

Secure Sharing of Student Credential DApp

Minor Project
SEM – VII, B. TECH
Dept. of Computer Science & Engineering

By

Rajan Gautam(19BCP101)
Kunj Kanani (19BCP167D)

Under the Supervision
of

Dr. Rutvij Jhaveri and Dr. Kaushal Shah



SCHOOL OF TECHNOLOGY
PANDIT DEENDAYAL ENERGY UNIVERSITY
GANDHINAGAR, GUJARAT, INDIA

July – December 2022

DECLARATION

We hereby declare that this written submission represents our ideas in our own words. Where others' ideas or phrases have been included, we have adequately cited and referenced the sources. We also declare that we have adhered to all academic honesty and integrity principles and have not misrepresented, fabricated, or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by Pandit Deendayal Energy University.



Kunj Kanani
19BCP167D
Computer Engineering
SoT, PDEU



Rajan Gautam
19BCP101
Computer Engineering
SoT, PDEU

APPROVAL SHEET

This report entitled Secure Sharing of Student Credential DApp by Rajan Gautam (19BCP101) and Kunj Kanani (19BCP167D) is recommended for the credits of Industrial Training.

Signature of Examiners

Signature of Supervisors

Date:

Place:

Abstract

Blockchain technology is a new booming technology in the market that is trustworthy, reliable, and distributed where the tempering of the data, once it has been added, becomes almost impossible. Advanced cryptographic algorithms ensure data integrity, and distributed systems provide data reliability. Anything in the blockchain can be traced, which also makes the system transparent, and different consensus mechanisms help to keep the data immutable. The exchange of student credentials, such as mark sheets, transcripts, degree certificates, experience certificates, LORs, etc., is an essential process in an educational environment, as multiple stakeholders, such as the student, institution, professors, and companies, are involved throughout a student's education and their placement. The credentials need to be shared with other institutions, companies, etc. As of now, the process is very tedious, complicated, lengthy, and not foolproof. This paper proposes a novel blockchain-based architecture in a decentralized application, as the information stored on the blockchain is trustworthy. The student's credentials are not directly available to any companies or anyone outside the organization without proper access granted.

Table of Contents

Introduction	- 1 -
Literature Review.....	- 4 -
Implementation.....	- 5 -
Stakeholders:	- 5 -
Admin Dashboard:	- 5 -
School Dashboard:.....	- 5 -
Professor Dashboard:	- 5 -
Company Dashboard:	- 5 -
Student Dashboard:.....	- 5 -
Contract Deployment:	- 6 -
IPFS	- 10 -
Web Application:	- 11 -
Create User	- 11 -
School Dashboard	- 12 -
Professor Dashboard	- 14 -
Company Dashboard	- 15 -
Student Dashboard.....	- 17 -
Admin Dashboard	- 19 -
Mobile Application	- 20 -
User Sign-in.....	- 20 -
Company Dashboard	- 23 -
Deployment:	- 27 -
Analysis:.....	- 28 -
Contract Deployment:	- 29 -
Create a User:	- 29 -
Add Students in School:.....	- 30 -

Add Professors in School:	- 30 -
Add Students Under Professor:	- 31 -
Upload Certificate:.....	- 31 -
Create Request:	- 32 -
Approve/Reject Request:	- 32 -
Conclusion	- 33 -
References	- 34 -

List of Figures

Figure 1: Proposed Architecture	- 2 -
Figure 2: Connect MetaMask to Remix IDE	- 6 -
Figure 3: Remix IDE with MetaMask	- 7 -
Figure 4: MetaMask Accounts.....	- 7 -
Figure 5: Deploying Contract on Mumbai Test Network.....	- 8 -
Figure 6: Transaction Details of the Contract Deployment	- 8 -
Figure 7: Available Contract Methods.....	- 9 -
Figure 8: Web3.Storage	- 10 -
Figure 9: Credi Manager Home Page	- 11 -
Figure 10: Registering User	- 12 -
Figure 11: School Dashboard.....	- 12 -
Figure 12: Student's Certificates	- 13 -
Figure 13: School Uploading Certificate for Student	- 14 -
Figure 14: Professor Dashboard.....	- 14 -
Figure 15: Professor Uploading Certificate	- 15 -
Figure 16: Company Dashboard	- 15 -
Figure 17: User Information with Request Option	- 16 -
Figure 18: Requests by Company	- 16 -
Figure 19: Student's Detail Page by Company	- 17 -
Figure 20: Student Dashboard	- 18 -
Figure 21: Student's Requests Page	- 18 -
Figure 22: Admin Dashboard.....	- 19 -
Figure 23: Students Page	- 19 -
Figure 24: Sign-in Page	- 20 -
Figure 25: Student Dashboard	- 21 -
Figure 26: Listing and Downloading Certificate	- 22 -
Figure 27: Request Page	- 23 -
Figure 28: Company Dashboard	- 24 -
Figure 29: Company's Request and Certificate Listing	- 25 -
Figure 30: Searching Student.....	- 26 -
Figure 31: Docker Image on Docker Hub.....	- 27 -

List of Tables

Table 1: Transaction Cost of Each Function.....	- 28 -
---	--------

Introduction

The fundamental credentials that remain with a person throughout their lifetime include transcripts, diploma and degree certificates, internship and training certificates, migration and transfer certificates, character certifications, letters of reference, etc. These credentials' issuance and dissemination are essential components of our educational ecosystem and are crucial to employers throughout the hiring process. Educational institutions use a variety of techniques to increase the security of the issued credentials, including assigning a unique identification number, placing a distinctive hologram, attaching a passport-sized photograph, printing the student's personal information, such as date of birth, place of birth, parents' names, registration/enrollment number, etc. Additionally, during the hiring process, businesses must confirm the credentials they get directly from the candidates. Organizations frequently get in touch with the parent university to verify the credentials applicants submitted. Such a procedure is expensive, time-consuming, and unpleasant.

Although several recent publications have highlighted the benefits and drawbacks of using blockchain technology in education, there is still a need to construct a working prototype of a student-credential sharing platform that can give services to all stakeholders in the education ecosystem.

We now know the issue, and one answer is to use blockchain. Said a blockchain is a set of data called blocks connected via encryption. Cryptography is a set of methods allowing two parties to communicate securely. I suppose everyone knows Bitcoin (BTC), a cryptocurrency (digital money) that reached an all-time high when a single BTC was worth \$60,000+. (April 2021). It may be used to purchase goods and services, but it keeps all sales records in an online ledger protected by powerful encryption.

Ethereum (ETH), a cryptocurrency similar to Bitcoin, was created in 2015 by a programmer named Vitalik Buterin. After Bitcoin, it is the second-largest cryptocurrency. The distinction between Ethereum and Bitcoin is that Ethereum is more than simply a cryptocurrency; it is ledger technology businesses utilize to create new applications. Both use blockchain technology, but Ethereum is significantly more resilient and conducive to innovation. These applications are built using "Smart Contracts," self-executing contracts written in code with the terms a buyer and seller define. That code, as well as the agreements, are disseminated over a decentralized blockchain network. This implies that blockchain does not store any information centrally. Instead, data has been replicated and distributed throughout a vast network of computers. Whenever a new block is added to the blockchain, each computer that contains information updates its blockchain to reflect the change. Solidity, Ethereum's

programming language, is used to create smart contracts. It is an object-oriented programming language used to implement smart contracts.

Using this motivation, a proposal for developing a decentralized and safe application for exchanging student credentials that are quick and validate each certificate in the form of a transaction in the Ethereum blockchain is offered. All stakeholders can securely interact with one another in a transactional way on the blockchain network. We have stakeholders in the form of schools for this DApp (Decentralized Application). Students will be registered in the school where they are enrolled; faculty members will be registered in the school where they work; recruiters who have access to the list of schools in the network; and the pupils at each school. However, we don't want any recruiter to be able to access a student's information, so we may include in the agreement an access request requirement in which a recruiter sends an email to a student requesting whether they can access their data, and once granted, that data is only available to that specific recruiter.

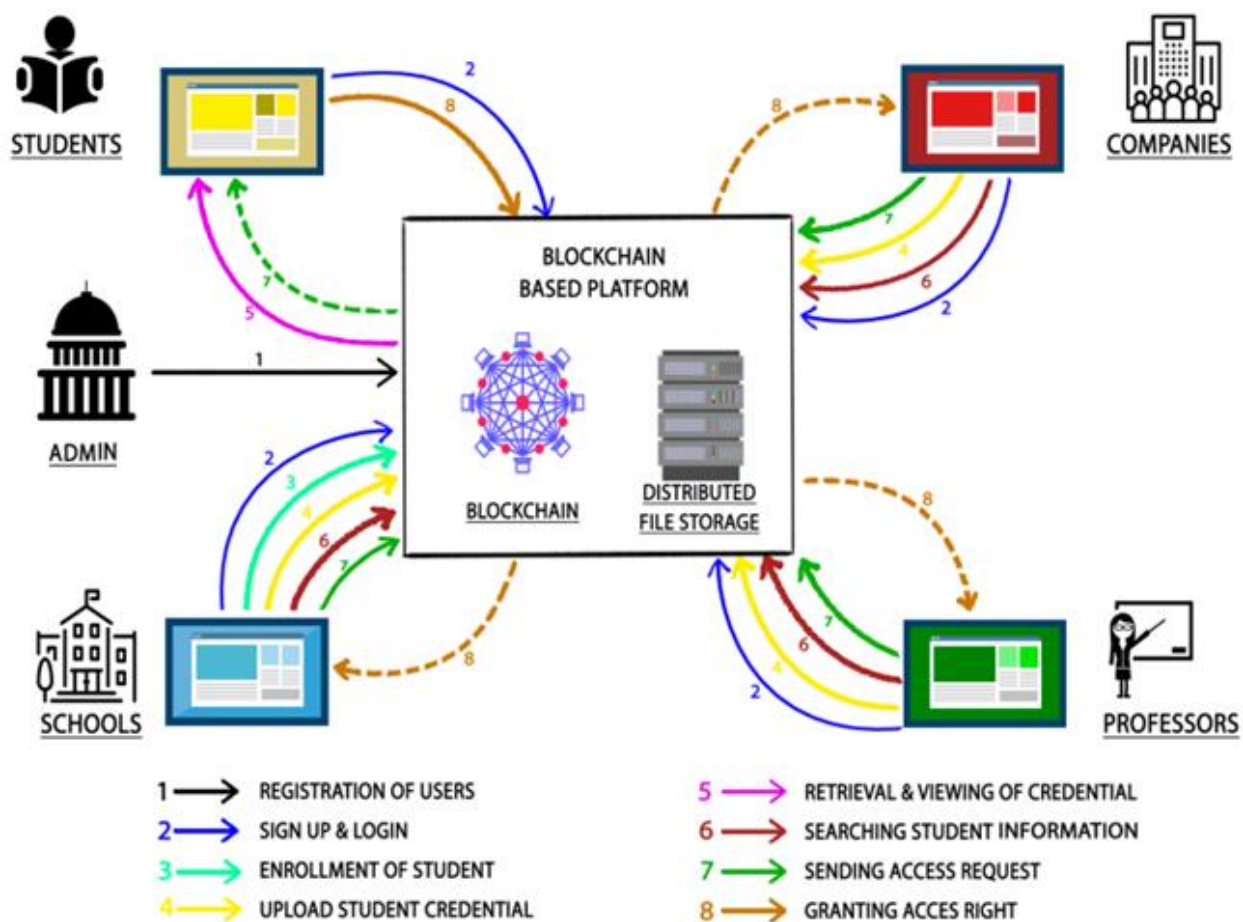


Figure 1: Proposed Architecture

According to the contract agreement, each stakeholder has a job to play. A software cryptocurrency wallet known as "Metamask" is utilized to carry out these functions. Users may access their wallets through a web browser extension or a mobile application, which can subsequently be used to engage with the decentralized service. To establish communication, various accounts will be created for each stakeholder using Metamask. Just as a school account will be created in Metamask and added to the blockchain network, a student account will be created and added to the blockchain network and then connected with the school account using its hash. The same is true for adding and linking a faculty inside a school.

On the other hand, a recruiter will be joined to the same blockchain network as everyone else and instantly linked with the school account, allowing access to the list of pupils registered in the school. On the other hand, a recruiter will not be able to see a student's credentials. They will issue an access request to the student's hash on the blockchain network to enable them access to their credentials. It secures the whole process since a student may compare a recruiter's hash, a unique identification for every material in the network, to see whether it is authentic.

We will deploy our smart contract using Remix IDE to test our application. Remix IDE is an online tool for writing, debugging, and deploying Ethereum smart contracts in a blockchain network. We will put our DApp to the test on the Mumbai test network, a blockchain network designed for developers to test their apps. It needs ethers, which are accessible at the Mumbai Test Network.

Literature Review

There have been a few studies on utilizing a blockchain-based architecture in education systems that outline the pros and drawbacks of adopting a blockchain-based credential issuing/sharing system that is readily accessible, verifiable, speedier, and less expensive. These methods, however, were recommended for particular schools. A paper developed these investigations into an integrated system, including all institutions and the certifications they provide. Students, the government, schools, professors, and businesses are all involved in its implementation. The government agency generates distinct identities for all parties. Accounts for all other stakeholders were formed based on their identities. Using a decentralized application comprised of institutions, students, professors, and so on can make things a lot more secure and more accessible for companies to have all academic details, letters of recommendation written by professors, and certificates from previous jobs/training in one place and verify at the same time as decentralized apps (DApp's) hold an immutable distributed ledger on a decentralized blockchain network.

Schools will have a list of enrolled pupils who will be given their certificates. When a new student enrolls in the school, they must provide the credentials from the preceding institution. A student may visit the system to examine their academic credentials and offer access to firms looking to recruit or a new school at the time of admission. Companies need students' permission to obtain certificates, letters of reference, and certifications from schools, instructors, or businesses.

After the training or internship, they may also award the student a certificate. On the other hand, professors are required to see a student's qualifications if they are considering applying for a Ph.D. or utilize the system to send their pupils a letter of reference. They tested the DApp on Rinkeby (Ethereum's test network) and obtained the results. Reading data from the blockchain requires very little time. Still, processing transactions and requests take some time, with a difference of 1.00337 seconds between the average turnaround time and the system's actual average time. The smart contracts on Ethereum and the communication lag are to blame for the 1.00337s delay.

The time needed to install smart contracts and process transactions on the blockchain is described in a study. However, the authors do not attempt to analyze the computational complexity of their built smart contracts or the underlying causes. Since these systems are utilized extensively, and the effectiveness of the planned smart contract has a significant impact on the efficacy of the final DApp installed on it, this is very significant in Blockchain technology research. We used their system and examined the developed smart contracts and their transactions' computational complexity.

Implementation

A prototype for this architecture is made using Ethereum, Metamask, web3.js, Next JS, and IPFS. We will start implementing each stakeholder functionality which will be set up using dashboards on the DApp.

Stakeholders:

Admin Dashboard:

- It will be able to add any type of stakeholder to the blockchain.
- It will be able to see all the users by different roles in the application.

School Dashboard:

- It will be able to add students to the list of enrolled students.
- Also, it will be able to upload credentials for the already enrolled students. When a credential is successfully uploaded to the IPFS, we get a hash value, and that is stored in the Ethereum blockchain so that only the intended stakeholders can see it.
- It can add professors or students to the school.

Professor Dashboard:

- They will be able to add students under him/her.
- They will be able to upload credentials to the enrolled students and view the credentials of the students.

Company Dashboard:

- Companies will be able to see Students' and Schools' details.
- They can request any students for certificates or credentials.
- They will be able to see the list of requests with the status.
- If granted permission by students, they will be able to access student credentials.

Student Dashboard:

- It will allow the student to view their credentials uploaded by the school and professors.
- It will show the access request to allow recruiters to access their profiles and see all their credentials.
- It will show the accept/reject response from the professor to whom they reached out to get a reference.
- It will give students an option to grant a recruiter permission to see their information or not.

All the functionalities of these dashboards are written in solidity language in Remix IDE, through which we set the environment we would like to deploy the DApp and deploy. Before deploying, we will need to have Metamask installed first in our system or as an extension in a web browser.

Contract Deployment:

Once the contract is made with all the methods and functionalities, we need to deploy it on the public network so everyone can use our application. We will use Remix and MetaMask to deploy our contract on the public Ethereum network. We are using Mumbai Test Network as our public network. We have gathered some faucets from Polygon faucet, which is required to pay a gas fee.

1. You first need to connect your MetaMask to Remix IDE for the further process. When you select Injected Provider – MetaMask from the dropdown under Environment, you will be asked to connect your account with Remix IDE. If you have multiple accounts, you must select which account you want to connect to; otherwise, it will ask for confirmation with the default account. Once you grant permission, it will be connected, and you can transact there.

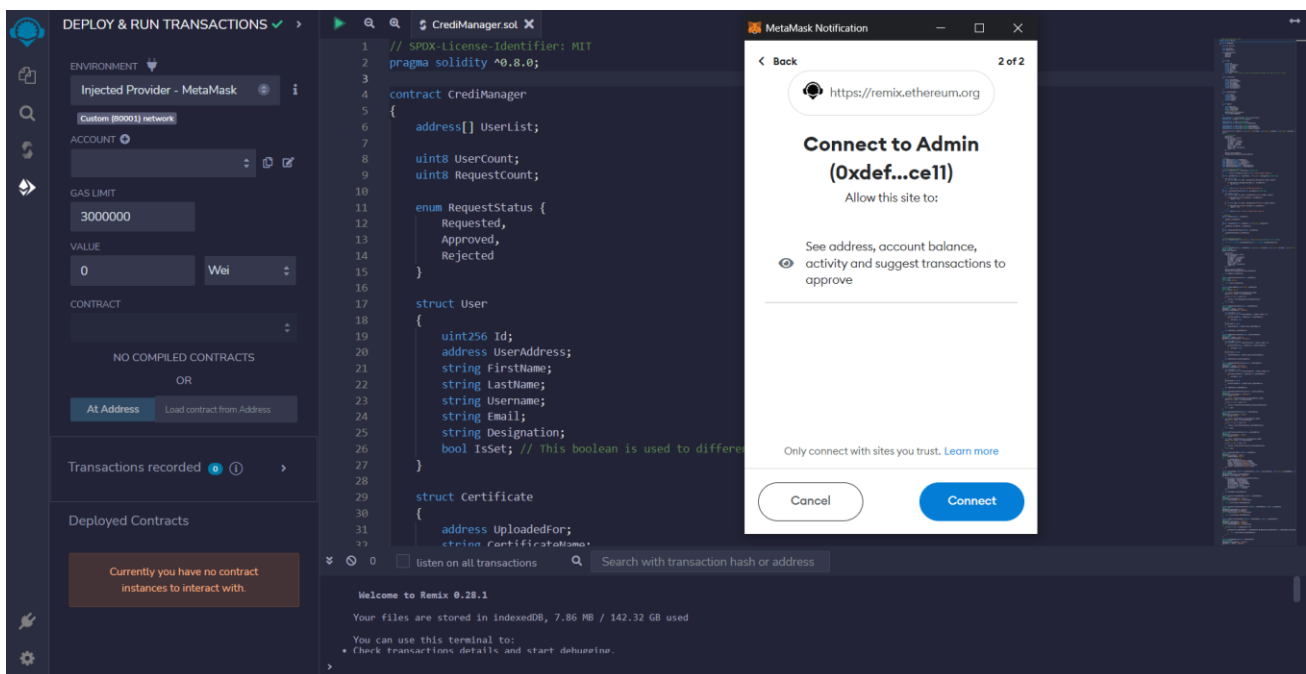


Figure 2: Connect MetaMask to Remix IDE

Once MetaMask is connected, you will see a green dot with connected written just before the account name.

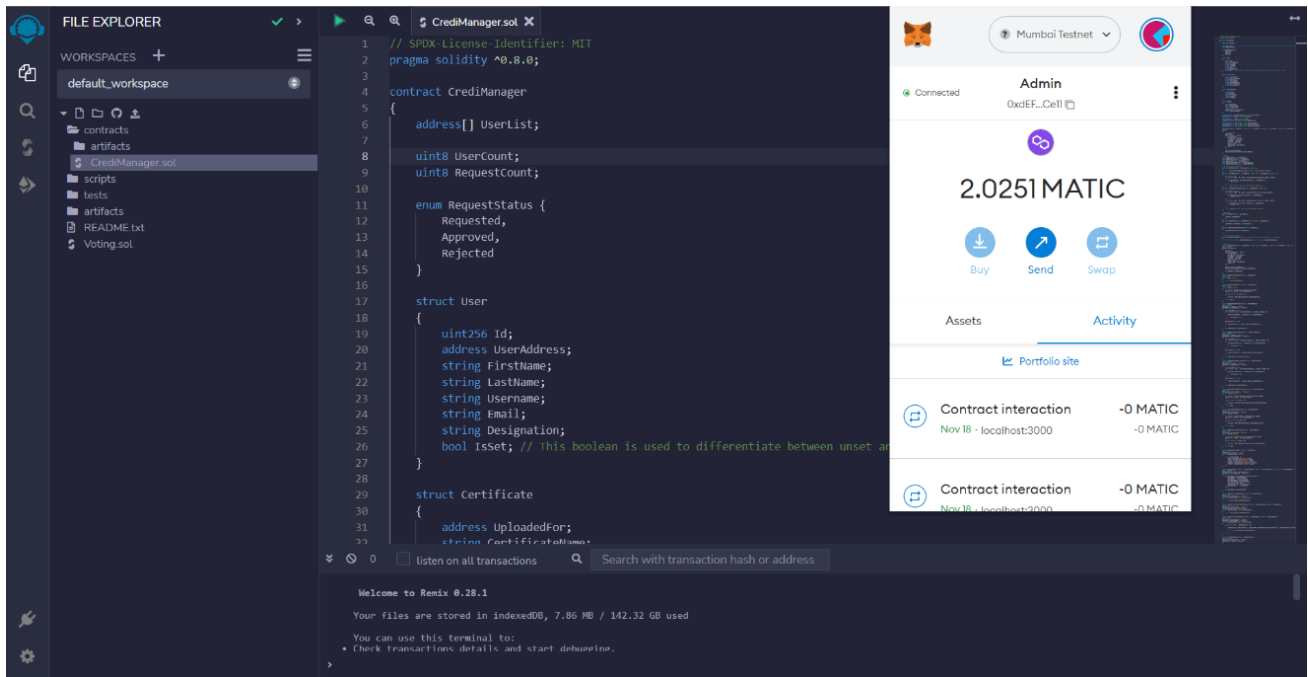


Figure 3: Remix IDE with MetaMask

- As we have five different stakeholders, we need to have at least five accounts to perform all the actions. You need to create five accounts on your MetaMask. We have created five accounts for Admin, School, Professor, Company, and Students.

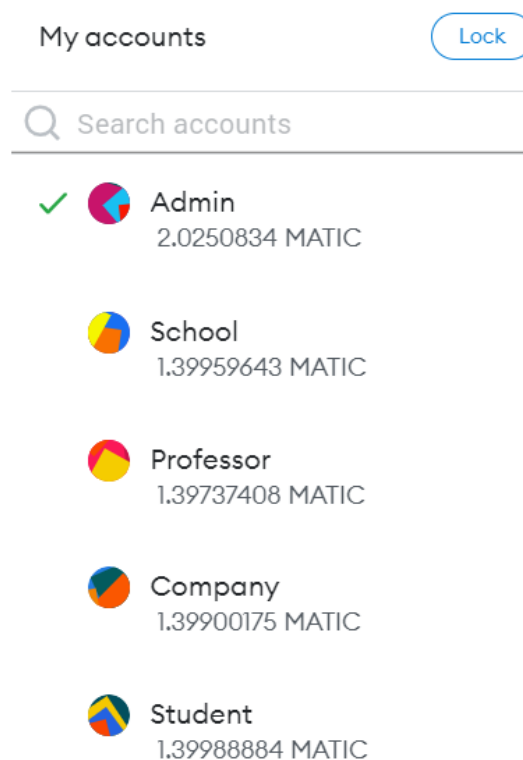


Figure 4: MetaMask Accounts

3. Our Smart Contract needs to be deployed now on Mumbai Test Network. For that, we will use our Admin Account to pay the gas fee. You need to go to the deployment section of Remix IDE and fill in the constructor parameters and click on deploy. Our Smart Contract takes user information to create the deployer as Admin of the contract.

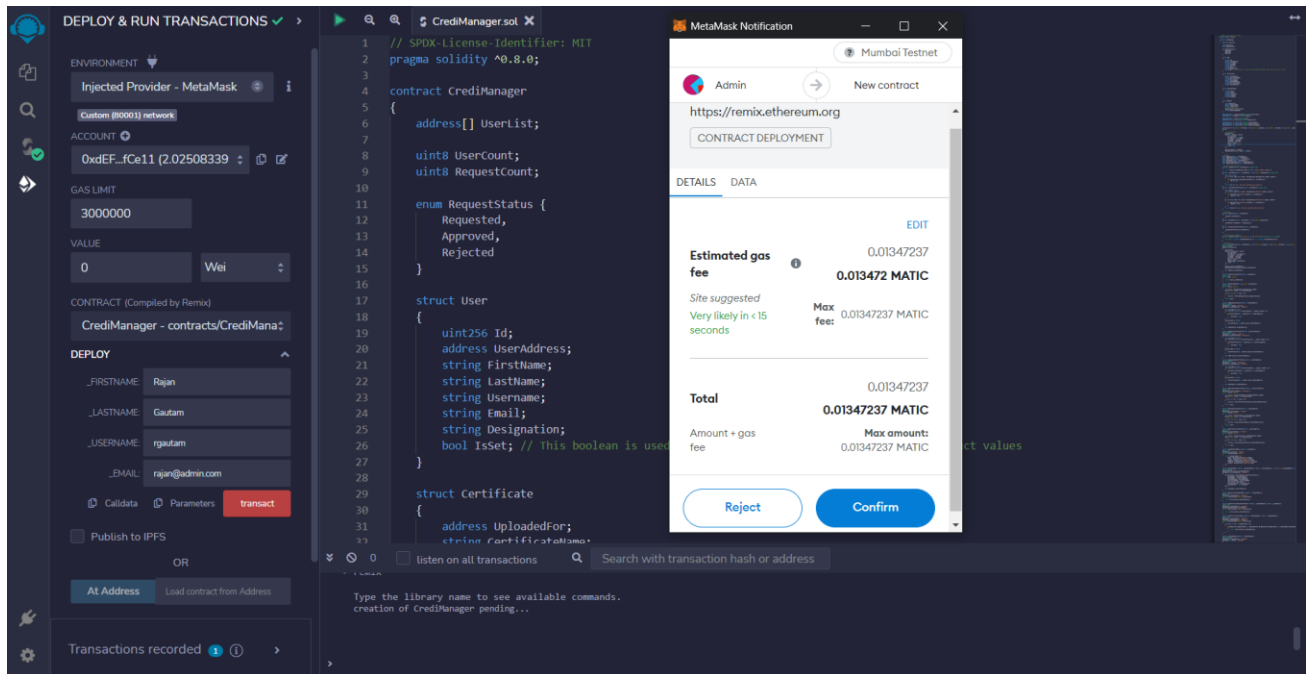


Figure 5: Deploying Contract on Mumbai Test Network

4. You can verify the contract deployment from Polygonscan. You will require the deployed contract address, which can be found under Deployed Contract.

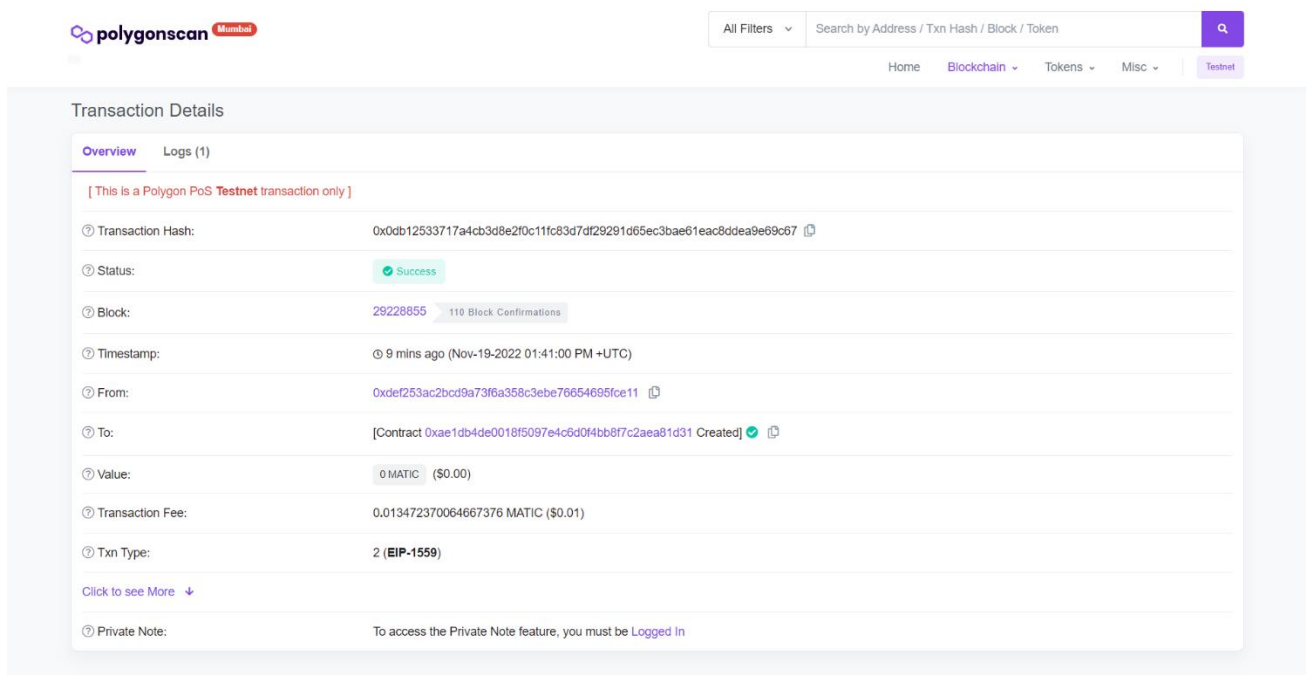


Figure 6: Transaction Details of the Contract Deployment

- Once the contract is deployed successfully, you can expand it under Deployed Contract and see all the available methods and public variables. You will get the contract address right there, you need to copy the contract address, and from the Compile tab, you need to copy the ABI. These two will be required in order to interact with the smart contract from the Web and Mobile Application.

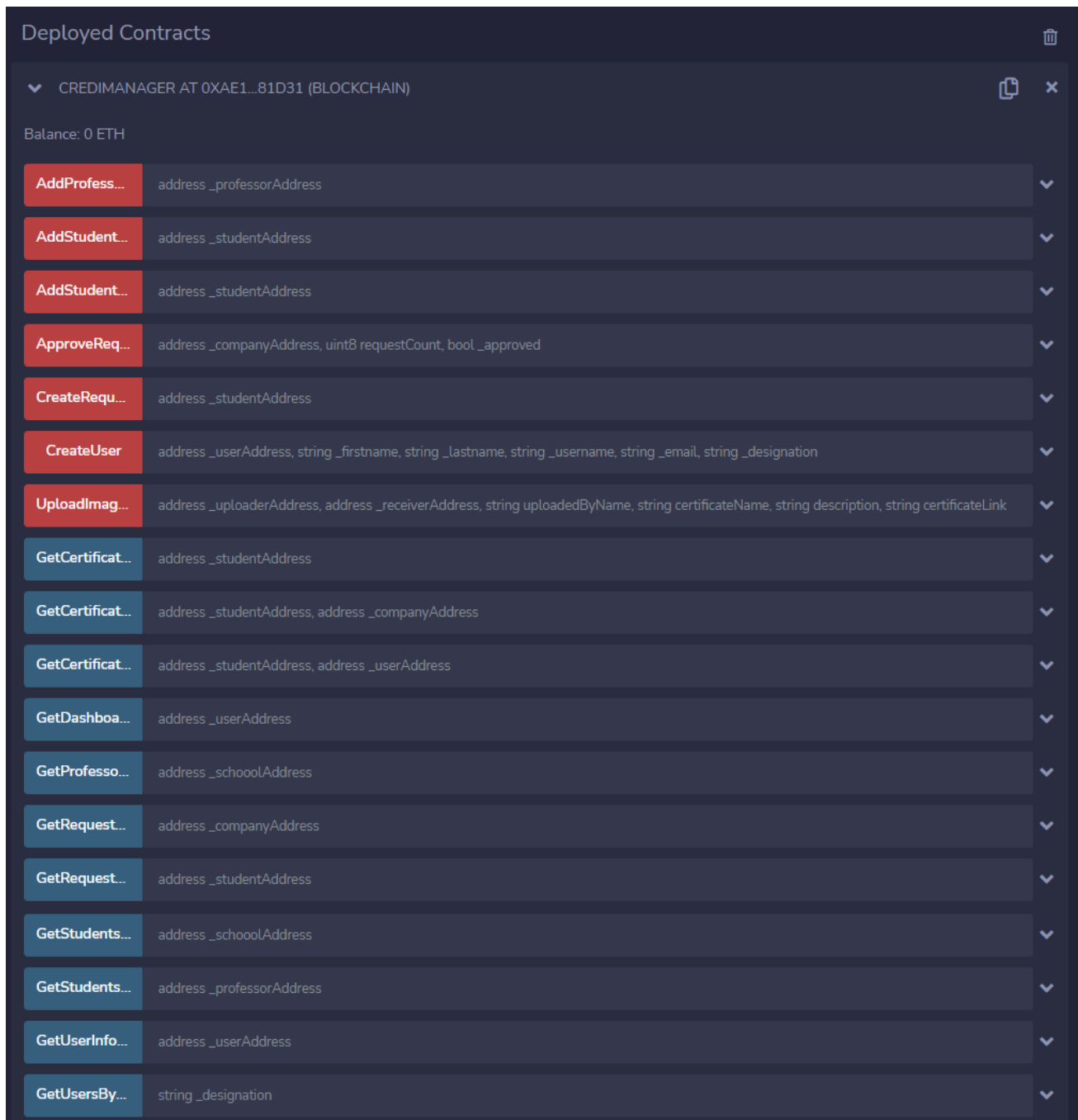


Figure 7: Available Contract Methods

With this, we are done from all fronts of contract deployment. Now, we are ready to work with the contract in the application.

IPFS

IPFS is a distributed system for storing and accessing files, websites, applications, and data. We will be using it to store the certificates. When we upload any files, it gives us the hash of the file, and with that we will be able to get the file. We are using web3.storage to interact with IPFS.

Account

Want more storage? Upgrade your plan here!

Your Plan: Free

Storage: 6.87 MiB of 5GiB used

Direct uploads 6.87 MiB

Upload more files

Files can also be uploaded directly using this web interface, as an alternative to using the API

Upload Files

Files Upload + Refresh Sort By Gateway

<input type="checkbox"/>	Name		CID		Complete file?	Storage Providers	Size	Date
<input type="checkbox"/>	Upload at 2022-11-20T09:47:18.282Z	✓	bafyb...isvjojja		Complete	Queuing...	404.77 KB	11/20/2022
<input type="checkbox"/>	Upload at 2022-11-19T15:33:13.008Z	✓	bafyb...tzldairy		Complete	Queuing...	145.94 KB	11/19/2022
<input type="checkbox"/>	Upload at 2022-11-18T17:34:14.943Z	✓	bafyb...ivefhkle		Complete	Queuing...	143.23 KB	11/18/2022
<input type="checkbox"/>	Upload at 2022-11-18T17:32:51.169Z	✓	bafyb...2wtlcv3e		Complete	Queuing...	140.08 KB	11/18/2022
<input type="checkbox"/>	Upload at 2022-11-18T14:37:49.027Z	✓	bafyb...psqajw3i		Complete	Stored (2)	188.65 KB	11/18/2022
<input type="checkbox"/>	Upload at 2022-11-18T17:31:29.491Z	✓	bafyb...6z4y2tki		Complete	Queuing...	135 KB	11/18/2022
<input type="checkbox"/>	Upload at 2022-11-11T19:18:17.667Z	✓	bafyb...binsokfi		Complete	Stored (3)	46.57 KB	11/12/2022
<input type="checkbox"/>	Upload at 2022-11-06T19:17:57.412Z	✓	bafyb...ao1n7y2y		Complete	Stored (3)	133.9 KB	11/7/2022
<input type="checkbox"/>	Upload at 2022-11-11T18:49:20.346Z	✓	bafyb...vyxegd4i		Complete	Stored (3)	5.54 MB	11/6/2022
<input type="checkbox"/>	Upload at 2022-11-11T19:51:36.808Z	✓	bafyb...it6344pa		Complete	Stored (3)	11.91 KB	11/6/2022

Delete Selected 1 2 Next View 10 results

Figure 8: Web3.Storage

Web Application:

We get the Contract Address and ABI from the Remix IDE. We use this contract address and ABI in the application to interact with the block created using the smart contract. Now, we used Next.JS to create a user interface through which each user will be able to interact with the blockchain.

To start with Next.JS Project, we must install Node.JS in our system. Once Node.JS is installed, we need to install npm; npm is a package manager for Node. It allows us to use and share JavaScript code with other developers around the world. It does this process quickly, securely, and reliably. The application starts on your local server on port 3000.

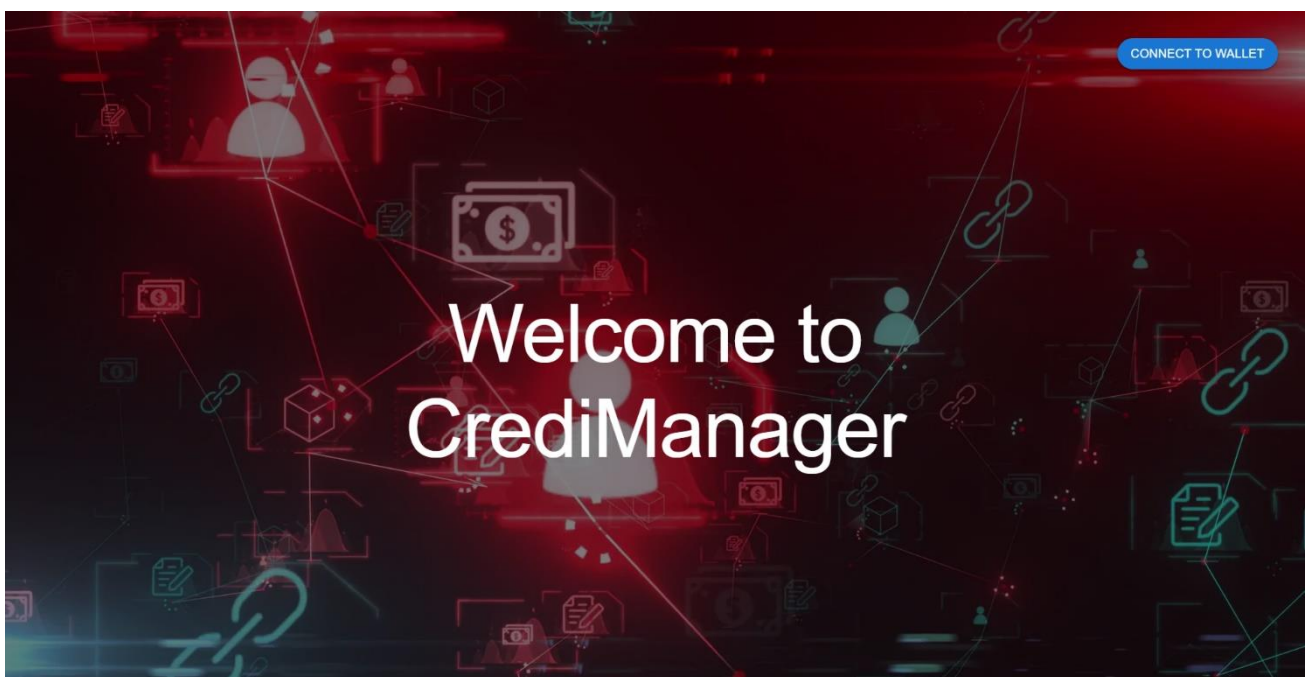


Figure 9: Credi Manager Home Page

As you can see a button at the right top corner to connect with wallet (In our case it's MetaMask). If you are not having any account in our blockchain, you will be asked to create your account. As we have already created accounts for Admin, School, Professor, Company, and Student. We will now use the functionalities added in each stakeholder content to integrate them together.

Create User

- Anyone can come and create account with different roles (School, Professor, Student, Company).
- Once they are registered successfully, they will be redirected to the dashboard page.

Register Yourself

First Name *

School

Last Name *

1

Username *

school1

Email *

school1@school.com

Role

School

None

Student

Professor

School

Company

Figure 10: Registering User

School Dashboard

CrediManager

Dashboard

Dashboard

Students

Professors

Schools

Companies

More

About Us

Contact Us

Help

© CrediManager 2022

Dashboard

Your Profile

Field	Value
User Address	0x720bE2bA09F1da5652BE94E19C14F29f64B20d4D
First Name	School
Last Name	1
Username	school1
Email	school1@school.com
Designation	School

Professors

Account *

ADD A PROFESSOR

S.N	Username	Email	Name	Action
1	professor1	professor1@professor.com	Professor 1	

Students

Account *

ADD A STUDENT

S.N	Username	Email	Name	Action
1	student1	student1@student.com	Student 1	

Figure 11: School Dashboard

- 12 -

- In the School dashboard, they will be able to see their details, added professors under school and added students under school. They will also be able to add professors or students under them.
- In order to add professor or student, they already need to have account with us with the proper role. One school can add multiple professors and students.
- They will also be able to see all the students, professors, and companies listed in the system.
- As you can see in the figure 10, one student and one professor are added in School 1.
- If the student is enrolled in the school, they will be able to view the uploaded certificates.

The screenshot displays the CredManager dashboard. On the left is a sidebar with navigation links: Dashboard, Students, Professors, Schools, Companies, More, About Us, Contact Us, and Help. The main content area has an orange header with the 'Dashboard' label and a user profile icon. Below the header, the 'User Information' section contains a table with the following data:

Field	Value
User Address	0x33D16Bd20E23B9c782eE12394baF5D6aA154a0Cf
First Name	Student
Last Name	1
Username	student1
Email	student1@student.com
Designation	Student

Below the user information is the 'Certificates' section, which includes an 'UPLOAD CERTIFICATE' button and three certificate thumbnails:

- AI**: AI For Everyone (deeplearning.ai)
- Excel**: Excel for Business (Macquarie University)
- Cyber Security**: Cyber Security - Coursera

Figure 12: Student's Certificates

- They will also get an option to upload certificate for students.

Upload Certificate

Certificate Name *

Description

User Address

First Name

Last Name

Username

Email

Designation

Section :		Shift : Day		
S.N.	Subject	F.M.	P.M.	Marks Obtained
1	Compulsory English	50	20	39
2	Physics	40	15	32
3	Chemistry	38	15	31
4	Biology	40	15	31
5	Mathematics	50	20	47
Total		218	85	180

Figure 13: School Uploading Certificate for Student

Professor Dashboard

- Professors can add students under them. Only after adding students under them, they will be able to see and upload certificates for students.
- They are also able to see all the students, professors, and companies listed in the system.
- One student is already added in the system as you can see in the figure 11.

Your Profile

Field	Value
User Address	0xbBF4c899b94AdE1B212cf11C4d8214636E196094
First Name	Professor
Last Name	1
Username	professor1
Email	professor1@professor.com
Designation	Professor

Students

Account *

S.N.	Username	Email	Name	Action
1	student1	student1@student.com	Student 1	<input type="button" value="eye icon"/>

Figure 14: Professor Dashboard

- If they view the student's detail page, they will be able to see an option to upload certificate for that student.

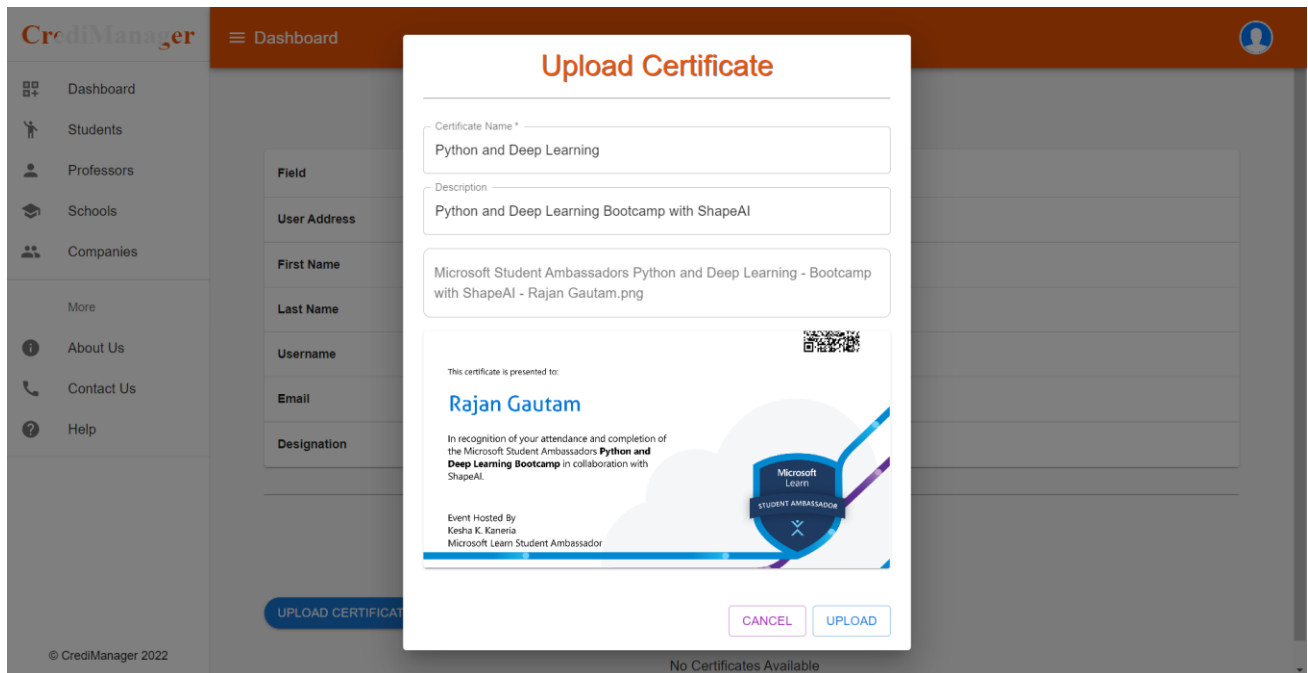


Figure 15: Professor Uploading Certificate

Company Dashboard

- Company can see their details on the dashboard.
- They can see the list of students and schools available in the system.

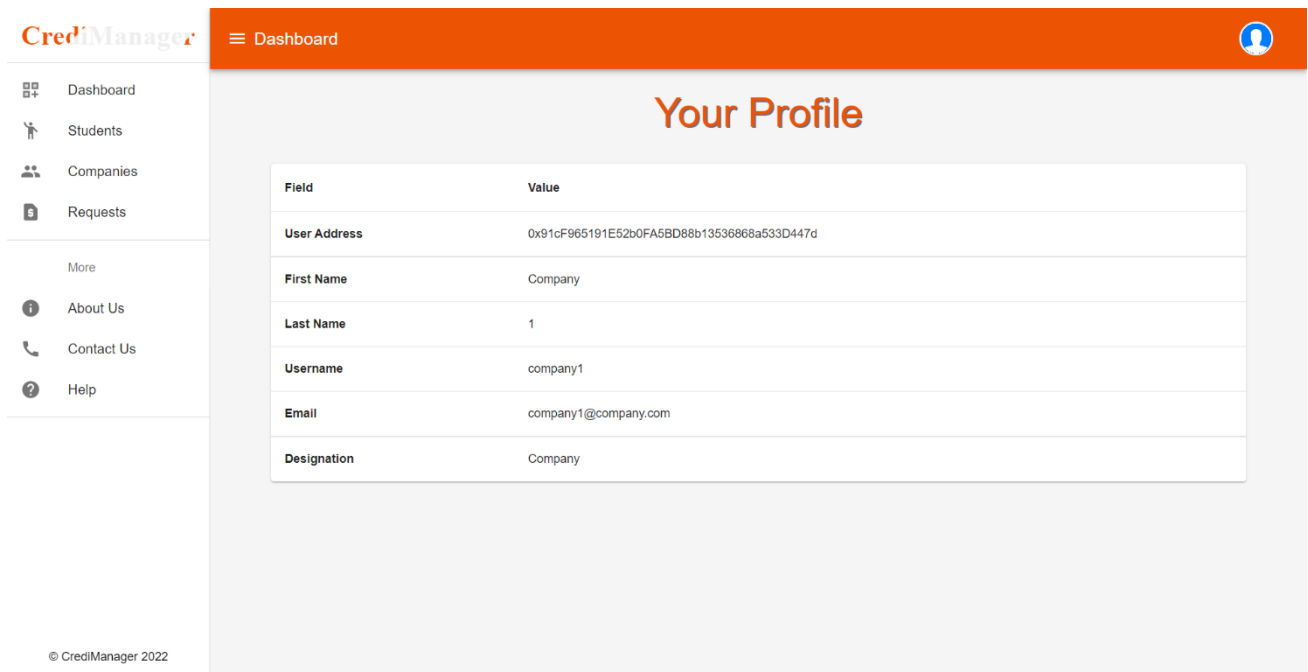
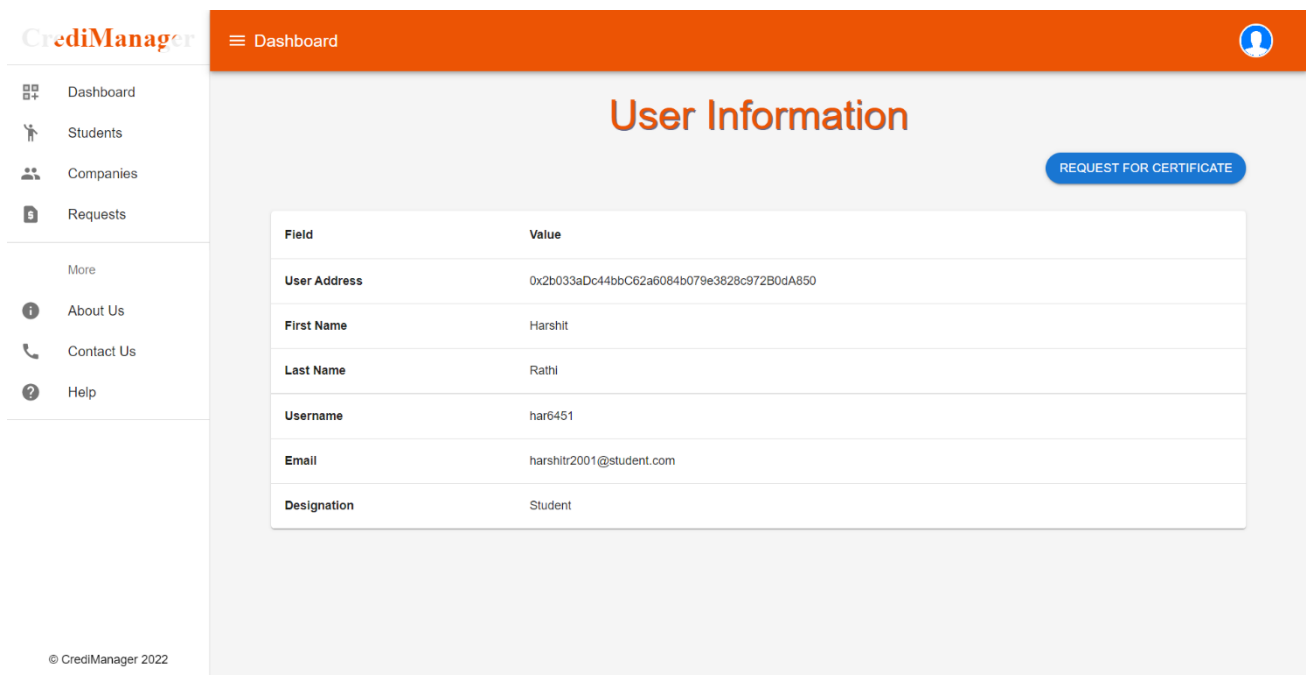


Figure 16: Company Dashboard

- If they open any student's profile, they will be able to request for certificate.



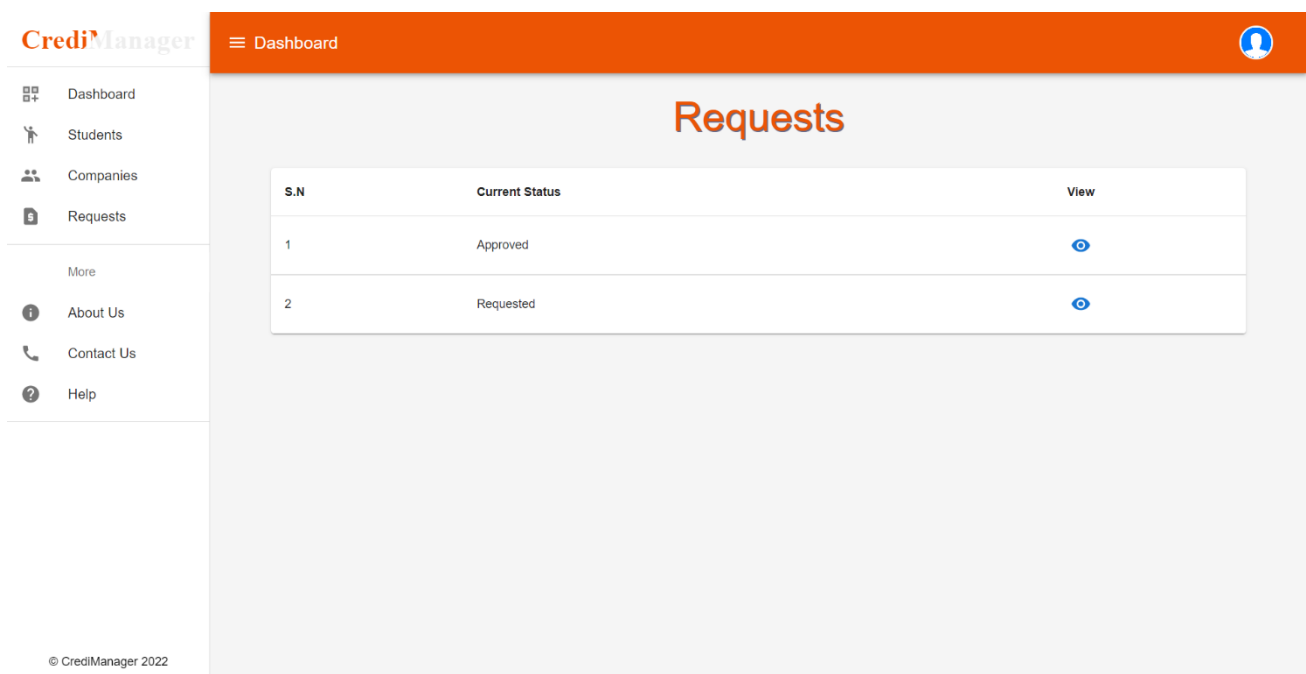
The screenshot shows the CredManager dashboard with the 'User Information' page selected. The page features a table with the following data:

Field	Value
User Address	0x2b033aDc44bbC62a6084b079e3828c972B0dA850
First Name	Harshit
Last Name	Rathi
Username	har6451
Email	harshitr2001@student.com
Designation	Student

A blue button labeled 'REQUEST FOR CERTIFICATE' is located in the top right corner of the page.

Figure 17: User Information with Request Option

- Once request is made successfully, all the requests can be seen in the requests tab.
- Request passes through three phases; Requested, Approved and Rejected. Unless request isn't accepted, they can't see student credentials.



The screenshot shows the CredManager dashboard with the 'Requests' page selected. The page features a table with the following data:

S.N	Current Status	View
1	Approved	View
2	Requested	View

Figure 18: Requests by Company

- If request is accepted, on the student's details page, they will be able to see the certificates.

CredManager Dashboard

User Information

Field	Value
User Address	0x33D16Bd20E23B9c782eE12394baF5D6aA154a0Cf
First Name	Student
Last Name	1
Username	student1
Email	student1@student.com
Designation	Student

Certificates

AI
AI For Everyone

Excel
Excel for Business

Cyber Security
Cyber Security - Coursera

© CredManager 2022

Figure 19: Student's Detail Page by Company

Student Dashboard

- Students will be able to see all the certificates uploaded by schools or professors for them on their dashboard.
- They will also be able to see basic details of other students and companies listed in the system.
- They will also be able to see the requests for certificates made by companies and do appropriate actions.



Dashboard

Students

Companies

Requests

More

About Us

Contact Us

Help

Your Profile

Field	Value
User Address	0x33D16Bd20E23B9c782eE12394baF5D6aA154a0Cf
First Name	Student
Last Name	1
Username	student1
Email	student1@student.com
Designation	Student

Your Certificates



AI
AI For Everyone



Excel
Excel for Business



Cyber Security
Cyber Security - Coursera

© CrediManager 2022

Figure 20: Student Dashboard

- They can either approve or reject the request made by any company.



Dashboard

Students

Companies

Requests



More

About Us

Contact Us

Help

Requests

S.N	Current Status	Company	Action
1	Approved		

© CrediManager 2022

Figure 21: Student's Requests Page

Admin Dashboard

- Only admin has rights to create another admin and create user on behalf of others.
- They are also able to see all the users enrolled in the system by their roles. They will also have a chart which shows the number of users in the blockchain.

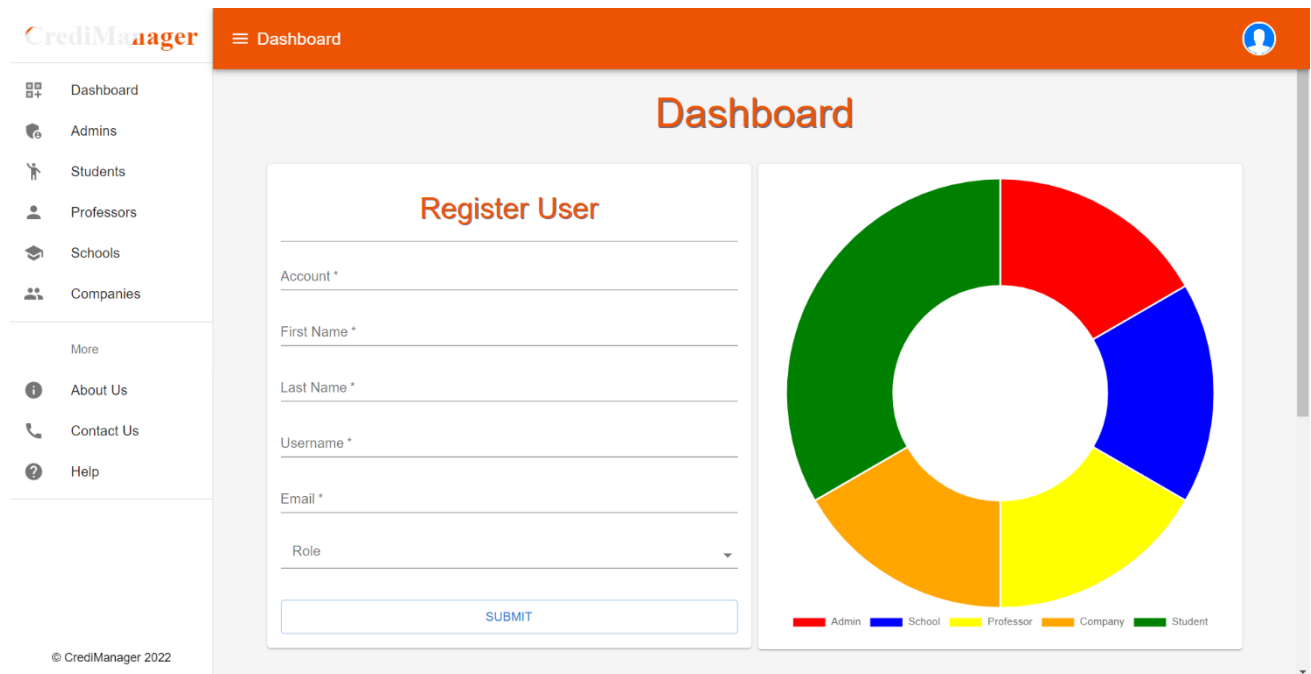


Figure 22: Admin Dashboard

- The list of Students, Admins, Professors, Schools and Companies is accessible to the admin.
- All other stakeholders are also able to view the appropriate list as seen in their navigation bar.

Students

S.N	Username	Email	Name	Action
1	student1	student1@student.com	Student 1	View
2	har6451	harshitr2001@student.com	Harshit Rathi	View

Figure 23: Students Page

Mobile Application

We get the Contract Address and ABI from the Remix IDE. We use this contract address and ABI in the application to interact with the block created using the smart contract. We used Flutter to create a user interface through which each user can interact with the blockchain.

To start with Flutter Project, we must set up Flutter in our system. It allows us to use and share Dart to build a mobile app. It does this process quickly, securely, and reliably. The application starts on your mobile.

User Sign-in

- In the mobile app, the User can sign in via a MetaMask address. It is very secure to log in via wallet.

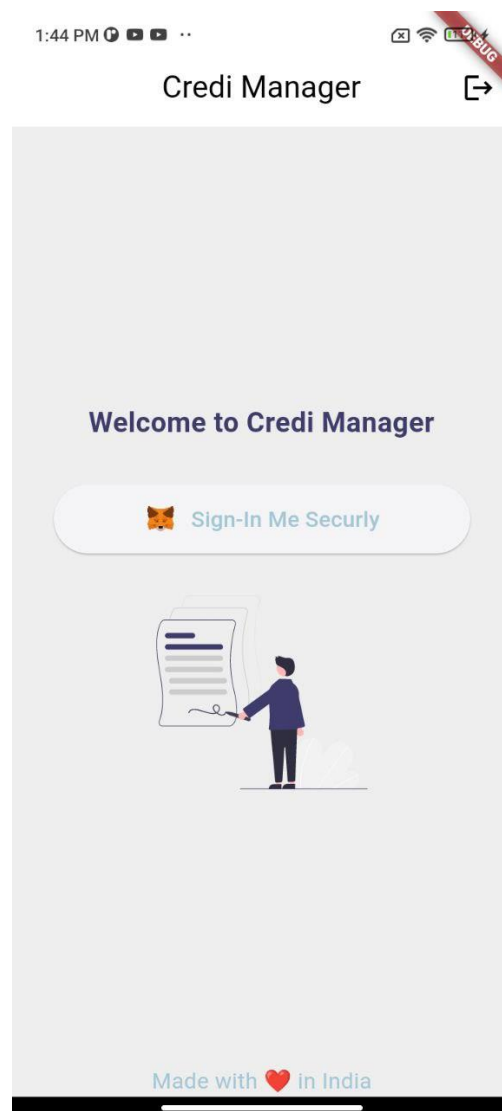


Figure 24: Sign-in Page

- In the dashboard, Students can able to see the total certificates, requested certificates, approved requests, and rejected request count.



Figure 25: Student Dashboard

- Students will be able to view a list of certificates assigned to them.
- Students can able to download the certificates.

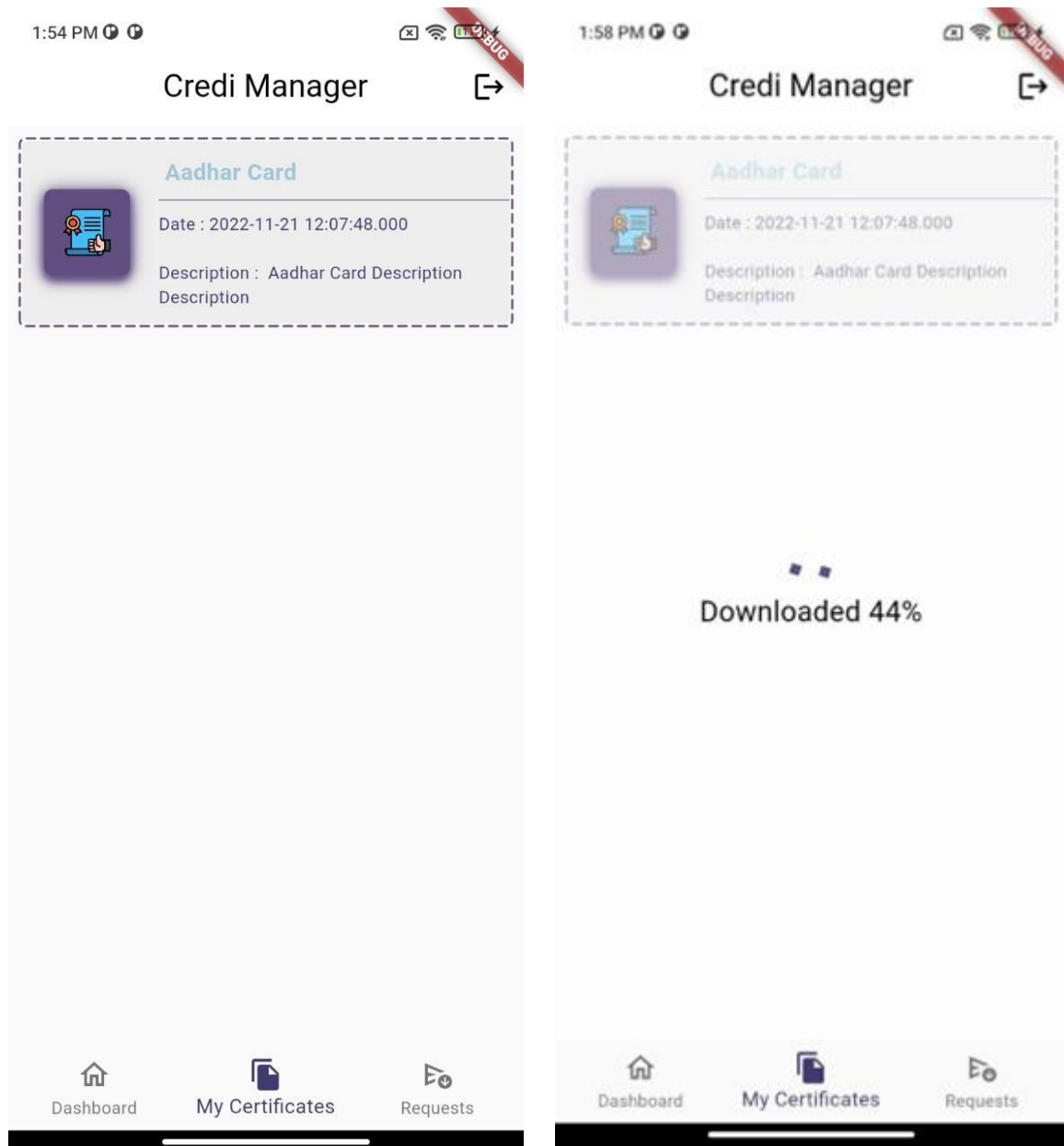


Figure 26: Listing and Downloading Certificate

- Students Can able to see the requests and able to approve and reject the requests.

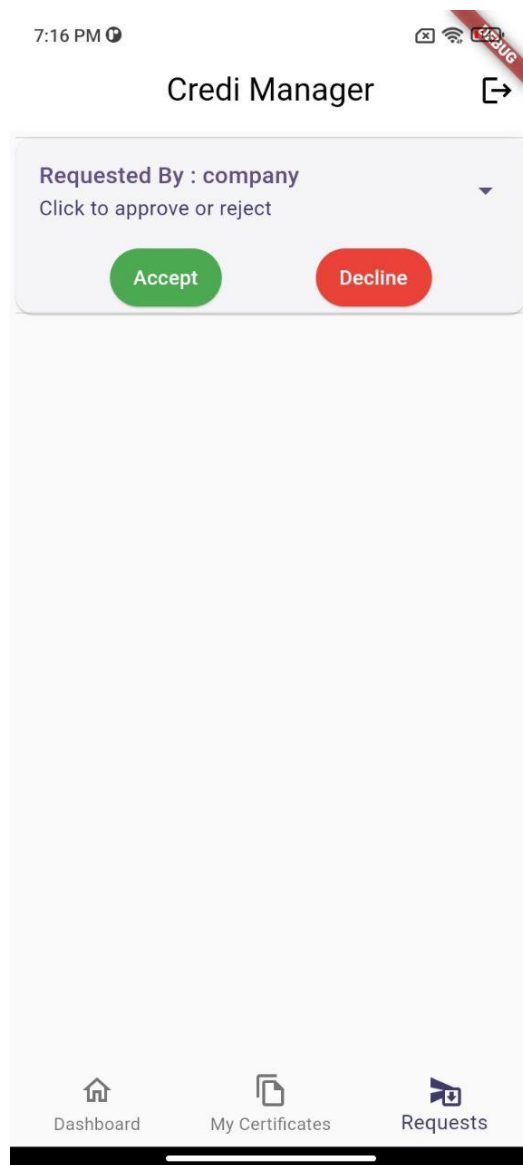


Figure 27: Request Page

Company Dashboard

- In the dashboard, Company can able to see the requested certificates, approved requests, and rejected request count.



Figure 28: Company Dashboard

- The company will be able to view a list of certificates approved to them.
- The company can able to view and download the certificates student vise.

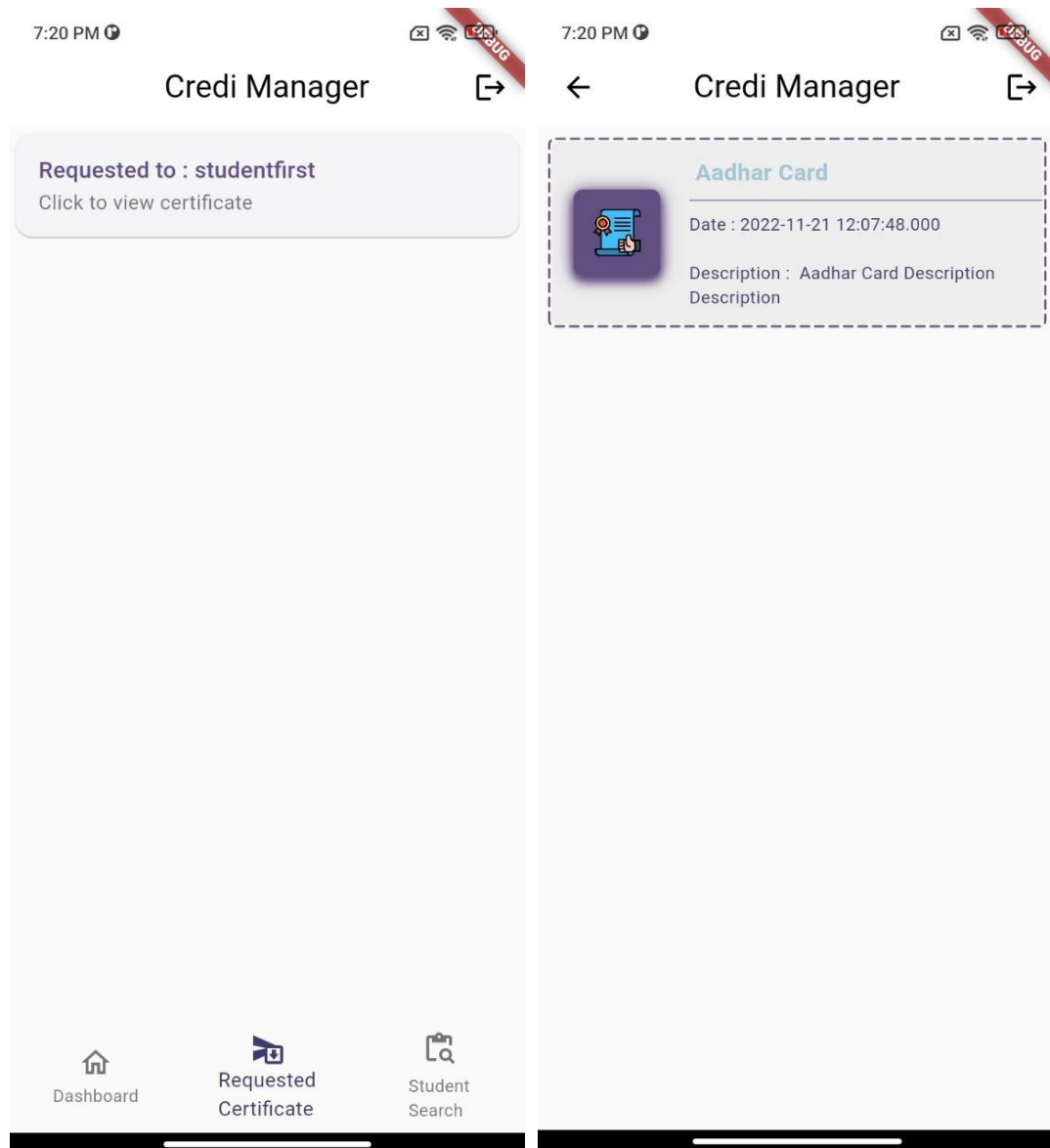


Figure 29: Company's Request and Certificate Listing

- The company will be able to search the student via username and send request for certificates.

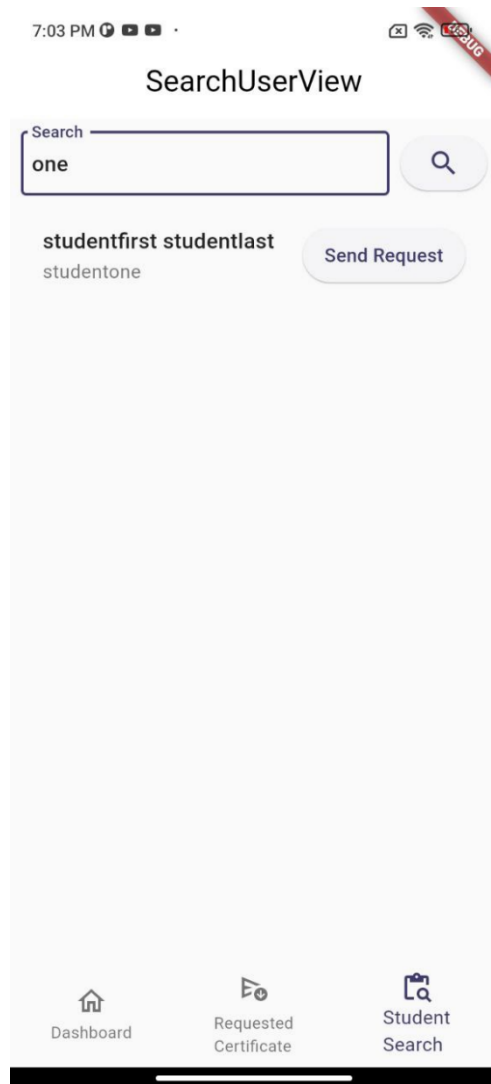


Figure 30: Searching Student

Deployment:

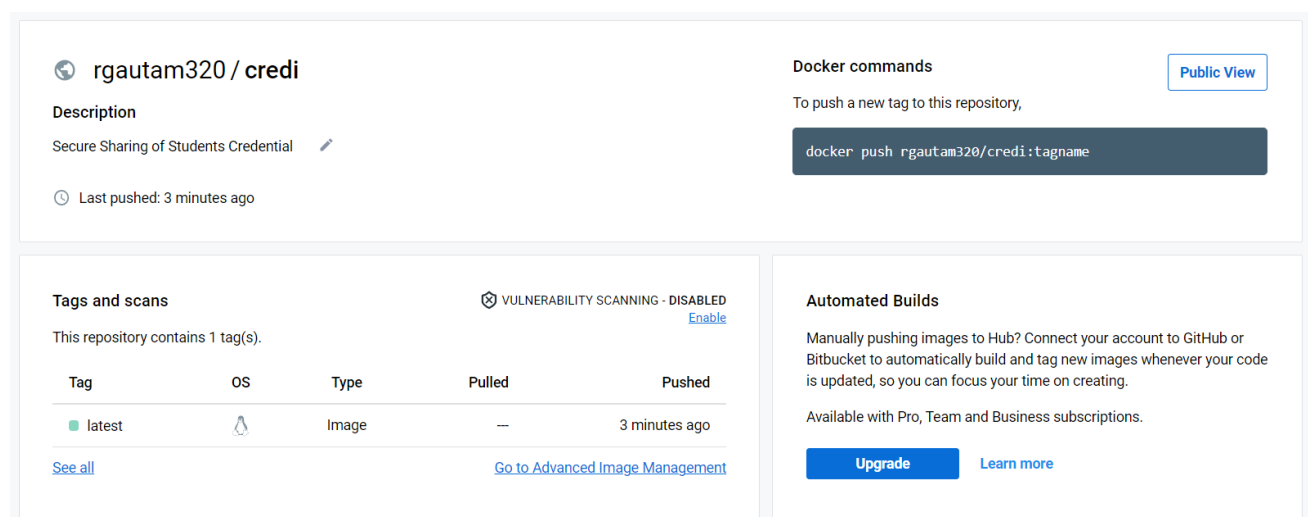
Source code of both Web Application and Mobile Application are managed using git. We have used GitHub to keep our code. You can find the code of both application in the following link.

Web Application: <https://github.com/rgautam320/CrediManager>

Mobile Application: <https://github.com/KunjKanani/CrediManager>

The Docker image has also been made and pushed to docker hub. Anyone can pull the image and run locally. The image can be found in the following link.

Docker Hub: <https://hub.docker.com/repository/docker/rgautam320/credi>



The screenshot shows the Docker Hub repository page for 'rgautam320 / credi'. The repository description is 'Secure Sharing of Students Credential'. It shows the latest tag 'latest' pushed 3 minutes ago. The page also includes sections for 'Tags and scans', 'Automated Builds', and 'Docker commands'. The 'Docker commands' section shows the command 'docker push rgautam320/credi:tagname'.

Tag	OS	Type	Pulled	Pushed
latest	linux	Image	—	3 minutes ago

Figure 31: Docker Image on Docker Hub

You have to follow the following steps to run Docker Image locally.

- **docker pull rgautam320/credi**
- **docker run --name credi 0.0.0.0:5000:3000 credi**

The Web Application is deployed in Vercel. There are two releases for the application; Preview and Production. The Preview release works on Local Ganache Network and the production one works in Mumbai Test Network. The web application can be found in the following link.

Web Application: <https://credi-manager.vercel.app/>

The demonstration of web application can be found here.

YouTube Link: <https://bit.ly/CrediManagerDemonstration>

Analysis:

We have written our smart contract with all the functions in one solidity file. It took a total of **0.013472 MATIC** or **0.94 INR** to deploy the smart contract in the Ethereum blockchain.

Table 1 shows all the payable functions executed in the smart contract; their transaction cost is in Ether and INR.

S.N.	Function Name	Transaction Cost (MATIC)	Transaction Cost (INR)
1	Contract Deployment	0.013472 MATIC	0.94 INR
2	Create a User	0.000717 MATIC	0.05 INR
3	Add Students in School	0.000202 MATIC	0.01 INR
4	Add Professor in School	0.000202 MATIC	0.01 INR
5	Add Student Under Professor	0.000717 MATIC	0.05 INR
6	Upload Certificate	0.00122 MATIC	0.09 INR
7	Create Request	0.000436 MATIC	0.03 INR
8	Approve or Reject Request	0.00018 MATIC	0.01 INR

Table 1: Transaction Cost of Each Function

Price conversion as of November 19, 2022.

The explanation for each of the functions is given below.

Contract Deployment:

- The contract deployment cost us 0.013472 MATIC or 0.94 INR. As we have all the methods in a single solidity file, that's why it costs this much.

```
constructor(string memory _firstname, string memory _lastname, string memory _username, string memory _email) payable
{
    UserCount++;
    Users[msg.sender] = User({
        Id: UserCount,
        UserAddress: msg.sender,
        FirstName: _firstname,
        LastName: _lastname,
        Username: _username,
        Email: _email,
        Designation: "Admin",
        IsSet: true
    });

    UserList.push(msg.sender);
    Designations["Admin"].push(msg.sender);
}
```

Create a User:

- This function is used to register users in the application. This is a payable function means you need to pay the gas fee to create your account.
- The cost of creating a user with all the details is 0.000717 MATIC or 0.05 INR.

```
function CreateUser(address _userAddress, string memory _firstname, string memory _lastname, string memory _username, string memory _email, string memory _designation) public payable
NotUser(_userAddress)
{
    UserCount++;
    Users[_userAddress] = User({
        Id: UserCount,
        UserAddress: _userAddress,
        FirstName: _firstname,
        LastName: _lastname,
        Username: _username,
        Email: _email,
        Designation: _designation,
        IsSet: true
    });

    UserList.push(_userAddress);
    Designations[_designation].push(_userAddress);

    emit AddUser(_userAddress);
}
```

Add Students in School:

- This function is accessible only to schools.
- With this method, schools will be able to enroll their students.
- The cost of adding students under school is 0.000202 MATIC or 0.01 INR.

```
function AddStudentInSchool(address _studentAddress)
    public payable
    CheckRole(msg.sender, "School")
    CheckRole(_studentAddress, "Student")
    {
        bool isAlready = false;
        for (uint256 i = 0; i < SchoolStudents[msg.sender].length; i++)
        {
            if(SchoolStudents[msg.sender][i] == _studentAddress)
            {
                isAlready = true;
            }
        }
        if(isAlready == false)
        {
            SchoolStudents[msg.sender].push(_studentAddress);
        }
        emit AddStudent(_studentAddress);
    }
```

Add Professors in School:

- This function is accessible only to Schools.
- With the help of this method, schools will be able to add professors under school.
- The cost of adding a professor under school is 0.000202 MATIC or 0.01 INR.

```
function AddProfessorInSchool(address _professorAddress)
    public payable
    CheckRole(msg.sender, "School")
    CheckRole(_professorAddress, "Professor")
    {
        bool isAlready = false;
        for (uint256 i = 0; i < SchoolProfessors[msg.sender].length; i++)
        {
            if(SchoolProfessors[msg.sender][i] == _professorAddress)
            {
                isAlready = true;
            }
        }
        if(isAlready == false)
        {
            SchoolProfessors[msg.sender].push(_professorAddress);
        }
        emit AddProfessor(_professorAddress);
    }
```

Add Students Under Professor:

- This function is accessible only to professors.
- With this method, professors will be able to enroll their students.
- The cost of adding students under school is 0.000717 MATIC or 0.05 INR.

```
function AddStudentUnderProfessor(address _studentAddress)
    public payable
    CheckRole(msg.sender, "Professor")
    CheckRole(_studentAddress, "Student")
    {
        bool isAlready = false;
        for (uint256 i = 0; i < ProfessorStudents[msg.sender].length; i++)
        {
            if(ProfessorStudents[msg.sender][i] == _studentAddress)
            {
                isAlready = true;
            }
        }
        if(isAlready == false)
        {
            ProfessorStudents[msg.sender].push(_studentAddress);
        }
        emit AddStudent(_studentAddress);
    }
```

Upload Certificate:

- This method is accessible to School or Professor to upload student certificates.
- The reason why this transaction costs more than most others is because of the file and interaction with IPFS.
- The cost of this method is 0.00122 MATIC or 0.09 INR.

```
function UploadImages (address _uploaderAddress, address _receiverAddress, string memory uploadedByName,
string memory certificateName, string memory description, string memory certificateLink)
    public payable
    CheckSchoolOrProfessor(_uploaderAddress)
    CheckRole(_receiverAddress, "Student")
    {
        Certificates[_receiverAddress].push(Certificate ({
            UploadedBy: _uploaderAddress,
            UploadedFor: _receiverAddress,
            UploadedByName: uploadedByName,
            CertificateName: certificateName,
            Description: description,
            CertificateLink: certificateLink,
            UploadedTime: block.timestamp
        }));
        emit Uploaded(_receiverAddress);
    }
```

Create Request:

- When the company wants access to the student certificates, they must request the certificate. And when a student approves the request, they can only view it.
- The cost of requesting permission is 0.000436 MATIC or 0.03 INR.

```
function CreateRequest(address _studentAddress)
    public payable
    CheckRole(_studentAddress, "Student")
    CheckRole(msg.sender, "Company")
    {
        Requests[RequestCount] = Request ({
            RequestId: RequestCount,
            RequestedBy: msg.sender,
            RequestedTo: _studentAddress,
            CurrentStatus: RequestStatus.Requested,
            RequestedByName: Users[msg.sender].FirstName
        });
        RequestCount++;
        emit RequestCreated(_studentAddress);
    }
```

Approve/Reject Request:

- Students can decide whether to grant permission or not. If can either Accept or Reject the request made by Company.
- The cost of accepting or rejecting a request is 0.00018 MATIC or 0.01 INR.
- This is lower than requesting permission because here, we are just updating the state of the request instead of creating.

```
function ApproveRequest(address _companyAddress, uint8 requestCount, bool _approved)
    public payable
    CheckRole(msg.sender, "Student")
    CheckRole(_companyAddress, "Company")
    {
        if(Requests[requestCount].RequestedTo == msg.sender && Requests[requestCount].RequestedBy ==
        _companyAddress)
        {
            if(_approved) {
                Requests[requestCount].CurrentStatus = RequestStatus.Approved;
            } else {
                Requests[requestCount].CurrentStatus = RequestStatus.Rejected;
            }
        }
        emit RequestApproved(_companyAddress);
    }
```


Conclusion

One incredible answer to the present sharing issue is a secure credential-sharing system built on a blockchain-based architecture. This structure is quicker, easier to use, and more productive than typical programs. The kind of file that has to be uploaded makes a major impact, and to some degree, our method worked better and quicker than the alternative, but it is also 0.47s slower. The DApp that uploads the same kind of file and size the least number of times than the other must be the most efficient in order to truly get efficiency. Along with a wide variety of scope and ideas to develop more and more with this technique, that is what we are anticipating.

References

- Raaj Anand Mishra & Team “Implementation and Analysis of Blockchain-Based DApp for Secure Sharing of Student Credentials,” *IEEE CCNC*, 2020.
- X. Tao, “The application and challenges of blockchain technology in educational practice,” *Modern Educational Technology*, vol. 1, p. 019, 2017.
- T. Nguyen, “Gradubique: An academic transcript database using blockchain architecture,” 2018.
- J. Rooksby and K. Dimitrov, “Trustless education? A blockchain system for university grades,” in *New Value Transactions: Understanding and Designing for Distributed Autonomous Organisations, Workshop at DIS*, 2017.
- J. Hope, “Give students ownership of credentials with blockchain technology,” *The Successful Registrar*, vol. 19, no. 1, pp. 1–7, 2019.
- M. Turkanovic, M. Holbl, K. Kosic, M. Hericko, and A. Kamisalic, “EduCTX: A blockchain-based higher education credit platform,” *IEEE Access*, vol. 6, pp. 5112–5127, 2018