherscan.io/address/0x5ffd210ef39b610444d3b6a9c72470c79acc0ff4#code>

## 7. 閱讀 Contract。



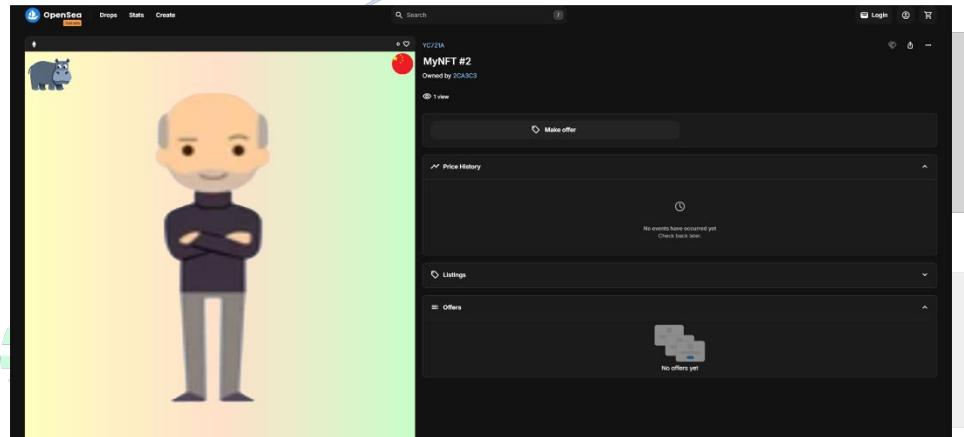
## 8. 進行鑄造。

Estimated changes

You send - 0.003 SepoliaETH

You receive + #1 0x5ffd2...c0ff4  
+ #2 0x5ffd2...c0ff4  
+ #3 0x5ffd2...c0ff4  
+ #4 0x5ffd2...c0ff4  
+ #5 0x5ffd2...c0ff4  
+ #6 0x5ffd2...c0ff4  
+ #7 0x5ffd2...c0ff4  
+ #8 0x5ffd2...c0ff4

- View Transaction Details (10 tokens)  
<https://sepolia.etherscan.io/tx/0x7b82517da5bf252954554ffd3d7396c44c21ec5318953ad2515c0556db08538f>
  - View Transaction (10 tokens)  
<https://sepolia.etherscan.io/address/0x2ca3c3f00d2e8e65fdbff1a5120ae6cca39dc593>
  - View Transaction Details (1000 tokens)  
<https://sepolia.etherscan.io/tx/0xbff927fcb15a011ef4df62450d1e98d1205ceffc964cfbb1a12d7e13609336c4e>
9. 進行空投。
- View Transaction (20 tokens)  
<https://sepolia.etherscan.io/tx/0x2523227fcf10dfc63baece5f7cdf7c236da8f81c327b5b473dad1b86f0c5bc61>
  - OpenSea URL  
<https://testnets.opensea.io/assets/sepolia/0x5ffd210ef39b610444d3b6a9c72470c79acc0ff4/1>



## Summary & Thoughts

其實我對於區塊鏈以及數位貨幣的交易就非常感興趣，因過去有相關的私鏈的開發經驗，所以更加的滿心期待，在陳博士的引導中，將許多概念重新建構，對於錢包及冷錢包等是當今滿常見的技術了，在幣圈相當盛行，對於有幸能夠學到，我感到非常的開心。對於整體過程，漸進式且化繁為簡的教學，讓在課堂中無法順利完成的部分，我們也能回家一一品嘗學習，中間也遇到不少插曲，例如在家做的過程，因 Google Chrome 會同步環境中的擴充套件，例如 MetaMask 和 IPFS，有時雖然在家裡製作，但是會因為同步的過程，導致學校電腦啟動 Google Chrome 時就被封鎖，不過這也成功讓我學習到如何在學校進行危機自救，這個過程真的值得一提，但就先不贅述，底下就附上相關的佐證，進行留念，這也是研究所修課的奇妙記憶。

編號	使用者帳號	事件名稱	被封鎖時間	事件IP	受影響IP數量
3.	113598043	違反學術網路使用規範(挖礦)	2025-03-18 16:10:10	140.124.182.4	2
6.	113598070	違反學術網路使用規範(挖礦)	2025-03-17 14:22:52	140.124.183.81	1
7.	113598045	違反學術網路使用規範(挖礦)	2025-03-17 14:22:46	140.124.182.142	1
12.	113C53004	違反學術網路使用規範(挖礦)	2025-03-12 13:46:27	140.124.131.33	1
18.	113AT8403	違反學術網路使用規範(挖礦)	2024-12-26 16:53:19	140.124.147.83	1
30.	111318184	違反學術網路使用規範(挖礦)	2024-09-11 16:50:55	140.124.44.189	1
30.	111398003	違反學術網路使用規範(挖礦)	2024-03-23 13:52:16	140.124.182.178	1
51.	111398003	違反學術網路使用規範(挖礦)	2024-03-18 10:09:38	140.124.182.178	1
53.	113598003	違反學術網路使用規範(挖礦)	2024-03-14 15:40:18	140.124.182.178	1
105.	108300215	違反學術網路使用規範(挖礦)	2023-02-23 16:12:12	140.124.32.121	1

顯示第 1 至 10 項結果，共 105 項

上一頁 1 下一頁

## HTTP ERROR 400 account is not active:t113598043@ntut.edu.tw

URI: /service/preauth  
STATUS: 400  
MESSAGE: account is not active:t113598043@ntut.edu.tw  
SERVLET: PreAuthServlet

請查詢 <https://oic.ntut.edu.tw/auth/account/>  
申請人(學號/員工編號): t113598043  
發送IP: 140.124.182.4  
處理說明: 你本校之預定辦連資安及系務指揮。  
未來回復狀態: 請即註銷狀態。

寄件者: “北科大計網中心-網管與資安安全管理系統”<[spothow\\_noreply@ntut.edu.tw](mailto:spothow_noreply@ntut.edu.tw)>

收件者: “張育丞”<[t113598043@ntut.edu.tw](mailto:t113598043@ntut.edu.tw)>

郵件標題: 2025-03-18 星期二 下午 4:32:40 <違反學術網路使用規範(挖礦)>因計網中心系統偵測，請盡速處理

請注意：此郵件是系統自動傳送。請勿直接回覆此信件！

張育丞 您好：

您的IP: 140.124.182.4 於 2025-03-18 16:10:10 因違反學術網路使用規範(挖礦)被計網中心偵測到。

違反校園網路使用規範或與連安全管理制度，該當屬即刻起無法連接校外網路，但依舊可使用校內網路。

為加強解鎖此問題避免對申請退件，請仔細詳閱以下步驟後申請解鎖

請於收到本信後7日內完成解鎖步驟，逾時您的IP所登記的網卡位址(MAC)將會被系統消除，屆時校內網路將無法正常連線。

若超過7日則需重新登記網卡位址(MAC)後，並於隔日上午六點前完成解鎖作業，以免再次遭到消除

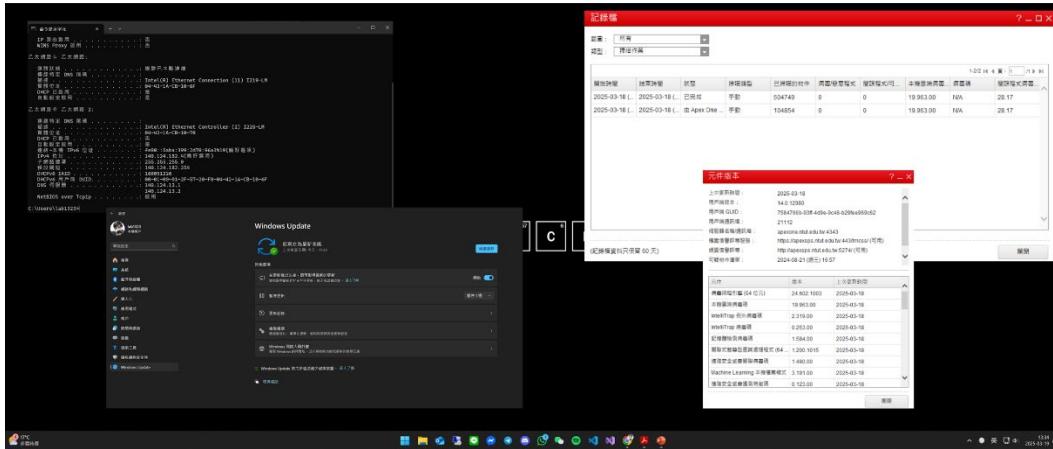
詳細解鎖方式請參照：  
本校使用者遇到資安事件封鎖檢舉時解鎖申請注意事項說明([https://oic.ntut.edu.tw/p/406\\_1004\\_00246\\_r179.php?Lang=zh\\_tw](https://oic.ntut.edu.tw/p/406_1004_00246_r179.php?Lang=zh_tw))

收到申請後將會對您的IP進行關聯，並至少審查一個工作天後才能進行解鎖。

無論是否通過或駁回均會以電子郵件形式告知，請耐心等候，謝謝。

因應資安公報資訊之資安事件處理程序，相關單位依託內容，將會作為本次資安事件單相關附件保存，以利來主管機關或檢調單位依法稽查之用。

任何有關本事件單之任何諮詢，請均以電子郵件書面往來，請勿以電話進行詢問，造成不便請見諒。



## Reference

- [1] MetaMask - Chrome Web Store, <https://chromewebstore.google.com/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn>
- [2] IPFS, <https://ipfs.tech/>
- [3] IPFS - Chrome Web Store, <https://chromewebstore.google.com/detail/ipfs-%E5%8D%9A%E5%8D%9A/nibojkomfdiaoajekhjakgkdhomnch>
- [4] Google Cloud Web3 - Ethereum Faucet, <https://cloud.google.com/application/web3/faucet/ethereum>
- [5] FileBase, <https://firebase.com>
- [6] Remix - Ethereum IDE, <https://remix.ethereum.org/>
- [7] OpenSea, <https://testnets.opensea.io/>