ProjectChain

By

MUHAMMED RAIHAN P A

(Reg No: 21352030)

Project Report

Submitted in partial fulfilment of the requirements for the award of the degree

MASTER OF COMPUTER APPLICATION

Under the Guidance of

Dr. M. Nandhini

ASSOCIATE PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF ENGINEERING AND TECHNOLOGY

PONDICHERRY UNIVERSITY

PUDUCHERRY-605014

BONAFIDE CERTIFICATE

This is to certify that the project work entitled "**ProjectChain**" is the bonafide work done by "**MUHAMMED RAIHAN P A** (Reg No: **21352030**)". In partial fulfilment for the award of degree of MASTER OF COMPUTER APPLICATION, from the Department of Computer Science, School of Engineering and Technology, of Pondicherry University. This Project work is original and not submitted earlier for the award of any degree /diploma in any University.

Guide

Dr. M. Nandhini
Associate Professor,
Department of Computer Science,
School of Engineering and Technology,
Pondicherry University,
Pondicherry - 605014

Head of the Department

Dr. S. Sivasathya
Professor,
Department of Computer Science,
School of Engineering and Technology,
Pondicherry University,
Pondicherry - 605014

Submitted for the Viva-Voce Examination held on 30/11/2022

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning those who made it possible, whose constant guidance and encouragement has crowned the efforts with success.

First and foremost, I thank the Almighty God for being with me over the years and guiding me to the heights of success. I extend sincere gratitude to my parents for their love, care and support. I express my sincere thanks to Prof. **Dr. T. Chitralekha**, Dean, School of Engineering & Technology, Pondicherry University, for giving me a chance to present my work and providing me with all the facilities for the successful completion of the same. I would like to express my deep sense of gratitude to Prof. **Dr. S. Sivasathya,** Head of the Department, Department of Computer Science, Pondicherry University, for her valuable guidance and for all the help rendered during the course of the project work and for its successful completion. Her professional advice, suggestions and timely guidance helped in refining the outcome.

I also express my deep sense of gratitude to **Dr. M. Nandhini,** Associate Professor, Department of Computer Science, Pondicherry University, Project guide for their timely check up of the progress and in turn encouraging me to complete in time with her valuable piece of advice. I am also thankful to Mr. **R.P. Seenivasan,** Assistant Professor, Department of Computer Science, Pondicherry University, for providing us with this wonderful opportunity and for their valuable guidance and moral support. I am extremely thankful for sharing expertise, and encouragement extended to me. Their regular guidance gave us the direction to work during the project.

I thank all the members of the Department of Computer Science and my friends for the timely help, advice and suggestions, which contributed either directly or indirectly to the success of the project

Muhammed Raihan P A

ABSTRACT

The need for security, integrity, transparency, and immutability is essential in the digital world. The ProjectChain focuses on establishing a decentralized distributed network that can hold up a chain of projects done by University scholars, using blockchain technology and IPFS, the decentralized file storage system.

The main objective of this project is to create a transparent record of projects done by scholars at the university so that every work would be unique and the ownership would solely belong to the actual authors. This is a react-based application and development is in Visual Studio Code with the help of the Ganache. Truffle will provide the framework for the blockchain. Ganache is an Ethereum simulator which provides some blockchain accounts which have fake ETHs in their wallets. The projects are stored in a decentralized file storage system called IPFS - InterPlanetary File System.

The web app will have user signup and login of interfaces. For the scholars who want to upload the project, there's an upload window where students can give the details of the project and upload the project. Details of three projects are mined into a single block. Students can view the projects and the chain and explore different projects through different tabs. The backend is supported by Solidity, whereas the frontend is developed using HTML, ReactJS and CSS. This blockchain focuses on increased security aspects. When a student uploads his project it won't be immediately reflected in the project chain. Only when 3 uploads are done by the scholars, the block is mined and the project is reflected in the ProjectChain. This app is a decentralized system that uses content-addressed storage using IPFS. In this protocol, each piece of data is assigned a unique content identifier (CID). All content-addressed data in IPFS can be found and retrieved based on this unique CID.

TABLE OF CONTENTS

CHAPTER 1

1. INTRODUCTION

The primary goal of this project is to establish an open database of all research projects completed by university scholars, ensuring that each one is original and that the authors alone are the rightful proprietors. In this way all the projects can be organized under a single platform. Immutability is one of the prominent features of this application, because once the project file is uploaded it can never be altered even by the owner of the application. This next level of security is achieved by blockchain technology and IPFS. IPFS is a distributed system for storing and accessing files, websites, applications, and data. It is decentralized making it possible to download a file from many locations that aren't managed by one organization. This uses content addressing instead of location addressing and provides a unique, content-derived identifier for the data, which we can use to retrieve the data from a variety of sources.

With Ganache's and Truffle's help, this react-based application was developed in Visual Studio Code. Truffle will provide the framework for the blockchain. The web application will offer interfaces for user registration and login. There is an upload window where students can input project information and upload their projects. Three projects' details were mined into a single block. Through several tabs, students can view the projects, the chain, and investigate various projects. Solidity supports the backend, while HTML, ReactJS, and CSS are used to develop the frontend. This blockchain is focused on improved security features. The project chain will not update right away when a student uploads his project. The block is mined and the project is reflected in the ProjectChain only once three uploads have been completed by the scholars.

1.1 MODULES

It contains mainly one module

1.1.1 Student

- Student signup
- Student login
- Upload
- ProjectChain
- Block Chain

1.2 OBJECTIVES

- To learn a user interface web application using ReactJS, HTML, CSS.
- To learn about blockchain development using Solidity.
- To develop a secure platform for students to upload and showcase their valuable projects.

1.3 MOTIVATION

- To make an application that can be used in the university.
- To learn development of web user interface applications in ReactJS.
- To study blockchain development using Solidity.

CHAPTER 2

2. PROBLEM DEFINITION

2.1 EXISTING SYSTEM

To store our files or Projects virtually or make available everywhere on the internet, cloud storage was the most practiced way of it. Cloud storage is a service model in which data is transmitted and stored on remote storage systems, where it is maintained, managed, backed up and made available to users over a network, typically, the internet. The current location-based HTTP system is subject to several disadvantages. All resources in HTTP are housed on a centralized single server. That server is always vulnerable to cybersecurity threats, such as DDoS attacks. It is also subject to failure or poor performance. And that server can be inactivated or censored based on what entity manages it, thereby eliminating the only source of the data housed on it.

2.2 PROPOSED SYSTEM

The proposed system aims at developing a fully functional distributed file system based on blockchain technology inside the university campus. The system enables a student to upload his research project files to the system by logging in to the app. The students who uploaded the projects will be the sole owner of the project and once it is uploaded it can never be altered, such that the project is immutable. The system requires authentication of the student. They can enter into the app by signing up with his/her credential.

In the proposed system, the blockchain is implemented by truffle framework and the smart contract is written in solidity. Once the contract is deployed in the network no one including the owner can change the nature of application and alter the application. When a student uploads the project it's stored on the IPFS and IPFS returns a hash value and when 3 projects are uploaded a block is created with a hash created from the hashes of 3 projects. And in this way projects can be stored on the blockchain more securely. In this application every user can see the transparent record of projects as the project chain.

CHAPTER 3

3. FEASIBILITY STUDY

3.1 TECHNICAL FEASIBILITY

Various available frameworks to develop web storage applications are analyzed and found blockchain is the most secure and immutable one among them. For storing data IPFS is found most efficient . For storing files IPFS provides comparatively cheaper storage space. Considering various other resources available at, the project is technically feasible for the development.

3.2. OPERATIONAL FEASIBILITY

Proposed project is found useful for students who research and easy to track for the authority of departments in the university. Hence the operational feasibility of the system is also good.

3.3. ECONOMIC FEASIBILITY

Since the system developed as part of the project work, there is no manual cost to spend for the proposed system. Also all hardware resources such as computer systems are already available, softwares such as visual studio code, react are open source and IPFS provides storage space at low cost which gives an indication that the system is economically feasible for the development.

CHAPTER 4

4. SOFTWARE REQUIREMENT SPECIFICATION

4.1. HARDWARE REQUIREMENT

Processor: intel core i3 or above

Hard Disk/SSD: 250GB

Network connection: LAN/MAN/WAN

Memory : 4GB RAM or HIGHER

4.2. SOFTWARE REQUIREMENT

Front End: ReactJS, HTML, CSS

Back End: Solidity, Truffle Framework, Ganache, IPFS

IDE : Visual studio code and Powershell/Terminal

Operating system: Windows/ Linux/Mac OS

Web Browser : A blockchain browser (Metamask)

4.2. DEPENDENCIES

- ➔ This project needs an intranet connection to access.
- ➔ The user must have an internet web browser to view the web pages.
- ➔ The web browser used by the client should have a Metamask extension to make it a blockchain browser
- ➔ It is going to be a purely client-server application.
- ➔ This system is very secure and decentralized, in nature.
- ➔ The server needs the truffle framework and Ganache in the backend.
- ➔ The client system needs web browsers.
- ➔ The smart contract is written on Solidity 0.5.0

CHAPTER 5

5. SYSTEM DESIGN

The second stage of the software life cycle is system design. The system develops in both a logical and physical state. While designing the required system, the user-oriented performance criteria is expanded into a design specification. When the requirement specification document for the software being developed is available, the design process starts. Designing is the process of bridging the gap between the final solution to satisfy the requirement and the requirement specification.

The process of design involves conceiving and planning out in mind and making a drawing, pattern or a sketch". The system design transforms a logical representation of what a given system is required to do into the physical reality during development. Important design factors such as reliability, response time, throughput of the system, maintainability, expandability etc., should be taken into account. Design constraints like cost, hardware limitations, standard compliance etc should also be dealt with. The task of' system design is to take the description and associate with it a specific set of facilities-men, machines (computing and other), accommodation, etc., to provide complete specifications of a workable system.

This new system must provide for all of the essential data processing and it may also do some of those tasks identified during the work of analysis as optional extras. It must work within the imposed constraints and show improvement over the existing system. At the outset of design a choice must be made between the main alternative approaches. Talks of "preliminary design" concerned with identification analysis and selection of the major design options are available for development and implementation of a system. These options are most readily distinguished in terms of the physical facilities to be used for the processing who or what does the work.

5.1 Input design

A user should give his/her credentials in order to register and login. These are the first inputs which are given in the application. And after that a user can upload his project. In that part the user has to give the name of the project, description of the project and has to upload the project files. These many inputs are given to the application in this system.

- ➔ Username, password - In login and Signup
- ➔ Project name and description
- ➔ Project Files

5.2 Output design

The output design should be efficient, intelligible so that the system relationship with the end user is improved and thereby enhancing the process of decision making. The output design is an ongoing activity almost from the beginning of the project, efficient and well defined output design improves the relation of the system and the user. The primary considerations in the design of the output are the requirement of the information and the objective of the end user. The system output may be of any of the following

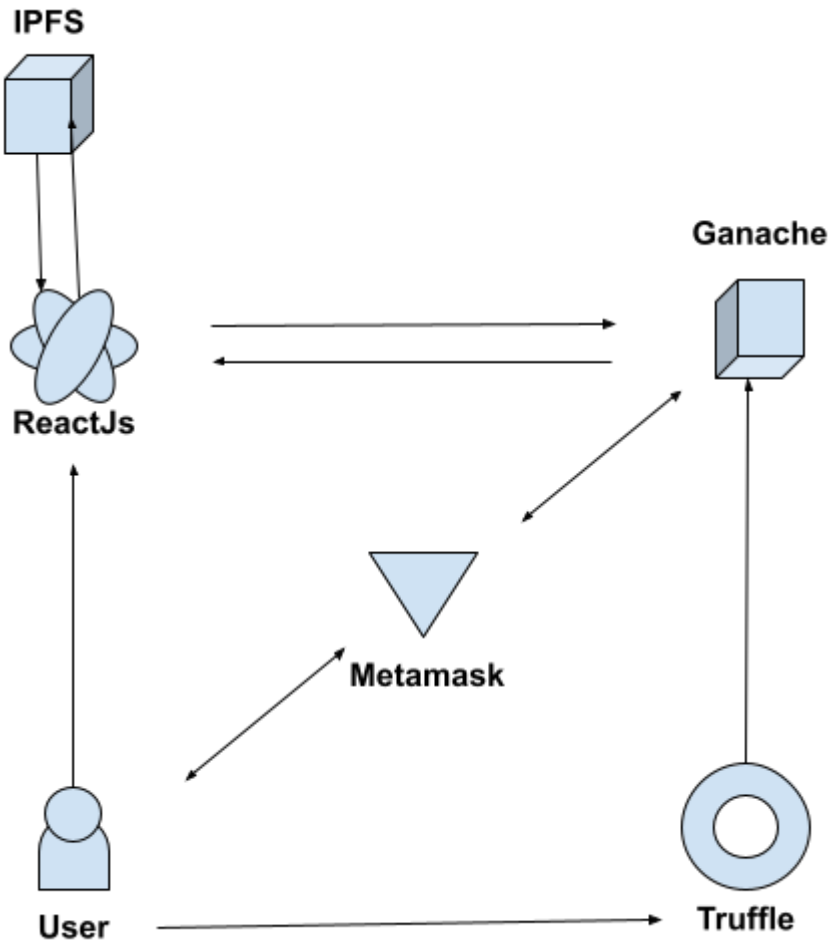
- A report
- A document
- A message

The output design specification is made in such a way that it is unambiguous and comprehensive. The approach to output design is very dependent on the type of output and nature of data. Special attention has to be made to data editing. The choice of appropriate output medium is also an important task. The output designed must be specified and documented, data items have to be accurately defined and arranged for clarity. The layout of the output will be normally specified on a layout chart.

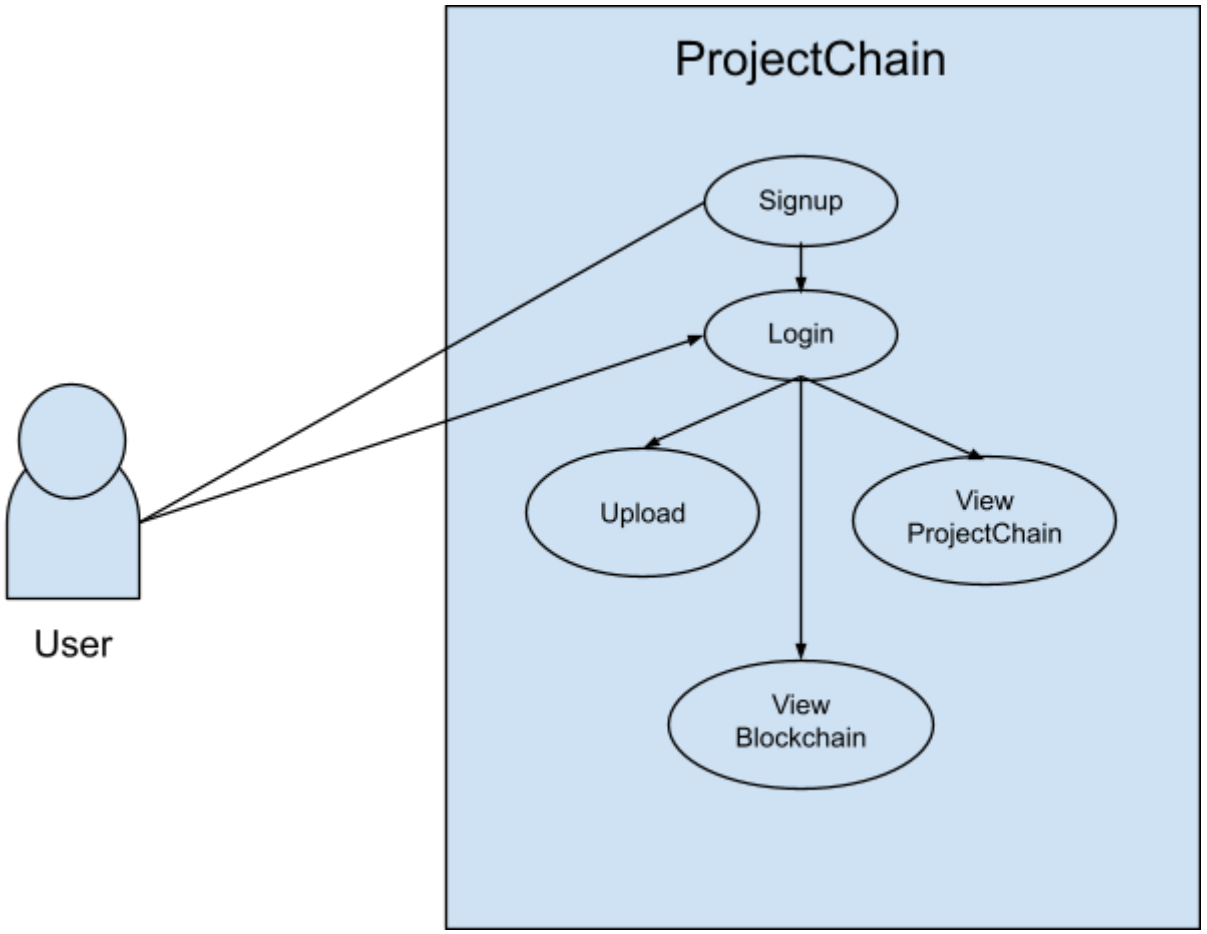
5.3 Architectural design

Describing the overall features of the software is concerned with defining the requirements and establishing the high level of the system. During architectural design, the various web pages and their interconnections are identified and designed. The major software components are identified and decomposed into processing modules and conceptual data structures and the interconnections among the modules are identified. The following modules :lfe identified in the proposed system.

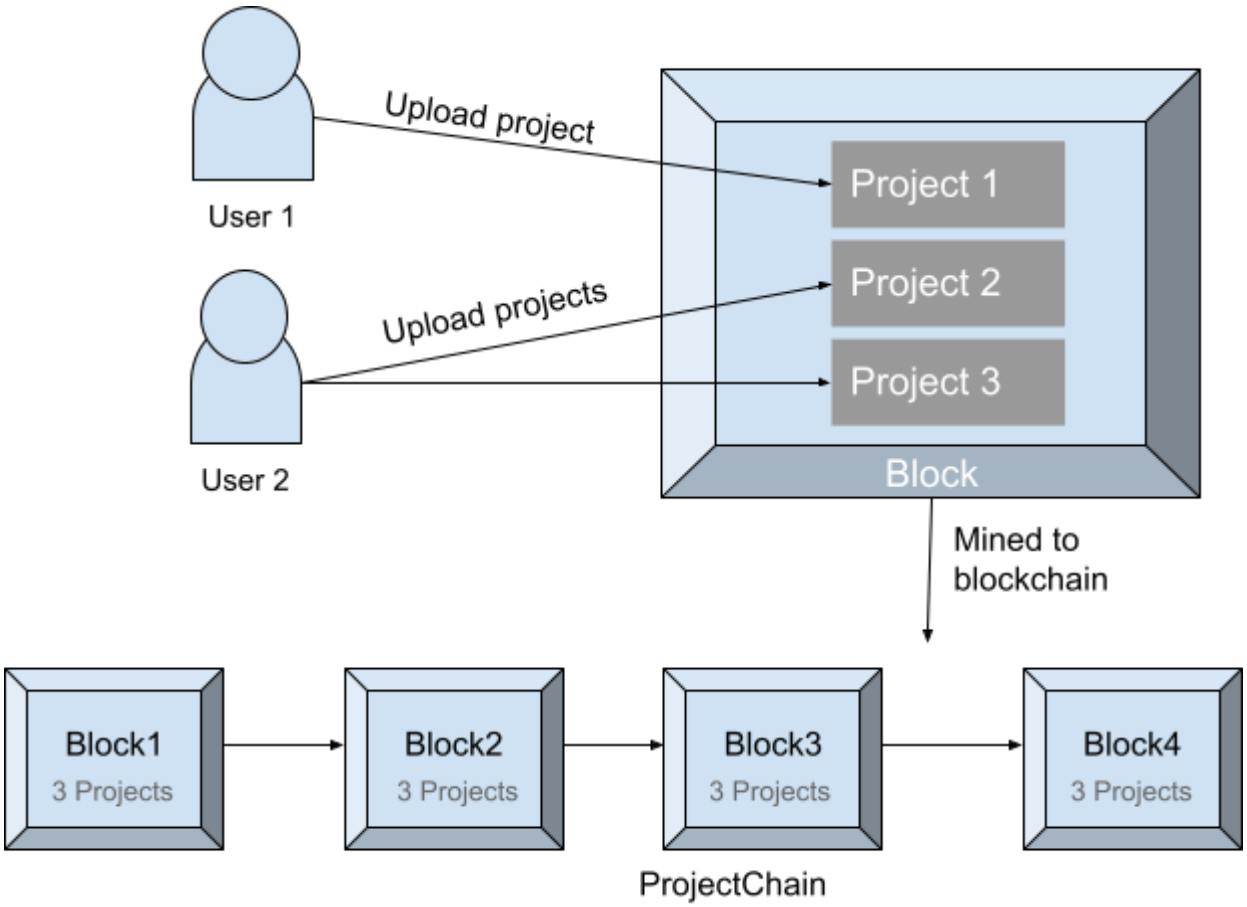
5.3.1 Block Diagram



5.3.2 USE CASE DIAGRAM



5.3.3 DETAILED DIAGRAM



CHAPTER 6

6. IMPLEMENTATION

System implementation, which entails developing compatible files, staff training, and hardware installation, is the process of turning a new system into an operational one. Avoiding organizational disruption is a crucial component of conversion. User education is essential for reducing resistance to change and giving it a chance to succeed. Conversion is followed by software maintenance to the extent that adjustments are required to maintain reliable operations in light of changes to the user's environment. Minor improvements or fixes to issues that arise late in the running of the system are frequently included in maintenance. We should either create the components from scratch or through composition during the implementation process. This advice can occasionally be found in the required document. Quality, performance, baselines, libraries, and debugging issues are addressed during the implementation phase. The product itself is the final output. The system is constructed in the implementation phase in accordance with the specifications from the earlier phases. Code writing, code reviews, tests, component selection for integration, configuration, and integration are all included in this.

This react-based application was created in Visual Studio Code with Ganache's assistance. The blockchain's framework will come from truffle. An Ethereum simulator called Ganache offers some blockchain accounts with fictitious ETHs in their wallets. The projects are kept in an IPFS (InterPlanetary File System) decentralized file storage system. The web application will offer interfaces for user registration and login. There is an upload box where students can enter project information and upload their projects for those scholars who choose to do so. Three projects' details were mined into a single block. Through several tabs, students can view the projects, the chain, and investigate various projects. Solidity supports the backend, while HTML, ReactJS, and CSS are used to create the frontend. This blockchain is concentrated on improved security features. The project chain will not update right away when a student uploads his project. The block is mined and the project is reflected in the ProjectChain only once three uploads have been completed by the scholars. This application employs content-addressed storage with IPFS and is decentralized.

Coding:-

ProjectChain.sol

```
pragma solidity ^0.5.0;
contract ProjectChain {
    string public name = "ProjectChain";
    uint256 public blockSize = 0;
    mapping(uint256 => Project) public projBlock;
    uint256 public blockCount = 0;
    mapping(uint256 => Block) public chain;
    uint256 public projCount = 0;
    mapping(uint256 => Project) public projects;
    uint256 public userCount = 0;
    mapping(uint256 => User) public users;
    struct User {
        uint256 userId;
        string userName;
        string userPwd;
        address payable walAdd;
        mapping(uint256 => Project)  myProjects;
    }
    struct Block {
        uint256 blockId;
        bytes32 blockHash;
        bytes32 preHash;
        uint256 proj1;
```

```
        uint256 proj2;
        uint256 proj3;
    }
    struct Project {
        uint256 projId;
        string prevHash;
        string projHash;
        uint256 projSize;
        string projType;
        string projName;
        string projDesc;
        uint256 uploadTime;
        address payable auther;
    }
        event UserRegistered(  uint256 userId,  string userName,  string userPwd,  address payable walAdd);
    event BlockMined( uint256 blockId,   bytes32 blockHash,  bytes32 preHash );
       event ProjUploaded( uint256 projId, string prevHash, string projHash, uint256 projSize, string projType, string projName, string projDes, uint256 uploadTime, address payable auther);
    constructor() public {}
    function regUser(string memory _userName, string memory _pwd) public {
        require(bytes(_userName).length > 0);
        require(bytes(_pwd).length > 0);
        require(msg.sender != address(0));
        userCount++;
        users[userCount] = User(userCount, _userName, _pwd, msg.sender);
        emit UserRegistered(userCount, _userName, _pwd, msg.sender);
    }
```

```
    function addToChain() public{
        blockSize=0;
        bytes32 pre;
        if(blockCount>0) pre = chain[blockCount].blockHash;
        blockCount++;
        chain[blockCount].blockId = blockCount;
        chain[blockCount].proj1 = projBlock[1].projId;
        chain[blockCount].proj2 = projBlock[2].projId;
        chain[blockCount].proj3 = projBlock[3].projId;
                                                    chain[blockCount].blockHash =
keccak256(abi.encodePacked(projBlock[1].projHash,projBlock[2].projHash,projBlock[3].proj
Hash));
        chain[blockCount].preHash = pre;
        projCount++;
        projects[projCount]=projBlock[1];
        projCount++;
        projects[projCount]=projBlock[2];
        projCount++;
        projects[projCount]=projBlock[3];
    }
    function uploadProj( string memory _projHash,   uint256 _projSize,
        string memory _projType,
        string memory _projName,
        string memory _projDesc
    ) public {
        require(bytes(_projHash).length > 0);
        require(bytes(_projType).length > 0);
        require(bytes(_projName).length > 0);
```

```
        require(bytes(_projDesc).length > 0);
        require(msg.sender != address(0));

        require(_projSize > 0);

        string memory prev;
        if (blockSize > 0) prev = projBlock[blockSize].projHash;
        else if (projCount > 0) prev = projects[projCount].projHash;
        else prev = "genesis";
        blockSize++;
        projBlock[blockSize] = Project(
            projCount+blockSize,
            prev,
            _projHash,
            _projSize,
            _projType,
            _projName,
            _projDesc,
            now,
            msg.sender
        );
        if(blockSize>=3){
        addToChain();
        }
            emit ProjUploaded( projCount, prev, _projHash, _projSize, _projType, _projName,
_projDesc, now, msg.sender );
    }
}
```

CHAPTER 7

7. CONCLUSION

The '**ProjectChain**' is very helpful as it will help a lot of scholars to store their valuable projects in a secure system. The app can be used on any desktop browser which has a metamask wallet with the help of the internet. The data and messages will be stored in IPFS and accessed by the user.

Advantages :-

- More secure than the currently existing cloud storage system.
- Blockchain technology makes the projects immutable, hence nobody even the owner of the system can alter the projects.
- Easily accessible because of the decentralized storage in IPFS.
- User-friendly interface with easily accessible upload window and viewing the chain.

Future Enhancements:-

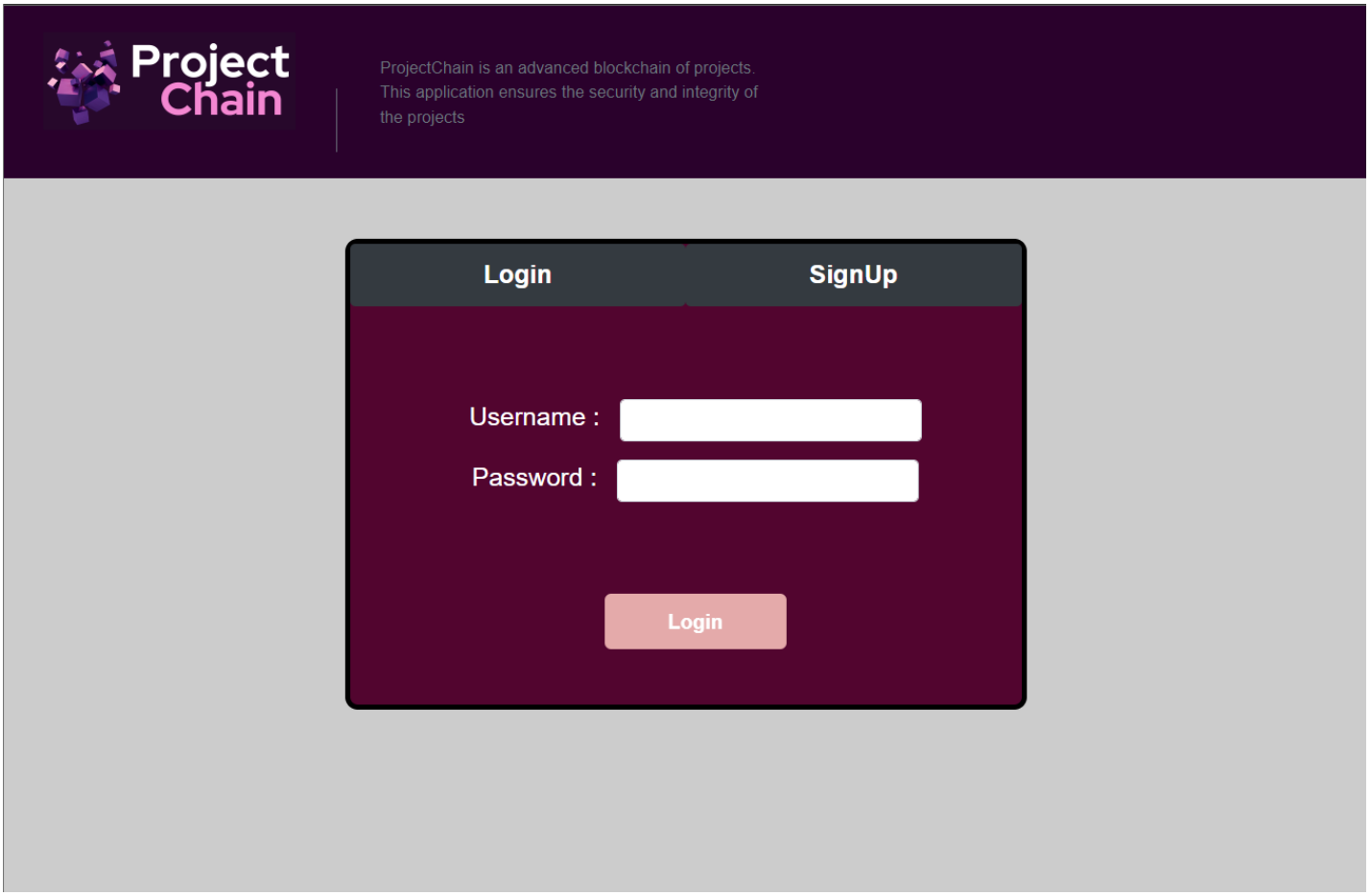
- Other users can request access to the Projects and once the owner approves they can view the project.
- Including validators from the university.
- More features including a student profile with his all attendance mined to the blockchain

REFERENCES

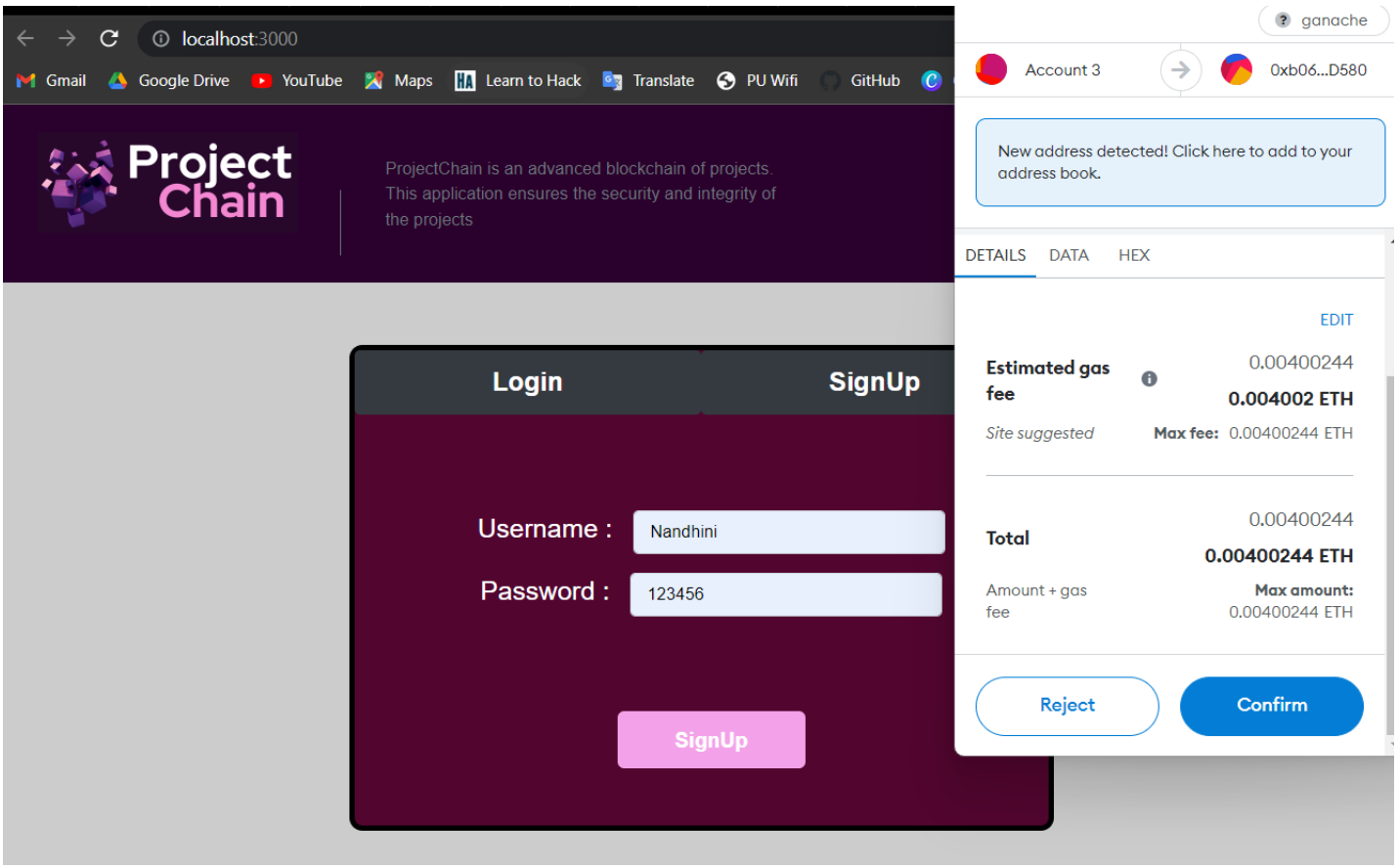
- https://app.infura.io/
- https://trufflesuite.com/
- https://trufflesuite.com/ganache/
- https://metamask.io/
- https://github.com/kvutien/Tutorial-ipfs-dapp
- https://itnext.io/storage-dapp-using-solidity-and-ipfs-62cf371b77dc
- https://reactjs.org/tutorial/tutorial.html
- https://ethereum.stackexchange.com/questions/64085/call-a-function-with-an-address-attached-solidity-react
- https://github.com/bhflm/ipfs-proxy-api
- https://www.becomebetterprogrammer.com/upload-files-using-react-ipfs-infura/

APPENDIX A

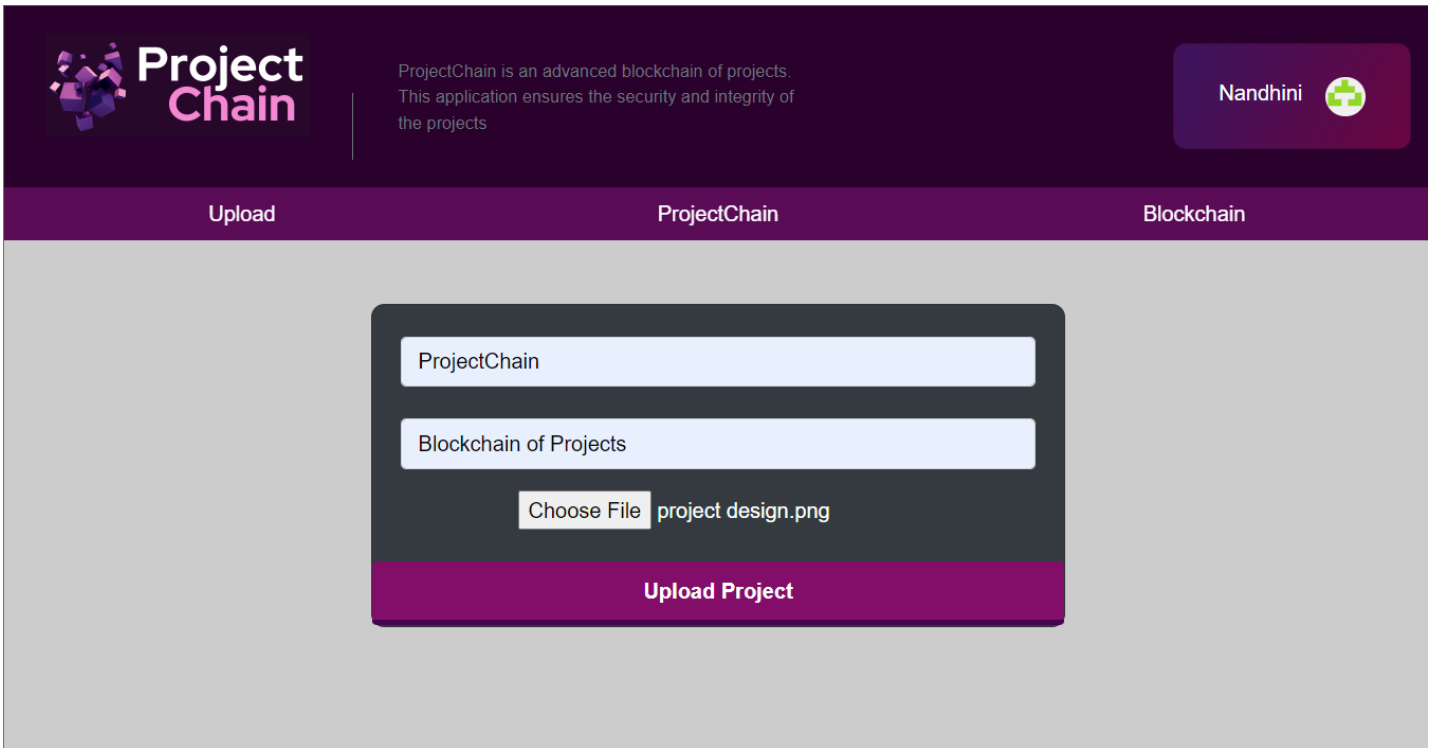
Screenshots:-



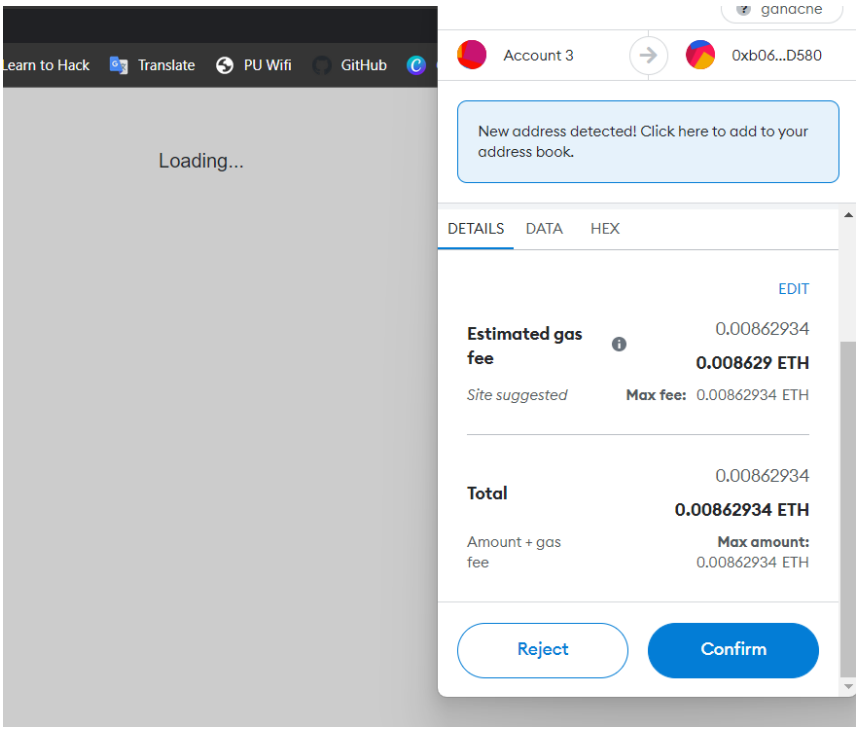
Login/Signup



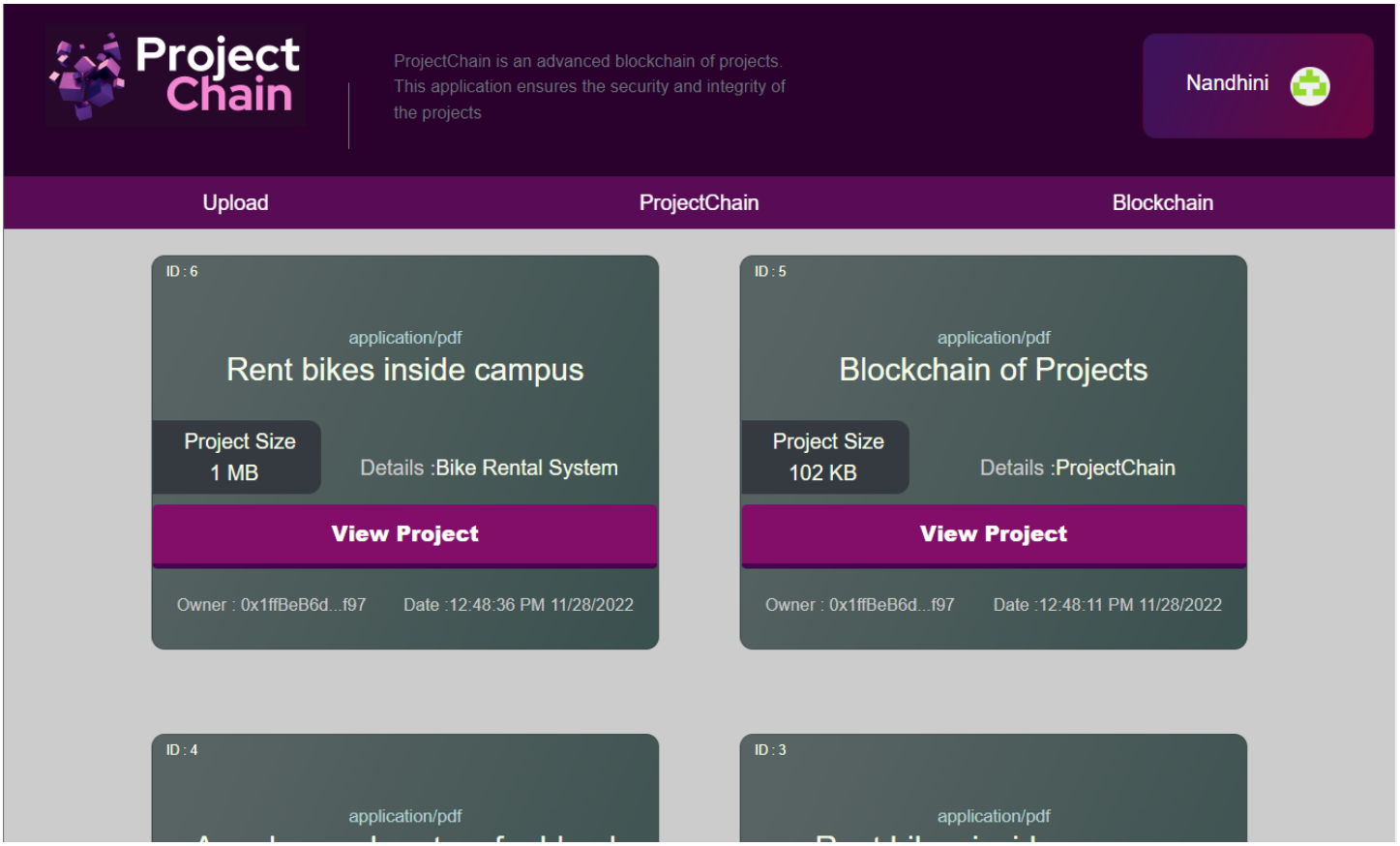
Signup



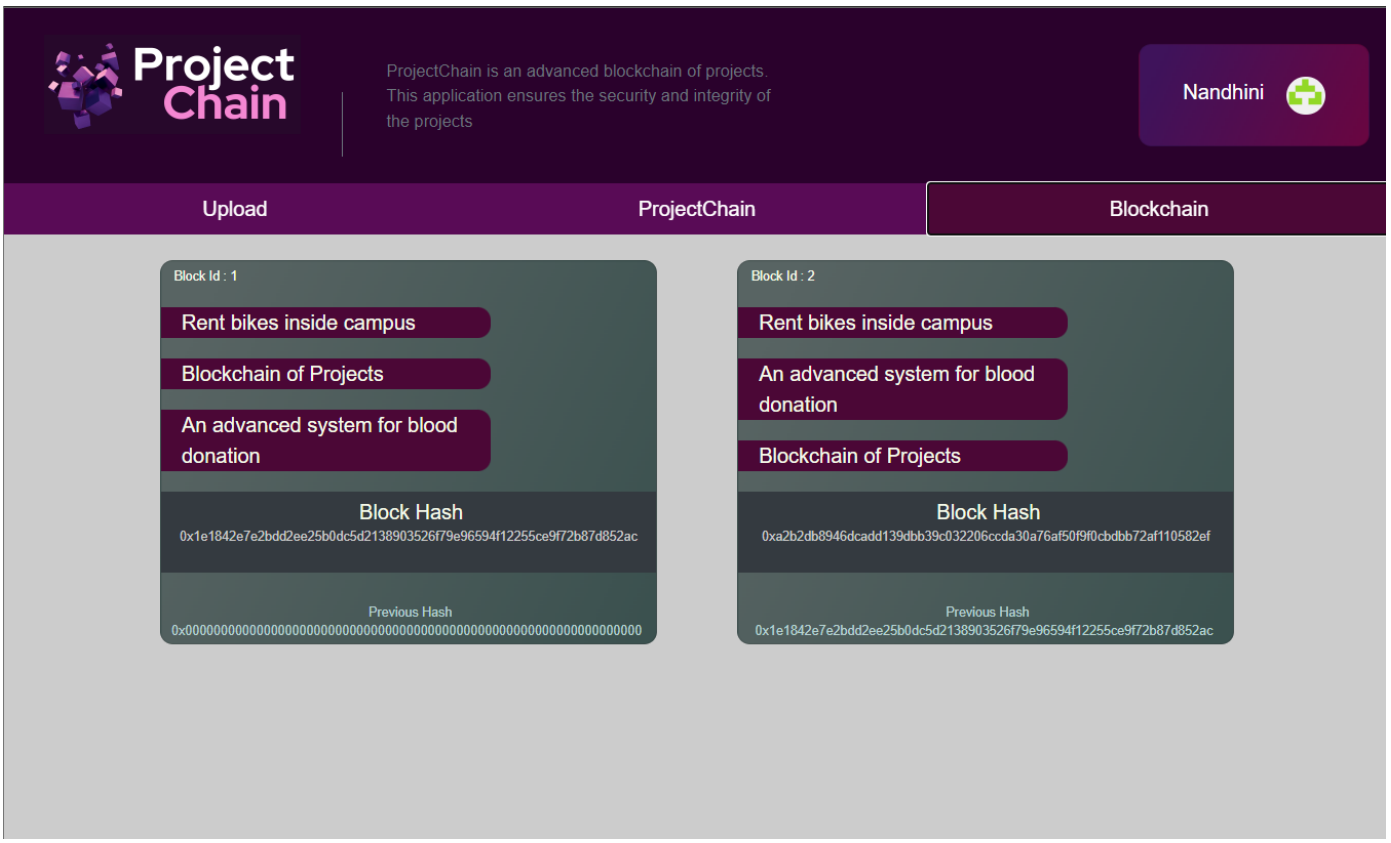
Logged in and ready to upload



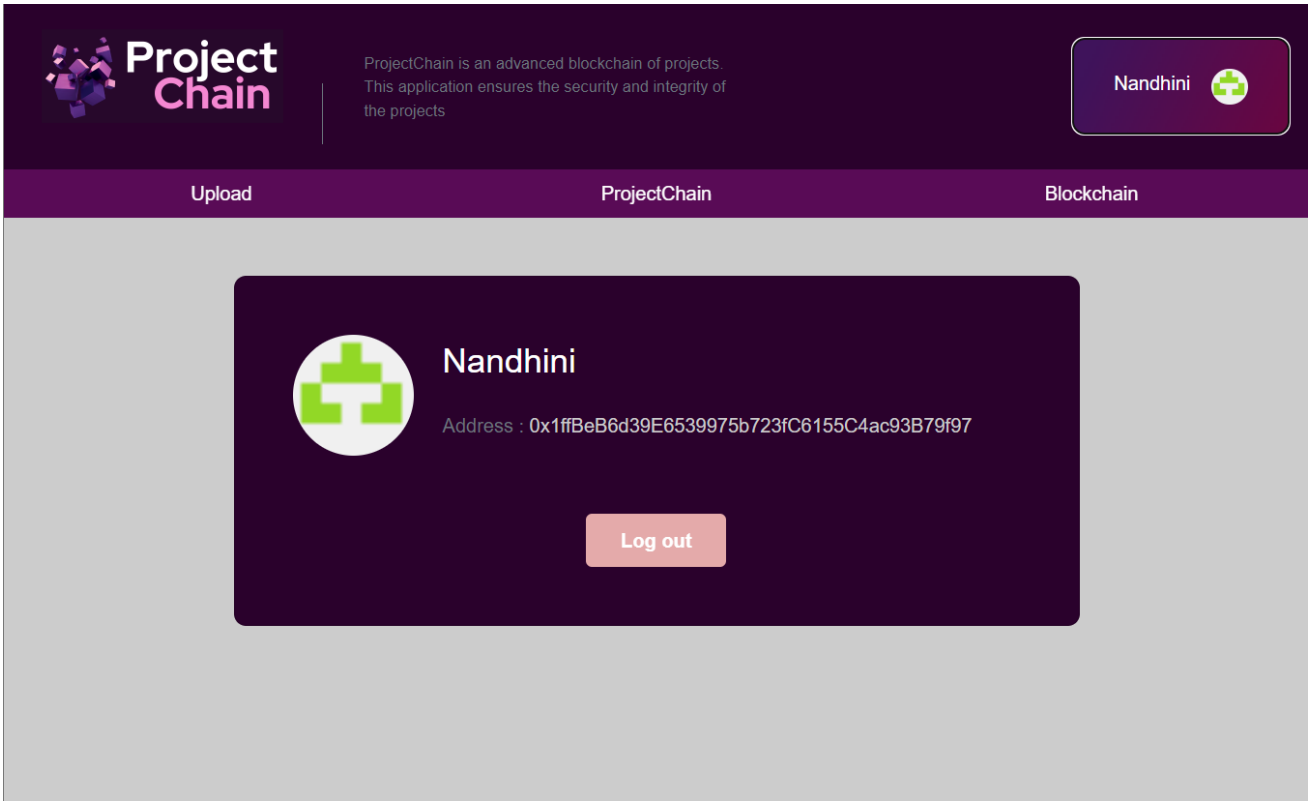
Uploading and confirming transaction



ProjectChain when 6 projects uploaded



Block Chain with 2 blocks



Profile view