National College of Ireland

Configuration Manual

MSc Research Project
MSc Cloud Computing

KrishnaMurthy Kowsik Gelli
Student ID: X23242817

School of Computing
National College of Ireland

Supervisor: Sai Emani

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	.....KrishnaMurthy Kowsik Gelli.....
Student ID:	...X23242817..... **Year:** ...2024.....
Programme:	...MSc Cloud Computing...
Module:	.....Research Project.....
Lecturer:	Sai Emani.....
Submission Due Date:	12-12-2024.....
Project Title:	Improving Security and Transparency in Data Sharing with Web3 Integration and Blockchain Smart Contracts for Amazon S3 Access.
Word Count:	..... **Page Count:** .....8.....

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ...Krishnamurthy Kowsik Gelli...

Date: .....12-12-2024.....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

KrishnaMurthy Kowsik Gelli
X23242817

1 Requirements

1.1 Tools and Software

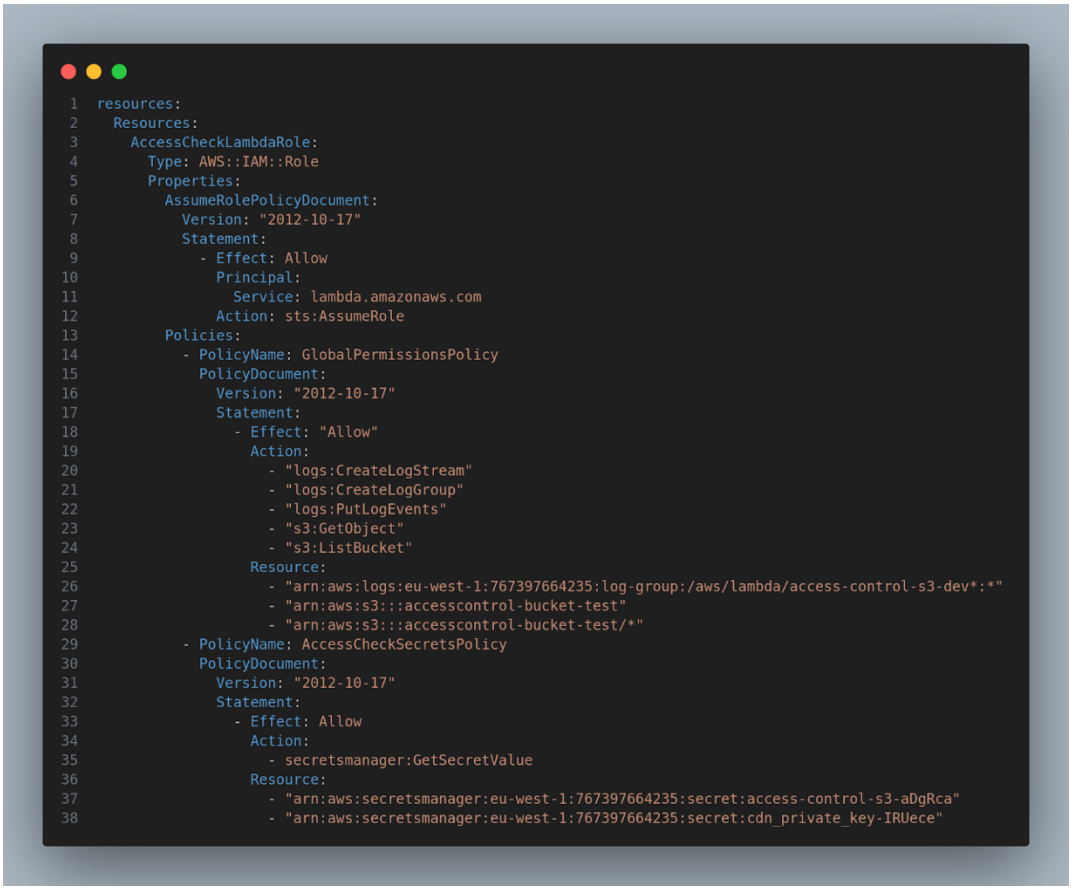
- Node.js (v18.19.1)
- npm (v9.2.0)
- Python
- AWS CLI and Management Console access
- Web3.js
- MetaMask wallet
- Remix IDE
- Serverless Framework

1.2 Installation Steps

1. Install Node.js and npm:
   - Visit https://nodejs.org/en/download/package-manager
   - Select your operating system and follow installation instructions
2. Install MetaMask:
   - Visit https://metamask.io/
   - Install the browser extension
   - Create a new wallet and securely store your 12-word seed phrase
3. Install Serverless Framework:

   ```
   npm i serverless -g
   ```

   - Clone the repository and switch to feature/V3 branch
   - git clone https://github.com/kowsikgelliie/access-control-serverless.git (Private Repo)
   - git checkout feature/V3
   - Open the code and check serverles.yml file



```yaml
resources:
  Resources:
    AccessCheckLambdaRole:
      Type: AWS::IAM::Role
      Properties:
        AssumeRolePolicyDocument:
          Version: "2012-10-17"
          Statement:
            - Effect: Allow
              Principal:
                Service: lambda.amazonaws.com
              Action: sts:AssumeRole
        Policies:
          - PolicyName: GlobalPermissionsPolicy
            PolicyDocument:
              Version: "2012-10-17"
              Statement:
                - Effect: "Allow"
                  Action:
                    - "logs:CreateLogStream"
                    - "logs:CreateLogGroup"
                    - "logs:PutLogEvents"
                    - "s3:GetObject"
                    - "s3:ListBucket"
                  Resource:
                    - "arn:aws:logs:eu-west-1:767397664235:log-group:/aws/lambda/access-control-s3-dev*:*"
                    - "arn:aws:s3:::accesscontrol-bucket-test"
                    - "arn:aws:s3:::accesscontrol-bucket-test/*"
          - PolicyName: AccessCheckSecretsPolicy
            PolicyDocument:
              Version: "2012-10-17"
              Statement:
                - Effect: Allow
                  Action:
                    - secretsmanager:GetSecretValue
                  Resource:
                    - "arn:aws:secretsmanager:eu-west-1:767397664235:secret:access-control-s3-aDgRca"
                    - "arn:aws:secretsmanager:eu-west-1:767397664235:secret:cdn_private_key-IRUece"
```

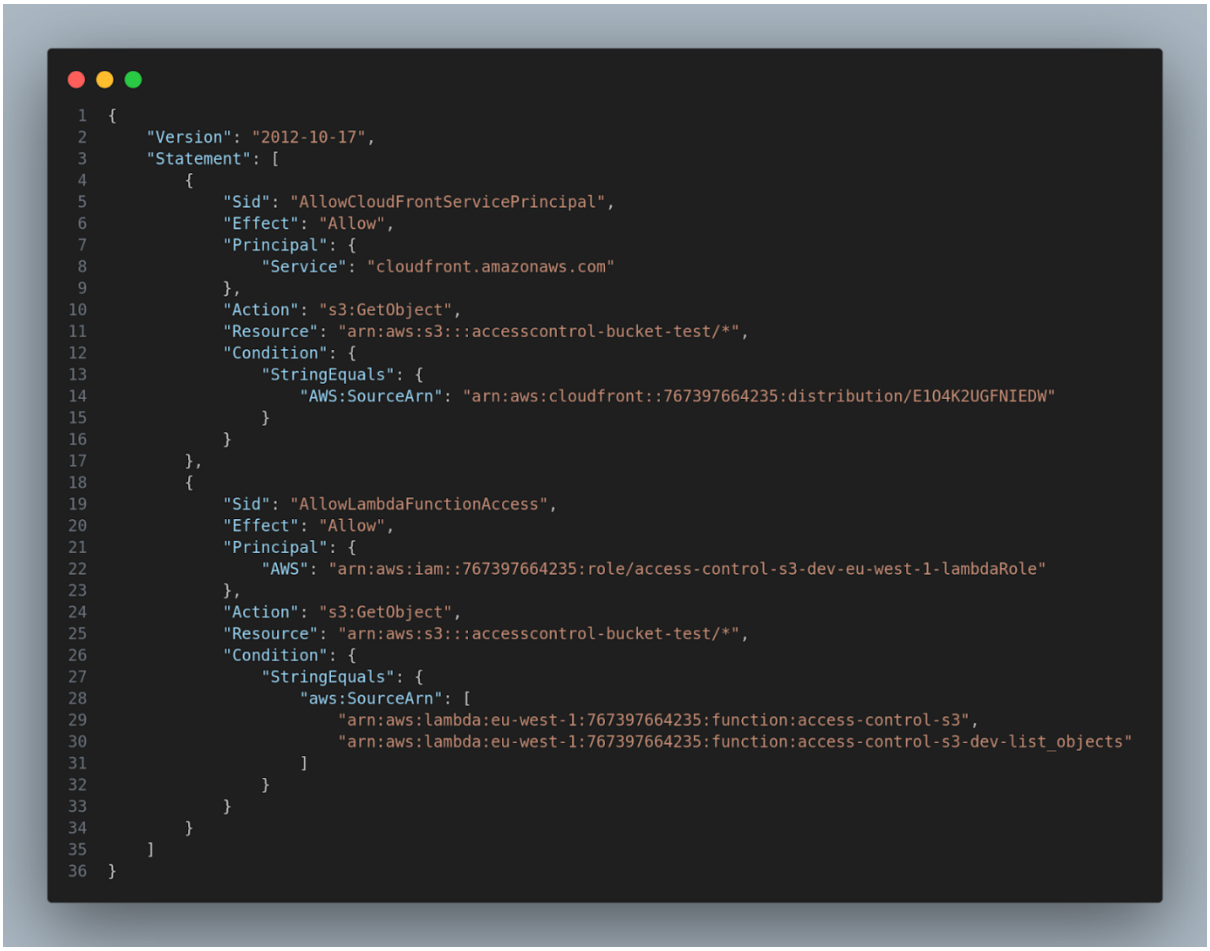
2 AWS Services Setup

2.1 S3 Bucket Setup

1. Create a new S3 bucket:

- Name: accesscontrol-bucket-test (or your preferred name)
- Settings:
    - Block all public access
    - Enable server-side encryption

2. Configure bucket policy:



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCloudFrontServicePrincipal",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudfront.amazonaws.com"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::accesscontrol-bucket-test/*",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceArn": "arn:aws:cloudfront::767397664235:distribution/E1O4K2UGFNIEDW"
                }
            }
        },
        {
            "Sid": "AllowLambdaFunctionAccess",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::767397664235:role/access-control-s3-dev-eu-west-1-lambdaRole"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::accesscontrol-bucket-test/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": [
                        "arn:aws:lambda:eu-west-1:767397664235:function:access-control-s3",
                        "arn:aws:lambda:eu-west-1:767397664235:function:access-control-s3-dev-list_objects"
                    ]
                }
            }
        }
    ]
}
```

3. To deploy the lambdas just run
   sls deploy
4. Configure aws cli with your access and secrets key before deployment.

```
kowsikgelli@kowsikgelli:~/Desktop/thesis/thesis-serverless/access-control-s3$ sls de
ploy

Deploying "access-control-s3" to stage "dev" (eu-west-1)

✔ Service deployed to stack access-control-s3-dev (212s)

endpoints:
  GET - https://[redacted].execute-api.eu-west-1.amazonaws.com/dev/hello
  GET - https://[redacted].execute-api.eu-west-1.amazonaws.com/dev/list_objects
  POST - https://[redacted].execute-api.eu-west-1.amazonaws.com/dev/access_check
functions:
  hello: access-control-s3-dev-hello (69 MB)
  list_objects: access-control-s3-dev-list_objects (69 MB)
  access_check: access-control-s3-dev-access_check (69 MB)

kowsikgelli@kowsikgelli:~/Desktop/thesis/thesis-serverless/access-control-s3$
```

5. Lambda Functions deployed as seen in above figure

2.2 CloudFront Configuration

1. Create CloudFront Key Pair:
   - Navigate to CloudFront → Settings → Key Management
   - Generate new key pair
   - Save private key (private_key.pem)

- Note the Key Pair ID

2. Create CloudFront Distribution:
- Origin Settings:
    - Origin Domain: Select your S3 bucket
    - Restrict Bucket Access: Yes
    - Origin Access Control: Allow Access Only

3. Default Cache Behavior:
    a. Viewer Protocol Policy: Redirect HTTP to HTTPS (for secure communication).
    b. Allowed Methods: GET, HEAD (recommended for read-only access).

4. Distribution Settings:
    a. Enable logging
    b. Restrict viewer access to signed URLs

2.3 AWS Secret Manager Setup

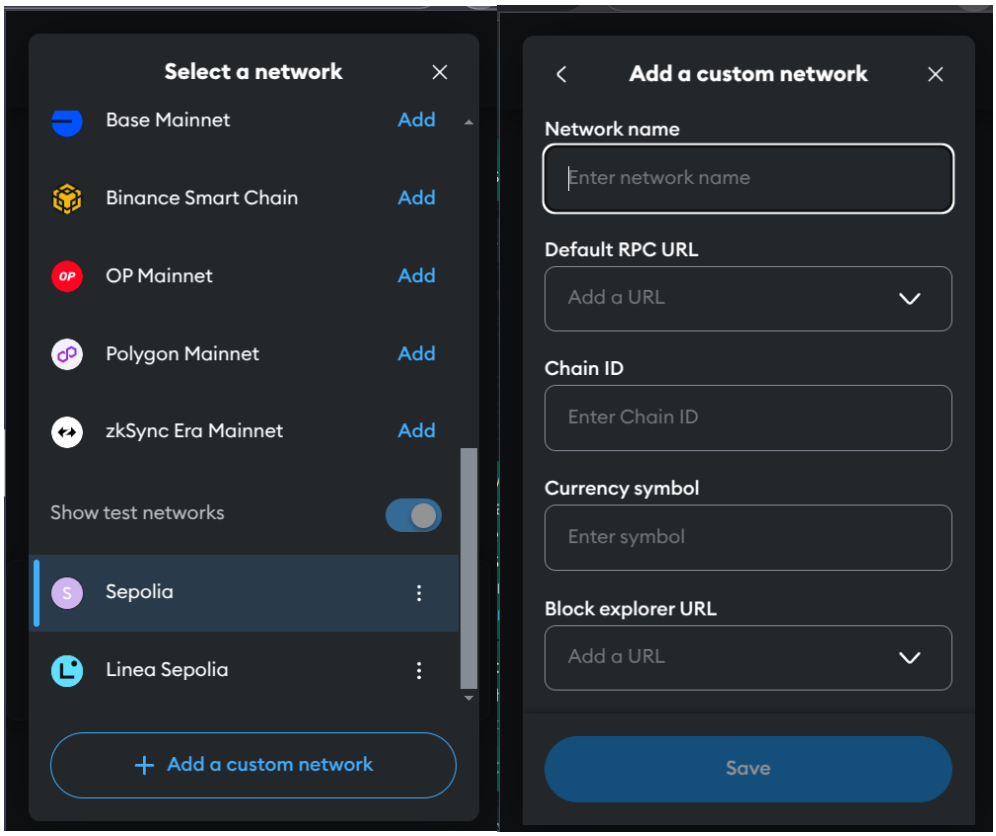
1. Create Application Secrets:
    - Type: Other type (Key/value pairs)
    - Store:
        - Contract ABI
        - Contract address
        - Other application secrets

2. Create CloudFront Private Key Secret:
    - Type: Plain text
    - Store the CloudFront private key (private_key.pem)

3 BlockChain Configuration

3.1 Polygon Amoy Testnet Setup

1. Add network to MetaMask:
    - Network Name: POLYGON AMOY TESTNET
    - RPC URL: https://rpc-amoy.polygon.technology/
    - Chain ID: 80002
    - Currency Symbol: POL
    - Block Explorer: https://www.oklink.com/amoy



2. Get testnet POL from faucets:
   - https://faucet.polygon.technology/
   - https://faucets.chain.link/polygon-amoy
   - https://learnweb3.io/faucets/polygon_amoy/

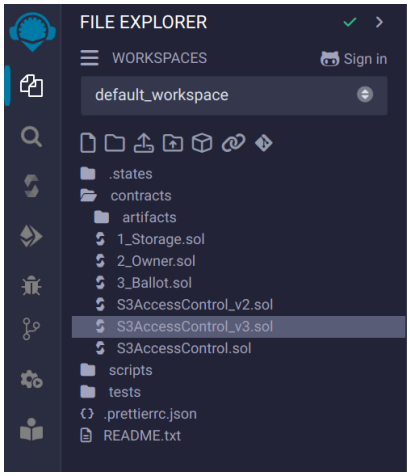
3.2 Smart Contract Deployment

1. Clone the contract repository:

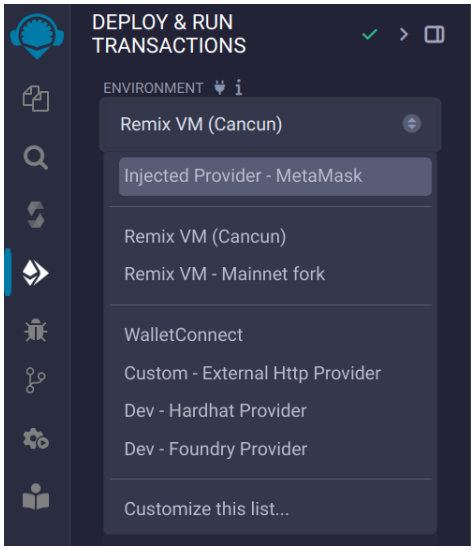
   bash

   git clone https://github.com/kowsikgelliie/access-control-contracts (private repo)

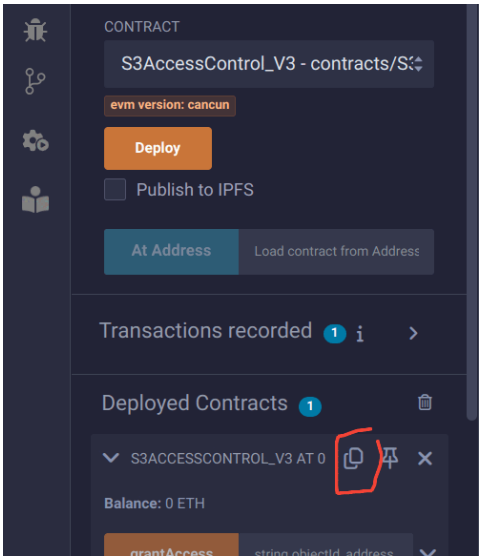
2. Deploy using Remix IDE:
   - Visit https://remix.ethereum.org/
   - Upload S3AccessControl_v3.sol



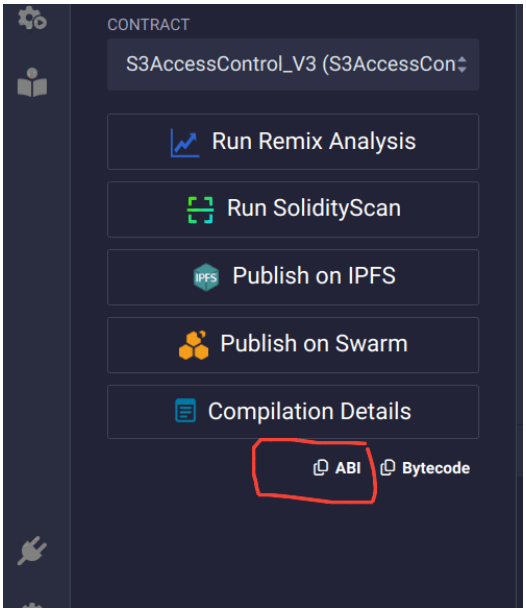
- Compile the contract
- Connect MetaMask (Injected Web3)



- Deploy and sign the transaction



- Save the contract address and ABI



- Store these in AWS Secrets Manager

4 Frontend React Setup

1. Clone the repository:

   bash
   git clone https://github.com/kowsikgelliie/thesis-dapp.git

2. Install dependencies:

   bash
   npm install

3. Configure environment: Create .env.development.local:

   REACT_APP_BACKEND_URL="your-api-gateway-url"

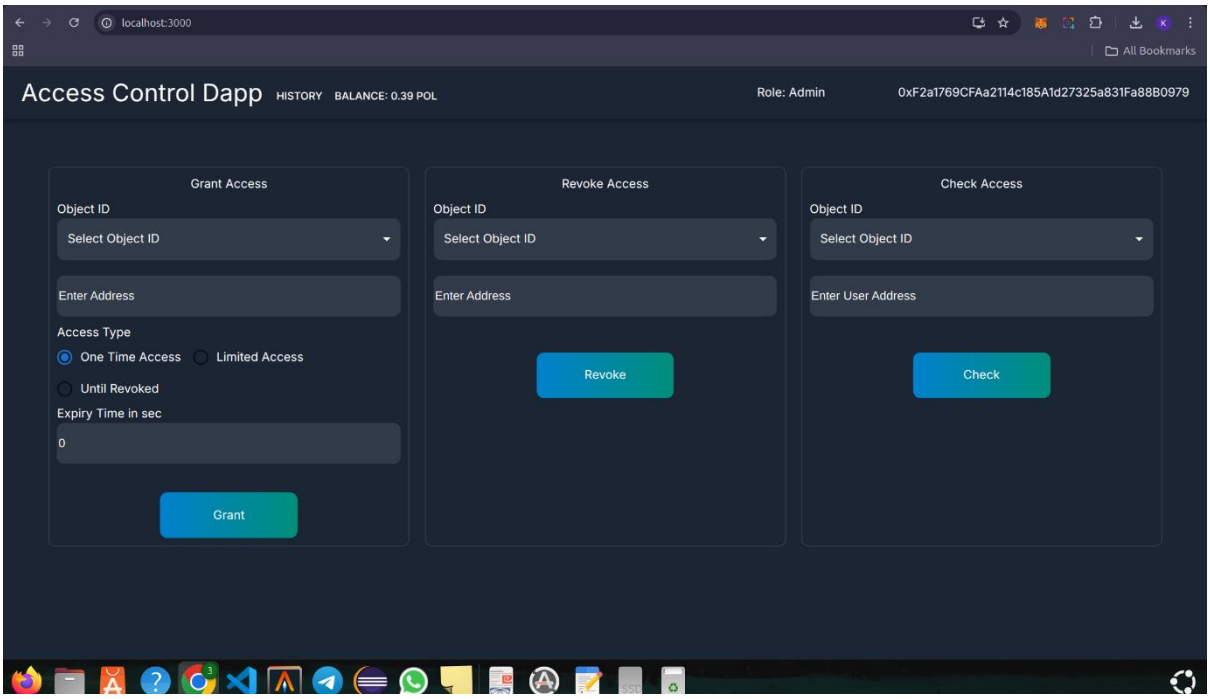
4. Start the application:

   bash
   npm run start

5. open http://localhost:3000 and connect to metamask. Make sure you are in polygon amoy testnet

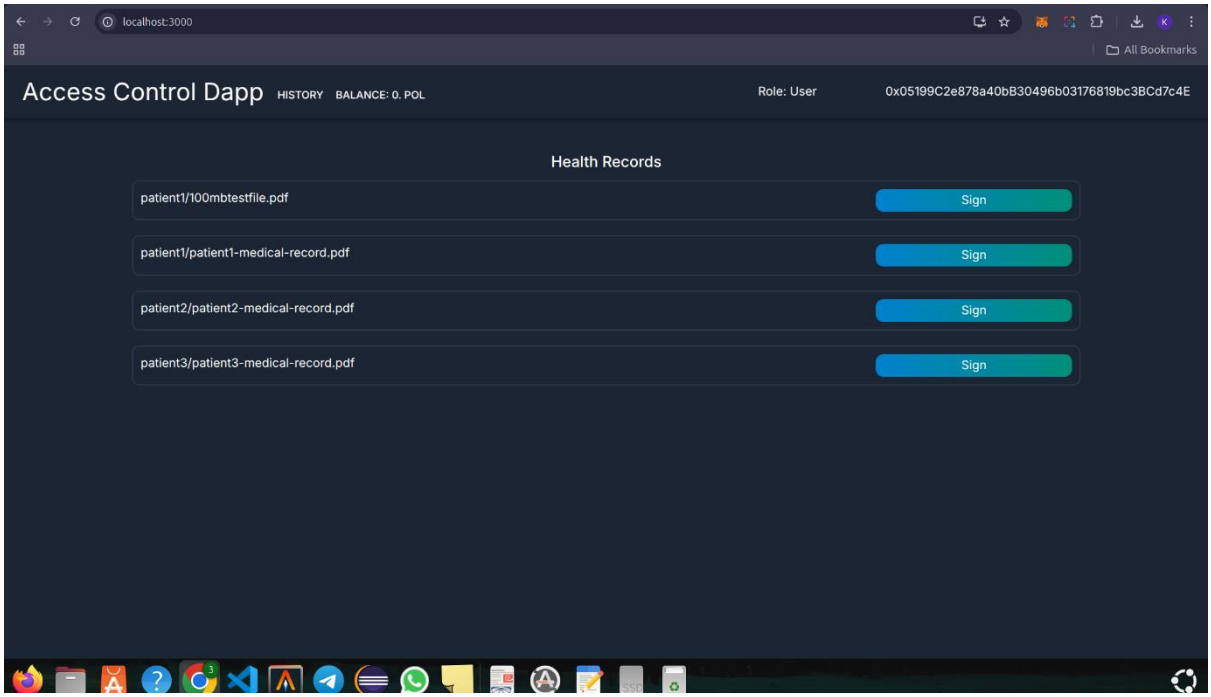
6. Select the account which deployed the contract to access the admin dashboard.

7. If everything works well, you will see admin dashboard to provide grant access and revoke access to objects.



Admin Dashboard

8. To access user dashboard, switch to different account in MetaMask which will change the role to user.



User Dashboard

9. The user can see list of objects and sign the transaction to download the object from this space.

5 Verification Steps

- Admin Dashboard:
    - Use the contract deployer account to connect MetaMask.
    - Check grant/revoke permissions access
    - Grant that a certain user address has access to test
- User Dashboard:
    - Now select different MetaMask account.
    - Verify object list display
    - Download an object with access and test
    - Check out how signed URL generation and access is verified.

6 Troubleshooting

- Contract Deployment Issues:
    - Try to keep sufficient POL in wallet.
    - Check out if your network is connected to Polygon Amoy
- Lambda Function Issues:
    - Check CloudWatch logs
    - Verify IAM permissions
    - Ensure that Secrets Manager access is valid
- Frontend Connection Issues:
    - Confirm API Gateway URL
    - Check MetaMask network
    - Check contract address and contract ABI