

B. TECH. PROJECT REPORT

On

Email Validation and Arbitration Framework Platform based on Blockchain

BY
ATCHE SRAVYA
B RUSHYA SREE REDDY



DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY INDORE
May 2021

Email Validation and Arbitration Framework Platform based on Blockchain

A PROJECT REPORT

*Submitted in partial fulfillment of the
requirements for the award of the degrees
of*

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Atche Sravya

B Rushya Sree Reddy

Guided by:

Dr. GOURINATH BANDA,

Associate Professor,

Computer Science and Engineering, IIT Indore



INDIAN INSTITUTE OF TECHNOLOGY INDORE

May 2021

Candidate's Declaration

We hereby declare that the project entitled “**Email Validation and Arbitration Framework Platform based on Blockchain**” submitted in partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science and Engineering completed under the supervision of **Dr. Gourinath Banda, Associate Professor, Computer Science and Engineering**, IIT Indore is an authentic work.

Further, we declare that we have not submitted this work for the award of any other degree elsewhere.

Atche Sravya 21/05/2021

B Rushya Sree Reddy 21/05/2021

Certificate by BTP Guide

It is certified that the above statement made by the students is correct to the best of my/our knowledge.

Dr. Gourinath Banda

Associate Professor

21/05/2021

Preface

This report on “Email Validation and Arbitration Framework Platform based on Blockchain” is prepared under the guidance of Dr. Gourinath Banda.

Atche Sravya

B Rushya Sree Reddy

B.Tech. IV Year

Discipline of Computer Science and Engineering

IIT Indore

Acknowledgments

This project is done in partnership with Atche Sravya/B Rushya Sree Reddy and I would like to thank her for the continuous efforts all through the work. We would like to express our sincere gratitude to our supervisor Dr. Gourinath Banda for providing us with his kind support, valuable guidance, comments, and suggestions throughout the course of the project. And furthermore for offering us the chance to deal with certain intriguing advanced technologies that assisted us with gaining excellent information. We might likewise want to thank every one of the individuals who helped in our task, it is their help and support, because of which we were able to finish the project and report on schedule.

Atche Sravya

B Rushya Sree Reddy

B.Tech. IV Year

Discipline of Computer Science and Engineering

IIT Indore

Abstract

E-mail is a form of documentary evidence that, like all other types of documentary evidence, may be admitted as evidence in court. However, the reliability of email as evidence, like that of other types of evidence, will be scrutinized. In this project, we try to create a third-party service that guarantees the trust and reliability of emails by offering a way to resolve conflicts.

Table of Contents

Candidate's Declaration	5
Certificate by BTP Guide	5
Preface	7
Acknowledgments	9
Abstract	11
List of Figures	15
1. Introduction	18
1.1 Current Architecture of Email Systems	18
1.2 Disadvantages of Centralized Architecture	19
1.3 Problem statement	20
2. Blockchain and its advantages	21
2.1 Introduction to Blockchain	21
2.2 Features of Blockchain	22
2.2.1 Decentralization	22
2.2.2 Immutability	22
2.2.3 Robust Encryption	23
2.3 Smart Contracts	23
2.4 Applications of Blockchain	24
3. Solution Architecture	25
3.1 Type of Blockchain used	25
3.2 Authentication Mechanism	26
3.3 Transaction Generation	27
3.4 Encryption Mechanism	28
3.5 Work Flow	31

4. Implementation Details	34
4.1 Tech Stack	34
4.2 User Interface Snapshots	37
5. Results	43
5.1 Conclusion	43
5.2 Future Work	44
Bibliography	45

List of Figures

Figure 3.1: Workflow Diagram

Figure 3.2: Sequence Diagram

Figure 4.1: Architectural Diagram

Figure 4.2: Registration Page

Figure 4.3: Login Page

Figure 4.4: General User Interface

Figure 4.5: Find My Email Page

Figure 4.6: Admin User Interface

Figure 4.7: Add Judiciary Page

Figure 4.8: Remove Judiciary Page

Figure 4.9: Decrypt Page at Admin

Figure 4.10: Judiciary User Interface

Figure 4.11: Decrypt Page at Judiciary

Figure 4.12: Validation Page

Chapter 1

Introduction

1.1 Current Architecture of Email Systems

Email is an internet-based system for sending and receiving personal and professional messages and it has been the most common way of communication. There are two subsystems in the email system architecture,

1. The user agents are used to read, send, compose, and reply to messages, as well as to view incoming messages, filter, search, and delete them. Some of the most common user agents include Google Gmail, Microsoft Outlook, Mozilla Firefox, and Apple Mail.
2. SMTP (Simple Mail Transfer Protocol) is used by message transfer agents to transmit messages from the source to the destination. They're often referred to as mail servers.

The following is the general email flow:

- When a user wants to send an email, he or she uses a user agent to generate an email that includes the recipient's address, message subject, message content, and other relevant information.
- Using a client-server architecture, the MTA now takes responsibility for transferring these emails at the receiver UA's end.
- The user or a timer at the receiving end triggers the UA. The user is notified by the UA that email is available. UA shows the user a list of the messages in their inbox. Now, the

user can choose which message he wants to read, and the text will appear on the screen or any other action he/she wants to perform on messages.

1.2 Disadvantages of Centralized Architecture

As email services are centralized in servers, which is evident from the email system architecture, users must have faith in the ESPs (email service providers) that will handle their email messages without encryption, as this is not mandated by email protocols. As the consumer relies solely on the ESP he/she has selected, this could result in a number of issues.

One of the most significant disadvantages of centralized architecture is the probability of a single point of failure, which occurs because individual user machines depend on a single central server to manage all processes. If the servers go down, their users would be unable to access their email accounts.

Repudiation: Since email messages can easily be forged, someone sending you a message can later deny sending it, and proving it is difficult. This has ramifications for business correspondence where emails are used as contracts.

Considering the centralized architecture, when a sender sends an email, it can take a long time for it to reach the recipient, or it may even get lost (though this is extremely rare), leaving the sender in the dark about the status of the email and also as all data on a centralized architecture would pass through one location, it's very simple to monitor and collect data across the network, which may make it vulnerable to illegal attacks. Furthermore, users must have faith in both their ESPs and the receivers' servers, assuming they do not tamper with the data they send.

1.3 Problem statement

As we've seen, attorneys must always go above and beyond to prove the authenticity of emails while also complying with other standards of proof that apply to other means of communication. The following are several scenarios in which the trustworthiness of email can be called into question.

1. A printed email is not admissible in court since the other side will simply dispute the validity of the email. You can easily modify the email address, timestamp, and message text by altering the HTML or in some other way nowadays. As a result, the other side will easily accuse you of altering the email and printing it.
2. Approximately 3% of all non-bulk emails never make it to their intended recipient. This means that cc-ing yourself on an outgoing email does not guarantee that it will meet the intended recipient. You will have to prove that your critical message was not included in those emails.
3. In the current email system architecture, power is concentrated in a single centralized authority that controls and manages the network. Hackers and data breaches are easy to come by with this centralized architecture. As a result, it isn't entirely trustworthy.

As a result, the trustworthiness of e-mail as evidence can be questioned. So, our goal is to comprehend the issues raised above and to design a solution to provide conflict resolution protocols that ensures the trust and reliability of emails by using Blockchain technology. All sensitive data will be encrypted, and there will be no compromises in the confidentiality, credibility, or protection of any Personal Information, including transmission, unlawful or unauthorized access, alteration, use, disclosure, acquisition, deletion, or destruction thereof.

Chapter 2

Blockchain and its advantages

2.1 Introduction to Blockchain

The concept of Blockchain is derived from Santoshi Nakamoto's 2008, Bitcoin cryptocurrency. Since then more than 2000 cryptocurrencies are available now in the market. In recent years the power of Blockchain has been unveiled and its usage is not just bounded to cryptocurrencies. They have been adopted in many applications of various domains.

Blockchain is a reliable, non-centralized, and difficult to use for fraudulent purposes type of database storage. It's a method of storing data in such a way that changing or deleting it is difficult or impossible. It's a chain of blocks with each one containing a cryptographic hash of the previous block, a time stamp, and the data for that block. Because of the nature of the cryptographic hash, data stored in a blockchain is inherently resistant to alteration: if one block of data is changed, all subsequent blocks should be regenerated with new hash values. Immutability is a crucial function of blockchain applications.

A blockchain is a decentralized ledger of transactions that is replicated and distributed throughout the blockchain's entire network of computer systems. It transfers the control and decision-making from a centralized authority (individual, organization, or group thereof) to a distributed network. Here, all the changes are reflected consistently for all the parties in the network. Since the parties using the blockchain do not need to trust a powerful third party to manage the business, this decentralization fosters trust among participants.

2.2 Features of Blockchain

2.2.1 Decentralization

Since the debut of Bitcoin in 2008, the first decentralized peer-to-peer electronic cash system, blockchain technology has advanced significantly. Anyone can participate in a decentralized network and transact on the ledger. The information does not move through a single point and instead travels through a variety of different points in a decentralized network architecture, which allows for greater privacy. This eliminates the issue of a single point of failure while also making network tracking much more difficult. Blockchain technology, because of its decentralized and trustless nature, will open up new doors and support businesses by improving security, transparency, and traceability.

2.2.2 Immutability

Immutability can be described as a blockchain ledger's ability to remain unchanged, or a blockchain's ability to remain unaltered, in terms of Blockchain technology.

A hash value or a cryptographic theory is used to process data present in every block. Each block contains a digital signature and a hash value for both itself and the previous block. This means that blocks are unremittingly coupled together. This feature assures that no one can alter or manipulate the data contained in the system or hack into it. Unaltering in Blockchain does not imply that data cannot be modified; instead, it is a measure of the complexity of changing data on a blockchain. Furthermore, any tampering is always detectable, guaranteeing the blockchain's credibility.

2.2.3 Robust Encryption

Hash functions and Asymmetric key algorithms are the most widely used cryptographic techniques in the blockchain. Hash functions are employed to provide each participant with the capability of a single view of the blockchain. The SHA-256 hashing technique is commonly used in blockchains as the hash function. Hash functions play an important role in connecting blocks and ensuring the integrity of the data stored within each block. Any change to the block data can cause inconsistency, causing the blockchain to be rendered invalid.

2.3 Smart Contracts

Blockchain was first used to build Bitcoin (a peer-to-peer digital payment system), but it has now expanded to include a broad variety of decentralized applications. Smart contracts are an enticing technology that can be implemented on top of the blockchain.

A smart contract is a piece of blockchain-based code that executes, facilitates, and enforces the covenant between untrustworthy parties. It can be thought of as a scheme that distributes digital assets to all or any of the parties involved when certain conditions are met. Smart contracts, unlike conventional contracts, have lower transaction costs as they do not require the intervention of a trusted third party to function. They can send or receive messages, transfer money from users/other contracts, read or write to its private storage and can also build new contracts. Smart contract's state is saved on the blockchain and is updated every time the contract is used. A fixed address of 20 bytes will be allocated to each contract uniquely. Smart contracts are immutable, which ensures that their code cannot be altered or upgraded after they have been added to the blockchain. They represent a digital contract or agreement.

Users should simply submit a transaction to the contract's address to initiate the contract. The transaction is made only after all the nodes in the network execute and reach a consensus. As

a result, the state of smart contracts will be changed. Smart contracts can be developed on a variety of blockchain platforms, including Ethereum, Bitcoin, and NXT, but Ethereum is the most common. This is due to Ethereum's language's Turing-completeness feature, which allows for more advanced and customized contracts to be created.

2.4 Applications of Blockchain

Blockchain technology is used in a variety of sectors, including healthcare, financial services, government, and travel, among others. This technology simplifies and streamlines the whole asset management and payment process, in the finance industry, by enabling an integrated trading lifecycle in which all parties have access to the same transaction details. This eliminates the need for intermediaries, while still ensuring transactional data confidentiality and performance. Blockchain has the potential to transform the healthcare industry by improving the security, privacy, and interoperability of clinical records. Through properly connecting and exchanging data with Blockchain, this technology has the potential to enhance government services and operations. This technology allows for efficient data processing across various agencies, as well as improved transparency and a better way to track and audit transactions. This Blockchain technology has a wide range of uses, including money transfers and the storage of essential documents such as identity cards or passports.

Chapter 3

Solution Architecture

3.1 Type of Blockchain used

Private and public blockchains are the two primary categories of blockchains. There are, however, several variants, such as Consortium and Hybrid blockchains. Before we get into the specifics of the various forms of blockchains, let's look at what they have in common. Any blockchain is made up of a group of nodes linked by a peer-to-peer (P2P) network. Every network node has a copy of the shared ledger, which is updated on a regular basis. Each node has the ability to verify transactions, send and receive messages, and generate new blocks.

Public Blockchain: A public blockchain is a permissionless, non-restrictive distributed ledger scheme. Anyone with internet access can sign up for a blockchain platform to become an approved node and join the network. A public blockchain node or user is allowed to access current and historical records, verify transactions or perform proof-of-work for an incoming block, and mine. Bitcoin, Ethereum, and Litecoin are only a few examples.

Private Blockchain: A private blockchain is a permissioned or restricted blockchain that can be used in a closed or secure network. Private blockchains are commonly used by enterprises or organizations where only chosen users are nodes in a blockchain network. The governing organization is in charge of the standard of compliance, authorizations, permits, and usability. As a result, private blockchains are similar to public blockchains in terms of functionality, but they have a smaller and more restricted network. Supply chain management, wealth ownership, digital identities, and other applications use private blockchain networks. Multichain and Hyperledger projects like Sawtooth, Fabric, Iroha, Corda, Ethereum, and other private blockchains are few examples here.

An average of 121 emails are sent/received every day by a businessperson around the world, with around 2.4 billion emails produced per second. Ethereum typically handles 10-15 transactions per second, with each block containing approximately 70 transactions. The average block time in Ethereum is 15 seconds, and this does not vary significantly over time. As a result, there will be approximately 5760 blocks every day. Given the huge disparity between the number of emails produced and the number of ethereum transactions possible per second, it is necessary to restrict access to this service to registered users rather than the general public.

Considering the reasons below we opted to use a Private Closed Blockchain.

- Only registered and verified users can log into the Blockchain network after a suitable authentication mechanism.
- The access and resources are controlled by the enterprise.
- There will be fewer nodes that participate in this private network, as a result, performance will be elevated and various efficient consensus mechanisms can be used.
- As an enterprise, you'll almost always have regulatory standards to meet, and getting leverage of our own facilities or infrastructure can help you meet these requirements more easily.
- The ability to connect nodes and resources on-demand can be a huge benefit to an enterprise.

3.2 Authentication Mechanism

Our solution is a third-party service provider which focuses on providing services only to the registered users. For registering or authenticating these users there is a need of having a separate authentication mechanism that stores the info of these registered users. So we are using a centralized database system for storing credentials that are required for the authentication mechanism.

The database will store the following elements for each user:

- Aadhar Number (Unique ID)
- Email
- First Name
- Last Name
- Password

Any database, or even a blockchain in combination with other smart contracts, may perform these functions. Though MongoDB is related to a centralized database, we used this to build the prototype for the sake of convenience and also because it is not the primary focus of this project. Any relational database can conveniently apply the data model used in this study. However, because of its simplicity, there are no significant advantages or drawbacks between the various options. The Aadhar Id is used to uniquely identify the user and the remaining details are the minimal requirements that are needed for the registration. The email and password here are used to log in to the user every time he/she wants to use the service.

3.3 Transaction Generation

The movement of an email message from the sender to the recipient's inbox is something that happens behind the scenes of general email flow. When a person or entity sends an email, the message spreads around the Internet from its point of origin, such as an email client where it was composed. It goes through several servers along the way, ensuring that it arrives at the correct location. So, when the sender sends an email, it goes from his mailbox to his SMTP server and passes through the network to reach the receiver's SMTP server, from where it gets downloaded into the receiver's mailbox where he/she can access his/her emails.

According to our proposed solution architecture, we are triggering the process to store emails whenever a new mail is being sent/received from the registered user. As discussed in

chapter 1, considering the disadvantages of the centralized architecture, there are possibilities of email that is being sent by the sender not reaching the receiver's mailbox. So, the actual process of generating a transaction for storing email will only be done after an acknowledgment is received from the receiver's end which confirms that mail has been received successfully. There are various apps and extensions that provide email tracking services that can help us with these acknowledgments. This is not the primary focus of this project and will be included in future work.

The details in blockchain for each email stored are as follows

- Url of encrypted email data that is stored on IPFS
- Timestamp
- Email Address
- Message-Id

Though the message-id for each email is unique, we extract message-id, and time stamp from the email using eml-parser and are using these details to uniquely identify the emails. Using web3.js we interact with the functions in the smart contract which is written on top of our blockchain. These function calls generate a new transaction whenever there is a change in the state of the smart contract which occurs while storing the required email details. These newly generated transactions are added into a transaction pool and upon successful mining, they will be added into a new block on the blockchain. More details about the encryption details and IPFS will be discussed in the upcoming sessions.

3.4 Encryption Mechanism

The practice of encoding and decoding data using mathematics and computing is known as cryptography. This technology is used for a variety of purposes,

- Including protecting network transfers, monitoring the creation of new currency units, and verifying the movement of digital assets and tokens. Cryptography, in its most basic form, is a method of sending encrypted communications between two or more parties.

- The sender encrypts/hides a message with a specific key and algorithm, then sends the encrypted message to the recipient, who decrypts it to reveal the initial message. The most crucial part of cryptography is encryption keys. They render a transaction, message, or information undecipherable to an unintended reader or receiver, allowing only the rightful owner to read and process it. These keys aid in the concealment of documents.

We have utilized the following cryptographic techniques for better privacy, validity, and assurance of information.

Hashing :

Hashing Algorithms (or Hash Functions) are intended to produce a hash value from a piece of information. Hash is a character string representation of a bit sequence. For a given function, the length of the hash is generally fixed. Whatever might be the size of the original data, it will be mapped to a hash of explicit length. Changing the original data by even the slightest bit results in a completely different hash value. This property is utilized to ensure that all transmitted data is intact. In blockchains, hashing is utilized widely and can be viewed as fundamental for their functioning. For instance, the strength of connections between blocks is generally guaranteed by composing the hash of the past block into the current block. Along these lines in the event that anybody endeavors to change anything in a block, they would likewise have to change all subsequent blocks as well, and they'll have to do so on a huge number of nodes at once.

Symmetric Encryption :

It encrypts the data at the source, sends the encrypted data to the intended receiver, and then decrypts the data at the receiver's endpoint, all using the same secret key. This strategy has the

benefit of being simple to implement with low computational costs, but it raises questions about protection issues with the shared key and issues of scalability.

Asymmetric Encryption :

Public-key cryptography utilizes two different keys known as public and private keys for the encryption and decoding mechanisms. The public key can be shared freely whereas the private key must be kept undisclosed and both of them are unique for each user. When the public key is used to encrypt some information it can be decrypted only by the concerned private key. This depends on some intriguing numerical properties and empowers two participants who have never met to safely share data.

Digital Signature :

A digital signature is a cryptographic system for displaying the validity of digital communications or records. A legitimate digital signature gives a recipient reason to assume that the message was made by the presumed sender (authentication), that the sender cannot refute sending the message (non-repudiation), and that the message was not modified on the way (integrity). Asymmetric cryptography is used in digital signatures. In several ways, digital signatures are similar to handwritten signatures, but correctly applied digital signatures are harder to counterfeit than signatures written by hand.

The encryption mechanism used while storing emails according to our designed solution architecture is as follows :

Level-1: Digitally Signing email content by user's private key

Level-2: Encryption of data returned in level-1 using the public key of the user

Level-3: Encryption of data returned in level-2 using the public key of Service Provider

The digital signature in Level-1 helps us in verifying the retrieved email content is neither tampered with nor corrupted. As the email data is being stored on a public blockchain/file storage system, these subsequent encryption layers using the user and service provider's public keys ensure that data is secured and is only accessible with the consent of both.

The decryption mechanism used while retrieving emails according to our designed solution architecture is as follows :

Level-1: Decrypt retrieved content using the private key of Service Provider

Level-2: Decrypt of content returned after Level-1 using the private key of the user

Level-3: Verify the email data using Digital Signature

Corresponding to these above-mentioned encryption layers, while retrieving the email data, initially it must be decrypted using the service provider's private key, and then this semi-encrypted must further be decrypted using the user's private key. Once this is done, the email data can now be verified using the digital signature. For this, the sender of the email who digitally signed the data must provide his private key, which will be used along with the hash of data to retrieve the address of the rightful owner. This retrieved address helps us in the verification process to check any tampering that might have occurred in transit.

3.5 Work Flow

There are three types of actors in this proposed solution architecture.

- 1) User who registers for this service
- 2) Judicial entity who can resolve the dispute
- 3) Administrator who manages the email requests

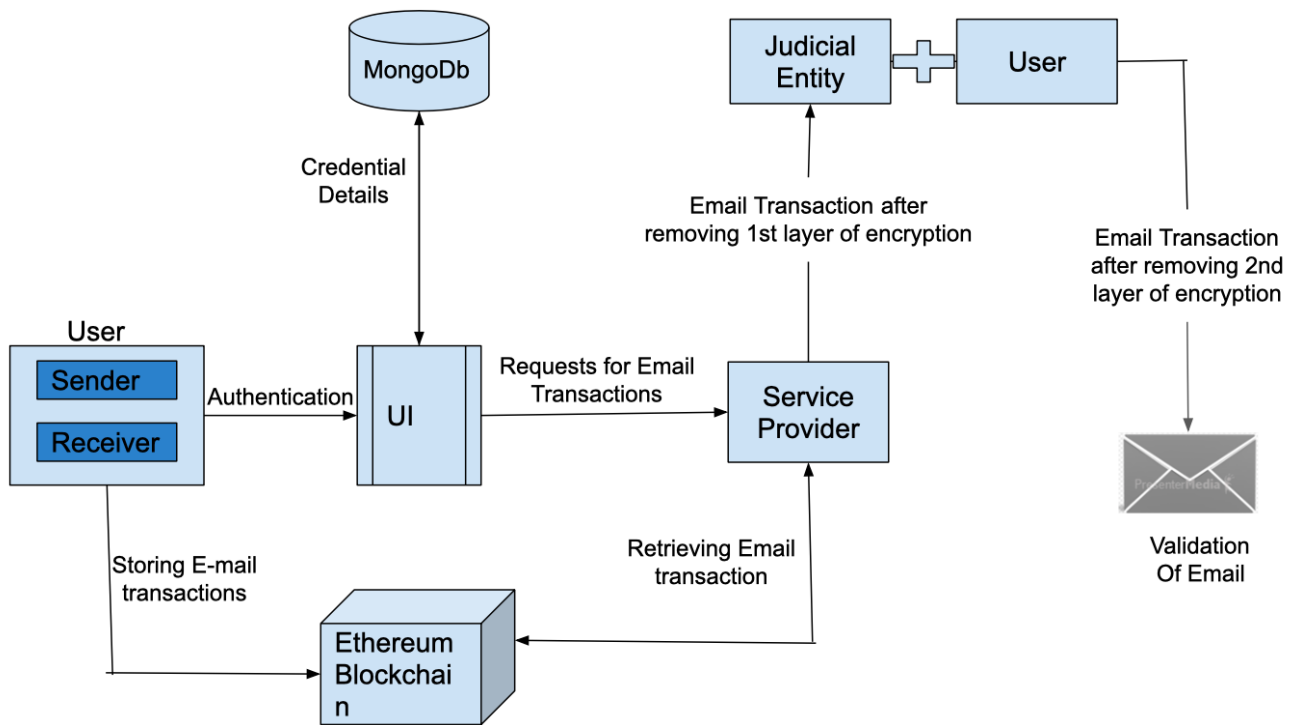


Figure 3.1: Workflow Diagram

Once the user registers for this service, they will be able to access the functionalities such as storing emails on the blockchain, placing requests for retrieving conflicted emails, and selecting the judiciary. Only the administrator has the authority to either add or remove judiciary after successful verification.

As discussed in section 1.2 in case of conflict scenarios that question the reliability of the email as evidence, the user can place a request for sending a copy of the valid/ legitimate email to some selected judiciary to prove his/her claim in the dispute by providing necessary details of the email (message Id, Timestamp).

With the email requests being submitted, all these requests are sent to the administrator to get processed for the next step. Then the service provider initiates the search process to find that particular email on the blockchain. If the email is found, the administrator of the service provider

selects the email request and uses his/her private key to decrypt the first layer of encryption, and sends this partially encrypted content to the concerned judiciary.

At the judiciary end, once the file with email data is received, the user uses his/her private key in the presence of the judiciary to decrypt the received file to get processed for the next step. After all of these processes are done, the judiciary checks for any discrepancies and validates the email data using the digital signature of the user and thus proceeds further in the dispute resolution.

In summary, the sequence diagram is as shown in the below diagram:

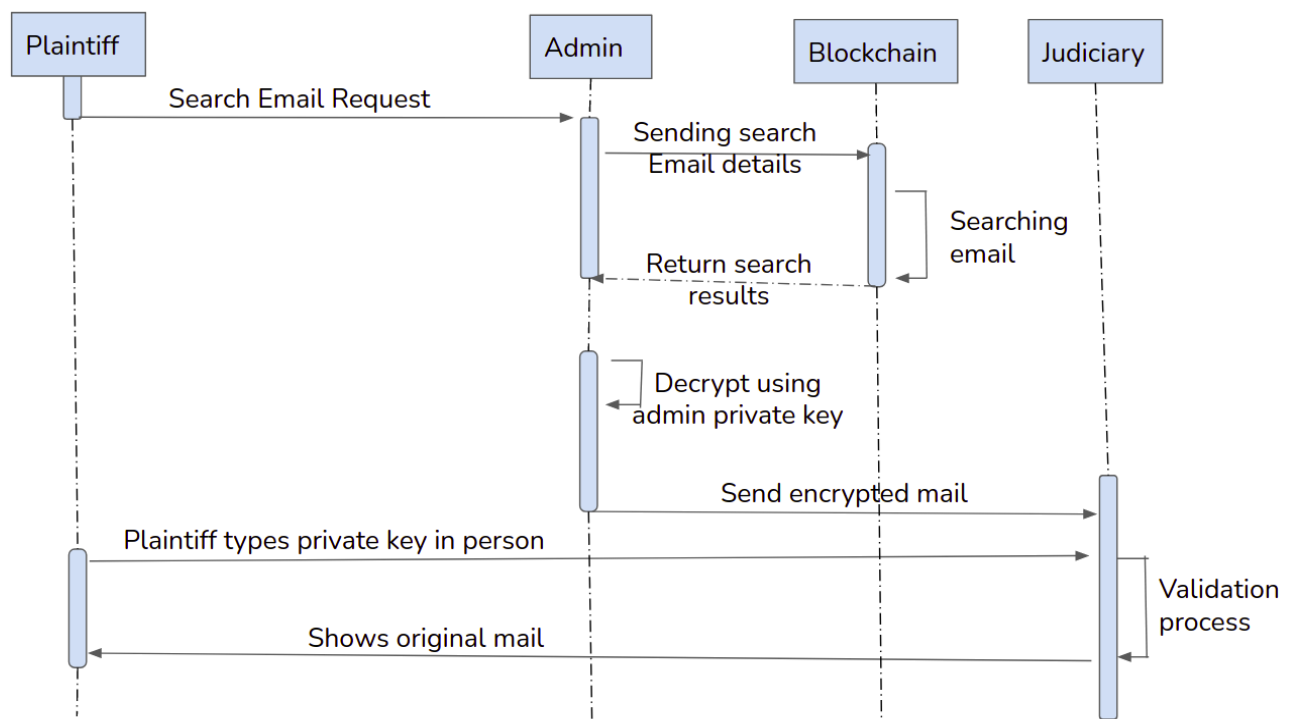


Figure 3.2: Sequence Diagram

Chapter 4

Implementation Details

4.1 Tech Stack

We present the key technologies used to build our solution architecture in this section, which relate to the architecture's components: Blockchain, Smart Contracts, File System, Web Server, Web Client, Hashing and Encryption, and Interaction with blockchain.

Ganache: Ganache is a private development blockchain that can be used to simulate the actions of a public blockchain. It will allow us to deploy smart contracts, build apps, and conduct tests in a secure and predictable environment. It will provide us with 10 external accounts with addresses on our local Ethereum network, each with 100 fake ether loaded down. It is available as a desktop application and a command-line utility for Windows, Mac, and Linux.

Remix IDE: It is a popular open-source platform that allows you to write Ethereum contracts using the Solidity programming language directly in your browser. This IDE includes modules for smart contract debugging, testing, and deployment, along with some other functionalities.

Solidity: It is a high-level object-oriented language for implementing smart contracts on top of various blockchain systems, most commonly Ethereum. The solidity programs that have been compiled are designed to run on the Ethereum Virtual Machine, or EVM. It is influenced by Python, C++, and JavaScript, and was created with the Ethereum Virtual Machine in mind (EVM).

IPFS: The InterPlanetary File System (IPFS) is a distributed file system protocol and peer-to-peer network for storing and exchanging files. In a global namespace linking all computing devices, IPFS uses content-addressing to uniquely define each file.

There is no hard limit on ethereum block size but usually, the transaction rate is limited to 15 transactions per second and around 70 transactions are added into each block. So the average block size on the ethereum blockchain network is anywhere between 20kb - 30kb in size. The average size of an email file is about 75 KB and it can go up to a maximum of 25MB. The gas price on Ethereum makes storing files on the blockchain costly, and the gas limit prevents massive files from being stored. So storing the entire email on the blockchain is not feasible. Using a distributed file system as a workaround is an option for this problem. We use the InterPlanetary File System (IPFS) to manage attached files in our proposal, restricting the blockchain to storing the unique hash of attached files to email transactions. Users would be able to attach as many and large files as they want while still ensuring data confidentiality. IPFS uses hashes to address all attached file data, storing them in the blockchain as a permanent and immutable link.

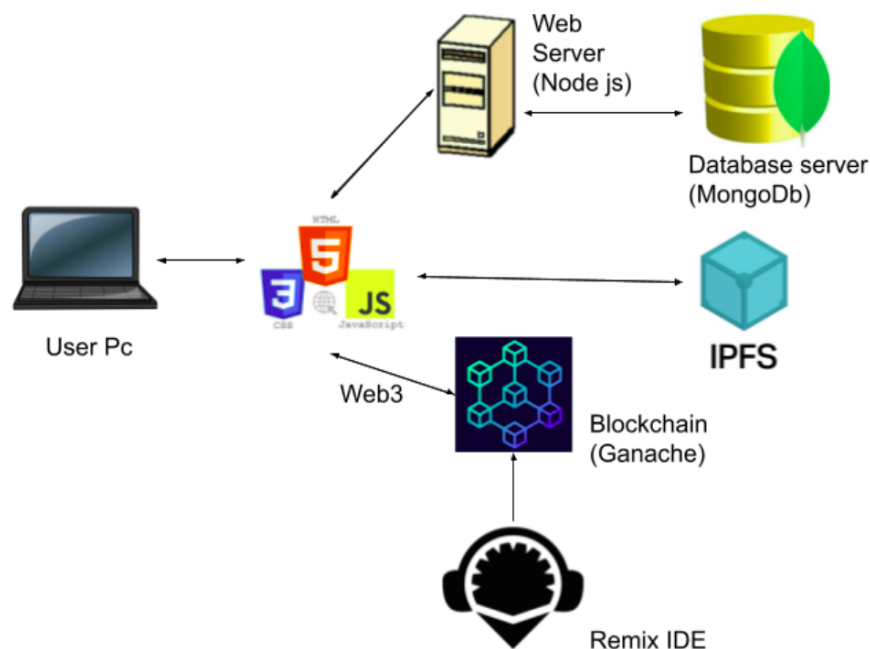


Figure 4.1 represents the architecture of our proposed system.

The basic framework of the sites is provided by HTML, and Cascading Style Sheets (CSS) is used to manage the appearance, styling, and composition of the HTML elements used. JavaScript is used to control the behavior of different elements. The server-side event-driven and JavaScript-based i/o frameworks are built with NodeJs and the express platform. As discussed in section 3.2 any database management system can be used to satisfy the requirements of our application. But, we have decided to use the MongoDB-based NoSQL database solely on the facilities that it provides for its integration with JavaScript and to perform frequent modifications to the model.

As mentioned in section 3.5 there are 3 actors in our application and each one has a different user interface having functions specific to his/her actions. We have used HTML, CSS, and Javascript to create these user interfaces. The web server built using Node.js receives all the API requests from web clients. Here this web server interacts with the MongoDB database which is used to store user credentials and requests to find emails. The actual data is encrypted using the javascript module eth-crypto which provides us with the necessary cryptographic techniques as mentioned in section 3.4. The hashing technique used in this module is Keccak-256. The encrypted email content is being stored on IPFS and the unique hash that it returns is stored on the blockchain(Ganache). We have used the web3 library to interact with the blockchain. This library is used to make function calls in smart contracts that run on the blockchain.

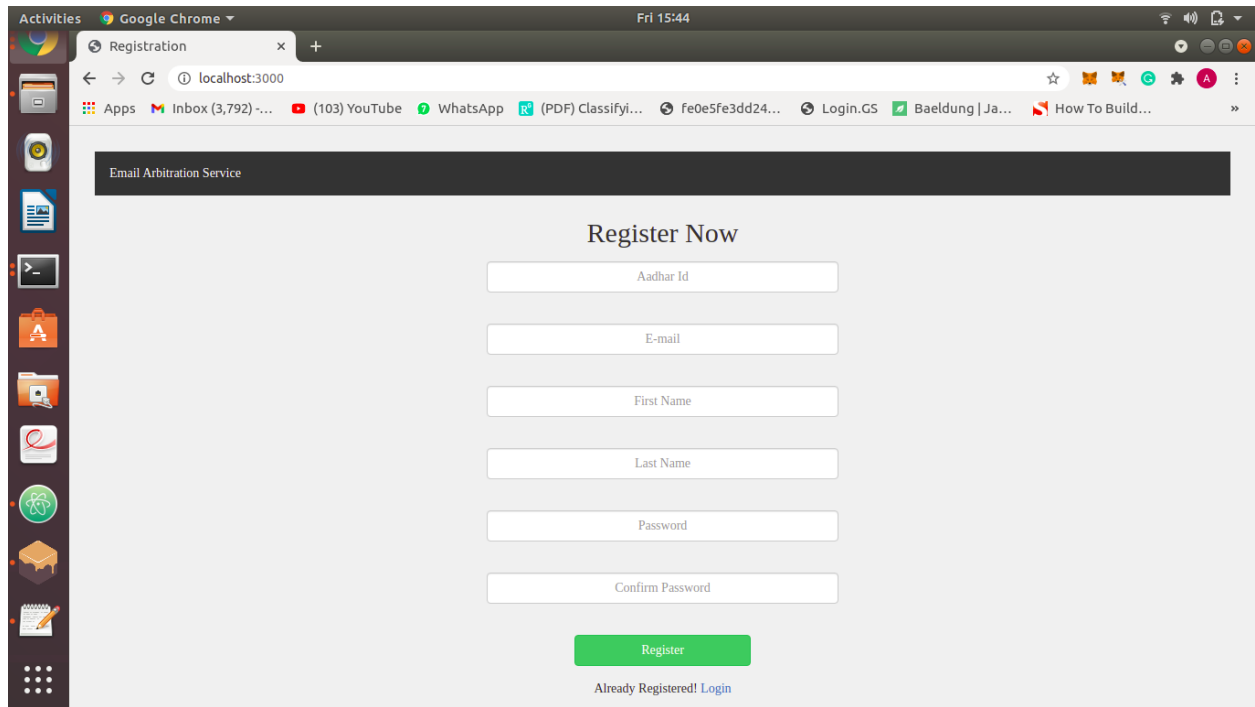
Below are a few functions that are written in the smart contract.

`getEmailTransactionCount()`: This method is used to retrieve the number of email transactions that are made up until the point.

`getEmailTransaction()`: Given required details, this method is used to retrieve all the details of email transactions.

`generateTransaction()`: This method is used to generate the transaction after adding emails to IPFS, for storing corresponding details to the blockchain.

4.2 User Interface Snapshots



The screenshot shows a web browser window with the title "Registration" and the URL "localhost:3000". The page header is "Email Arbitration Service". The main heading is "Register Now". Below the heading are six input fields: "Aadhar Id", "E-mail", "First Name", "Last Name", "Password", and "Confirm Password". A green "Register" button is at the bottom. Below the button is the text "Already Registered! [Login](#)".

Activities Google Chrome Fri 15:44

Registration

localhost:3000

Apps Inbox (3,792) ... (103) YouTube WhatsApp (PDF) Classifyf... fe0e5fe3dd24... Login.GS Baeldung | Ja... How To Build...

Email Arbitration Service

Register Now

Aadhar Id

E-mail

First Name

Last Name

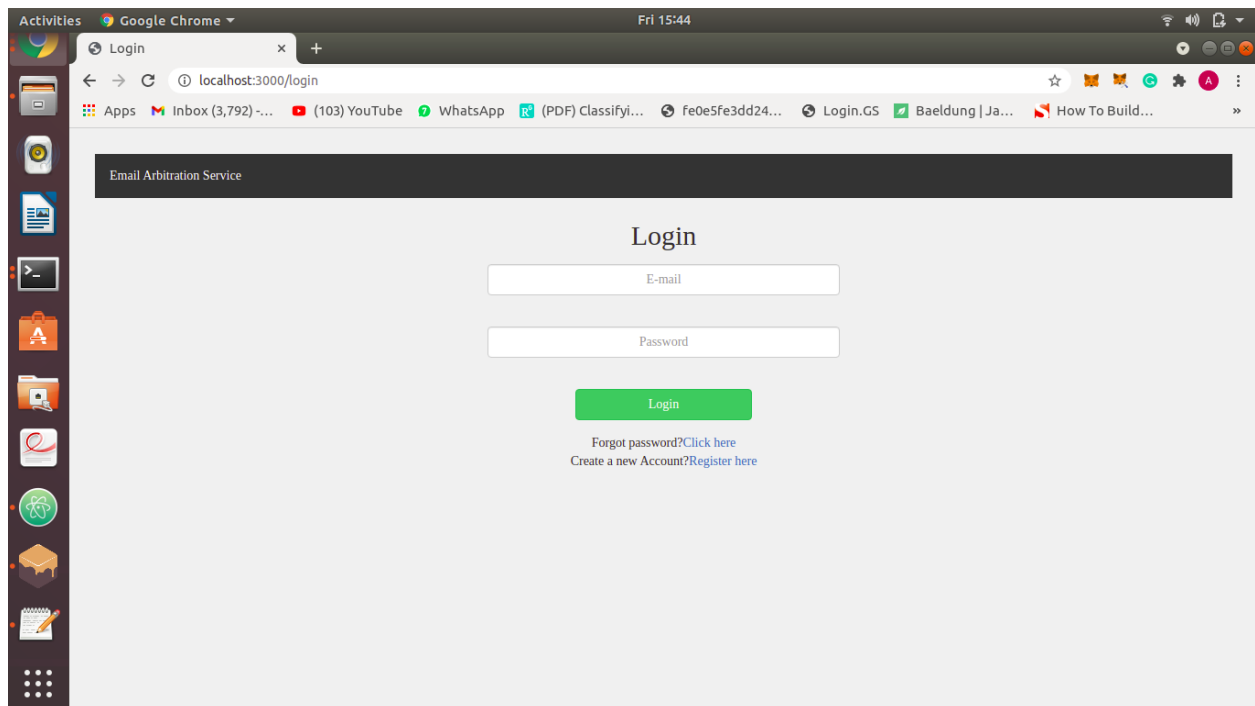
Password

Confirm Password

Register

Already Registered! [Login](#)

Figure 4.2: Registration Page



The screenshot shows a web browser window with the title "Login" and the URL "localhost:3000/login". The page header is "Email Arbitration Service". The main heading is "Login". Below the heading are two input fields: "E-mail" and "Password". A green "Login" button is at the bottom. Below the button are two links: "Forgot password? [Click here](#)" and "Create a new Account? [Register here](#)".

Activities Google Chrome Fri 15:44

Login

localhost:3000/login

Apps Inbox (3,792) ... (103) YouTube WhatsApp (PDF) Classifyf... fe0e5fe3dd24... Login.GS Baeldung | Ja... How To Build...

Email Arbitration Service

Login

E-mail

Password

Login

Forgot password? [Click here](#)

Create a new Account? [Register here](#)

Figure 4.3: Login Page

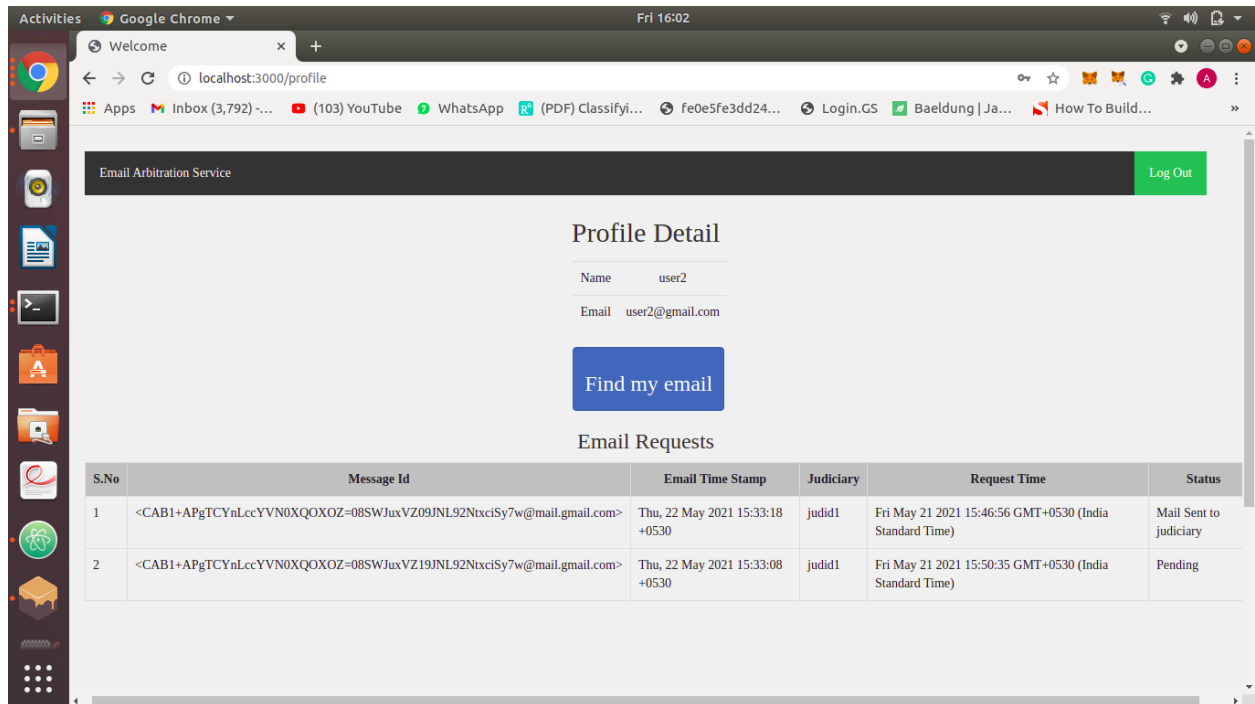


Figure 4.4: General User Interface

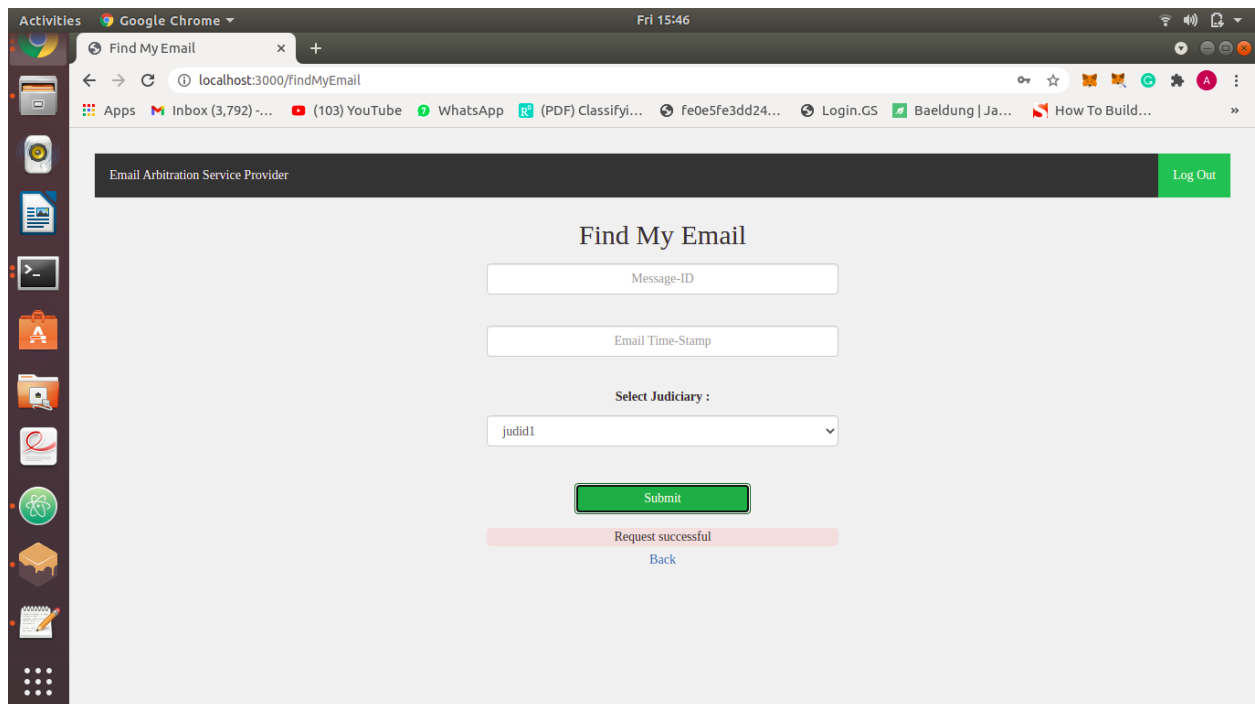


Figure 4.5: Find My Email Page

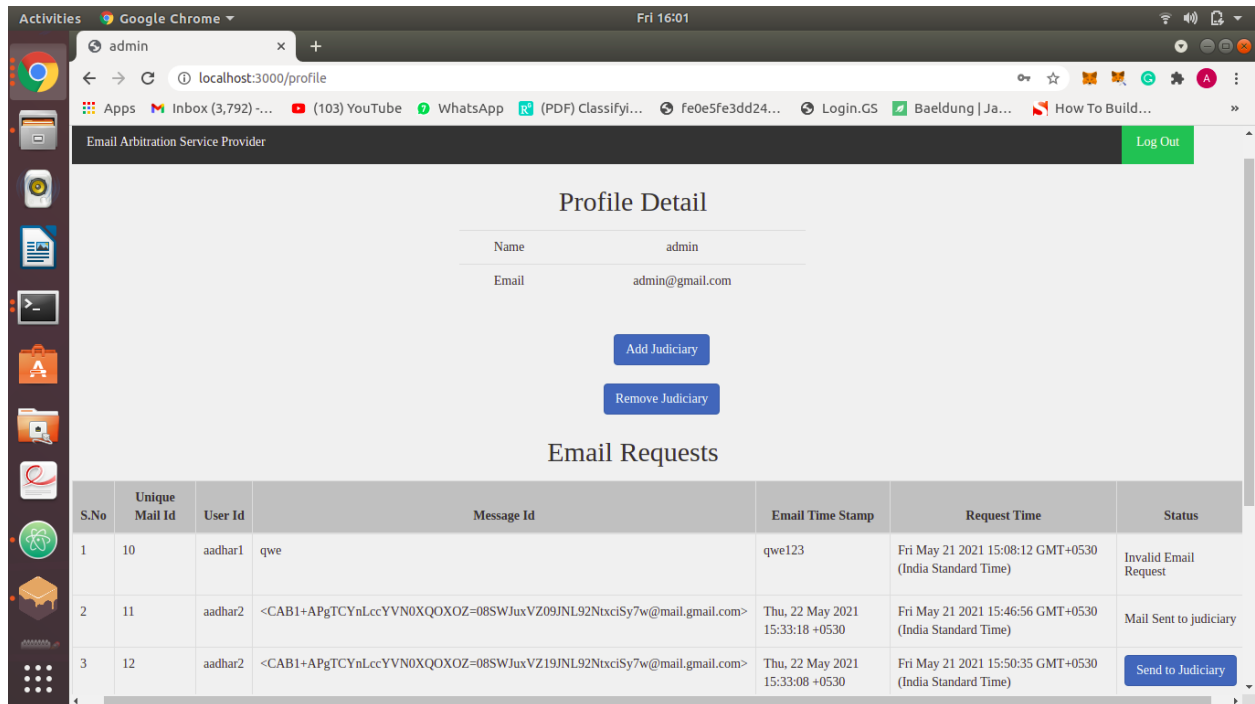


Figure 4.6: Admin User Interface

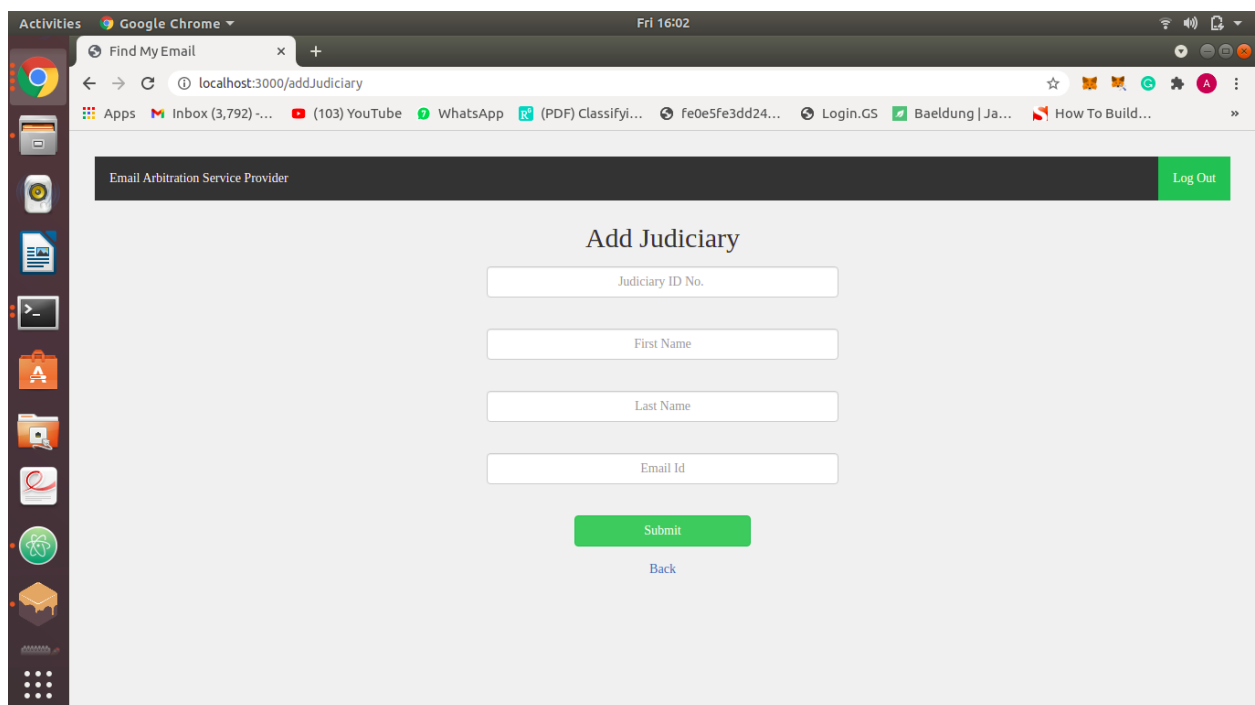


Figure 4.7: Add Judiciary Page

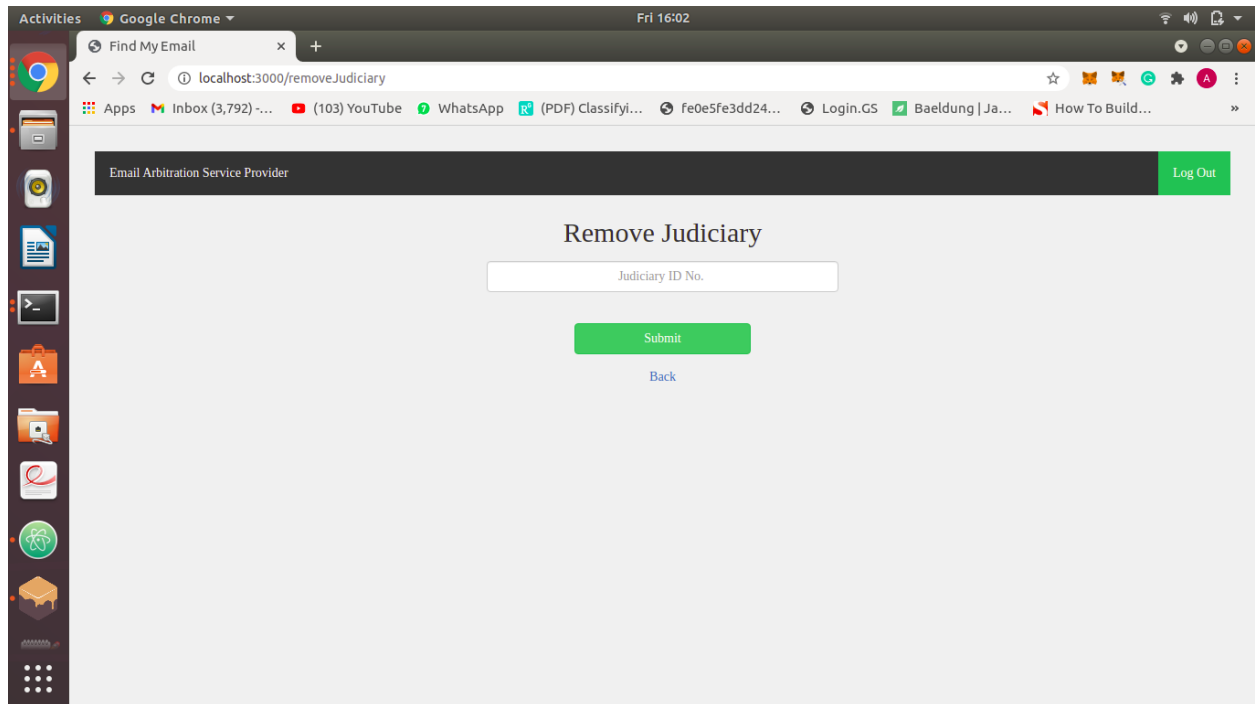


Figure 4.8: Remove Judiciary Page

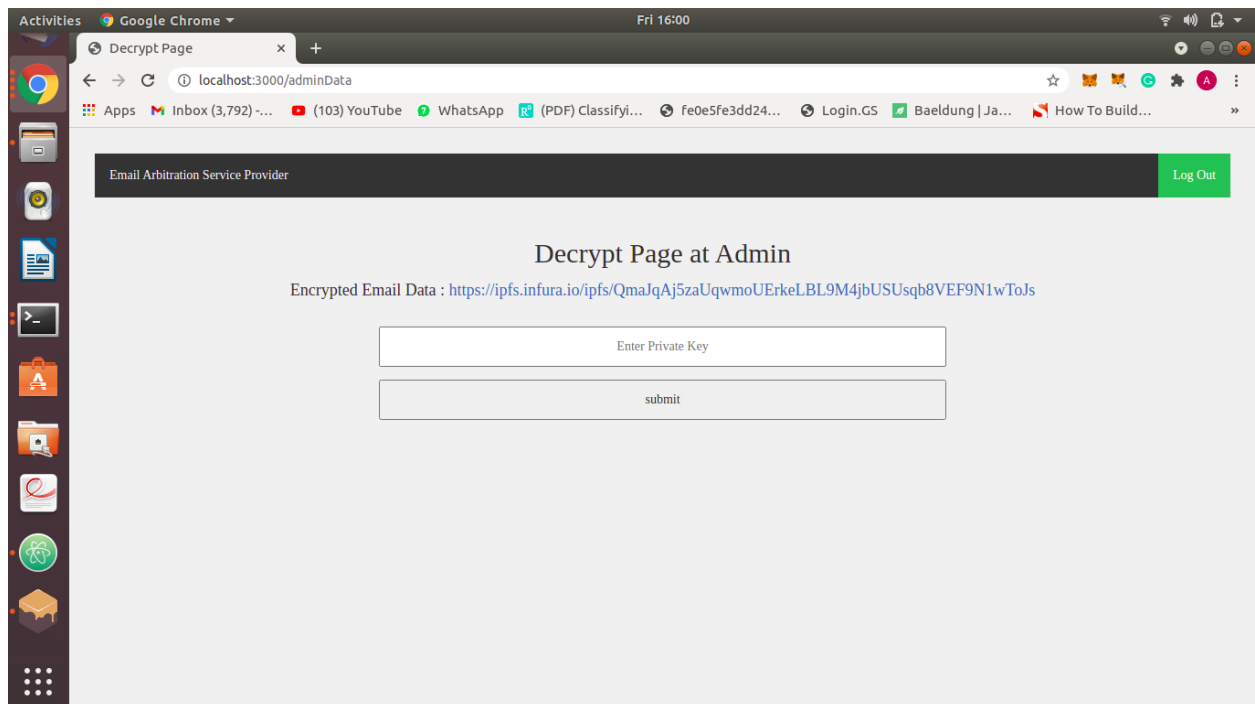


Figure 4.9: Decrypt Page at Admin

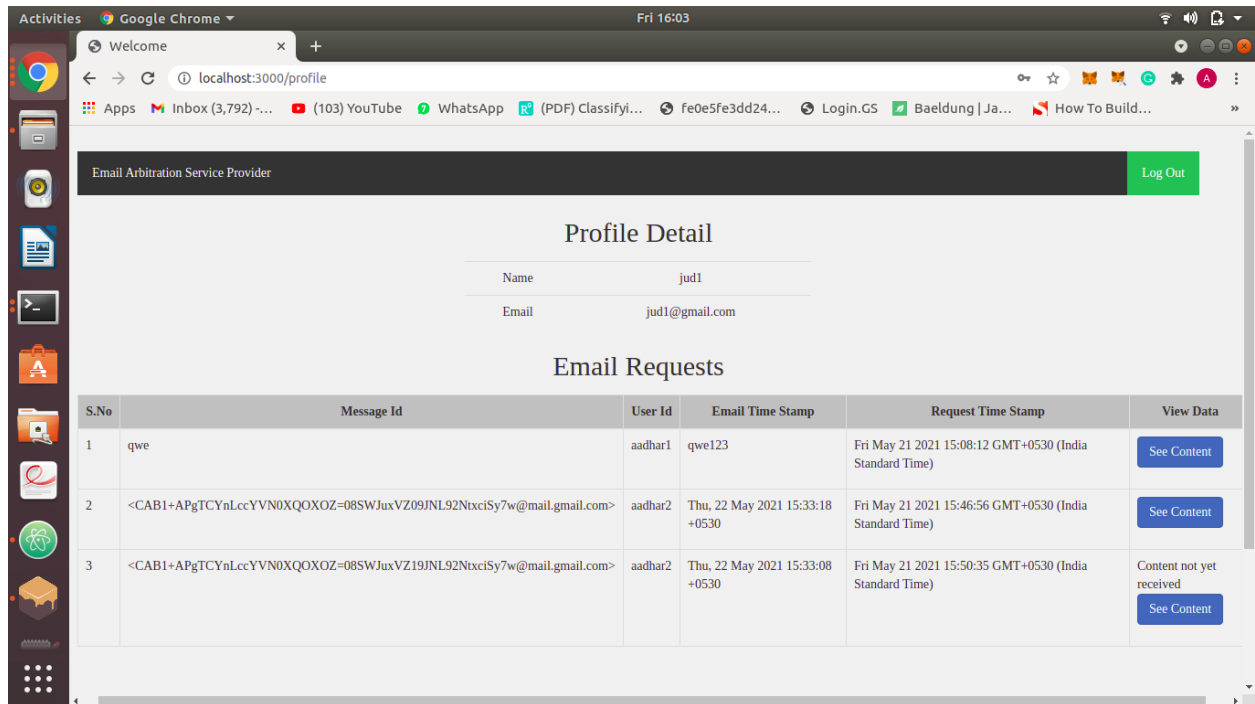


Figure 4.10: Judiciary User Interface

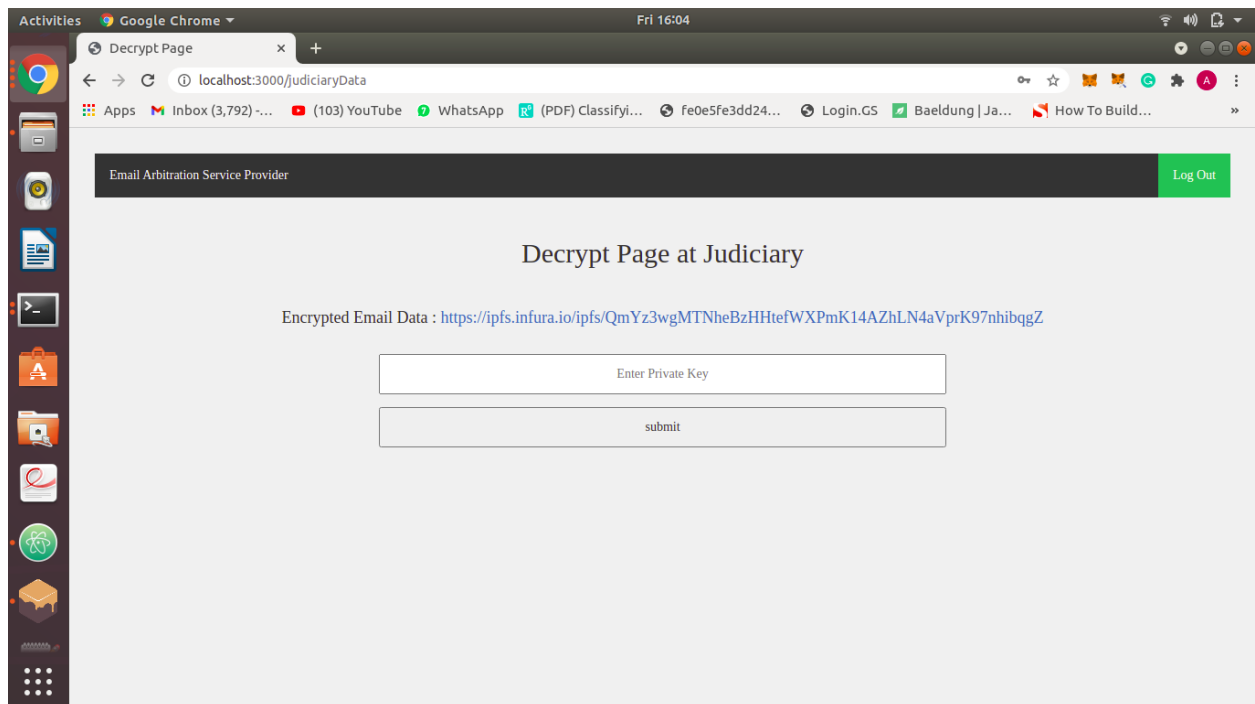


Figure 4.11: Decrypt Page at Judiciary

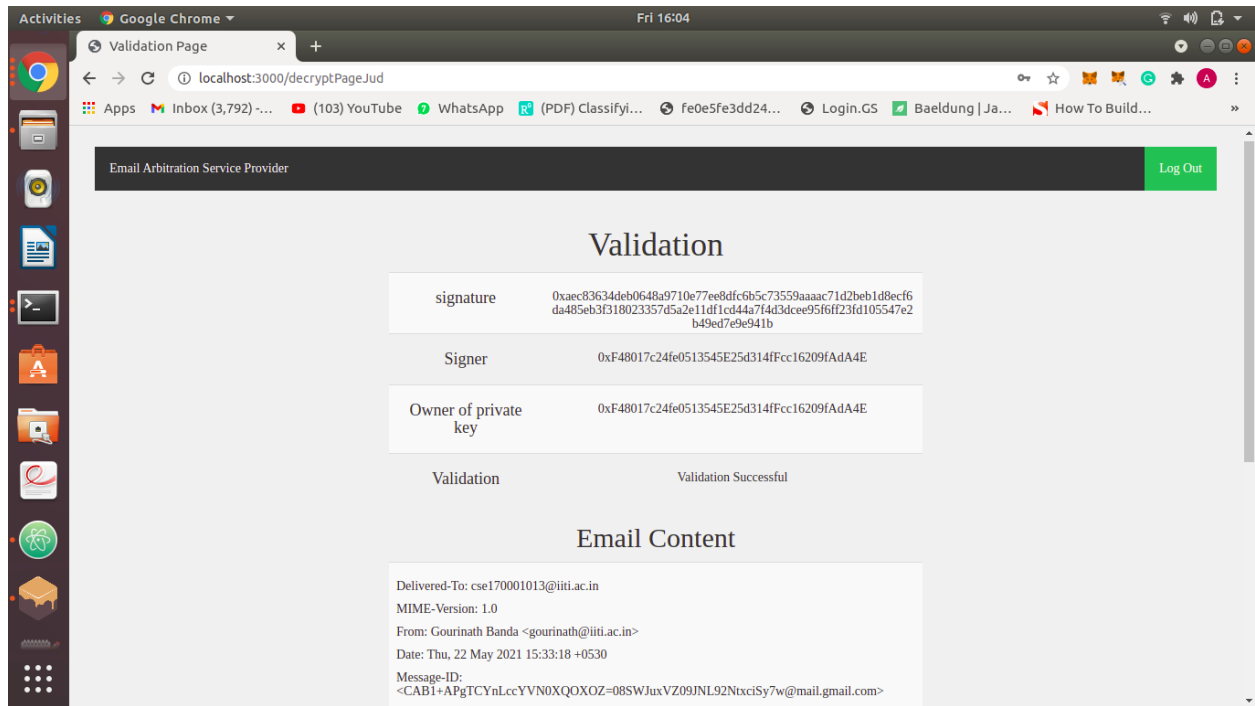


Figure 4.12: Validation Page

Chapter 5

Results

5.1 Conclusion

We have examined the current email architecture and its drawbacks. Our work proposes an email validation and arbitration framework platform based on blockchain to store emails in a decentralized system where no one, in particular, will have the power to read, alter or destroy their messages thus helping in conflict resolution.

The combination of smart contracts with blockchain can provide a distributed ledger to manage email transactions. To overcome the problem of storing large email files on the blockchain, we utilized the facility provided by IPFS. Thus, in this work, we proposed an architecture that minimizes the drawbacks of current alternatives by leveraging the functionalities provided by the above-mentioned technologies.

We deployed the smart contract on a personal ethereum blockchain that addresses the main requirements of this problem like decentralization, encryption, immutability, conflict resolution. And finally, we implemented a client interface that makes the service accessible to all three entities (General user, Judiciary, and Administrator).

5.2 Future Work

In the present proposed architecture, the attachments of emails were not distinguished from the plain text content of an email. So in future work, the foremost step includes working on storing and retrieving attachments of emails. As attachments as large as 25 MB can be sent through email, this brings the trouble of encrypting/decrypting large email files which may drastically slow down the processes. We know that asymmetric cryptography is a slow process compared to symmetric encryption and also requires more computational resources. So using this asymmetric encryption is not recommended in such cases.

To overcome this drawback, we can generate a random key of fixed size and symmetrically encrypt the large email file using this randomly generated key. In the next steps, we can apply all the layers of asymmetric cryptography as discussed in the proposed architecture on the randomly generated key and add it to the email file. As the size of this random key would be very less as compared to the email file, asymmetric encryption on this key takes much less computational time and resources. Implementing this would be a potential future work.

As discussed in section 3.3, the acknowledgment mechanism after receiving the email at the receiver end is needed to be incorporated in the solution architecture.

Bibliography

- [1] Gurpal Singh Chhabra, Dilpreet Singh Bajwa, Review of Email System, Security Protocols and Email Forensics.
- [2] Sabah Al-Fedaghi, Hadeel Alnasser, Modeling Network Security: Case Study of Email System
- [3] Jose Chamadoira Gonza lezl, Vincente Garcia-D iazl, Edward Rolando Nu n ez-Valdezl, Alberto Go mez Go mez2, Rube n Gonza lez Crespo3, Replacing email protocols with blockchain-based smart contracts.
- [4] Wikipedia Infrastructure of current email architecture.
- [5] M. Tariq Banday, Effectiveness and Limitations of E-mail security protocol.
- [6] Satoshi Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System.
- [7] Fran Casino, Thomas K Dasaklis, Constantinos Patsakis, A systematic literature review of blockchain-based applications: Current status, classification and open issues.
- [8] WEI CAI, ZEHUA WANG, JASON B. ERNST, ZHEN HONG, CHEN FENG, VICTOR C. M. LEUNG, Decentralized Applications: The Blockchain-Empowered Software System
- [9] Wood, G., Ethereum: A secure decentralised generalised transaction ledger. Ethereum Proj. Yellow Pap. 151, 1–32 (2014)
- [10] Truffle Suite, Ganache Overview
- [11] Sheping Zhai^{1,2}, Yuanyuan Yang^{1*}, Jing Li¹, Cheng Qiu¹ and Jiangming Zhao¹, Research on the Application of Cryptography on the Blockchain
- [12] Thomas kitsantas¹ , Athanasios Vazakidis² and Evangelos Chytis³, A Review of Blockchain Technology and Its Applications in the Business Environment
- [13] Ethereum Blockchain Explorer Site.
- [14] Yining Hu, Madhusanka Liyanage, Ahsan Manzoor, Kanchana Thilakarathna, Guillaume Jourjon, Blockchain-based Smart Contracts - Applications and Challenges
- [15] Fran Casino, Eugenia Politou, Efthimios Alepis and Constantinos, Patsakis, Immutability and Decentralized storage.