Analysis and Result of Smart Complaint Redressal System Using Ethereum Blockchain

Akhilesh R1, Rachith R Naik1, Suhani1, Vineeth Kumar1, Sneha N S\*

*Department of Computer Science and Engineering,*

*Shri Madhwa Vadiraja Institute of Technology and Management Bantakal.*

***Abstract:* In today’s world, more focus is given on the availability of the websites and also the various applications present in the android market. We manage our daily work on time, precisely, very fast and with our satisfaction. So we are using various technologies in our life to fulfill our daily work. In India, there is no direct and efficient way of communication between the government and the public, for solving a problem i.e for getting a problem solved at our place, we may have to wait for three months, but it can probably be solved sooner. Nowadays, the scenario has changed. Some applications are available, which allow users to register their complaint. But there are some problems related to its transparency. This paper proposes an Ethereum blockchain application that will help people to register their complaints and get updates about the complaint. Adoption of blockchain technology makes the application more secure, transparent and immutable.**

*Keywords: Ethereum blockchain, interplanetary file system.*

1. INTRODUCTION

An application named Sakala is a scheme that assures timely delivery of government services to citizens. In all, Sakala offers 375 services under various departments which are sought after by the citizens. In Short, it is an Act to empower the citizens in getting what they want in the time defined.

Whenever the request for the service is made, the citizen receives an acknowledgement slip with a unique 15-digit number called the Guarantee of Services to Citizen (GSC) number.

This system also has a mobile interface. Citizens can check the status of their application by sending a SMS from a mobile phone by typing their 15-digit GSC number. The system will send a reply back to them with the current status of the application, the system also sends message updates every time the status of the complaint changes. Citizens are encouraged to give their mobile numbers at the time of registrations so that they can be automatically intimated about the interim status of their applications. Hence their visits to offices are reduced.

The drawback of this application is that, when any user wants to register a complaint, it requires proper documents of the complainer to launch that complaint. If the complaint is of public interest, it is not so feasible to add the complainer's personal information. No transparency is maintained when any complaint is registered which is having public interest and no module to prioritizing the complaint based on citizens’ response. No provision to accept user ratings/feedback after the problem is solved or if not solved.

In the “Udupi Help” application, the public can report complaints related to any issues in their wards. Once the complaint is submitted, it will reach the natural calamities authority of the government bodies. The complaints are later forwarded to respective officials to take actions.

Apart from Udupi city municipal council officials it is forwarded to the forest department, MESCOM, health department, police and other department officials to address the grievances registered in the application. This application can be used by the people belonging to the Municipality region only. This is the drawback of this application. Udupi city includes the gram panchayat region as well. People living in the gram panchayat region cannot register their complaints through this application. There should be some provision for them to address their problems.

This paper proposes a Blockchain-based technology application that will help citizens under the jurisdiction of a municipal corporation and gram panchayat to register their complaints about the various problems they are facing in their ward, with more transparency and security. It has an automated system that manages all the tasks that one has to face during his/her visit for the complaint. One can register their valid complaint, wait for the validation and be updated about the complaint status [1]. The application manages all the background processes like prioritizing complaint type, hosting complaints, notifying user’s status.

OUR CONTRIBUTION

The proposed system is as follows

1. A secured and transparent complaint redressal system is proposed in this paper using the ethereum blockchain. The aim is to build a more connected digital world through the decentralized application. The complainer will be well informed about the complaint registered, not only about the status of the complaint but also the exact information about who is currently handling the problem and when exactly it will get solved. Hence, the best platform to develop a decentralized web application is Ethereum Blockchain. It provides tools to build a decentralized web application.

2. The process is initiated when a user registers a complaint. Complaint is registered by adding required details and hence it is stored in ethereum blockchain. The complaint number will be assigned to the user to track the status.

3. Only when the data is stored in the blockchain, the officials can view the complaint registered and take necessary actions regarding the complaint. Every action performed on the registered complaint can be tracked by the user.

II.  SYSTEM BACKGROUND

*A. Blockchain Technology*

Blockchain is the underlying technology used in bitcoin in which mutual trust is established between the two transacting parties without the need for a third party. Blockchain is a decentralized and distributed database that stores the transaction details where each record is stored in terms of blocks that are timestamped, and every block contains the data, hash of the previous block and block number. Verification of data, security in case of data transmission and trust between the entities are achieved through cryptography and consensus algorithms. These blocks are tamper-proof as they are time-stamped and verified before adding to blockchain. Computations of the corresponding data are also distributed throughout the blockchain network and each node in the network contains a copy of the transaction database [2]. In Spite of storing transaction details, blockchain networks can store self-executing scripts that describe the behavior of the transaction. This helps to develop various blockchain applications and these applications are not controlled by any authority and need not depend on any intermediary entity for verification and authentication, the network takes care of all the requirements [3]. The unconfirmed transactions are added to the block by a process known as mining. Every node which adds blocks to blockchain uses proof of work protocol to ensure the integrity of the blockchain. In the case of proof of work protocol, the mining node has to find a random number or nonce using which the hash of that block will be computed. The hash value obtained should satisfy the requirement of the network. Nodes that first calculate the proof of work will add the block containing unconfirmed transactions to the longest chain [4].

*B. Overview of Ethereum blockchain*

A Smart Contract is a computer program to implement a sequence of events that follows particular rules. It runs in a blockchain and the transaction is permanently recorded and stored in a transparent environment. The transaction record will remain immutable. Ethereum blockchain smart contract program is written using a language called ‘solidity’. Although blockchain was initially used for transferring money and for conducting multiple transactions but now it can be used to run special code snippets called smart contracts which includes all the business logic regarding how the transaction should behave and these contracts once deployed are immutable. The changes can be reverted back only by deploying a new instance of smart contracts [5].

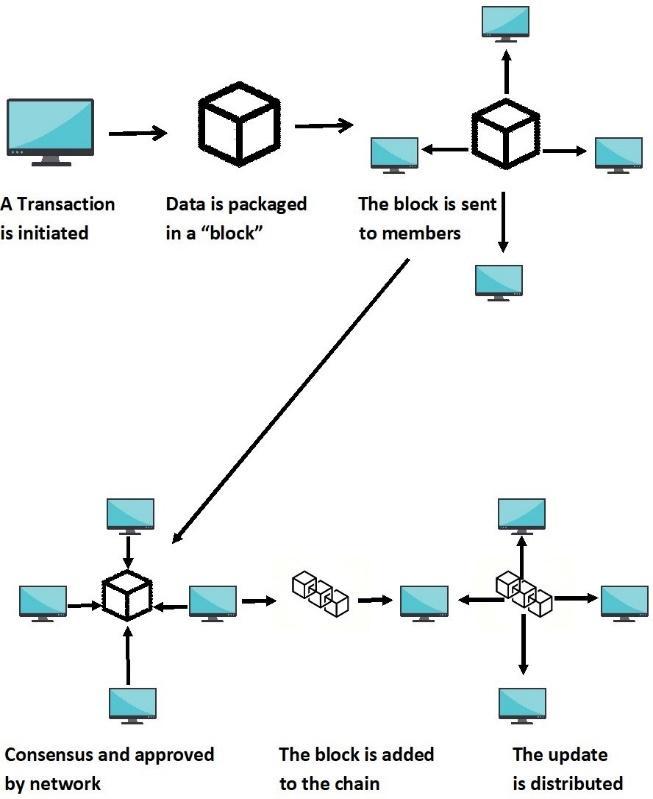


 Figure 1: Working of Ethereum blockchain

Whenever the smart contract is fed into a solidity compiler it gives two files that are, bytecode and ABI. Since smart contracts deployed in the Ethereum network are in the form of byte code, the JavaScript code snippets written by the user cannot directly interact with the deployed smart contracts. ABI acts as an interface or translator for user-written JavaScript code to interact with the deployed smart contracts as shown in fig 1.

The bytecode is deployed onto the test network with the help of web3js. The contract properties are then accessed and tested using various test frameworks before deploying into the actual network [5].

*C. File storage in interplanetary file system*

The interplanetary file system is a distributed file system that looks out for decentralizing the web so as to make it faster and more efficient. The system works like this, it connects all the computing systems that are connected as a node with the same system of files. This system doesn't have ownership to the management side since, whenever a file is being uploaded, its hash value is shared with the peer and these peers can access files using a hash address. Permission less blockchain like Ethereum are open to the public and are validated every time for every transaction. This system is used for storage platforms for data sharing. Its advantage is that it has a good performance and higher availability.

IPFS is used in this system as it is a Distributed application. This DApp will upload the user complaint data to the IPFS and a hash on the reference of the data stored in IPFS is returned. The returned hash value is saved inside the Ethereum network node. Once the hash value is pushed into an Ethereum blockchain, the user will get a transaction receipt [6].

IPFS is content addressed which means data can be accessed quickly by the source which is near to the node and holds the copy of the content address. IPFS uses a Distributed hash table (DHT) to store the data. To search any content, a request is made to these peers. Once the content is found again the request is made to find the location of the peer to obtain the required content.

The DHT is essentially a database consisting of keys to values and keys are cryptographic hashes, where the table is split across the peers in the distributed network. It uses node identifiers and keys, both need to have the same length with a distance metric to easily store and retrieve information. When searching for a value, a node attempts to find nodes that are more likely to the key and requests the value from them. It uses the buckets to keep track of nodes within the network. The buckets are organized such that any node of the network has accurate information of its near environment. Node’s knowledge about the other node in the network decreases as the distance increases. A node contacts other nodes that are closer to the key, in order to find the value associated with a key. It replies back either by sending the value or by referring to the node that is much closer to the key. This process continues until the required key is found. A node determines a set of nodes that are closest to the key in order to write a value. This makes the process of writing values to and retrieve them from the network more efficient. Keys need to be updated regularly, they are valid for a certain amount of time [7].

*D. Metamask*

Metamask is a browser based tool for ethereum blockchain which converts normal browsers into blockchain browsers. It is used as a browser extension, supports many browsers like Google Chrome, opera, firefox etc. It is also used to manage personal ethereum accounts. Ethereum Dapps can be run on the browser instead of running the entire Ethereum node. It provides easy access to the ethereum network. Metamask provides a graphical user interface through which users can manage their accounts. Users can manage multiple accounts using metamask and each account is composed of mainly 3 components: unique account address, private and a public key pair. These user accounts along with private public key pairs are generated by the metamask using a mnemonic which is generated while installing metamask. All the accounts are generated using a single mnemonic. Users can create accounts in different test networks including the main network with the help of metamask and the accounts in different networks are decoupled from each other, which means users cannot use the assets of one network in another network. Whenever the user performs the transaction metamask provides a secure interface through which the user can review his transaction [8].

*E. Nodejs*

Node js is an open source server-side cross platform. To develop a smart contract in a private or local blockchain requires Node Package Manager (NPM). NPM includes Node.js. Node.js is built on Chrome's JavaScript runtime to build fast and scalable network applications easily. It makes use of an event-driven, non-blocking input/output model that makes the application lightweight and very efficient, which can run across distributed devices [8].

*F. Web3.JS*

Web3.jsis a Javascript library used to communicate with the smart contract which is deployed in the blockchain. It checks whether the account is present in the network and also helps to validate the accounts in the metamask. It is used to access the Transaction ID and transaction hash.We can use web 3 instances to access various deployed contracts in various blockchain networks. The newly created web3 instance has to be configured to interact with the deployed contract. The configuration of the web3 is done using a provider which acts as a communication layer between web3 and some specific ethereum network. Provider Can be used to communicate with the deployed smart contracts Provider tells web3 to instance to which network it has to connect. Every provider which we use to connect to different ethereum networks has identical set of methods and these methods help web3 to send requests to specific networks and to receive responses from that network. Using web 3 we can access the user accounts and can send transactions into the network web3 acts as a portal through which our javascript can interact with the deployed contract.

III. RELATED WORK

In 2015 Trupti Bomble et al proposed a system, where the complainer will have mobile application. She/he can register a complaint related to a specific zone where he/she finds a problem. Users can embed an image with the complaint. If the complaint is not followed up in the specified period of time, then the alarm will buzz on the mobile in order to notify that the complaint is registered long back and no action has been taken against it. The proposed system also generates emails to notify the user regarding the status of the complaint[9].

In 2016 Devika Radhakrishnan et al proposed a system, where users can lodge the complaint anytime and anywhere using this application. There are many sectors each have their individual login section. When the user launches a complaint, it is forwarded to the grievance officer. The process of forwarding the complaints among officer and employee will be kept hidden from the user[10].

In 2018 Fibina F.Maheen et al proposed a system, where citizens should register themselves on a website with their details. While lodging a complaint he/she has to give certain information regarding the complaint along with the image/video of the problem as evidence. The image will be geotagged, it contains the information about the location of the complaint site in the form of longitude and latitude that helps in finding the exact position of the site. In addition to this the user has to provide a keyword related to the nature of the problem, which will be helpful in determining the corresponding department. The department officials can login and they can see to the problem, take necessary actions and update the status of the problems registered. The system also sends daily notifications about day to day activities. So the citizens can view the status of the complaints by these alerts[11].

In 2019 Sudeep J et al proposed an application, it is built on normal client - server architecture. This application is divided into three sections: Citizens, Web Page and Corporator. Users get registered and logins to application and upload images of grievances in the area. The uploaded image will be accessed by the corporate officials and approve it by confirming whether the image is fake or real. Once it is confirmed it will be reported to the concerned corporator where he/she can view the image, region and complete details. The Corporators can take action by communicating with the respective people[12].

VI. METHODOLOGY

Complaint redressal system allows citizens to register their complaints regarding the various issues in their city and helps to get it solved. Users can fill in the complaint details in the provided interface and upload the picture for the grievance. And also any user can vote the already registered complaint in order to increase the priority of the complaint. Registered complaints will be redirected to the respective government department based on the location entered by the user and government officials should look over the addressed problem. The department officials should update the status on the basis of day to day activities regarding the complaint to the complainer.

When the complaint is reviewed by the officials, if the complaint is genuine then the complaint is accepted otherwise the complaint is rejected.

The adoption of Ethereum in the complaint redressal system helps citizens to build a more connected digital world. It provides a lot of advantages like integrity and transparency thus it is tamper proof. The complainer will be well informed about the complaint registered.

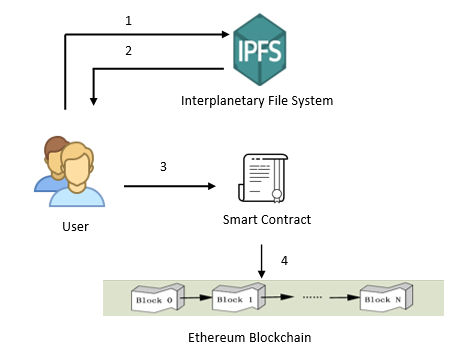


Figure 2: Working of the proposed system

The description of the fig 2 is as follows,

1. User registered complaint details contain images. The image is uploaded to the interplanetary file system (IPFS).

2. IPFS will return the hash on the reference of the stored image.

3. When a user registers the complaint the deployed smart contract will execute.

4. Complaint details and the image hash is stored in ethereum blockchain.

*B. Architecture of a proposed system*

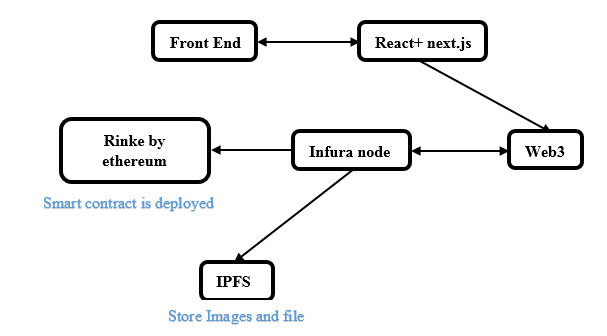


Figure 3: Architecture of a system

In this application, the front end is designed using next js which is an open source framework for developing web applications based on Vue and react js. Front end interacts with the blockchain using the web3 library. Users can add data to the blockchain using the front end. Whenever the user tries to add data to the blockchain a transaction is created by web3 and the corresponding transaction is sent to the rinkeby network. The data transaction is confirmed through the metamask.

Web3 uses infura api which provides access to the infura node hosted on the rinkeby network and using this node web3 will submit transactions to the rinkeby network. This network will process the transaction and the corresponding response is sent to the front end.

While accessing data from the blockchain, the front end will submit a call to rinkeby network using web3 which in turn uses infura api to access contracts residing in the rinkeby network. Response is then displayed in the user interface.

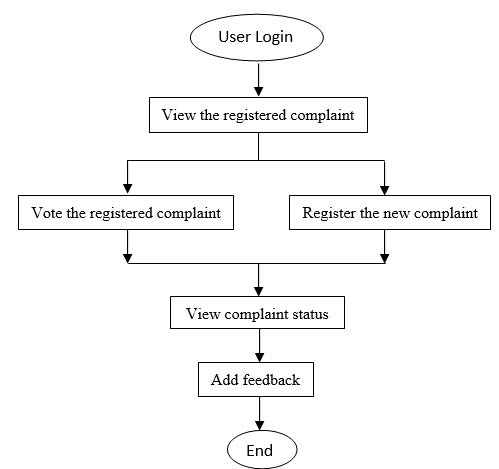
IPFS (Interplanetary File System) is a distributed storage system that helps to develop a decentralized application.

When a user tries to register the complaint through this application image uploaded by the user will be stored in IPFS.

Other details of the complaints like location, complaint description and complaint type are securely stored in a node of Rinke By Ethereum blockchain along with IPFS hash of the image as shown in fig 3. Finally the respective government authorities can view the complaint and take necessary action depending on the complaint.

There are 2 modules in our scheme,namely user module and Government official module.

*1. User module*

Figure 4 : Flow diagram for User activities

All the users of the application can view the complaint registered by other users so that transparency can be maintained. Any other user can vote the registered complaint in order to increase the priority of the complaint and can get it solved early. Users can register the complaint using the complaint registration form. After the complaint is submitted, the user can view the complaint status and can give feedback at the end as shown in fig 4.

*2. Government Officers module*

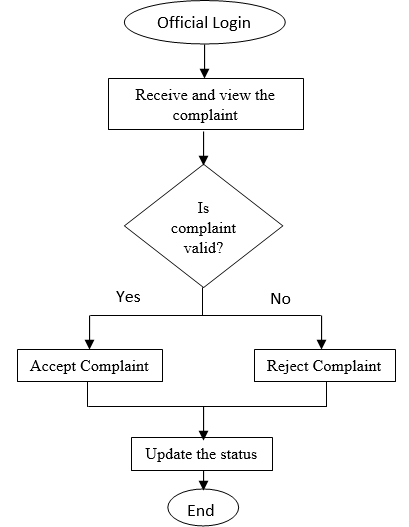


Figure 5: Flow diagram of Official activities

In the Government Officers module, officials should login to the application. The complaint registered by the user is accepted by the officers if the complaint is genuine otherwise the complaint is rejected and the status of the complaint is updated as shown in fig 5. The status updated at the official side is visible at the user end.

*A. Smart contract design*

In the proposed system, the interaction between the user and government officials is achieved through the Ethereum Blockchain technology. The smart contracts are deployed to perform the transactions. Each transaction performed by the users are recorded in the blockchain nodes.

In this section, we describe a scheme of algorithm logic related to smart contracts. In the Ethereum blockchain, smart contracts are created and compiled in solidity and deployed on the RinkeBy Test network.

In this scheme, there are 2 contracts deployed namely ComplaintFactory and Complaint. Each contract contains functions and variables.

**Contract 1:** Complaint Factory

This Contract is used to create an instance of a contract named Complaint and the address returned after deploying each instance is stored in an array. This array can be later used to track the complaint address and details. An array *deployedComplaints* holds all the addresses of the complaint.

It contains functions *createComplaint()* and *getDeployedComplaints().*

|  |
| --- |
| Algorithm 1: *createComplaint(all\_complaintDetails)* |
| Input: *all\_complaintDetails*  Output: *address\_of\_the\_complaintStored* |
| address *complaintAddress* = new Complaint(*all\_complaintDetails)*  *deployedComplaints.push(deployedComplaints)* |

1. *CreateComplaint()*: This function complaint details as input from the front end entered by the user. The main functionality is that it creates instances of contract named Complaint by passing complaint details as the parameter. Address returned creating the complaint is pushed into an array *deployedComplaints.*

|  |
| --- |
| Algorithm 2: *getDeployedComplaint(address[])* |
| Input: *address[]*  Output: *deployedComplaints* |
| return *deployedComplaints* |

2. *getDeployedComplaints()*: Length of an array *deployedComplaints* is returned*.*. All the complaints stored in the network can be retrieved by parsing through this array of addresses.

**Contract 2**: Complaint

This contract contains a structure that stores a complaint log, date and time. Any modification done on the complaint is recorded with the timestamp. It contains many variables each one is assigned with the complaint details entered by the user. Each time when any user creates a complaint the variables get initialized with the values.

The functions defined in the contract are listed as follows:

3. *getSummary()* : When invoked it returns complaint details such as complaint owner, complaint title, description, location, image hash, total votes of the complaint, phone number of complainer.

|  |
| --- |
| Algorithm 4:*Vote(voteMappings,userPhonenumber)* |
| Input: *voteMappings,userPhonenumber*  Output: Increment the total number of Votes |
| if *userPhonenumber* not in *votersArray*  Add *userPhonenumber* to *votersArray*  *totalVotes=totalVotes*+1 |

4. *Vote(voteMappings, userPhonenumber)*: When this function is invoked, it checks whether the sender of transaction has an entry in mapping votes, if this condition fails the transaction will be reverted back. If the condition is satisfied then an entry for the sender is made in mapping and the total number of votes is incremented.

|  |
| --- |
| Algorithm 5: *updateComplaintScope(date, time, compScope)* |
| Input: *date,time,compScope*  Output: Complaint status is updated |
| iflength(*compScope*) == 0  *complaintScope=compScope*  *addLog(date,time,compScope)* |

5. *updateComplaintScope(date, time, compScope):* When this method is called it first checks whether the complaint state has changed or not. If the state is changed, the transaction fails and it is reverted back, if not , the complaint scope variable will be updated with the received status value and the log will be created with time,date and event.

|  |
| --- |
| Algorithm 6: *closeComplaint(date, time)* |
| Input: *date,time*  Output: creating log of complaint |
| ifcomplaint is not *completed*  *completed=true*  *addLog(date, time, “Complaint solved”)* |

6. *closeComplaint()* : When this function is invoked it checks whether complaint has been finalized or not(if the complaint variable is set to true indicates complaint has been solved) If complaint is not finalized the function will finalize the complaint by assigning true value to completed variable and log will be created with date time and event. If the initial check fails and the complaint is already finalized then the transaction fails.

In our proposed system, a user needs to enter the active mobile number in order to register a complaint. The mobile number verified through the OTP(One-Time-Password). If the verification is successful in that case the complaint is received. The user can also view his complaint by entering the registered mobile number and the number is verified.

VII. RESULTS

Our proposed system consists of 2 modules: 1. User module 2. Government Officers module. The following images show the interfaces provided for both the modules.

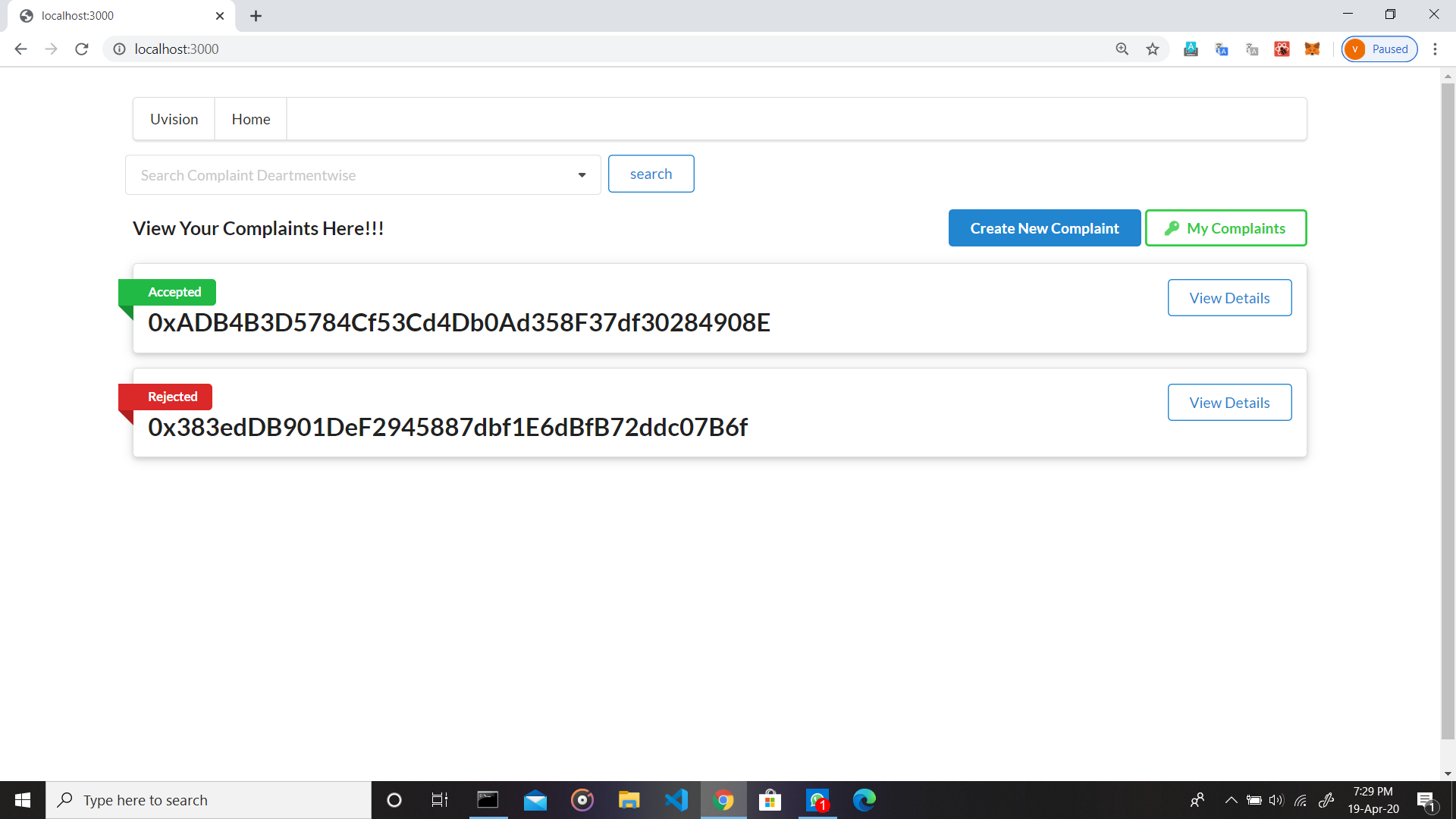


Figure 6: Home Page

When a user enters the application, the home page is available as shown in fig 6. It consists of all the registered complaints. Any user can view the details about the complaints except the complainers identity. The user can search the complaint registered in any area/ward by entering the ward name in the search box. One can register a new complaint by clicking the “Add new Complaint” button. While adding the complaint user needs to enter the mobile number. The complaint will get registered in case of successful verification of the number through OTP(One-Time-Password).

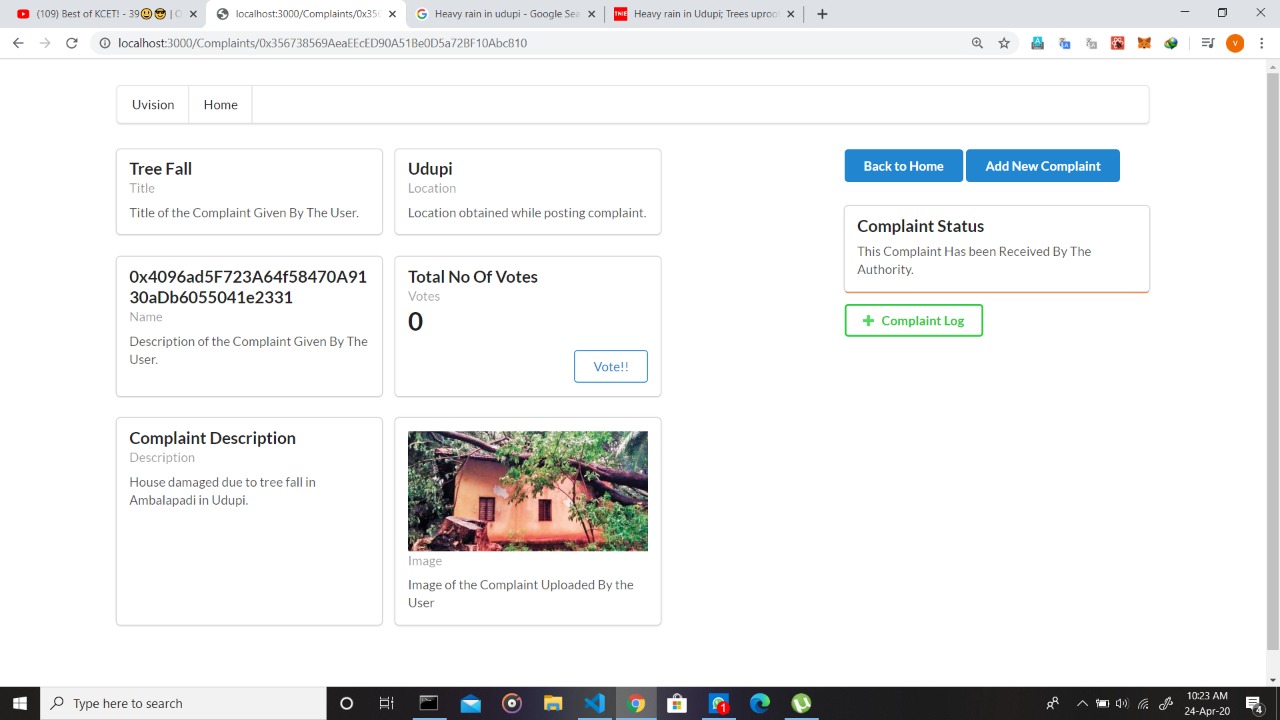


Figure 7: Complaint Details

The user can view complete details of his complaint by entering the mobile number as shown in fig 7.

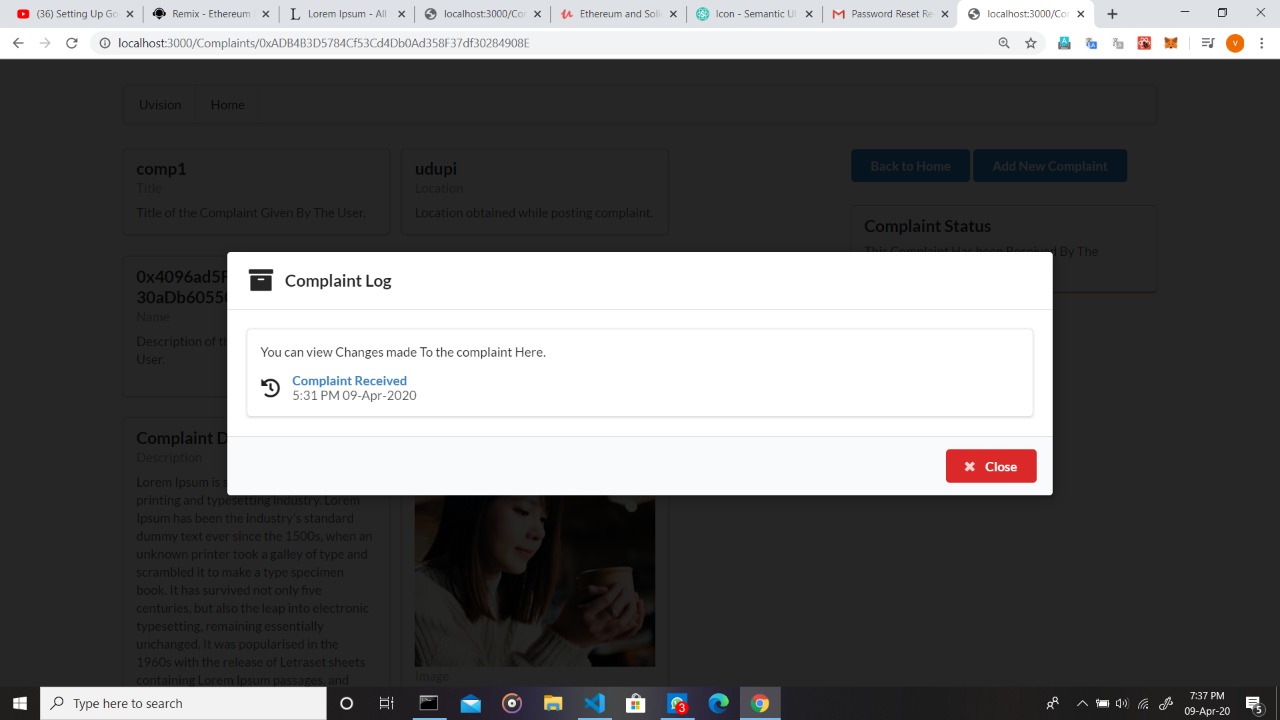


Figure 8: Complaint Log

Complaint Log will show the details of the complaint status along with the complaint updated date and time as shown in fig 8.

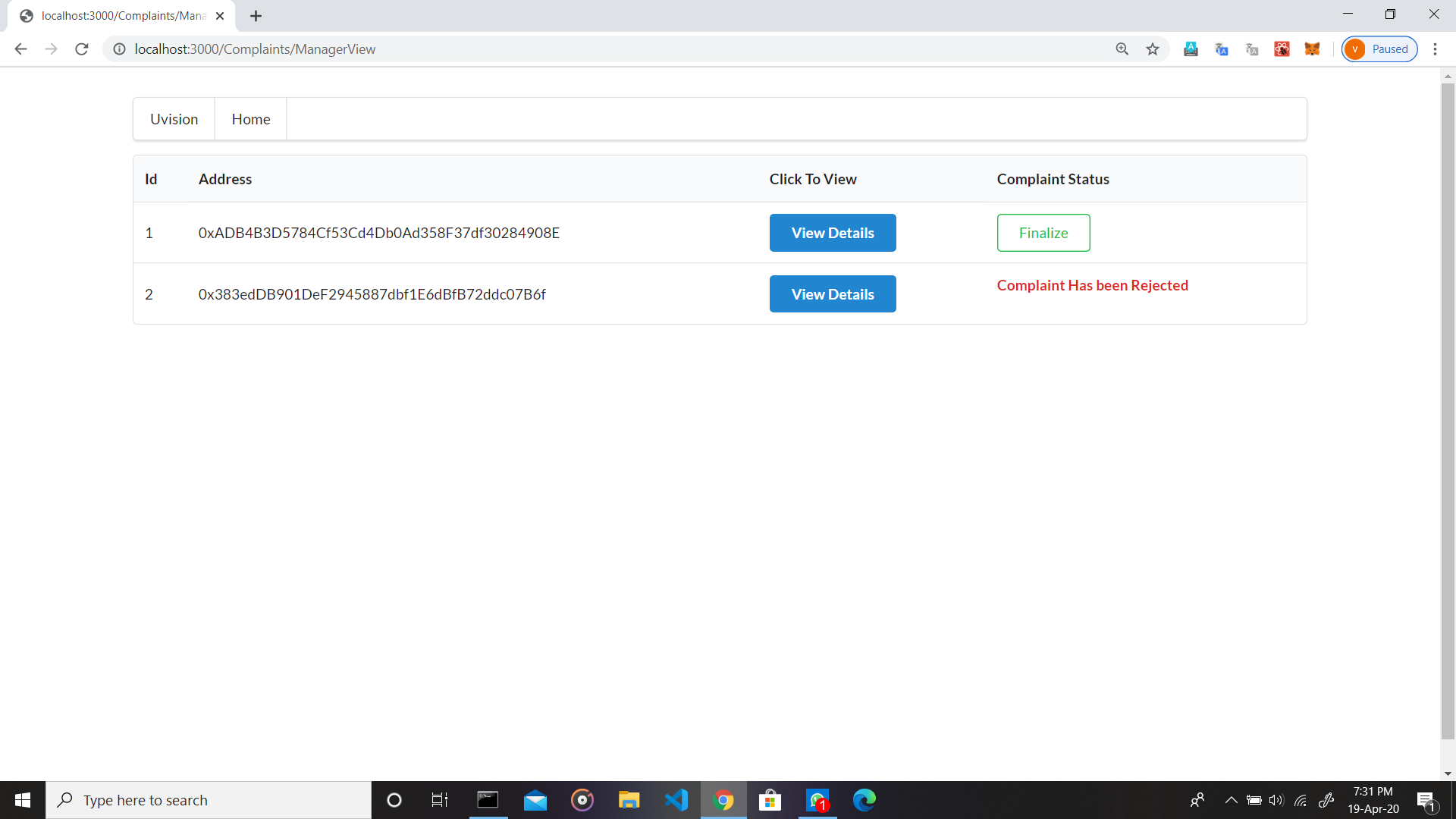


Figure 9: Government Officer module

At the Government Officer module, all the complaints which come under their jurisdiction will be listed based on the complaint location as shown in fig 9. The complete details of the complaint are visible to them. The required action will be taken on the basis of the complaint.

VIII. CONCLUSION

To overcome the drawback of the existing system of grievance redressal, this paper proposes a new web application based on blockchain technology. Using ethereum blockchain technology makes the complaint redressal systems more transparent and accountable. This application improves the people’s trust in the government. As each user keeps their copy of data with themselves, a middle man cannot modify or tamper the data.

REFERENCE

|  |  |
| --- | --- |
| [1] | Shubham Patil, Shreekant Khadsan, Saurabh Virkar, Kartik Dhankude, Mr Ramdas Jare “Integrated Municipal Service Application,” in ISSN 2278-1021 *International Journal of Advanced Research in Computer and Communication Engineering* Vol. 7, Issue 11, November 2018. |
| [2] | Ying Ma, Yu Sun, Yunjie Lei, Nan Qin, Junwen Lu, “A survey of blockchain technology on security, privacy, and trust in crowdsourcing services” in *Special Issue on Trust, Privacy and Security in Crowdsourcing Computing.* |
| [3] | Yinsheng Li, “Emerging blockchain‐based applications and techniques”. |
| [4] | Tsung-Ting Kuo, Hyeon-Eui Kim, and Lucila Ohno-Machado, “Blockchain distributed ledger technologies for biomedical and health care applications” in *Journal of the American Medical Informatics Association.* |
| [5] | Krishnendu Chatterjee, Amir Kafshdar Goharshady, and Yaron Velner, “*Quantitative Analysis of Smart Contracts*” |
| [6] | Constantinos Patsakis, Fran Casino, “Hydras and IPFS: a decentralised playground for malware” in the International *Journal of Information Security.* |
| [7] | Steichen, M., Fiz, B., Norvill, R., Shbair, W., & State, R. (2018), “Blockchain-Based, Decentralized Access Control for IPFS” in *2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing, Smart Data, Blockchain, Computer and Information Technology, Congress on Cybermatics.* |
| [8] | Kumar Bhosale, Kadaya Akbarabbas, Jadhav Deepak, Awani Sankhe, “Blockchain based Secure Data Storage” in *International Research Journal of Engineering and Technology (IRJET)* e-ISSN: 2395-0056 Volume: 06 Issue: 03 | Mar 2019. |
| [9] | Trupti Bomble et al,”Android Based Complaint Management System for Municipal Corporation” in *International Journal of Engineering Research and Applications* ISSN : 2248-9622, Vol. 5, Issue 4, ( Part -3) April 2015, pp.64-66 . |
| [10] | Devika Radhakrishnan, Nisarg Gandhewar, Ruchita Narnaware,Prayas Pagade, Arpan Tiwari and Pooja, “Smart Complaint Management System” in *International Journal of Trend in Research and Development,* Volume 3(6), ISSN: 2394-9333 Nov-Dec 2016. |
| [11] | Fibina F.Maheen, Sumithra M.D ,“Development of Smart Complaint Portal based on Geotagging and Proximity Search” in *International Research Journal of Engineering and Technology* Volume: 05 Issue: 07 | July 2018 e-ISSN: 2395-0056. |
| [12] | Sudeep J, Abhiram R, Adithya U S Vaidya, Rajath R Urs, Vallabh Joshi,” Smart Application for Complaint Registration” in *International Research Journal of Engineering and Technology (IRJET)* Volume: 06 Issue: 05 | May 2019. |