Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiement :

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

NFT Creation and Deployment

Step 1: Prepare NFT Metadata

1. Open VS Code (Visual Studio Code).
2. Create a JSON file named nft_metadata.json with your NFT metadata:

```
{
  "name": "CUTM Badge #2",
  "description": "NFT demo for Blockchain Studnets on Sepolia.",
  "image": "https://brown-important-alpaca-
785.mypinata.cloud/ipfs/bafkreih42pxk2yk6jzxi24nzhn34ycfxislehun6mgkws4ulsn3vg
h5mb4",
  "attributes": [
    {
      "trait_type": "Department",
      "value": "CSE"
    },
    {
      "trait_type": "Campus",
      "value": "BBSR"
    }
  ]
}
```

Step 2: Upload NFT Image to IPFS via Pinata

1. Go to Pinata and log in.
2. Navigate to the API Keys section and generate a new key with appropriate access.
3. Upload your NFT image (e.g., ganesha.png) to IPFS using the Pinata upload tool.
4. Copy the CID or the IPFS URL of the uploaded image.
5. Paste the IPFS image link (e.g., ipfs://<CID>/ganesha.png) into the image field of your nft_metadata.json.

Step 3: Upload Metadata JSON to IPFS

1. After updating your metadata file with the correct image IPFS link, upload nft_metadata.json to Pinata.
2. Copy the CID or IPFS URL of this metadata file.
   Example: ipfs://<CID>/nft_metadata.json

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

Step 4: Write the Smart Contract in Solidity

1. Open Remix IDE.
2. Create a new file named NFT.sol.
3. Paste the following contract code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.24;

import "@openzeppelin/contracts/token/ERC721/extensions/ERC721URIStorage.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract Ganesha is ERC721URIStorage, Ownable {
   uint256 private _nextId;

   constructor(string memory name_, string memory symbol_, address initialOwner)
      ERC721(name_, symbol_)
      Ownable(initialOwner)
   {}

   function mintTo(address to, string memory metadataURI) external onlyOwner
returns (uint256) {
      _nextId += 1;
      uint256 tokenId = _nextId;
      _safeMint(to, tokenId);
      _setTokenURI(tokenId, metadataURI);
      return tokenId;
   }

   function totalMinted() external view returns (uint256) {
      return _nextId;
   }
}
```

Coding Phase: Pseudo Code / Flow Chart / Algorithm

Step 5: Compile the Contract

1. In Remix, navigate to the Solidity Compiler tab.
2. Select the appropriate compiler version (^0.8.24).
3. Click Compile NFT.sol.

Step 6: Deploy the Contract

1. Go to the Deploy & Run Transactions tab.
2. Select the Injected Provider - MetaMask environment to connect your wallet.
3. Provide the constructor parameters:
   - name_: e.g., Ganesha
   - symbol_: e.g., GNSH
   - initialOwner: Your MetaMask address
4. Click Deploy and approve the transaction in MetaMask
   .

Step 7: Mint the NFT

1. After deployment, use the contract's mintTo function:
   - to: Your MetaMask address
   - metadataURI: The IPFS URL of your nft_metadata.json file (e.g., ipfs://<CID>/nft_metadata.json)
2. Click Transact and approve the minting transaction.

Step 8: View Your NFT

- Once the NFT is minted, it should be visible in your MetaMask wallet under NFTs (if supported).

* Softwares used

Pinata
Remix IDE
VScode

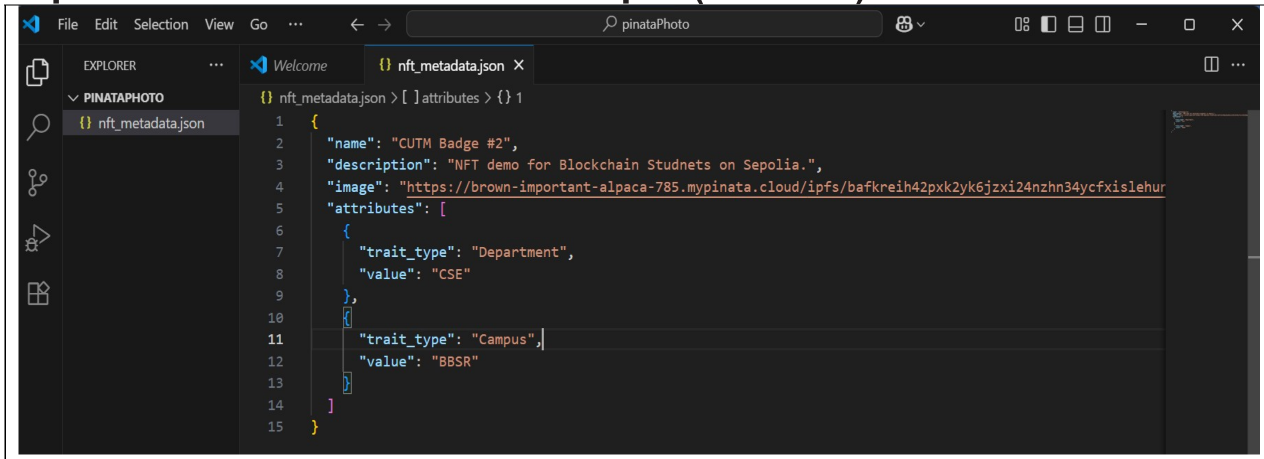
*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Testing Phase: Compilation of Code (error detection)

No Error

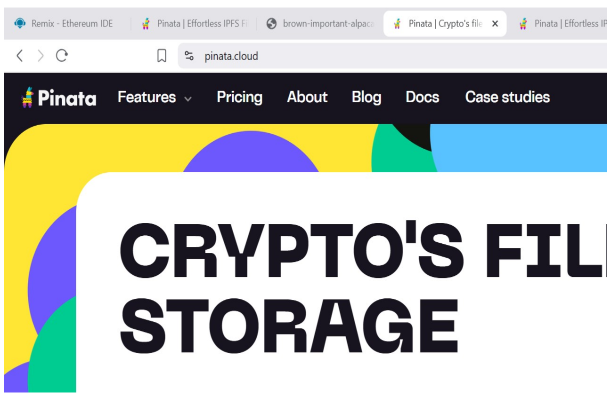
*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

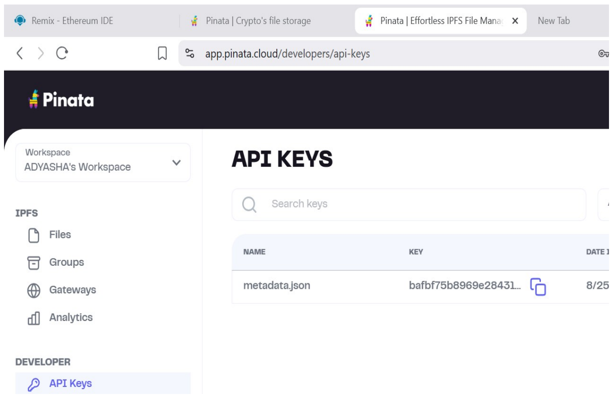
* Implementation Phase: Final Output (no error)



nft_metadata.json > [ ] attributes > { } 1

```json
{
  "name": "CUTM Badge #2",
  "description": "NFT demo for Blockchain Studnets on Sepolia.",
  "image": "https://brown-important-alpaca-785.mypinata.cloud/ipfs/bafkreih42pxk2yk6jzxi24nzhn34ycfxislehur
  "attributes": [
    {
      "trait_type": "Department",
      "value": "CSE"
    },
    {
      "trait_type": "Campus",
      "value": "BBSR"
    }
  ]
}
```



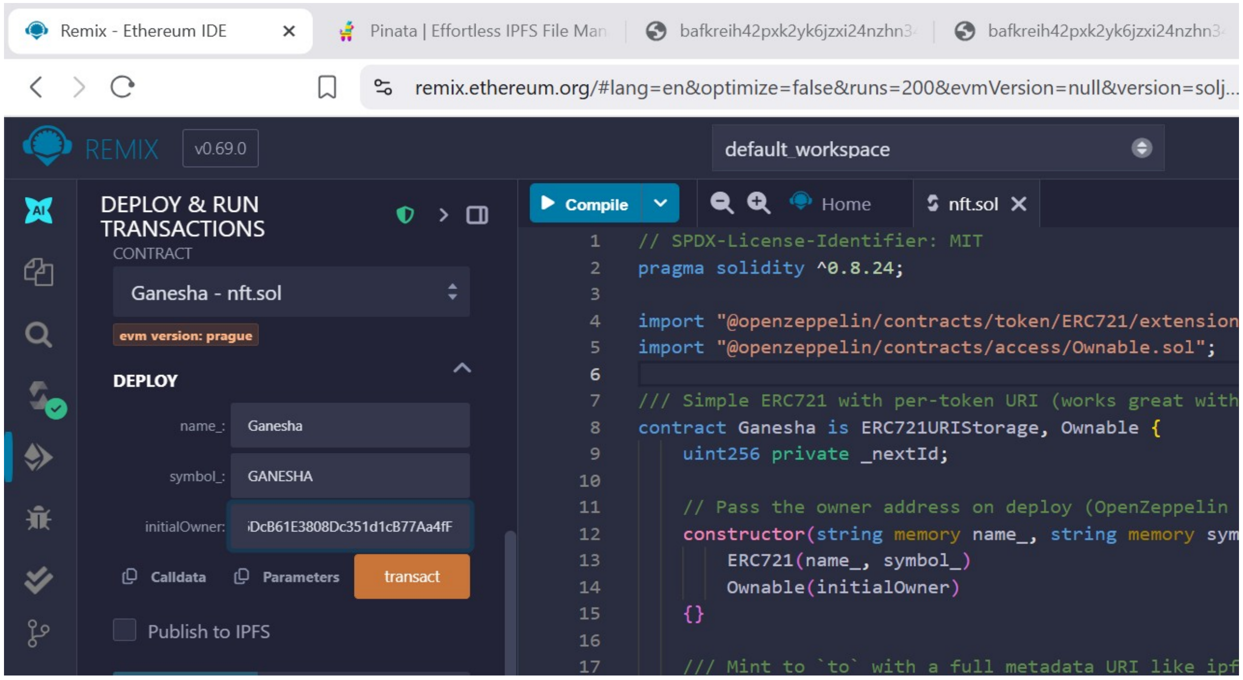


app.pinata.cloud/developers/api-keys

Workspace: ADYASHA's Workspace

API KEYS

NAME	KEY	DATE
metadata.json	bafbf75b8969e28431...	8/25



remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=solj...

DEPLOY & RUN TRANSACTIONS

CONTRACT: Ganesha - nft.sol

evm version: prague

DEPLOY

name_: Ganesha

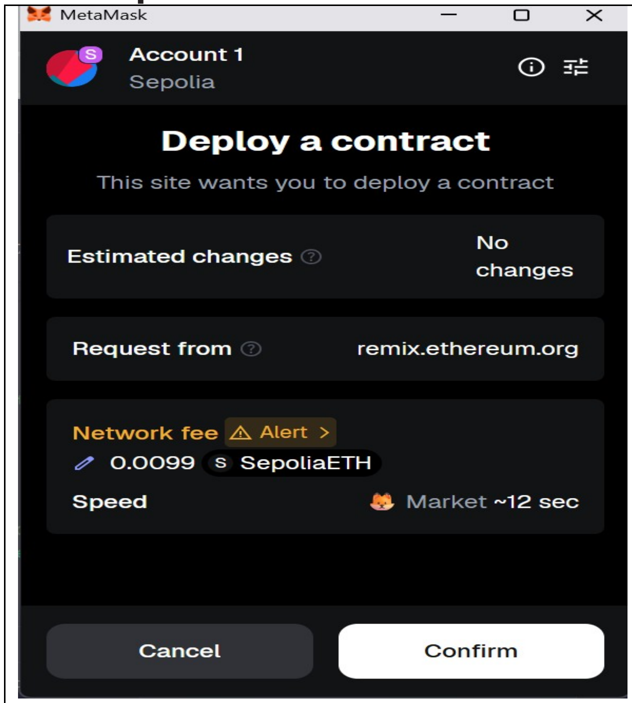
symbol_: GANESHA

initialOwner: iDcB61E3808Dc351d1cB77Aa4fF

```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.24;

import "@openzeppelin/contracts/token/ERC721/extension
import "@openzeppelin/contracts/access/Ownable.sol";

/// Simple ERC721 with per-token URI (works great with
contract Ganesha is ERC721URIStorage, Ownable {
    uint256 private _nextId;

    // Pass the owner address on deploy (OpenZeppelin
    constructor(string memory name_, string memory sym
        ERC721(name_, symbol_)
        Ownable(initialOwner)
    {}

    /// Mint to `to` with a full metadata URI like ipf
```

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

Implementation Phase: Final Output (no error)



MetaMask

Account 1
Sepolia

Deploy a contract

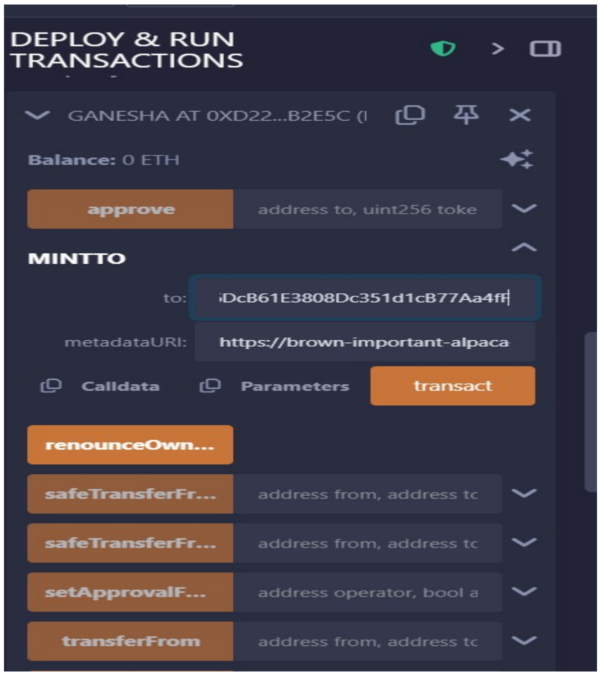
This site wants you to deploy a contract

Estimated changes: No changes

Request from: remix.ethereum.org

Network fee ⚠ Alert
0.0099 SepoliaETH

Speed: Market ~12 sec



DEPLOY & RUN TRANSACTIONS

GANESHA AT 0XD22...B2E5C (I

Balance: 0 ETH

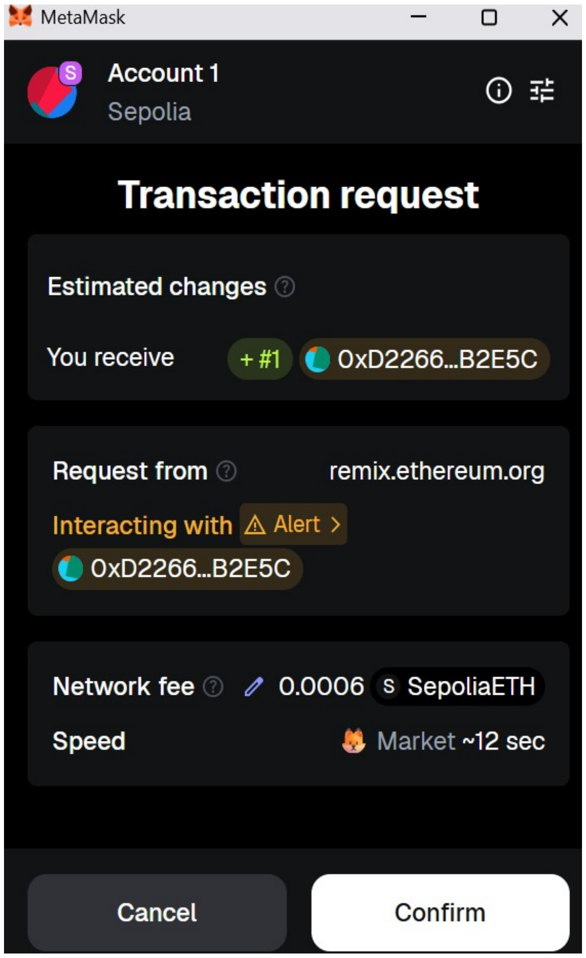
approve — address to, uint256 toke

MINTTO

to: DcB61E3808Dc351d1cB77Aa4fF

metadataURI: https://brown-important-alpaca

Calldata
Parameters
transact
renounceOwn...
safeTransferFr... address from, address tc
safeTransferFr... address from, address tc
setApprovalF... address operator, bool a
transferFrom address from, address tc



MetaMask

Account 1
Sepolia

Transaction request

Estimated changes

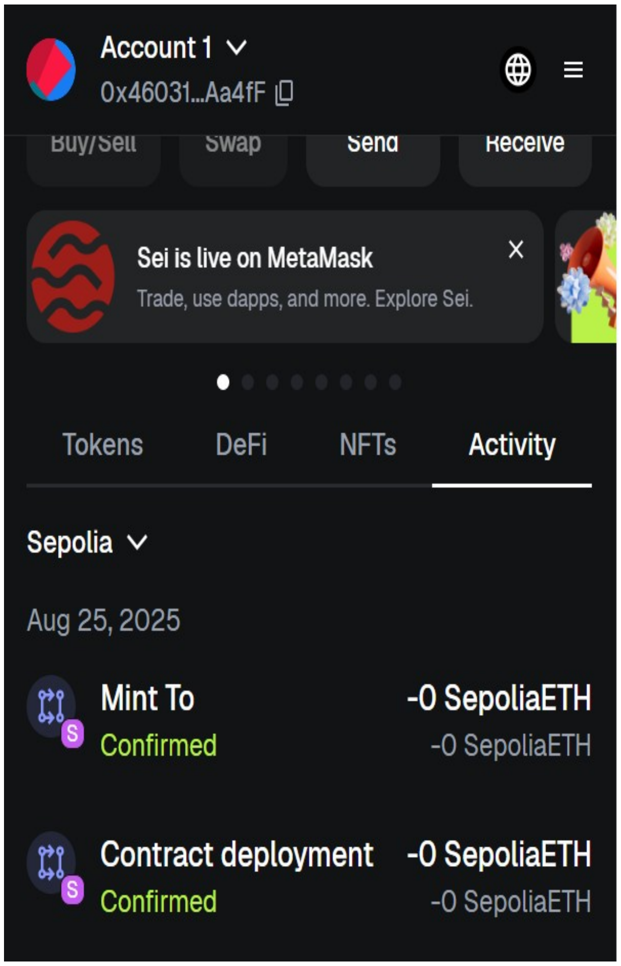
You receive: + #1 0XD2266...B2E5C

Request from: remix.ethereum.org

Interacting with ⚠ Alert
0XD2266...B2E5C

Network fee: 0.0006 SepoliaETH

Speed: Market ~12 sec



Account 1
0x46031...Aa4fF

Sei is live on MetaMask
Trade, use dapps, and more. Explore Sei.

Sepolia

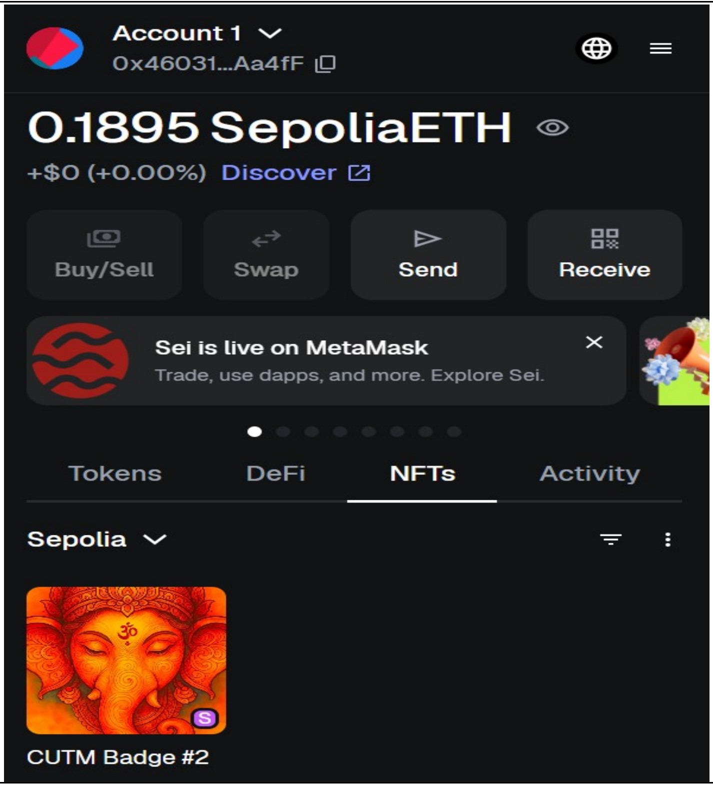
Aug 25, 2025

Mint To — Confirmed — -0 SepoliaETH / -0 SepoliaETH

Contract deployment — Confirmed — -0 SepoliaETH / -0 SepoliaETH

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.

* Implementation Phase: Final Output (no error)



Account 1

0x46031...Aa4fF

0.1895 SepoliaETH

+$0 (+0.00%) Discover

Sei is live on MetaMask

Trade, use dapps, and more. Explore Sei.

Sepolia

CUTM Badge #2

* Observations

- Metadata was properly structured and uploaded to IPFS using Pinata.
- Smart contract used OpenZeppelin's ERC721URIStorage for NFT functionality.
- Image and metadata were stored on IPFS, ensuring decentralization.
- Contract deployed successfully using Remix and MetaMask.
- NFT was minted correctly and appeared in the MetaMask wallet.
- The process followed best practices for secure and efficient NFT deployment.

ASSESMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	**50**		

Signature of student:

Name :

Signature of the Faculty:

Regn. No. :